Internet architecture and history

Why did the Internet win?

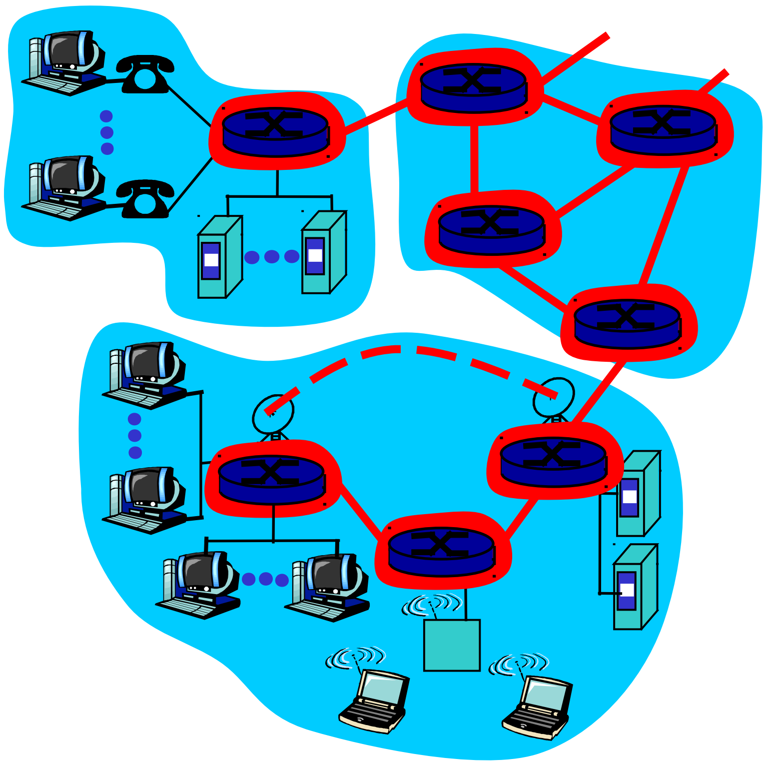
- Packet switching over circuit switching
- End-to-end principle and "Hourglass" design
- Layering of functionality
- Distributed design, decentralized control
- Superior organizational process

Packet switching versus Circuit switching

- Analogy
  - Zip cars vs. privately owned cars
- Zip cars (packet-switching)
  - Many users share a single car
  - Large demand for cars causes users to delay usage
  - Car is more efficiently used
- Privately owned cars (circuit-switching)
  - Single user
  - Guaranteed access for user
  - Car is not used as efficiently

Packet vs. circuit switching

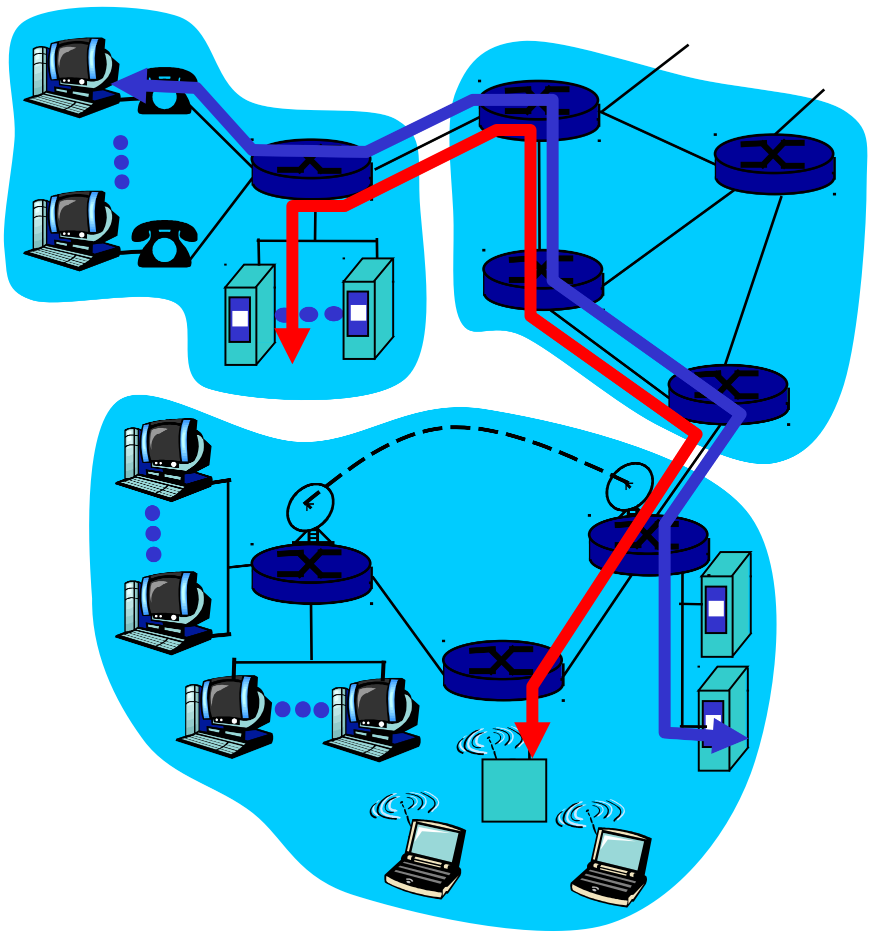
- mesh of interconnected routers
- *the* fundamental question: how is data transferred through net?



