# CS 204:
# Layering

Jiasi Chen

Lectures: MWF 12:10-1pm in WCH 139

http://www.cs.ucr.edu/~jiasi/teaching/cs204_spring16/

# Overview

- How to read
- History
- Layering
  - Inter-LAN (small)
  - Inter-LAN (large)
  - Evolution of layers
- Paper discussion

Q: How to design the Internet from the ground up?

# Why You Need to Read

- For class!
- To understand classic papers in the field
- To keep up with the field
- To review conference/journal papers
- To review drafts of your co-authors for further discussion
- To become a better writer for your own papers

# First Pass

- Read
  - Abstract and introduction
  - Section headings
  - Conclusions

- Be able to understand
  - Category: What type of paper is it?
  - Context: What body of work does it relate to?
  - Correctness: Do the assumptions seem valid?
  - Contributions: What are the main research contributions?
  - Clarity: Is the paper well-written?

# Second Pass

- Read
  - Read paper fully, but ignore proofs
  - Figures
  - Mark relevant references

- Be able to understand
  - Key ideas
  - I find it helpful to write 2 bullet points at the top for future reference

- When to do this
  - Possibly relevant paper to your research

# Third Pass

- Read
    - All details fully
    - Imagine you were writing the paper, and question every assumption
    - Note ideas for future work

- When to do this
    - Reviewing a conference/journal paper
    - Key related work to your own paper

- Time: several hours

# Internet history

## *1961-1972: Early packet-switching principles*

- 1961: Kleinrock - queueing theory shows effectiveness of packet-switching

- 1964: Baran - packet-switching in military nets

- 1967: ARPAnet conceived by Advanced Research Projects Agency

- 1969: first ARPAnet node operational

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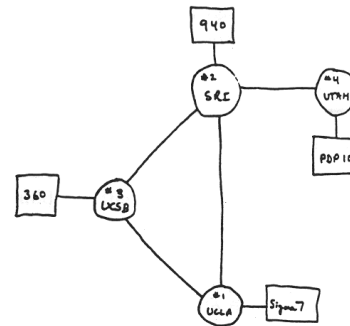


THE ARPA NETWORK

# Internet history

*1972-1980: Internetworking, new and proprietary nets*

- 1970: ALOHAnet satellite network in Hawaii

- 1974: Cerf and Kahn - architecture for interconnecting networks

- 1976: Ethernet at Xerox PARC

- 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, 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

# Internet history

*2005-present*

- ~750 million hosts
  - Smartphones and tablets
- Aggressive deployment of broadband access
- Increasing ubiquity of high-speed wireless access
- Emergence of online social networks:
  - Facebook: soon one billion users
- Service providers (Google, Microsoft) create their own networks
  - Bypass Internet, providing "instantaneous" access to search, emai, etc.
- E-commerce, universities, enterprises running their services in "cloud" (eg, Amazon EC2)

# Overview

- How to read
- History
- Layering
  - Inter-LAN (small)
  - Inter-LAN (large)
  - Evolution of layers
- Paper discussion

Q: How to design the Internet from the ground up?

# Inter-LAN connectivity

- Circuit switching
- Packet switching
  - Datagram routing
  - Source routing
  - Virtual circuits

