Introduction to Computer Communications
Computer Communications
Communications

Yuval Shavitt

Mon: 10:00 – 12:00

Wed: 16:00 – 18:00 option 18:00 – 20:00

Office hours:

Mon. 12:00 – 13:00 @ room 303 S/W Eng. Bldg.

T.A.s:

Eli Brosh, room 310 S/W Eng. Bldg.

Zki Lotker

Final Exam: Mon, June 16th.

Moed Bet: Mon, Aug. 18th.

What is the course about?

- Data Networks
- How information is transferred between terminals.
- Issues, Principles, Protocols, Tools

Covered topics:

- Queueing Systems
- Multiple Access Protocols and their performance
- Routing
- Flow Control
- ARQ protocols
- Traffic Management/Engineering

Final Mark structure

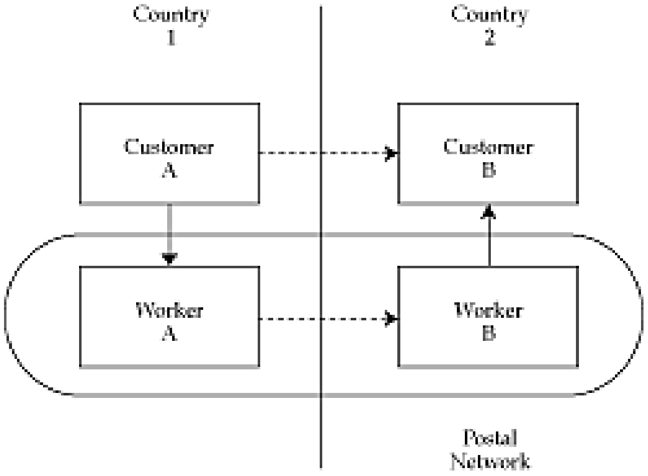
Final exam:	70-90%
Home assignments	10-30%

Sources

- General
  - D. Bertsekas and R. Gallager. *Data Networks,* 2nd Ed., 1992. P-H.
  - S. Keshav. *An Engineering Approach to Computer Networking. 1997. E-W*
  - J.F. Kurose and K.W. Ross. *Computer Networking.* 2000, E-W.
- Multiple Access
  - R. Rom and M. Sidi. *Multiple Access Protocols*. 1990. Springer-Verlag
- Queueing Systems
  - L. Kleinrock. *Queueing Systems*, *Vol. 1*. 1975. Wiley

Protocol Layering

Peer entities



- Customer A and B are *peers*
- Postal worker A and B are *peers*

Protocols

- A *protocol* is a set of rules and formats that govern the communication between communicating peers
  - set of valid messages
  - meaning of each message
- A protocol is necessary for any function that requires cooperation between peers

Example

- Exchange a file over a network that corrupts packets
  - but doesn't lose or reorder them
- A simple protocol
  - send file as a series of packets
  - send a *checksum*
  - receiver sends OK or not-OK message
  - sender waits for OK message
  - if no response, resends entire file
- Problems
  - single bit corruption requires retransmission of entire file
  - what if link goes down?
  - what if not-OK message itself is corrupted?

What does a protocol tell us?

- *Syntax* of a message
  - what fields does it contain?
  - in what format?
- *Semantics* of a message
  - what does a message mean?
  - for example, not-OK message means receiver got a corrupted file
- *Actions* to take on receipt of a message
  - for example, on receiving not-OK message, retransmit the entire file

Another way to view a protocol

- As providing a *service*
- The example protocol provides *reliable file transfer service*
- Peer entities use a protocol to provide a service to a higher-level peer entity
  - for example, postal workers use a protocol to present customers with the abstraction of an *unreliable letter transfer* service

Protocol layering

- A network that provides many services needs many protocols
- Turns out that some services are independent
- But others depend on each other
- Protocol A may use protocol B as a *step* in its execution
  - for example, packet transfer is one step in the execution of the example reliable file transfer protocol
- This form of dependency is called *layering*
  - reliable file transfer is *layered* above packet transfer protocol
  - like a subroutine

Some terminology

- *Service access point (SAP)*
    - interface between an upper layer and a lower layer
- *Protocol data units (PDUs)*
    - packets exchanged between peer entities
- *Service data units (SDUs)*
    - packets handed to a layer by an upper layer
- PDU = SDU + optional header or trailer
- Example
    - letter transfer service
    - protocol data unit between customers = letter
    - service data unit for postal service = letter
    - protocol data unit = mailbag (aggregation of letters)
    - (what is the SDU header?)