Circuit Switching

End-end resources reserved for "call"

- network resources (e.g., bandwidth) divided into "pieces"
  - link bandwidth, switch capacity
  - pieces allocated to calls
  - resource piece *idle* if not used by owning call
    - dedicated resources: no sharing
- circuit-like (guaranteed) performance
- call setup and admission control required



Case study: Circuit Switching

- 1890-current: Phone network
  - Fixed bit rate
  - Mostly voice
  - Not fault-tolerant
  - Components extremely reliable
  - Global application-level knowledge throughout network
  - Admission control at local switching station (dial-tone)

Packet Switching

each end-end data stream divided into *packets*

- user A, B packets *share* network resources
- each packet uses full link bandwidth
- resources used *as needed*

Bandwidth division into "pieces"

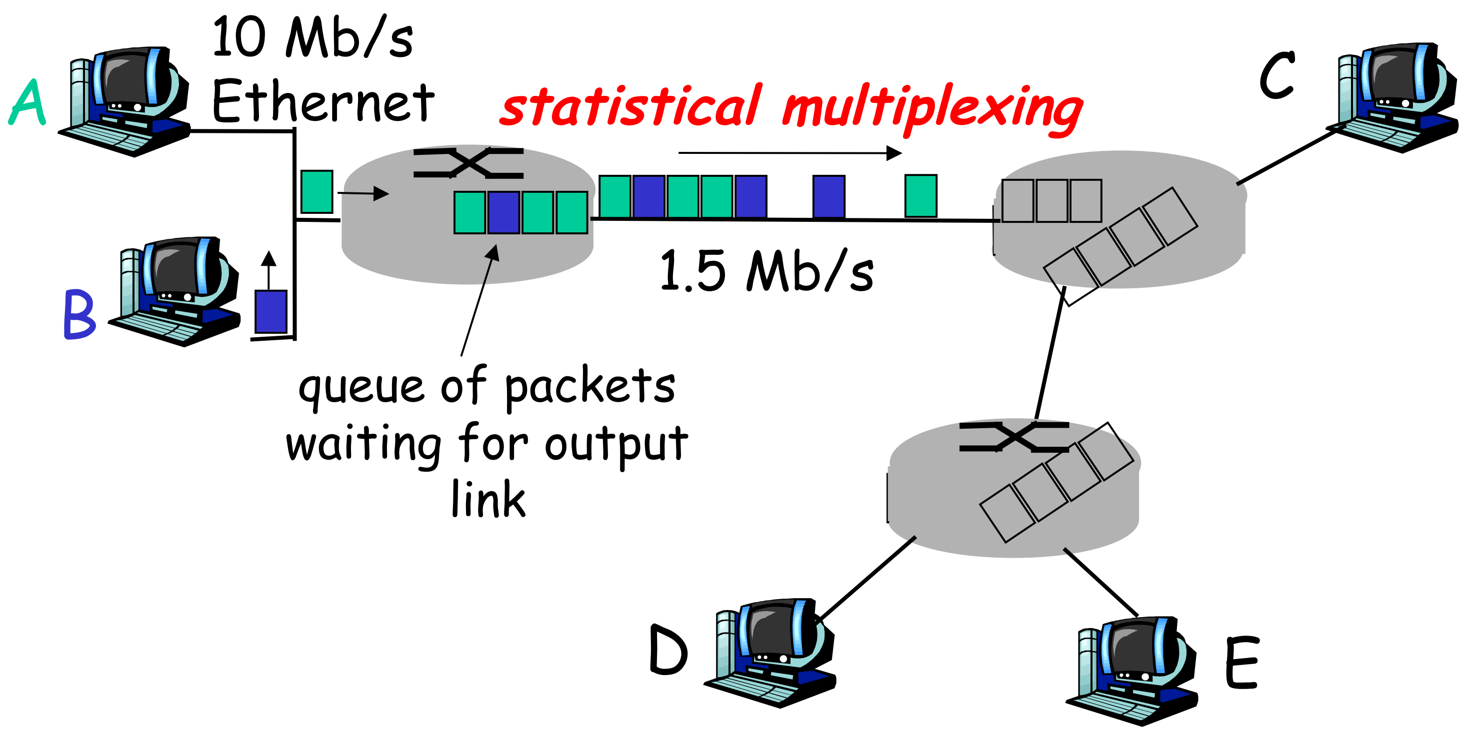