# Circuit switching

**end-end resources allocated to, reserved for "call" between source & dest:**

- In diagram, each link has four circuits.
  - call gets 2$^{nd}$ circuit in top link and 1$^{st}$ circuit in right link.
- dedicated resources: no sharing
  - circuit-like (guaranteed) performance
- circuit segment idle if not used by call *(no sharing)*
- Commonly used in traditional telephone networks
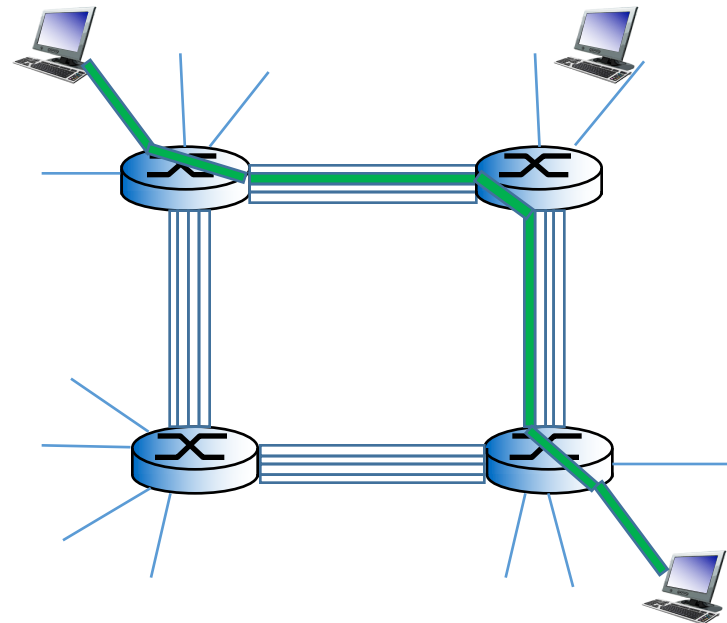
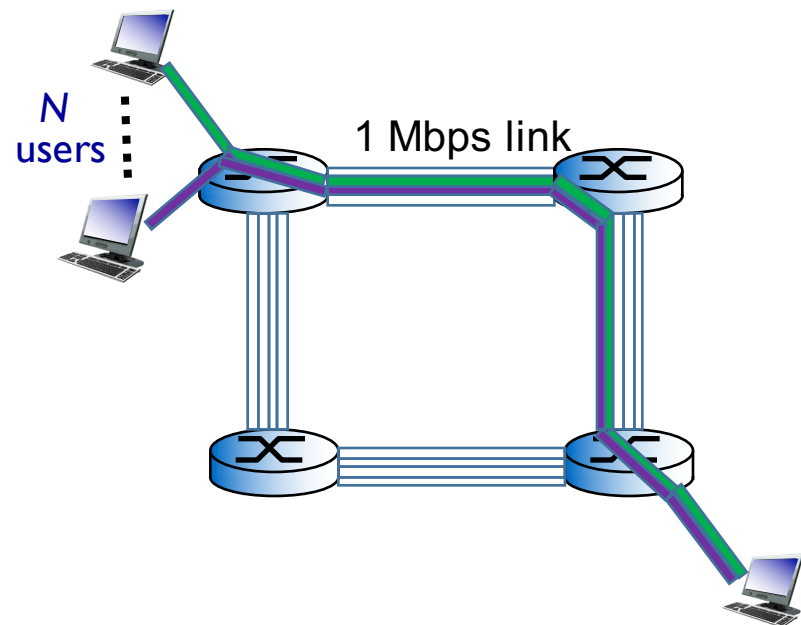# Packet-switching: store-and-forward



N
users
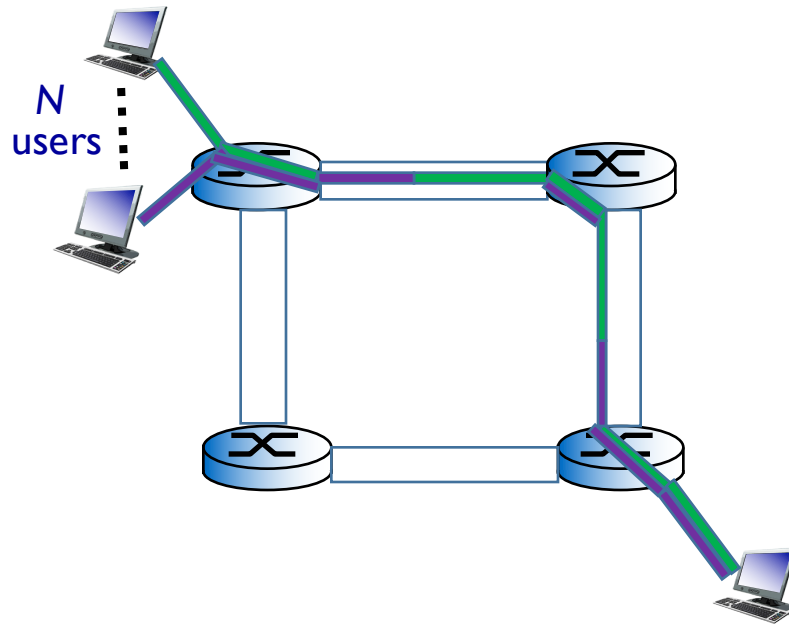
# Packet switching versus circuit switching

*packet switching allows more users to use network!*

example:

- 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 at same time is less than .0004



$N$ users

1 Mbps link

*Q:* how did we get value 0.0004?