Protocol stack

- A set of protocol layers
- Each layer uses the layer below and provides a service to the layer above
- Key idea
  - once we define a service provided by a layer, we need know nothing more about the details of *how* the layer actually implements the service
  - information hiding
  - decouples changes

The importance of being layered

- Breaks up a complex problem into smaller manageable pieces
  - can compose simple service to provide complex ones
- Abstraction of implementation details
  - separation of implementation and specification
  - can change implementation as long as service interface is maintained
- Can reuse functionality
  - upper layers can share lower layer functionality
  - example: DNS

Problems with layering

- Layering hides information
  - if it didn't then changes to one layer could require changes everywhere
    - ☞ *layering violation*
- But sometimes hidden information can be used to improve performance
  - for example, flow control protocol may think packet loss is always because of network congestion
  - if it is, instead, due to a lossy link, the flow control breaks
  - this is because we hid information about reason of packet loss from flow control protocol

Layering

- There is a tension between information-hiding (abstraction) and achieving good performance
- Art of protocol design is to leak enough information to allow good performance
  - but not so much that small changes in one layer need changes to other layers

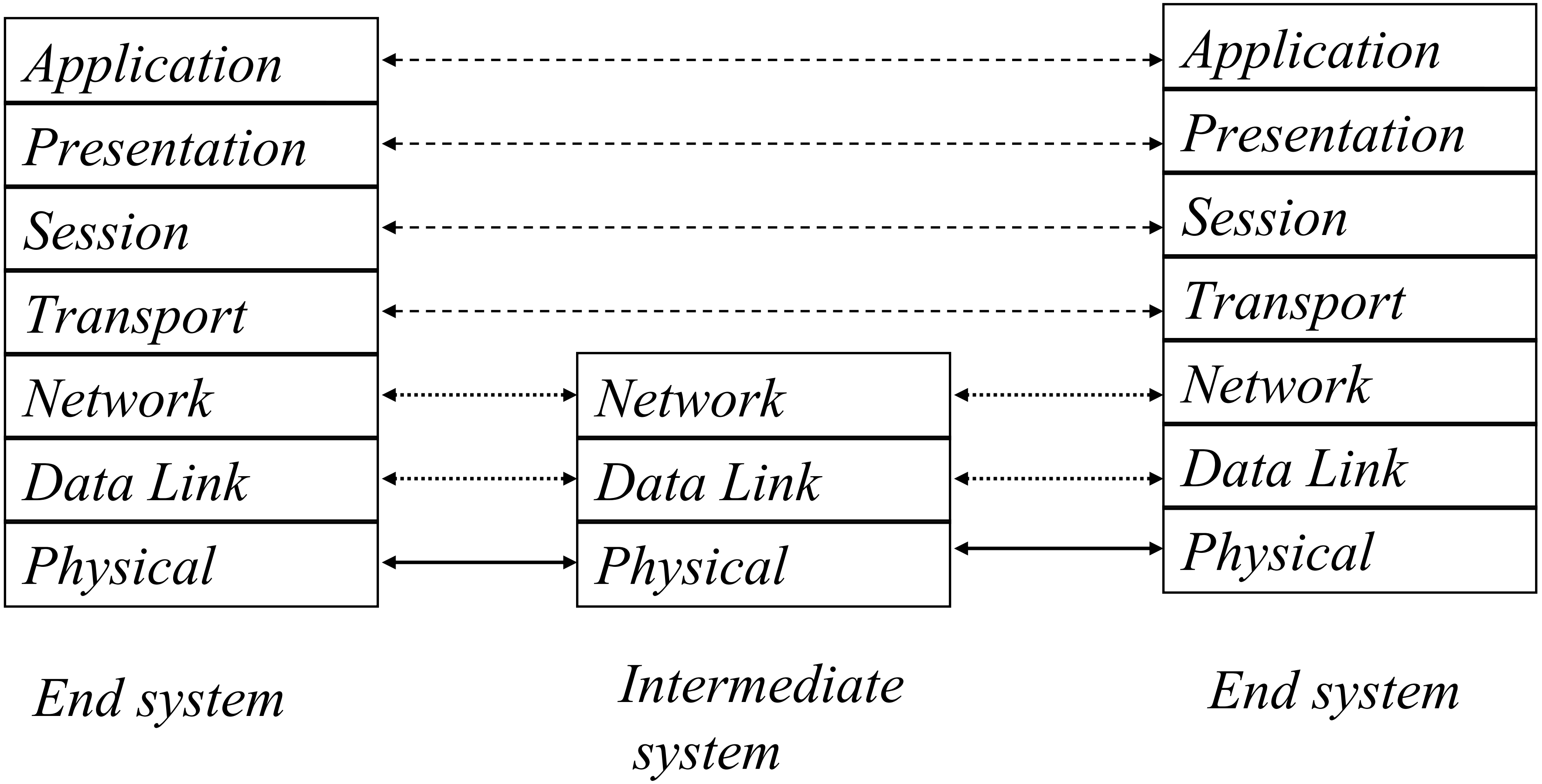
ISO OSI reference model

- A set of protocols is *open* if
  - protocol details are publicly available
  - changes are managed by an organization whose membership and transactions are open to the public
- A system that implements open protocols is called an *open system*
- International Organization for Standards (ISO) prescribes a standard to connect open systems
  - *open system interconnect (OSI)*
- Has greatly influenced thinking on protocol stacks

