Master Kurs Rechnernetze
Computer Networks
IN2097

Prof. Dr.-Ing. Georg Carle
Dr. Thomas Fuhrmann

Institut für Informatik
Technische Universität München
http://www.net.in.tum.de

Course Outline

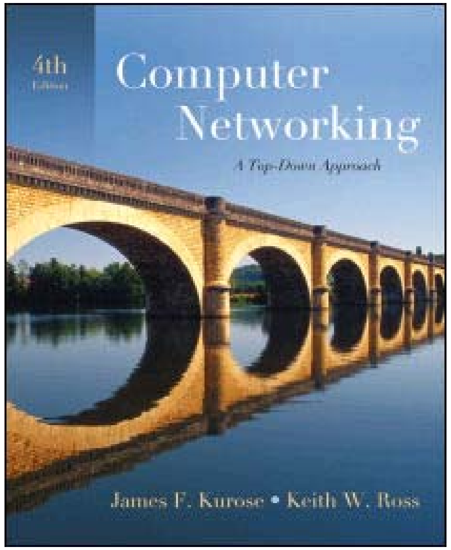
- Part 1: Internet protocols
  - Link Layer protocols
  - Network Layer protocols
  - Transport Layer protocols
  - Application Layer protocols
- Part 2: Advanced Computer Networks Principles
  - *review:* packet-, circuit-switching primer
  - *common themes:* signaling, indirection, virtualization, multiplexing, randomization, scalability
  - *implementation principles:* techniques
  - *network architecture:* the big picture, synthesis
  - *network algorithmics:* self stabilization (routing examples), broadcast/controlled flooding (link state broadcast, ad hoc routing), routing and congestion control: an optimization viewpoint
  - *network simulation* (time permitting)

Part 2
Advanced Computer Networks

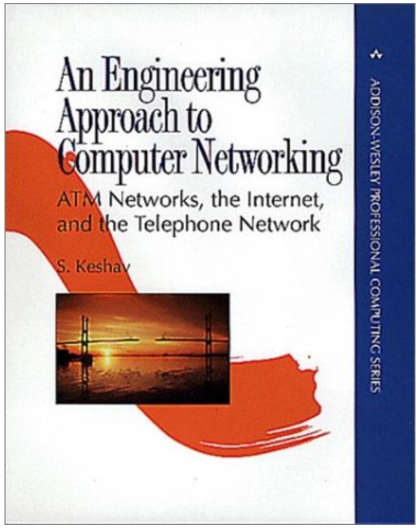
Acknowledgements:
Jim Kurose, University of Massachusetts, Amherst

Additional relevant books

- J. F. Kurose & K. W. Ross, *Computer Networking: A Top-Down Approach Featuring the Internet*, 2007, 4th edition, Addison Wesley
  - Innovation: Presentation of Protocols Top-Down
  - Statements of key persons in networking research



- S. Keshav: *An Engineering Approach to Computer Networking*, 1999, Addison-Wesley
  - Very good architectural and quantitative treatment of computer networks
  - Motivation of many design decisions



Part 2 Educational Goals

Advanced, fundamental networking principles

- foundational material: long half life
- mix of theory and practice
- explain design tradeoffs and design decisions

Course Part 2 - Topics

- *review:* packet-, circuit-switching primer
- *common themes:* signaling, indirection, virtualization, multiplexing, randomization, scalability
- *implementation principles:* techniques
- *network architecture:* the big picture, synthesis
- *network algorithmics:* self stabilization (routing examples), broadcast/controlled flooding (link state broadcast, ad hoc routing), routing and congestion control: an optimization viewpoint
- *network simulation* (time permitting)

Networking Review

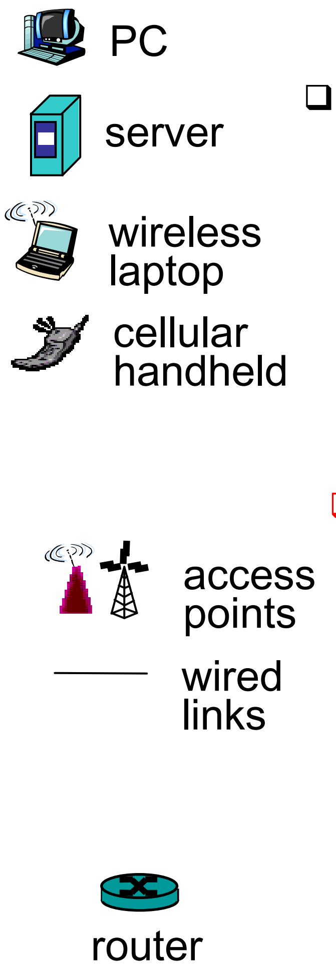
Overview:

- overview
- error control
- flow control
- congestion control
- routing
- LANs
- addressing
- synthesis:
  - control timescales

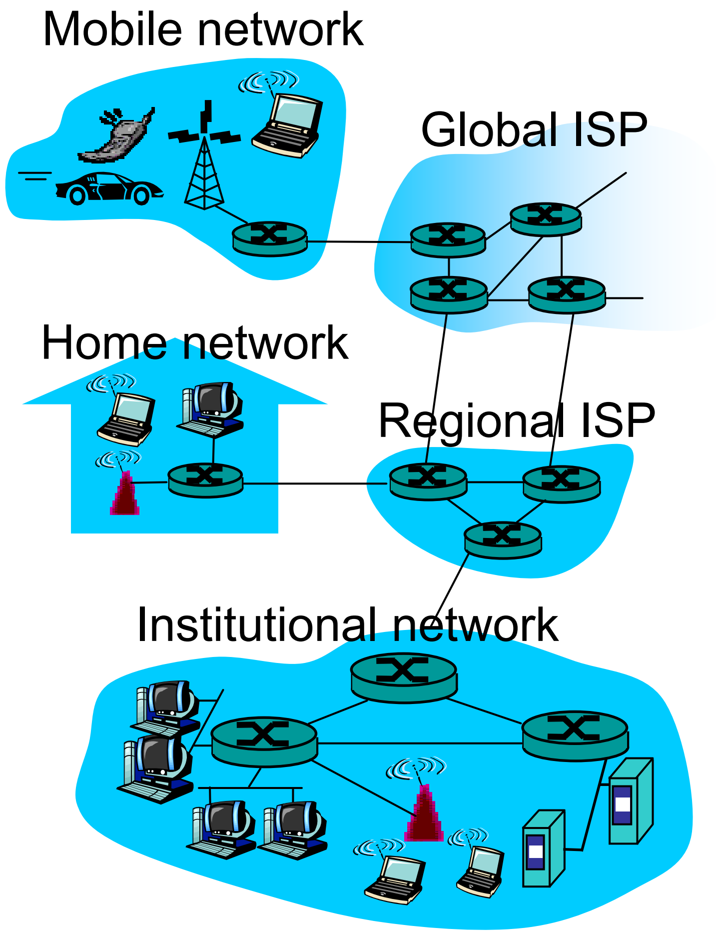
Goals:

- review key topics
- independent of specific protocols

What’s the Internet: “nuts and bolts” view

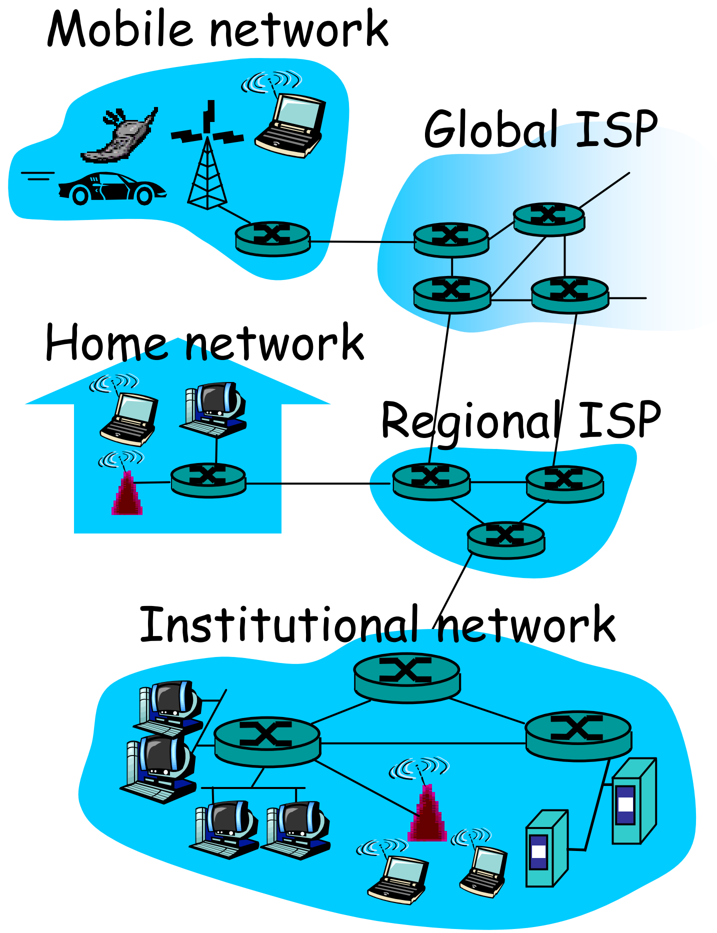


- millions of connected computing devices: *hosts = end systems*
  - running *network apps*
- *communication links*
  - fiber, copper, radio, satellite
  - transmission rate = *bandwidth*
- *routers:* forward packets (chunks of data)



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

- *protocols* control sending, receiving of msgs
  - e.g., TCP, IP, HTTP, Skype, Ethernet
- *Internet:* "network of networks"
  - loosely hierarchical
  - public Internet versus private intranet
- Internet standards
  - RFC: Request for comments
  - IETF: Internet Engineering Task Force

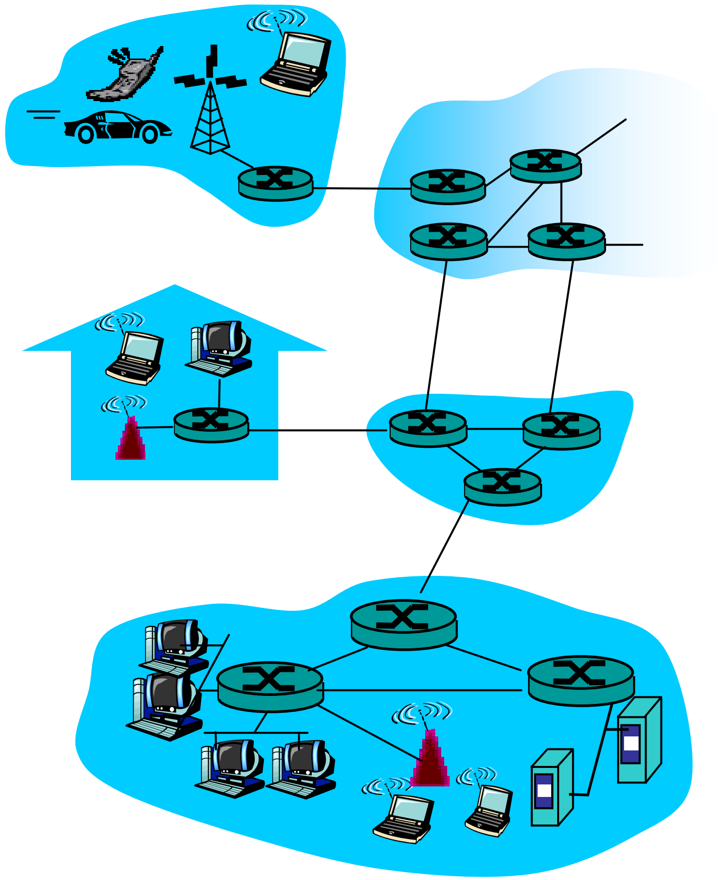


A closer look at network structure:

- network edge: applications and hosts

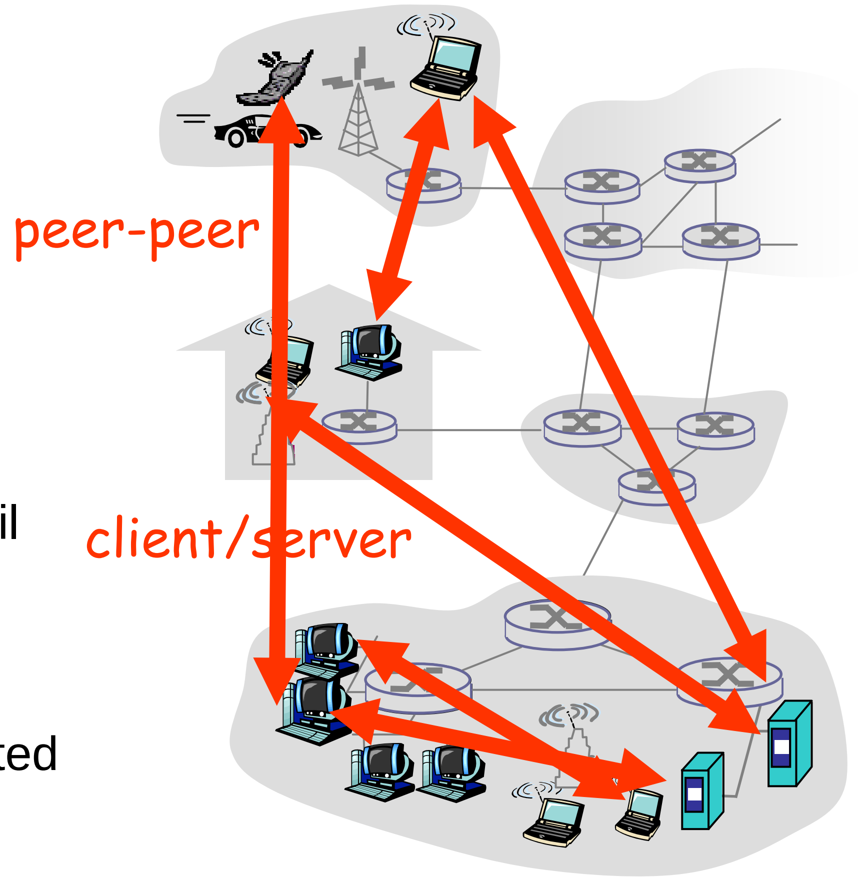
- access networks, physical media: wired, wireless communication links

- network core:
  - interconnected routers
  - network of networks



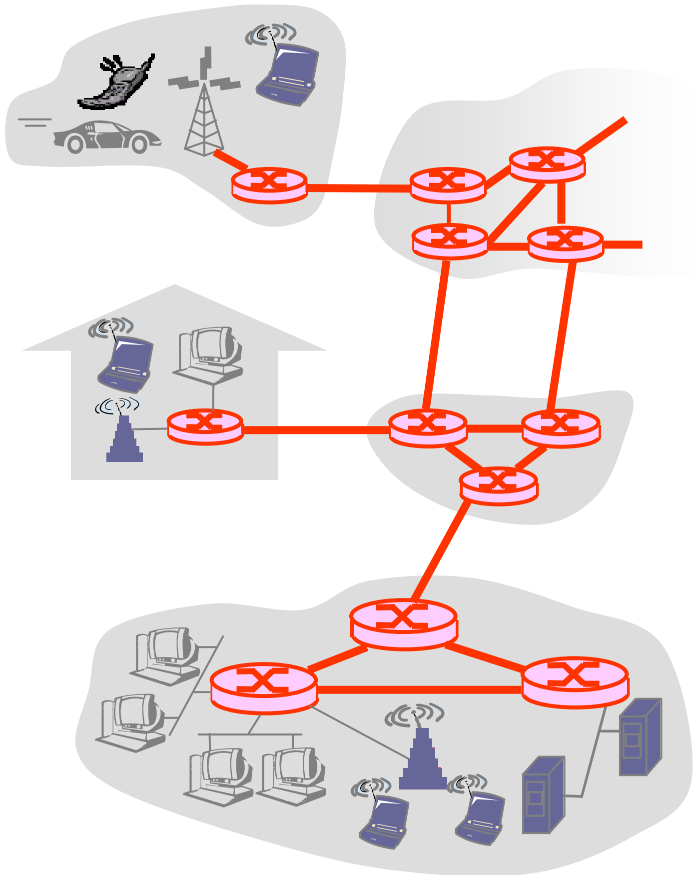
The network edge:

- end systems (hosts):
  - run application programs
  - e.g. Web, email
  - at “edge of network”
- client/server model
  - client host requests, receives service from always-on server
  - e.g. Web browser/server; email client/server
- peer-peer model:
  - minimal (or no) use of dedicated servers
  - e.g. Skype, BitTorrent