Dedicated allocation

Resource reservation

congestion:

- aggregate resource demand can exceed amount available
- packets queue, wait for link use
- store and forward: packets move one hop at a time

Packet Switching: Statistical Multiplexing

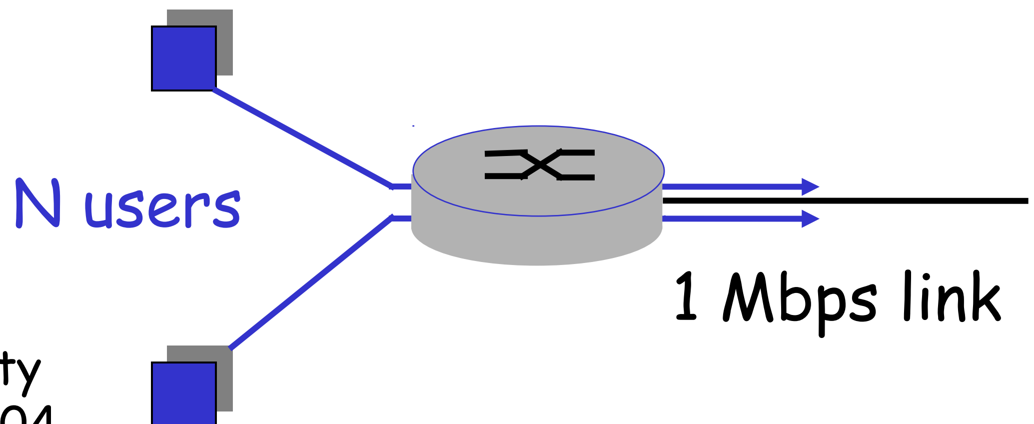


Sequence of A & B packets does not have fixed pattern, shared on demand ➨ ***statistical multiplexing***.

Packet switching versus circuit switching

Packet switching allows more users to use network

- N users over 1 Mb/s link
- each user:
  - 100 kb/s when "active"
  - active 10% of time
- circuit-switching:
  - 10 users
- packet switching:
  - with 35 users, probability > 10 active less than .0004
  - Allows more users to use network
  - "Statistical multiplexing gain"



Packet switching versus circuit switching

Is packet switching a “slam dunk winner?”