# Types of packet switching

- Datagram routing
  - Packet header contains destination
  - Switches perform routing

D| data

D → 1    D → 2

0      1      2

H      S1     S2     D

- Source routing
  - Packet header contains list of switches to traverse

S1, S2 | data

0      1      2

H      S1     S2     D

- Virtual circuit
  - First send a control message
  - When switches receive this control message, set up routes and reserve resources

D |

flag |data

Flag → 1    Flag → 2
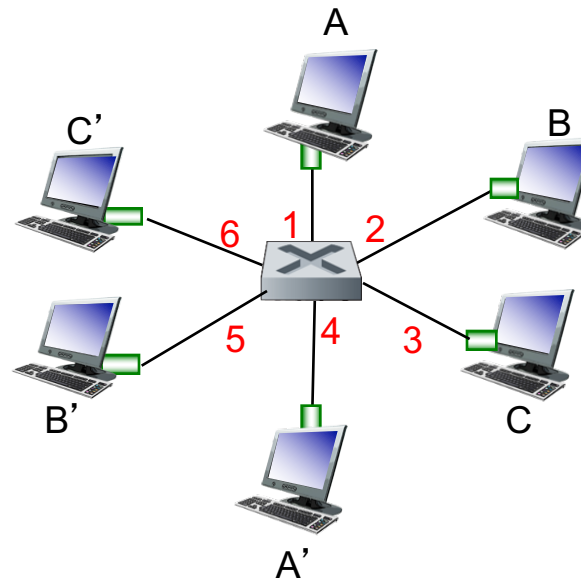
0      1      2

H      S1     S2     D
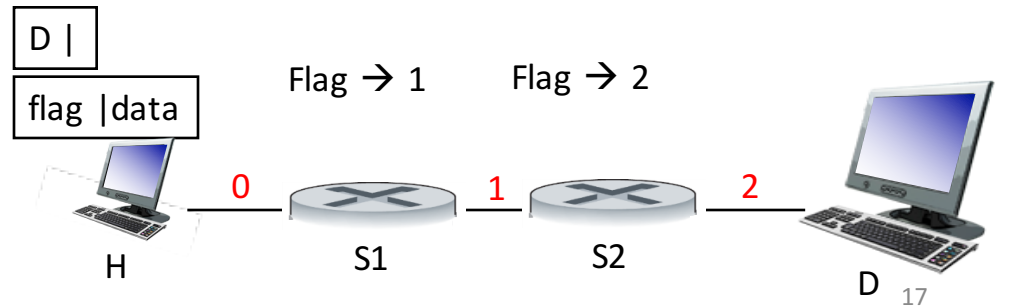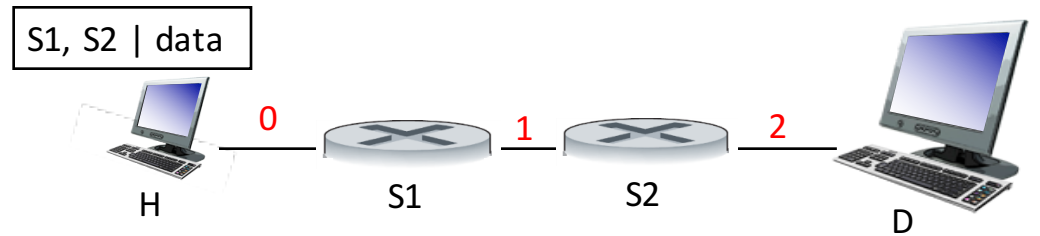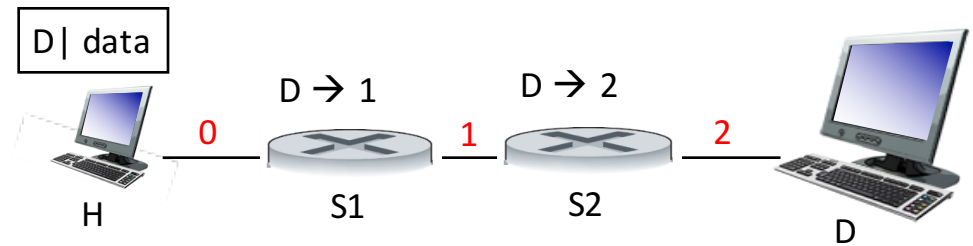
# Switch forwarding table

*Q:* how does switch know A'
reachable via interface 4, B'
reachable via interface 5?