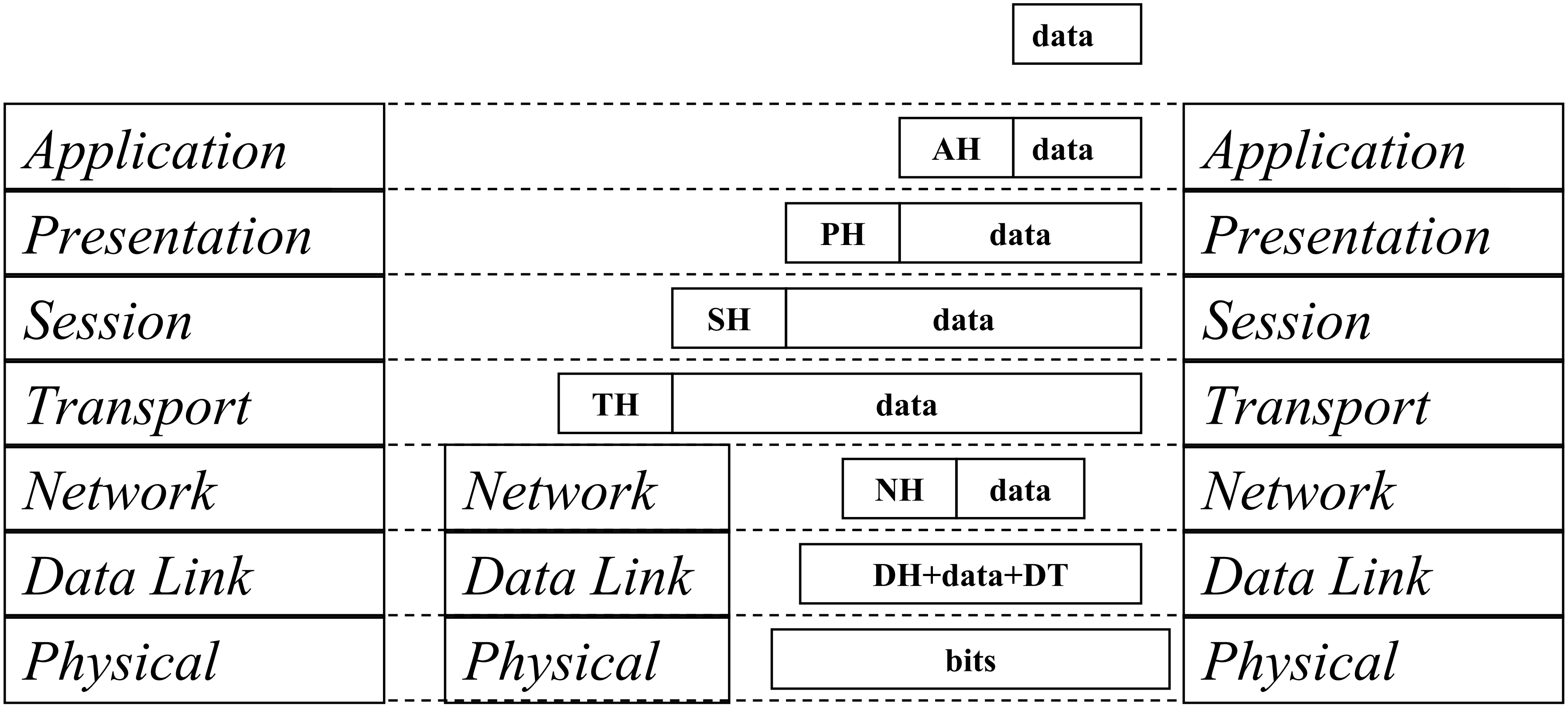
ISO OSI

- *Reference model*
  - formally defines what is meant by a layer, a service etc.
- *Service architecture*
  - describes the services provided by each layer and the service access point
- *Protocol architecture*
  - set of protocols that implement the service architecture
  - compliant service architectures may still use non-compliant protocol architectures

The seven Layers



The seven Layers - protocol stack



Physical layer

- **Moves bits between physically connected end-systems**
- Standard prescribes
  - coding scheme to represent a bit
  - shapes and sizes of connectors
  - bit-level synchronization
- Postal network
  - technology for moving letters from one point to another (trains, planes, vans, bicycles, ships…)
- Internet
  - technology to move bits on a wire, wireless link, satellite channel etc.

Datalink layer

- Reliable communication over a single link.
- Introduces the notion of a *frame*
  - set of bits that belong together
- *Idle* markers tell us that a link is not carrying a frame
- *Begin* and *end* markers delimit a frame
- On a broadcast link (such as Ethernet)
  - end-system must receive only bits meant for it
  - need datalink-layer address
  - also need to decide who gets to speak next
  - these functions are provided by *Medium Access sublayer (MAC)*

Datalink layer (contd.)