The Network Core

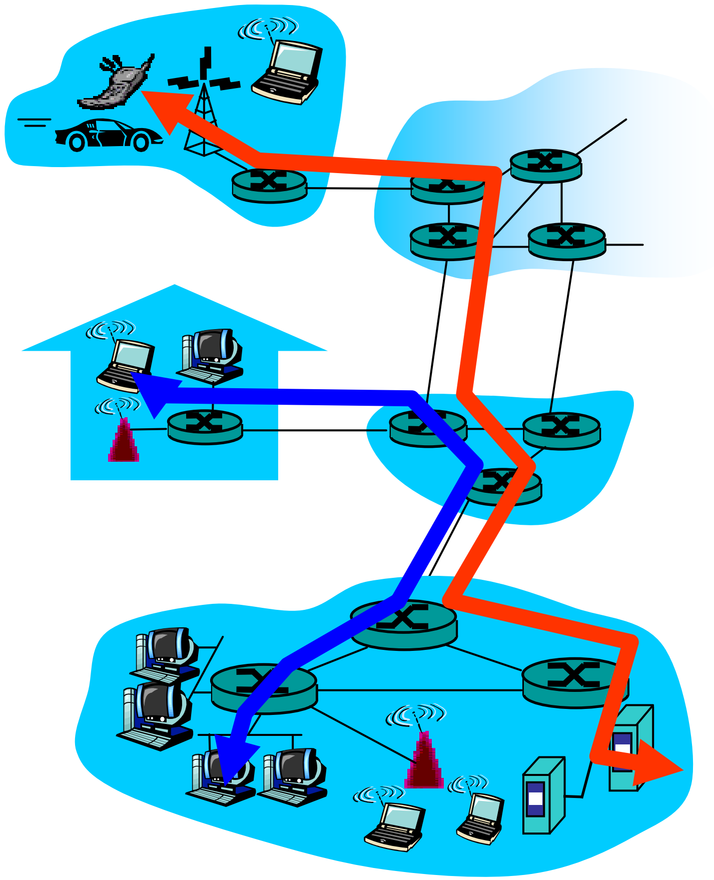
- mesh of interconnected routers
- *the* fundamental question: how is data transferred through net?
  - circuit switching: dedicated circuit per call: telephone net
  - packet-switching: data sent thru net in discrete “chunks”



Network Core: Circuit Switching

End-end resources reserved for "call"

- link bandwidth, switch capacity
- dedicated resources: no sharing
- circuit-like (guaranteed) performance
- call setup required



Network Core: Circuit Switching

network resources (e.g., bandwidth) divided into "pieces"

- pieces allocated to calls
- resource piece *idle* if not used by owning call *(no sharing)*

- Question: how is bandwidth divided into "pieces"?
  - Bandwidth of a link is divided, using schedulers
  - Access to these pieces of bandwith is controlled (policers)

Network Core: 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*

resource contention:

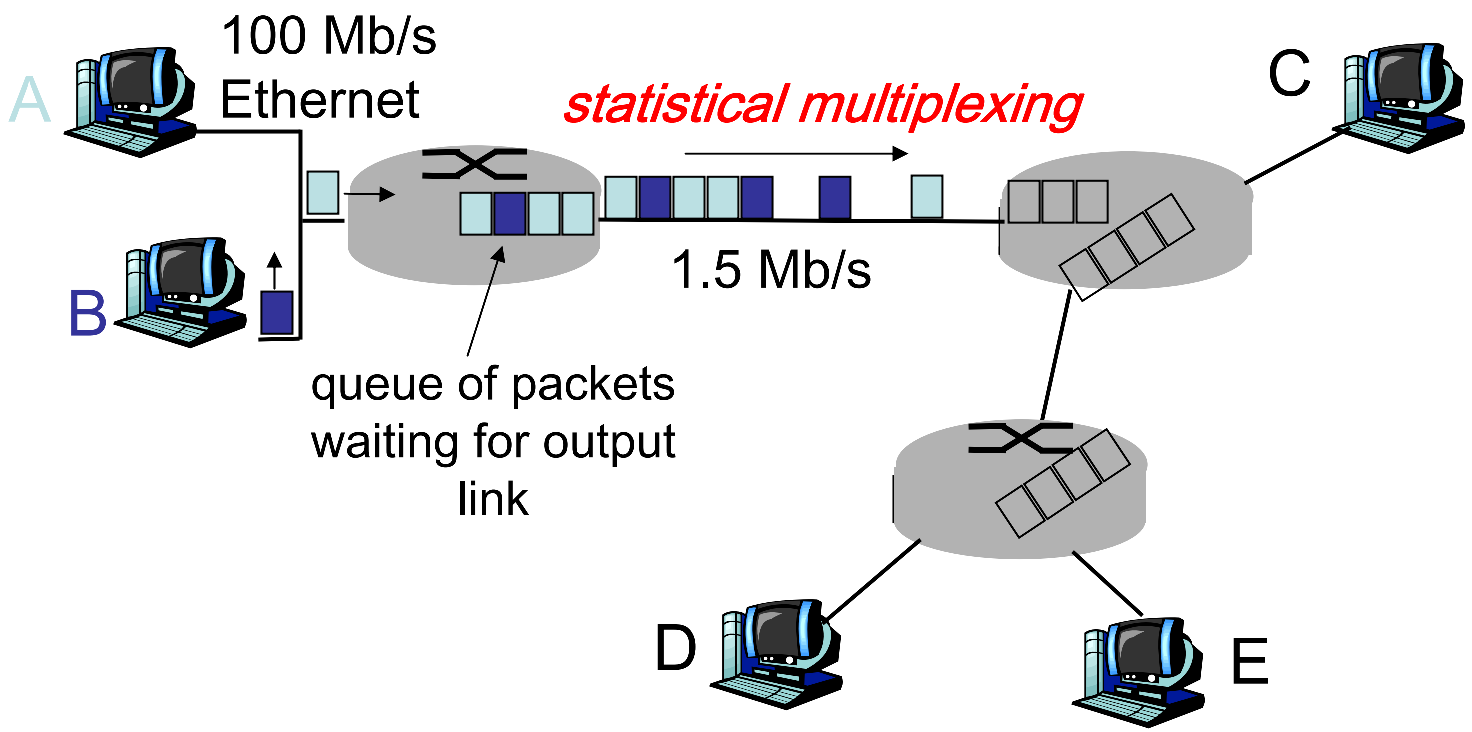
- aggregate resource demand can exceed amount available
- congestion: packets queue, wait for link use
- store and forward: packets move one hop at a time
  - Node receives complete packet before forwarding

No bandwidth division into “pieces”

No dedicated allocation

No resource reservation

Packet Switching: Statistical Multiplexing



Question: why packet switching?

- Sequence of A & B packets does not have fixed pattern, bandwidth shared on demand ➔ ***statistical multiplexing***.
- Circuit switching typically would lead to lower pieces of bandwith (and higher delay) compared to packet switching.

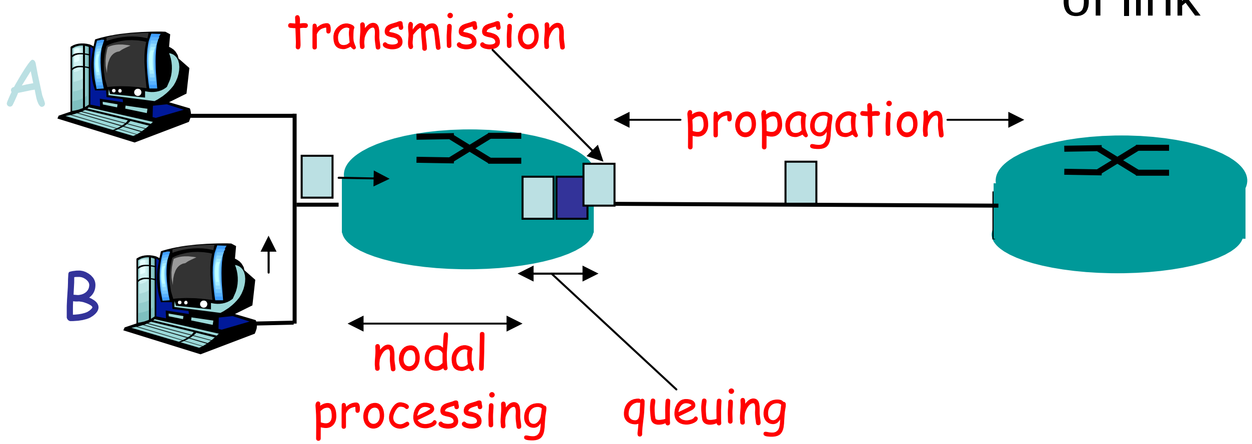
Packet switching versus circuit switching

Question: Is packet switching an obvious winner?