❖ <u>A:</u> each switch has a switch
table, each entry:

- (MAC address of host, interface
  to reach host, time stamp)

- looks like a routing table!

<u>Q:</u> how are entries created,
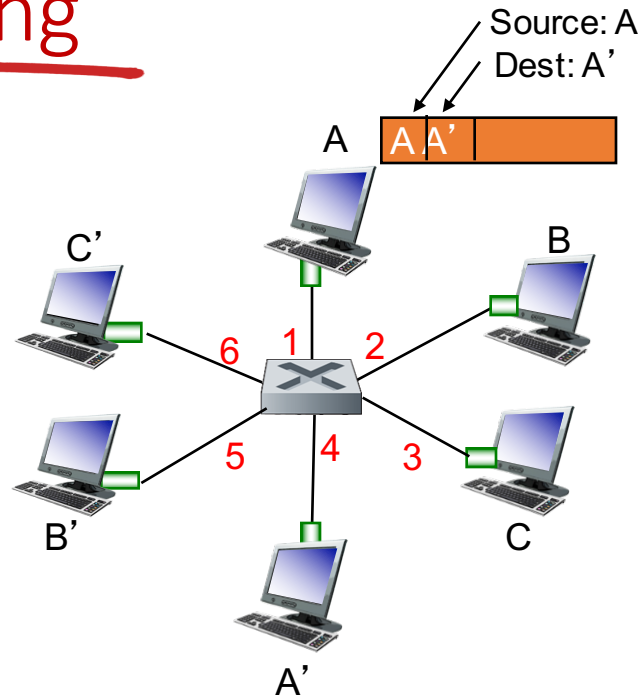maintained in switch table?

- something like a routing
  protocol?



*switch with six interfaces*
*(1,2,3,4,5,6)*

# Switch: self-learning

- switch *learns* which hosts can be reached through which interfaces
  - when frame received, switch "learns" location of sender: incoming LAN segment
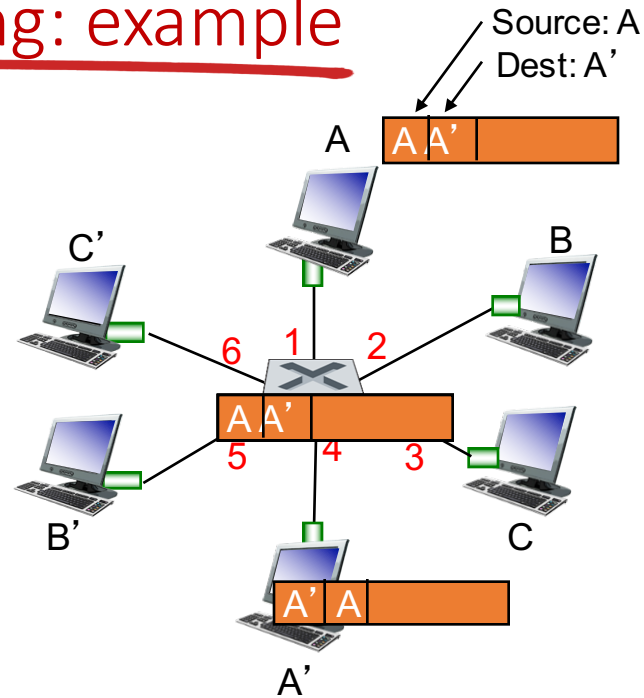  - records sender/location pair in switch table

Source: A
Dest: A'

A  A A'

| MAC addr | interface | TTL |
|----------|-----------|-----|
| A | 1 | 60 |
| | | |

*Switch table (initially empty)*

# Self-learning, forwarding: example

- frame destination, A',
  locaton unknown: *flood*

❖ destination A location
  known: selectively send
  on just one link



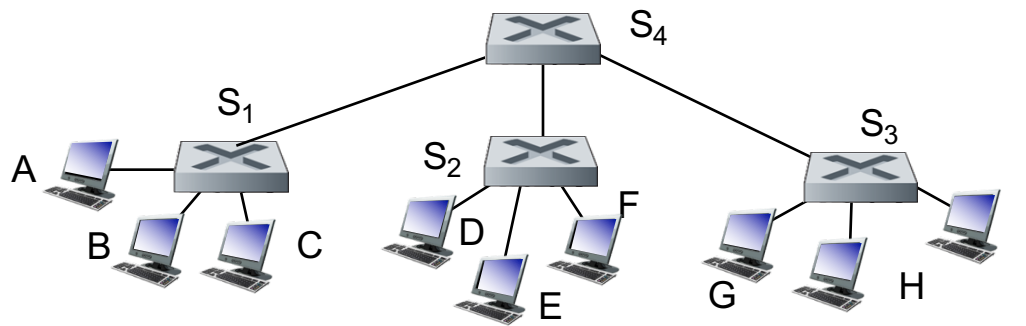| MAC addr | interface | TTL |
|----------|-----------|-----|
| A | 1 | 60 |
| A' | 4 | 60 |
|  |  |  |