- Datalink layer protocols are the first layer of software
- Very dependent on underlying physical link properties
- Usually bundle both physical and datalink layer on *host adaptor card*
  - example: Ethernet
- Postal service
  - mail bag ‘frames’ letters
- Internet
  - a variety of datalink layer protocols
  - most common is Ethernet
  - others are FDDI, SONET, HDLC

Network layer

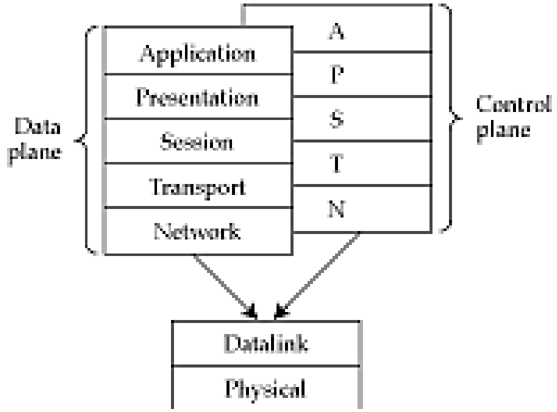
- Carrying data from source to destination.
- Logically concatenates a set of links to form the abstraction of an end-to-end link
- Allows an end-system to communicate with any other end-system by computing a route between them
- Hides idiosyncrasies of datalink layer
- Provides unique network-wide addresses
- Found both in end-systems and in intermediate systems
- At end-systems primarily hides details of datalink layer
  - segmentation and reassembly
  - error detection

Network layer (contd.)

- At intermediate systems
  - participates in routing protocol to create routing tables
  - responsible for forwarding packets
  - scheduling the transmission order of packets
  - choosing which packets to drop

Two types of network layers

- In datagram networks
  - provides both routing and data forwarding
- In connection-oriented network
  - we distinguish between data plane and control plane
  - data plane only forwards and schedules data (touches every byte)
  - control plane responsible for routing, call-establishment, call-teardown (doesn't touch data bytes)



Network layer

- Postal network
  - set up internal routing tables
  - forward letters from source to destination
  - static routing
  - multiple qualities of service
- Internet
  - network layer is provided by Internet Protocol
  - found in all end-systems and intermediate systems
  - provides abstraction of end-to-end link
  - segmentation and reassembly
  - packet-forwarding, routing, scheduling
  - unique IP addresses
  - can be layered over anything, but only best-effort service

Transport layer

- Reliable end-to-end communication.
- Network provides a ‘raw’ end-to-end service
- Transport layer creates the abstraction of an *error-controlled*, *flow-controlled* and *multiplexed* end-to-end link
- Error control
  - message will reach destination despite packet loss, corruption and duplication
  - retransmit lost packets; detect, discard, and retransmit corrupted packets; detect and discard duplicated packets
- Flow control
  - match transmission rate to rate currently sustainable on the path to destination, and at the destination itself

Transport layer (contd.)

- Multiplexes multiple applications to the same end-to-end connection
  - adds an application-specific identifier (*port number)* so that receiving end-system can hand in incoming packet to the correct application
- Some transport layers provide fewer services
  - e.g. simple error detection, no flow control, and no retransmission
  - *lightweight transport layer*

Transport layer (contd.)

- Postal system
  - doesn't have a transport layer
  - implemented, if at all, by customers
  - detect lost letters (how?) and retransmit them
- Internet
  - two popular protocols are TCP and UDP
  - TCP provides error control, flow control, multiplexing
  - UDP provides only multiplexing

Session layer

- Not common
- Provides *full-duplex service, expedited data delivery,* and *session synchronization*
- Token management.
- Duplex
  - if transport layer is simplex, concatenates two transport endpoints together
- Expedited data delivery
  - allows some messages to skip ahead in end-system queues, by using a separate low-delay transport layer endpoint
- Synchronization
  - allows users to place marks in data stream and to roll back to a prespecified mark

Example

- Postal network
  - suppose a company has separate shipping and receiving clerks
  - chief clerk can manage both to provide abstraction of a duplex service
  - chief clerk may also send some messages using a courier (expedited service)
  - chief clerk can arrange to have a set of messages either delivered all at once, or not at all
- Internet
  - doesn’t have a standard session layer

Presentation layer

- Unlike other layers which deal with *headers* presentation layer touches the application data
- Hides data representation differences between applications
  - e.g. *endian-ness*
  - *characters (ASCII, unicode, EBCDIC.)*
- Can also encrypt data
- Usually *ad hoc*
- Postal network
  - translator translates contents before giving it to chief clerk
- Internet
  - no standard presentation layer
  - only defines network byte order for 2- and 4-byte integers

Application layer

- The set of applications that use the network
- Doesn't provide services to any other layer
- Postal network
  - the person who uses the postal system
  - suppose manager wants to send a set of recall letters
  - translator translates letters going abroad
  - chief clerk sends some priority mail, and some by regular mail
  - mail clerk sends a message, retransmits if not acked
  - postal system computes a route and forwards the letters
  - datalink layer: letters carried by planes, trains, automobiles
  - physical layer: the letter itself