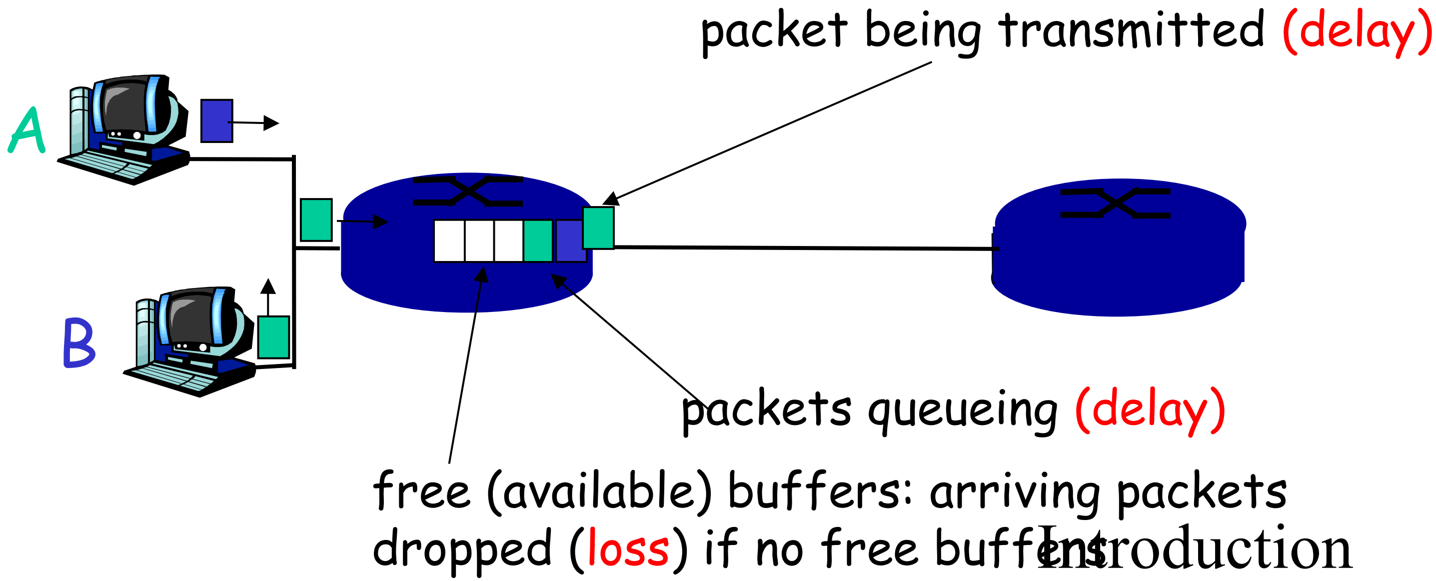
- Great for bursty data
  - resource sharing
  - simpler, no call setup
- Bad for applications with hard resource requirements
  - Excessive congestion: packet delay and loss
  - Need protocols to deal with packet loss/congestion
  - Applications must be written to handle congestion

Problems with packet switching

Packet loss and queuing delay

packets *queue* in router buffers

- packet arrival rate exceeds output link capacity
- packets queue, wait for turn
- packet arrives to full queue, it is dropped (aka lost)
  - lost packet may be retransmitted by previous node, by source end system, or not retransmitted at all