*switch table
(initially empty)*

# Interconnecting switches

❖ switches can be connected together



**Q:** sending from A to G - how does $S_1$ know to forward frame destined to F via $S_4$ and $S_3$?

❖ **A:** self learning! (works exactly the same as in single-switch case!)

# Overview

- How to read
- History
- Layering
  - Inter-LAN (small)
  - Inter-LAN (large)
  - Evolution of layers
- Paper discussion

Q: How to design the Internet from the ground up?

# Inter-networking

- *Goal*: Design a scalable network infrastructure that connects different smaller networks together, to enable hosts on different networks to talk to each other.


- Key challenges with the LAN approach:
  1. Scaling up
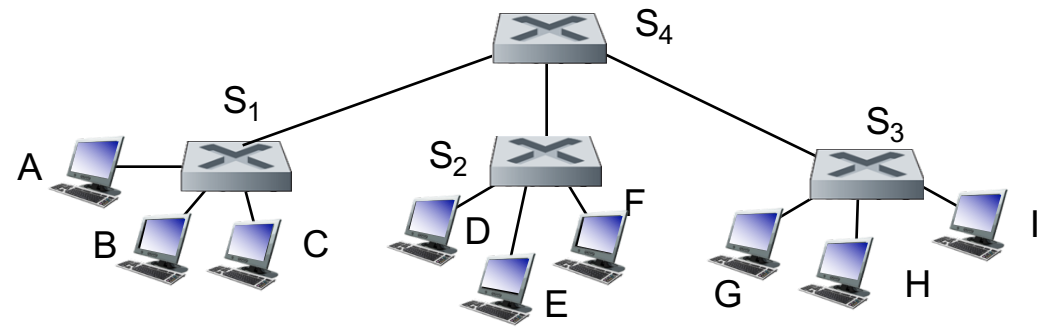  2. Heterogeneity