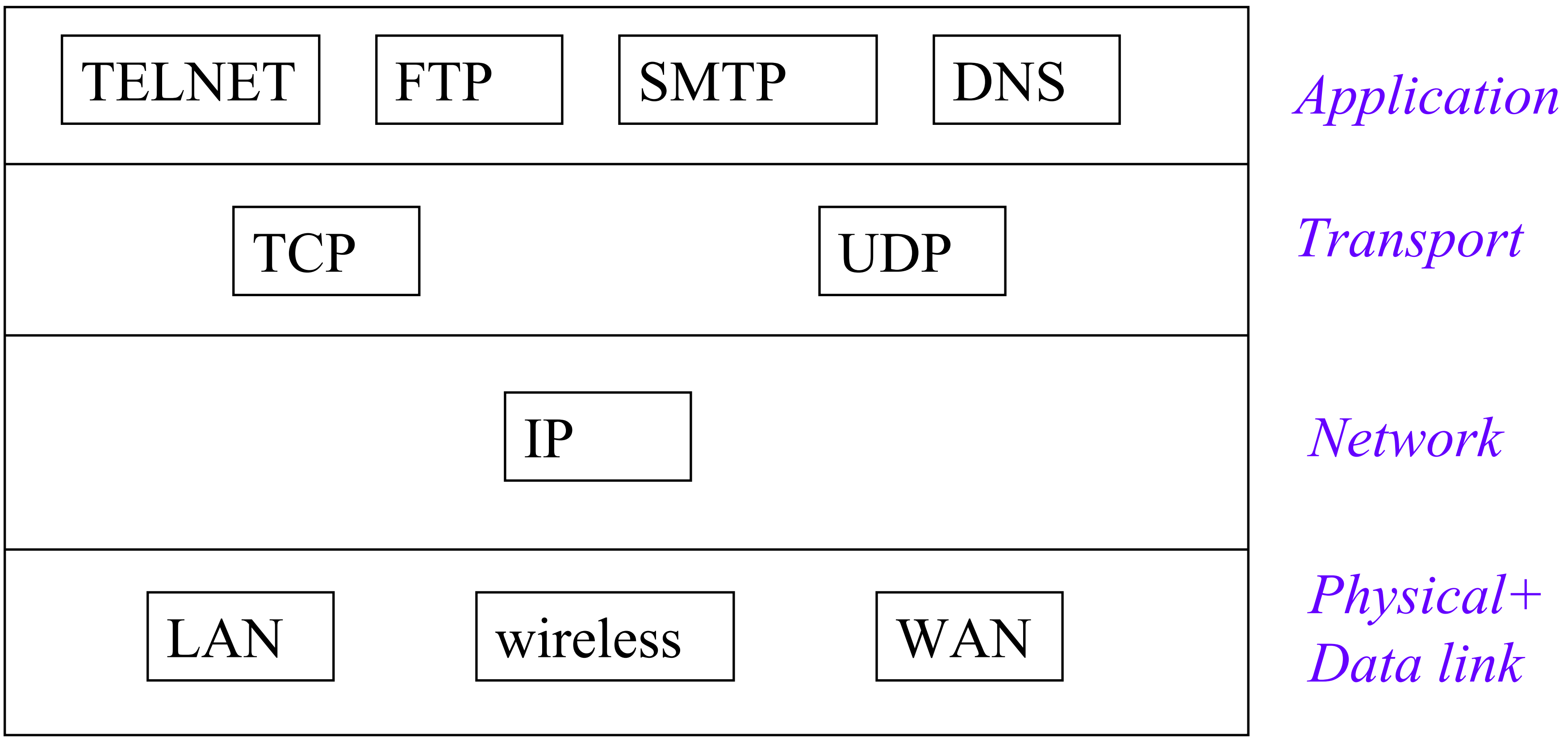
Layering

- We have broken a complex problem into smaller, simpler pieces
- Provides the application with *sophisticated* services
- Each layer provides a clean abstraction to the layer above

Why seven layers?

- Need a top and a bottom -- 2
- Need to hide physical link, so need datalink -- 3
- Need both end-to-end and hop-by-hop actions; so need at least the network and transport layers -- 5
- Session and presentation layers are not so important, and are often ignored
- So, we need at least 5, and 7 seems to be excessive
- Note that we can place functions in different layers

TCP/IP Protocols



Remarks on Layering

- Layer mixing (TCP/IP)
- Functionality duplications: checksums, encryption,…
- What is a layer *x* function?
- the *end-to-end principle*:
  - the network is fast and dumb, the intelligence is in the edges
  - thus, inside the networks we only have layers 1-3,
  - and, every function that can be done end-to-end will not be done inside the network.

The *End-to-End Principle* - Reality Check

- L4-7 switching
- **Computation is done in the network**:
  - firewalls,
  - proxies,
  - gateways (multimedia/wireless),
  - NAT,
  - etc.
- Maybe making the network smart is OK?
  - we do it anyway
  - it can optimize operation
  - it give us flexibility and tailorability
- Is it time for Active Networks?