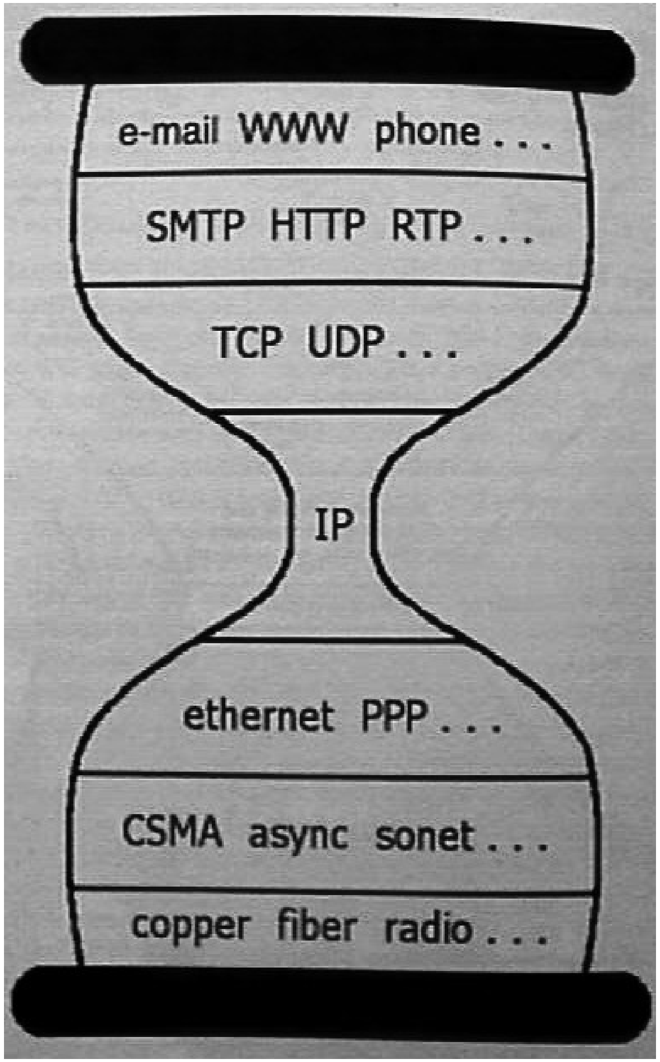
Case study: Packet Switching

- 1970/80s-current: Internet network
  - Variable bit rate
  - Mostly data
  - Fault-tolerant
  - Components not extremely reliable (versus phone components)
  - Distributed control and management

Why did the Internet win?

- Packet switching over circuit switching
- End-to-end principle and “Hourglass” design
- Layering of functionality
- Distributed design, decentralized control
- Superior organizational process

End-to-end principle and Hourglass design



Keep it simple, stupid!

- One, very simple protocol to run it all

End-to-end principle

- Where to put the functionality?
  - In the network? At the edges?
- End-to-end functions best handled by end-to-end protocols
  - Network provides basic service: data transport
  - Intelligence and applications located in or close to devices at the edge
- Leads to innovation at the edges
  - Phone network: dumb edge devices, intelligent network
  - Internet: dumb network, intelligent edge devices

Hourglass design

- End-to-end principle leads to “Hourglass” design of protocols
- Only one protocol at the Internet level
  - Minimal required elements at narrowest point
- IP – Internet Protocol
  - RFC 791 and 1812
  - Unreliable datagram service
  - Addressing and connectionless connectivity
  - Like the post office of old!

Hourglass design

- Simplicity allowed fast deployment of multi-vendor, multi-provider public network
  - Ease of implementation
  - Limited hardware requirements (important in 1970s)
    - Is it relevant now with today's semiconductor speeds?
  - Rapid development leads to eventual economies of scale
- Designed independently of hardware
  - No link-layer specific functions
  - Hardware addresses decoupled from IP addresses
  - IP header contains no data/physical link specific information (e.g. wired LAN, WiFi, 3G, etc.)
  - Allows IP to run over any fabric

Hourglass design

- Waist expands at transport layer
  - Network layer = host to host communication
  - Transport layer = application to application communication
- Two dominant services layered above IP
- TCP – Transmission Control Protocol
  - Connection-oriented service
  - RFC 793
- UDP – User Datagram Protocol
  - Connectionless service
  - RFC 768

Hourglass design

- TCP – Transmission Control Protocol
  - Reliable, in-order data transfer
    - Acknowledgements and retransmissions of lost data
  - Flow control
    - Sender won't overwhelm receiver
  - Congestion control
    - Senders won't overwhelm network
- UDP – User Datagram Protocol
  - Unreliable data transfer
  - No flow control
  - No congestion control

Hourglass design

- What uses TCP?
  - HTTP (Web), SMTP (E-mail transmission), IMAP, POP (E-mail access)
- What uses (mainly) UDP?
  - DNS, NTP (network time protocol), Highly interactive on-line games (First-Person Shooters)
  - Many protocols can use both
- Check out /etc/services on *nix or C:\WIN*\system32\services
- IANA
  - http://www.iana.org/assignments/port-numbers

Hourglass design

- Question?
  - Are TCP, UDP, and IP enough?
  - What other functionality would applications need?