# Why scaling up doesn't work
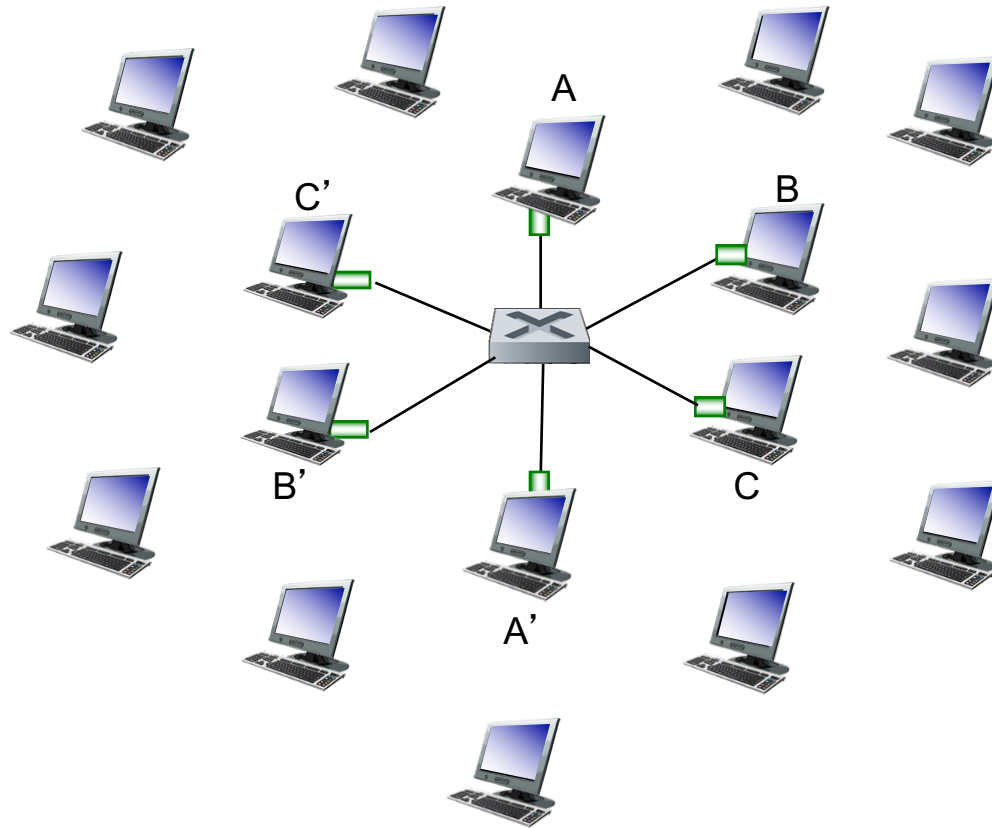
# What is LAN heterogeneity?

- Sources of heterogeneity
  - Addressing
  - Bandwidth and latency
  - Packet size
  - Loss rates
  - Packet routing

- Gateways provide translation between LANs

# Options for gateway functionality

1.  Translation: translate between different LAN "languages"
    - Updates: translation may fail if LANs get updated or new features are added
    - Scalability: have to translate between many LANs

2.  Unified network layer: define some common "words" that everyone has to understand
    - This is the Clark paper you read!

# Universality goals

- IP-over-everything
  - Common set of names (IP addresses) and routing protocols so that gateways know how to behave

- Best-effort
  - No special treatment of different packets (ignoring QoS)
  - No loss recovery (at the network layer)

- End-to-end
  - Complicated functionality (e.g. reliability in the transport layer) implemented in the end host
  - Network gateways kept simple

# Robustness goals

- Soft-state inside the network
  - Definition: information that times out (goes away) unless refreshed
  - Easily recover from errors
  - E.g., routing protocols automatically update themselves periodically

- Fate sharing of end hosts
  - If end hosts go down, state is lost
  - If gateway fails, network can recover (soft state)

- Conservative transmission / liberal reception
  - "Be conservative in what you send; be liberal in what you accept"
  - E.g. sender receives ACK for unknown packets; silently drops

# Weaknesses

- Rely on end-hosts to behave
    - Buggy implementation
    - Greedy senders (non-TCP)
    - Malicious

- Administration and management tools not very mature

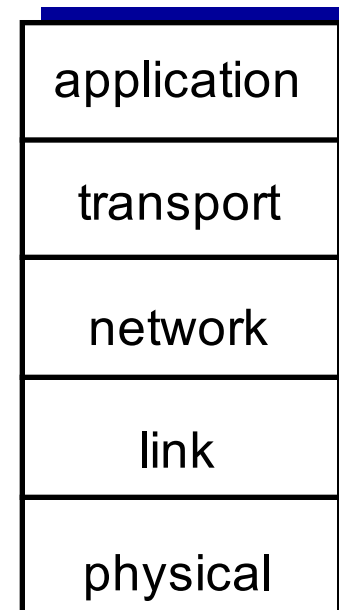- Weak accounting and pricing tools

# Overview

- How to read
- History
- Layering
  - Inter-LAN (small)
  - Inter-LAN (large)
  - Evolution of layers
- Paper discussion

Q: How to design the Internet from the ground up?