The Telephone Network

The Good Old Ubiquitous Network

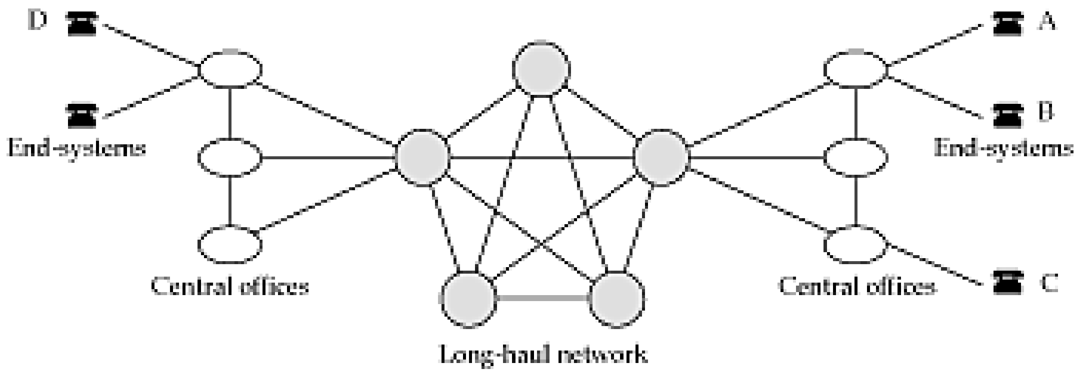
Is it a computer network?

- Specialized to carry voice
- Also carries
  - telemetry
  - video
  - fax
  - modem calls
- Internally, uses digital *samples*
- Switches and switch controllers are special purpose computers
- Principles in its design apply to more general computer networks

Concepts

- Single basic service: two-way voice
  - low end-to-end delay
  - guarantee that an accepted call will run to completion
- Endpoints connected by a *circuit*
  - like an electrical circuit
  - signals flow both ways (*full duplex*)
  - associated with bandwidth and buffer *resources*

The big picture



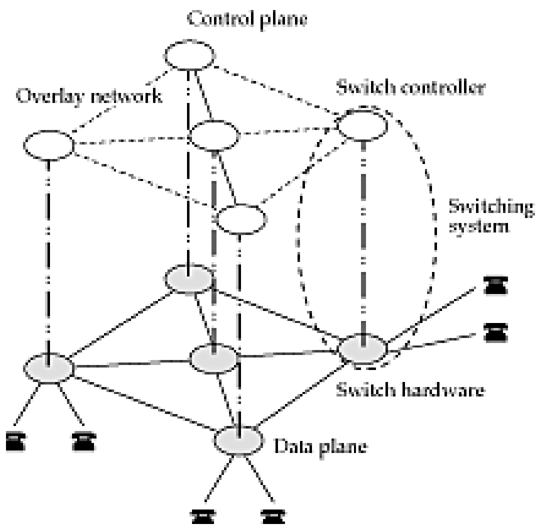
- Fully connected core
  - simple routing
  - telephone number is a hint about how to route a call
    - but not for 800/888/900 numbers
  - hierarchically allocated telephone number space

The pieces

1. End systems: telephones, faxes, …
2. Transmission
3. Switching
4. Signaling

Switching

- Problem:
  - each user can potentially call any other user
  - can't have direct lines!
- Switches establish temporary *circuits*
- Switching systems come in two parts: switch and switch controller

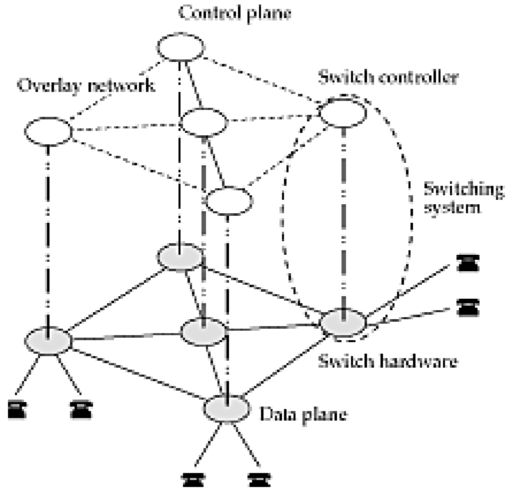


Signaling

- Recall that a switching system has a switch and a switch controller
- Switch controller is in the *control* plane
  - does not touch voice samples
- Manages the network
  - call routing (collect *dialstring* and forward call)
  - alarms (ring bell at receiver)
  - billing
  - directory lookup (for 800/888 calls)

Signaling network

- Switch controllers are special purpose computers
- Linked by their own internal computer network
  - *Common Channel Interoffice Signaling (CCIS) network*
- Earlier design used *in-band* tones, but was severely hacked
- Also was very rigid (why?)
- Messages on CCIS conform to *Signaling System 7 (SS7)* spec.