Hourglass design

- Security?
  - IPsec/SSL/TLS
- Quality-of-service?
  - RSVP, int-serv, diff-serv
- Reliable, out-of-order delivery service?
  - SCTP
- Handling greedy sources?

End-to-end principle and the Hourglass design

- The good
  - Basic network functionality allowed for extremely quick adoption and deployment using simple devices
- The bad
  - New network features and functionality are impossible to deploy, requiring widespread adoption within the network
  - IP Multicast, QoS

Why did the Internet win?

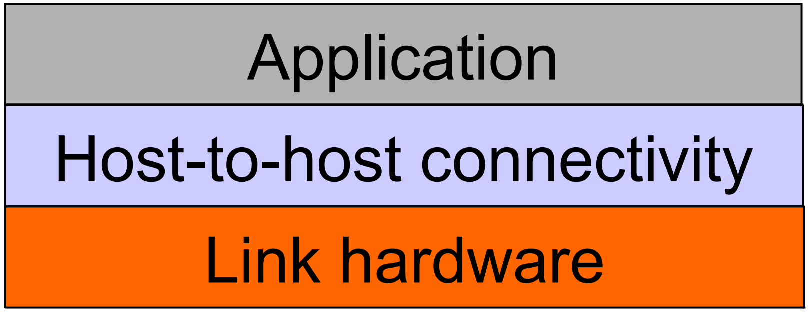
- Packet switching over circuit switching
- End-to-end principle and "Hourglass" design
- Layering of functionality
- Distributed design, decentralized control
- Superior organizational process

Layering

- Modular approach to network functionality
  - Simplifies complex systems
    - Each layer relies on services from layer below and exports services to layer above
  - Hides implementation
  - Eases maintenance and updating of system
    - Layer implementations can change without disturbing other layers (black box)

Layering

- Examples:
  - Topology and physical configuration hidden by network-layer routing
    - Applications require no knowledge of routes
      - e.g. web servers do not need to calculate routes to clients
    - New applications deployed without coordination with network operators or operating system vendors