# Internet protocol stack

- *application:* supporting network applications
  - FTP, SMTP, HTTP
- *transport:* process-process data transfer
  - TCP, UDP
- *network:* routing of datagrams from source to destination
  - IP, routing protocols
- *link:* data transfer between neighboring network elements
  - Ethernet, 802.111 (WiFi), PPP
- *physical:* bits "on the wire"
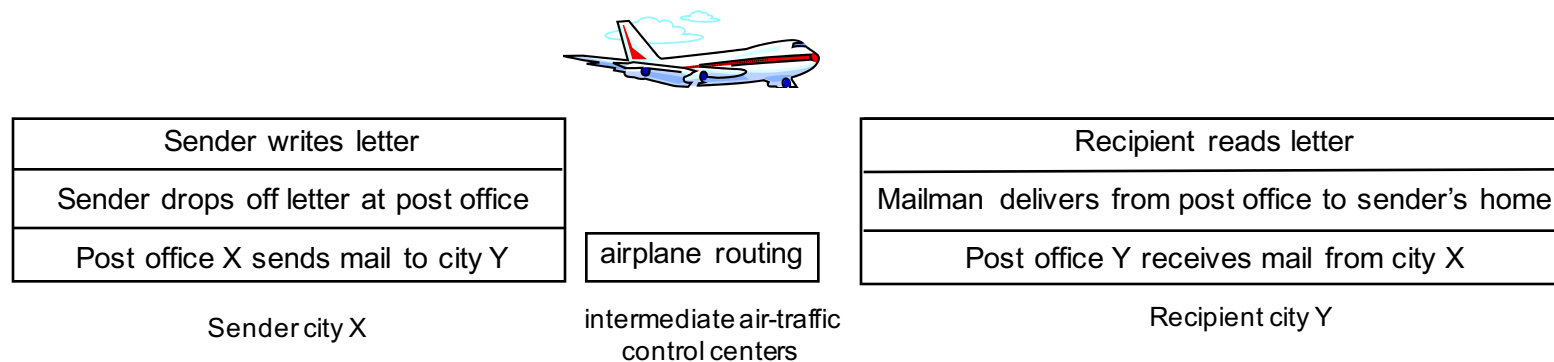
| application |
| --- |
| transport |
| network |
| link |
| physical |

# Layering of post office functionality



| Sender writes letter |
| Sender drops off letter at post office |
| Post office X sends mail to city Y |

Sender city X

| airplane routing |

intermediate air-traffic
control centers

| Recipient reads letter |
| Mailman delivers from post office to sender's home |
| Post office Y receives mail from city X |

Recipient city Y

*layers:* each layer implements a service
- via its own internal-layer actions
- relying on services provided by layer below

# Layering of post office functionality



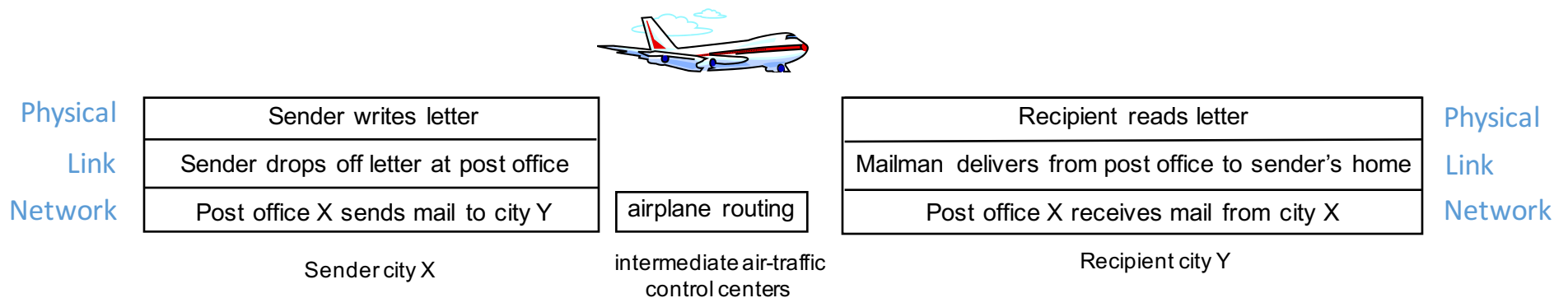| | | | | | |
|---|---|---|---|---|---|
| Physical | Sender writes letter | | | Recipient reads letter | Physical |
| Link | Sender drops off letter at post office | | | Mailman delivers from post office to sender's home | Link |
| Network | Post office X sends mail to city Y | airplane routing | | Post office X receives mail from city X | Network |
| | Sender city X | intermediate air-traffic control centers | | Recipient city Y | |

*layers:* each layer implements a service
- via its own internal-layer actions
- relying on services provided by layer below

Transport: Delivery via UPS (signature required) or USPS (no signature required)
Application: the contents of the letter, e.g. photo, video, novel