Challenges for the telephone network

- Multimedia
  - simultaneously transmit voice/data/video over the network
  - people seem to want it
  - existing network can't handle it
    - bandwidth requirements
    - *burstiness* in traffic (TSI can't skip input)
    - change in statistical behavior
- Backward compatibility of new services
  - huge existing infrastructure
  - idiosyncrasies
- Regulation
  - stifles innovation

Challenges

- Competition
  - future telephone networks will no longer be monopolies
  - how to manage the transition?
- Inefficiencies in the system
  - an accumulation of cruft
  - special-purpose systems of the past
  - ‘legacy’ systems
  - need to change them without breaking the network

The Internet

The network of networks

My how you’ve grown!

- The Internet has doubled in size every year since 1969
- In July 2000: **93,000,000** hosts
- Currently about 79 new hosts join every minute.
- Soon, everyone who has a phone is likely to also have an email account
  - Pacific Bell telephone directories are planning to include email addresses in white pages

What does it look like?

- Loose collection of networks organized into a multilevel hierarchy
  - 10-100 machines connected to a *hub* or a *router*
    - service providers also provide direct dialup access
    - or over a wireless link
  - 10s of routers on a *department backbone*
  - 10s of department backbones connected to *campus backbone*
  - 10s of campus backbones connected to *regional service providers*
  - 100s of regional service providers connected by *national backbone*
  - 10s of national backbones connected by *international trunks*

A Routing Example

```
[12:02pm]bakara:~> traceroute www.bell-labs.com
traceroute to www.bell-labs.com (204.178.16.43), 30 hops max, 40 byte packets
 1  * cisco.eng.tau.ac.il (132.66.48.1)  80.615 ms  12.670 ms
 2  kir.tau.ac.il (132.66.4.129)  1.166 ms  1.299 ms  1.512 ms
 3  tel-aviv.tau.ac.il (132.66.4.1)  2.239 ms  2.437 ms  1.978 ms
 4  gp1-mag.ilan.net.il (128.139.198.80)  2.569 ms  2.626 ms  2.261 ms
 5  tau-gp2-fe-i1.ilan.net.il (192.114.99.49)  3.068 ms  3.678 ms  2.953 ms
 6  chi-gp3-0.ilan.net.il (192.114.99.65)  186.263 ms  186.351 ms  187.445 ms
 7  chi-gp4-fe-i1.ilan.net.il (192.114.101.50)  188.897 ms  185.726 ms  186.886 ms
 8  207.112.240.113 (207.112.240.113)  191.239 ms  189.516 ms  189.222 ms
 9  p3-1.nchicago2-br2.bbnplanet.net (4.0.6.1)  189.429 ms  191.585 ms  196.991 ms
10  p1-0.nchicago2-br1.bbnplanet.net (4.0.1.145)  189.862 ms  190.833 ms  188.593 ms
11  p6-2.chcgil1-br2.bbnplanet.net (4.0.5.209)  193.674 ms  188.560 ms  195.565 ms
12  p4-0.chcgil1-br1.bbnplanet.net (4.24.5.225)  199.049 ms  187.375 ms  187.873 ms
13  so-4-1-0.chcgil2-br1.bbnplanet.net (4.24.9.69)  191.858 ms  195.277 ms  190.537 ms
14  p10-0.nycmny1-nbr1.bbnplanet.net (4.24.9.66)  212.121 ms  210.855 ms  214.814 ms
15  p1-0.nycmny1-br1.bbnplanet.net (4.24.10.82)  210.330 ms  207.278 ms  209.605 ms
16  p4-0.nycmny1-br2.bbnplanet.net (4.24.6.226)  211.959 ms  214.922 ms  212.422 ms
17  p7-0.nycmny1-ba2.bbnplanet.net (4.24.6.234)  214.183 ms  216.112 ms  207.337 ms
18  192.205.32.153 (192.205.32.153)  209.345 ms  211.558 ms  215.922 ms
19  gbr3-p50.n54ny.ip.att.net (12.123.1.122)  211.942 ms  206.762 ms  207.737 ms
20  gbr5-p60.n54ny.ip.att.net (12.122.5.105)  210.346 ms  207.418 ms *
21  ar4-p380.n54ny.ip.att.net (12.123.1.149)  208.590 ms  209.746 ms  223.931 ms
22  12.126.222.246 (12.126.222.246)  208.821 ms  405.484 ms  303.498 ms
23  207.140.138.66 (207.140.138.66)  228.353 ms  209.717 ms  588.706 ms
24  www.bell-labs.com (204.178.16.43)  350.651 ms  211.687 ms  210.527 ms
```

Intranet, Internet, and Extranet

- Intranets are administered by a single entity
  - e.g. Cornell campus network
- Internet is administered by a coalition of entities
  - name services, backbone services, routing services etc.
- Extranet is a marketing term
  - refers to exterior customers who can access privileged Intranet services
  - e.g. Cornell could provide ‘extranet’ services to Ithaca college

What holds the Internet together?