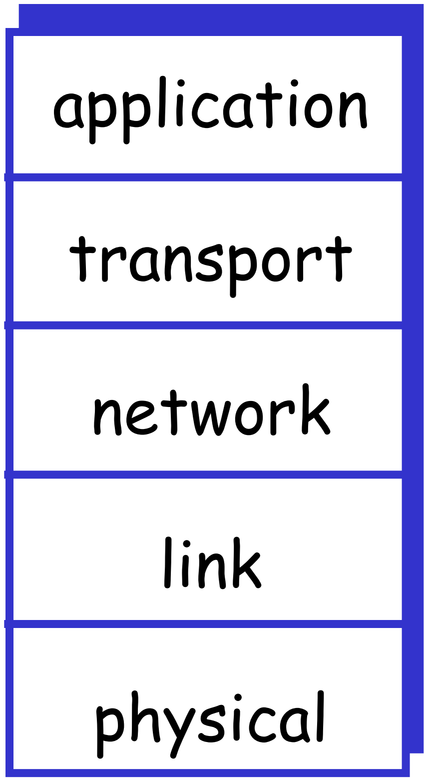


Layering essential in Protocols

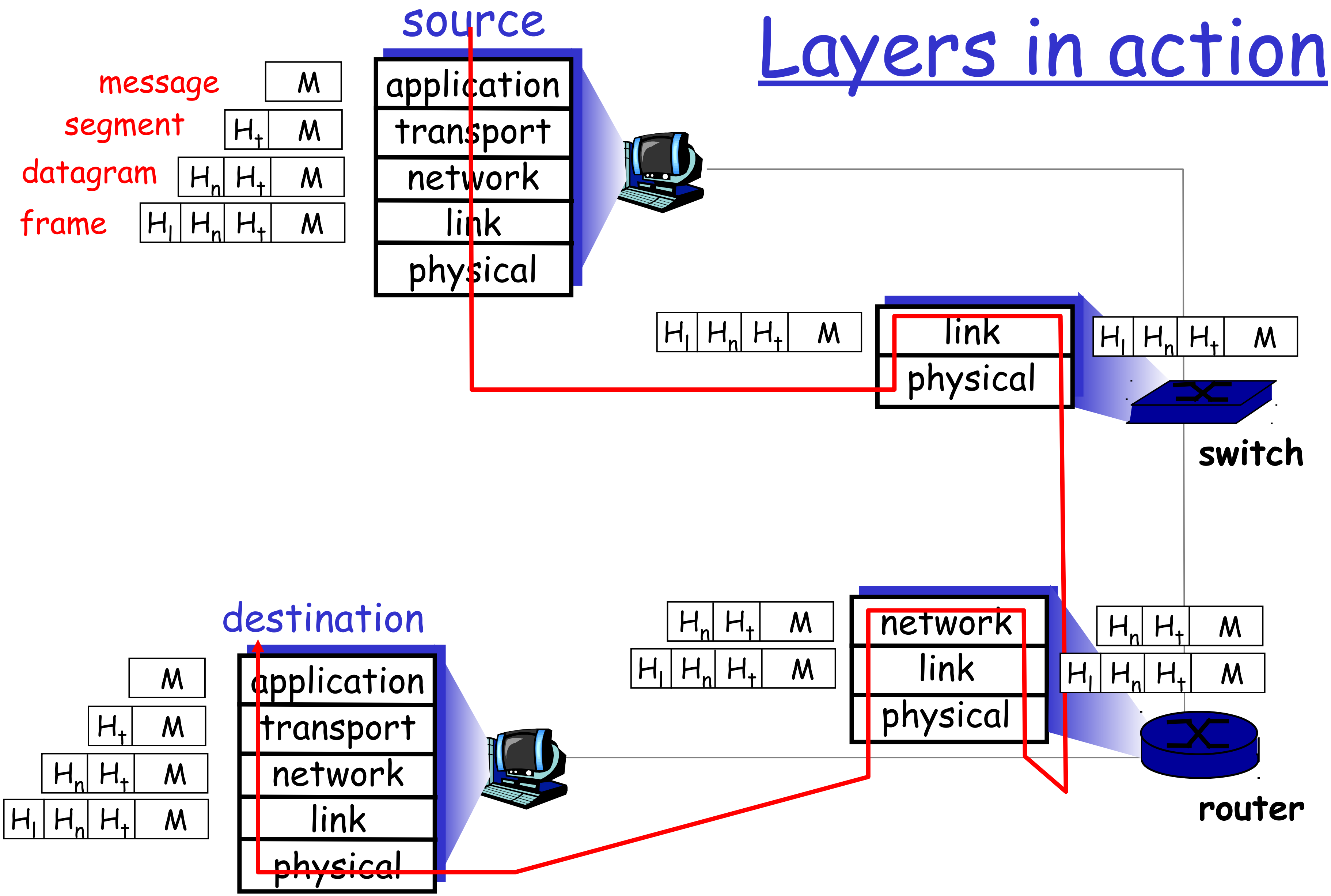
- Set of rules governing communication between network elements (applications, hosts, routers)
- Protocols specify:
  - Interface to higher layers (API)
  - Interface to peer
    - Format and order of messages
    - Actions taken on receipt of a message

Layering: Internet protocols

- application:
  - FTP, SMTP, HTTP
  - e.g. URL requests and responses
- transport: process-process data transfer
  - TCP, UDP
  - e.g. how those requests and responses are broken up into network packets
- network: routing of datagrams from source to destination
  - IP
  - e.g. how to deliver those packets to their destinations
- link: data transfer between neighboring network elements
  - Ethernet, 802.11
  - e.g. how to deliver those packets to the next hop
- physical: bits "on the wire"



Layers in action



Why did the Internet win?