# The narrow waist of IP
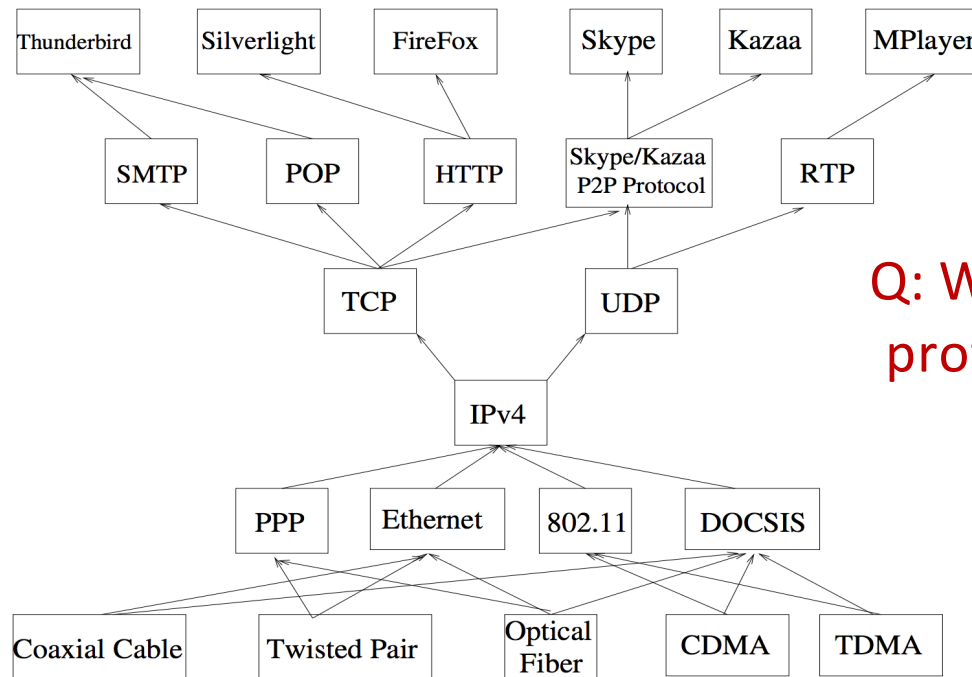


Q: Why does the Internet protocol stack resemble an hourglass?

"The Evolution of Layered Protocol Stacks Leads to an Hourglass-Shaped Architecture", *SIGCOMM* 2011.

# Why the narrow waist?

- IP is a global address, so no need for two naming systems?

- Lower layers are diverse (e.g. wireless, optical, cable), higher layers are also diverse (e.g. voice, video, file transfer), so IP layer in the middle must be more general (and hence unique)?

- Analytic birth/death model?
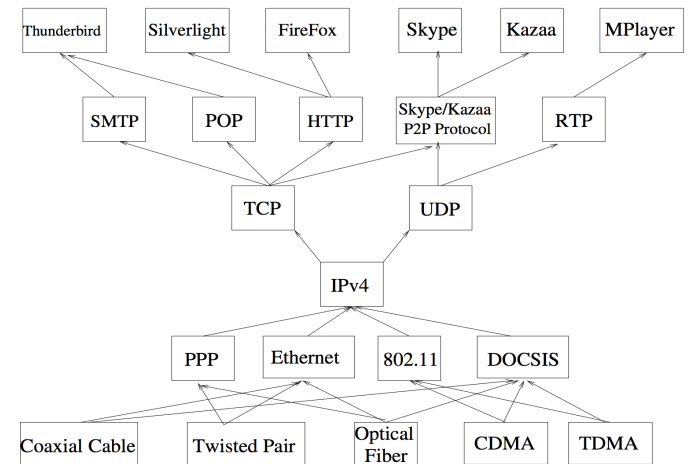
# Overview

- How to read
- History
- Layering
  - Inter-LAN (small)
  - Inter-LAN (large)
  - Evolution of layers
- Paper discussion

Q: How to design the Internet from the ground up?

# Design goals

1. Effective techniques for multiplexed utilization of existing local area networks
2. Reliability in the face of gateway failure
3. Support for multiple types of communication
4. Accommodate a variety of local area networks
5. Distributed management
6. Cost-effective
7. Easy host attachment
8. Accountable resource usage

# Consequences of the design goals

1. Effective techniques for multiplexed utilization of existing local area networks
   - Packet-switched networks
2. Reliability in the face of gateway failure
   - State contained in the end host, only soft state in the network
3. Support for multiple types of communication
   - Different transport-layer protocols (e.g. TCP, UDP)
4. Accommodate a variety of local area networks
   - Best-effort service
5. Distributed management
   - Multiple tier-1 ISPs
   - Intra-domain and inter-domain routing

# Discussion

- What would happen if the design goals were in a different order?

- How would we re-design a network with more emphasis on security? Network management?
  - Or without modifying existing networks too much?