- Addressing
  - how to refer to a machine on the Internet
- Routing
  - how to get there
- Internet Protocol (IP)
  - what to speak to be understood

Example: joining the Internet

- How can people talk to you?
  - get an IP address from your administrator
- How do you know where to send your data?
  - if you only have a single external connection, then no problem
  - otherwise, need to speak a routing protocol to decide next hop
- How to format data?
  - use the IP format so that intermediate routers can understand the destination address
- If you meet these criteria--you're on the Internet!
- Decentralized, distributed, and chaotic
  - but it scales (why?)

What lies at the heart?

- Two key technical innovations
  - packets
  - store and forward

Packets

- Self-descriptive data
  - packet = data + metadata (header)
- Packet vs. sample
  - samples are not self descriptive
  - to forward a sample, we have to know *where* it came from and *when*
  - can't store it!
  - hard to handle bursts of data

Store and forward

- Metadata allows us to forward packets when we want
- E.g. letters at a post office headed for main post office
  - address labels allow us to forward them in batches
- Efficient use of critical resources
- Three problems
  - hard to control delay within network
  - switches need memory for buffers
  - convergence of flows can lead to congestion

Key features of the Internet

- Addressing
- Routing
- Endpoint control

Addressing

- Internet addresses are called IP addresses
- Refer to a *host interface*: need one IP address per interface
- Addresses are structured as a two-part hierarchy
  - network number
  - host number

135.105.53	*100*

An interesting problem

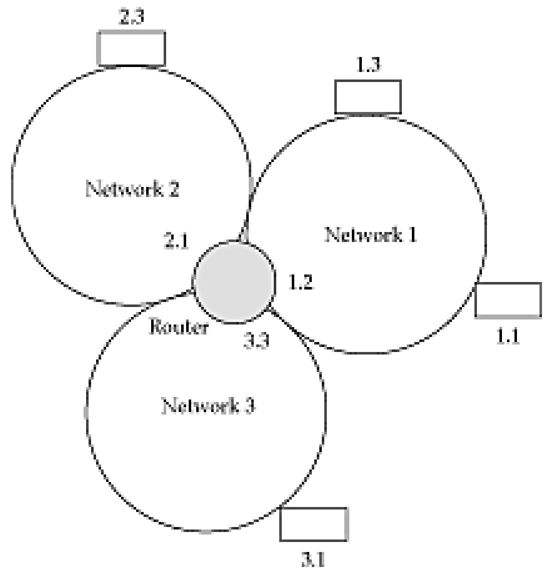
- How many bits to assign to host number and how many to network number?
- If many networks, each with a few hosts, then more bits to network number
- And *vice versa*
- But designer’s couldn’t predict the future
- Decided three sets of partitions of bits
  - class A: 8 bits network, 24 bits host
  - class B: 16 bits each
  - class C: 24 bits network, 8 bits host

Addressing (contd.)

- To distinguish among them
  - use leading bit
  - first bit 0 => class A
  - first bits 10 => class B
  - first bits 110 => class C
  - (what class address is 132.66.48.1?)
- Problem
  - if you want more than 256 hosts in your network, need to get a class B, which allows 64K hosts => wasted address space
- Solution
  - associate *every* address with a *mask* that indicates partition point
  - *CIDR*

Routing

- How to get to a destination given its IP address?
- We need to know the next hop to reach a particular network number
  - this is called a *routing table*
  - computing routing tables is non-trivial
- Simplified example



Default routes

- Strictly speaking, need next hop information for every network in the Internet
    - > 100,000 now
- Instead, keep detailed routes only for local neighborhood
- For unknown destinations, use a *default* router
- Reduces size of routing tables at the expense of non-optimal paths

Endpoint control

- Key design philosophy: "the end-to-end principle"
  - do as much as possible at the endpoint
  - dumb network

  - exactly the opposite philosophy of telephone network
- Layer above IP compensates for network defects
  - Transmission Control Protocol (TCP)
- Can run over any available link technology
    - ☞ but no quality of service
    - ☞ modification to TCP requires a change at every endpoint
    - ☞ (how does this differ from telephone network?)

Challenges

- IP address space shortage
  - because of free distribution of inefficient Class B addresses
  - decentralized control => hard to recover addresses, once handed out
- Decentralization
  - allows scaling, but makes *reliability* next to impossible
  - cannot guarantee that a route exists, much less bandwidth or buffer resources
  - single points of failure can cause a major disaster
    - and there is no control over who can join!
  - hard to guarantee security
    - end-to-end encryption is a partial solution
    - who manages keys?

Challenges (contd.)

- Decentralization (contd.)
  - no uniform solution for accounting and billing
    - ☞ can’t even reliably identify individual users
  - no equivalent of white or yellow pages
    - ☞ hard to reliably discover a user’s email address
  - nonoptimal routing
    - ☞ each administrative makes a locally optimal decision

Challenges (contd).

- Multimedia
  - requires network to support quality of service of some sort
    - hard to integrate into current architecture
    - store-and-forward => shared buffers => traffic interaction => hard to provide service quality
  - requires endpoint to signal to the network what it wants
    - but Internet does not have a simple way to identify streams of packets
    - nor are routers required to cooperate in providing quality
    - and what about pricing!