- Packet switching over circuit switching
- End-to-end principle and "Hourglass" design
- Layering of functionality
- Distributed design, decentralized control
- Superior organizational process

Distributed design and control

- Requirements from DARPA
  - Must survive a nuclear attack
- Reliability
  - Intelligent aggregation of unreliable components
  - Alternate paths, adaptivity
- Distributed management & control of networks
  - Allows individual networks to independently develop without large amounts of coordination
  - Exceptions: TLDs and TLD servers, IP address allocation (ICANN)

Why did the Internet win?

- Packet switching over circuit switching
- End-to-end principle and “Hourglass” design
- Layering of functionality
- Distributed design, decentralized control
- Superior organizational process

Superior organizational process

- IAB/IETF process allowed for quick specification, implementation, and deployment of new standards
  - Free and easy download of standards
  - Rough consensus and running code
  - 2 interoperable implementations
  - Bake-offs
  - http://www.ietf.org/
- ISO/OSI
  - Large cost to obtain copy of standards
  - Slow approval process
  - Standards measured by the inch!

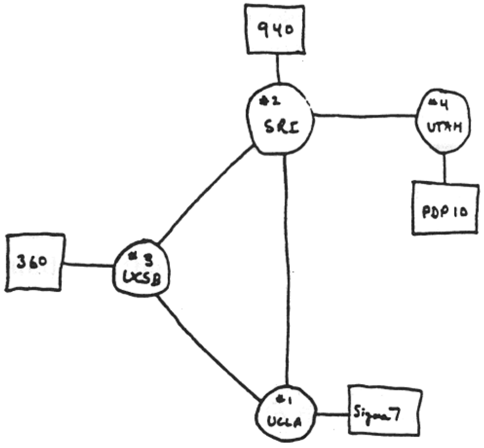
Internet history

Internet History

1961-1972: Early packet-switching principles

- 1961: Kleinrock - queueing theory shows effectiveness of packet-switching
- 1964: Baran - packet-switching in early military nets
- 1967: ARPAnet conceived by Advanced Research Projects Agency
- 1969: first ARPAnet node operational

- 1972:
  - ARPAnet public demonstration
  - NCP (Network Control Protocol) first host-host protocol
  - first e-mail program
  - ARPAnet has 15 nodes



THE ARPA NETWORK

Internet History

1972-1980: Internetworking, new and proprietary nets

- 1970's: proprietary network architectures developed: DECnet, SNA, XNA
- 1974: Cerf and Kahn - architecture for interconnecting networks
- 1976: Ethernet at Xerox PARC
- 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
- 1983: smtp e-mail protocol defined
- 1983: DNS defined for name-to-IP-address translation
- 1985: ftp protocol defined
- 1988: TCP congestion control

- Late 1980s, Early 1990s: new national networks: Csnet, BITnet, NSFnet, Minitel
  - 100,000 hosts connected to confederation of networks

Internet History

1990, 2000's: commercialization, the Web, new apps

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

Late 1990's – 2000's:

- more killer apps: instant messaging, P2P file sharing
- network security to forefront
- est. 50 million host, 100 million+ users
- backbone links running at Gbps

References

- National Research Council "The Internet's Coming of Age"
  - http://www.nap.edu/html/coming_of_age/
- RFC 1958
  - http://www.ietf.org/rfc/rfc1958.txt
- J. H. Saltzer, D. P. Reed and D. D. Clark "End-to-end arguments in system design", Transactions on Computer Systems, Vol. 2, No. 4, 1984
  - http://www.acm.org/pubs/citations/journals/tocs/1984-2-4/p277-saltzer/
- D. Clark, "The design philosophy of the DARPA Internet", SIGCOMM 1988, August 16 - 18, 1988.
  - http://www.acm.org/pubs/citations/proceedings/comm/52324/p106-clark/