- Great for bursty data
  - resource sharing
  - no call setup
- Possibly excessive congestion: packet delay and loss
  - protocols needed for reliable data transfer, congestion control
- Q: How to provide circuit-like behavior?
  - bandwidth guarantees needed for audio/video applications - still an unsolved problem! (more later)

Delay in packet-switched networks

- ❑ Packets experience delay on end-to-end path
- ❑ Four sources of delay at each hop



1) Nodal processing delay:
   - process protocol
   - check for bit errors
   - determine output link

2) Queuing delay
   - time waiting at output link for transmission
   - depends on congestion of link

Delay in packet-switched networks

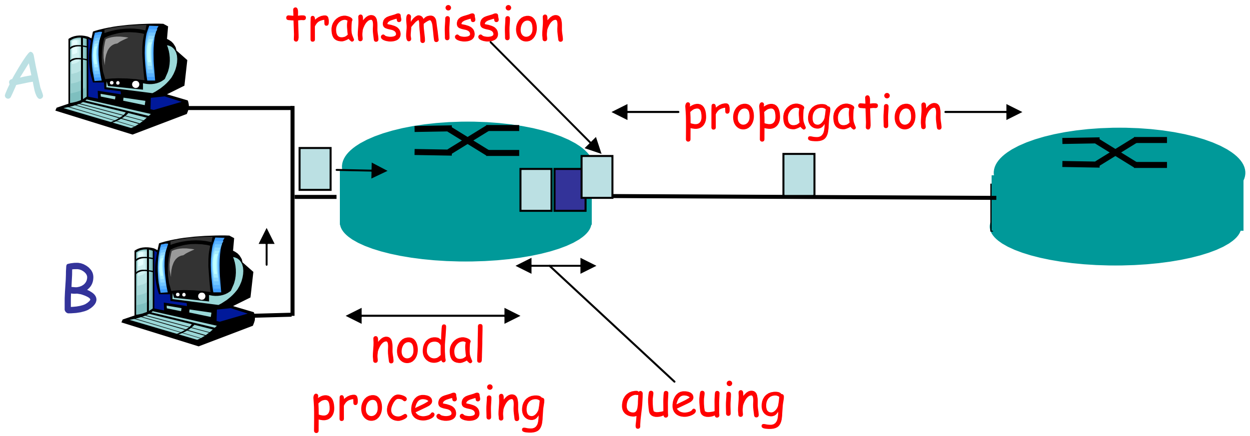
3) Transmission delay:

- R=link bandwidth (bps)
- L=packet length (bits)
- time to send bits into link = L/R

4) Propagation delay:

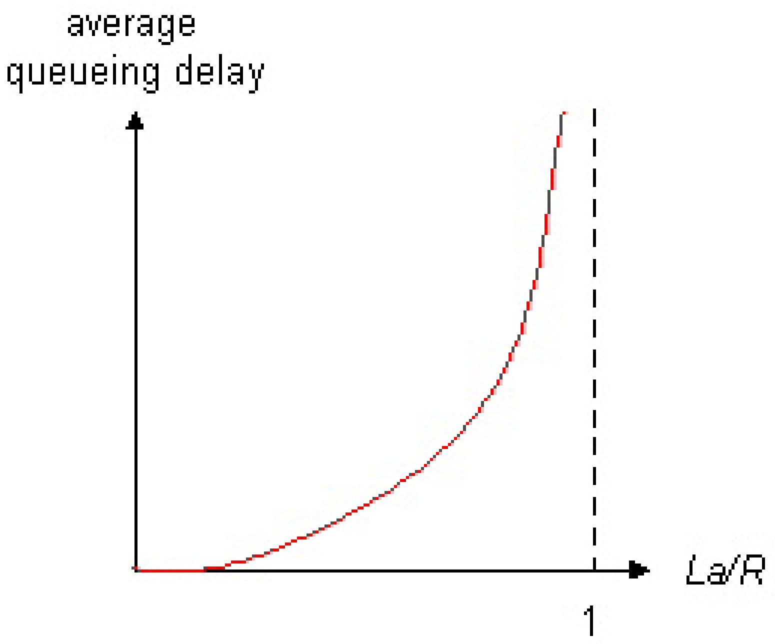
- s = propagation speed in medium ($\sim 2/3 * 3 \times 10^8$ m/sec)
- d = length of physical link
- propagation delay = d/s

Note: s and R are *very* different quantities!



Queuing delay (revisited)

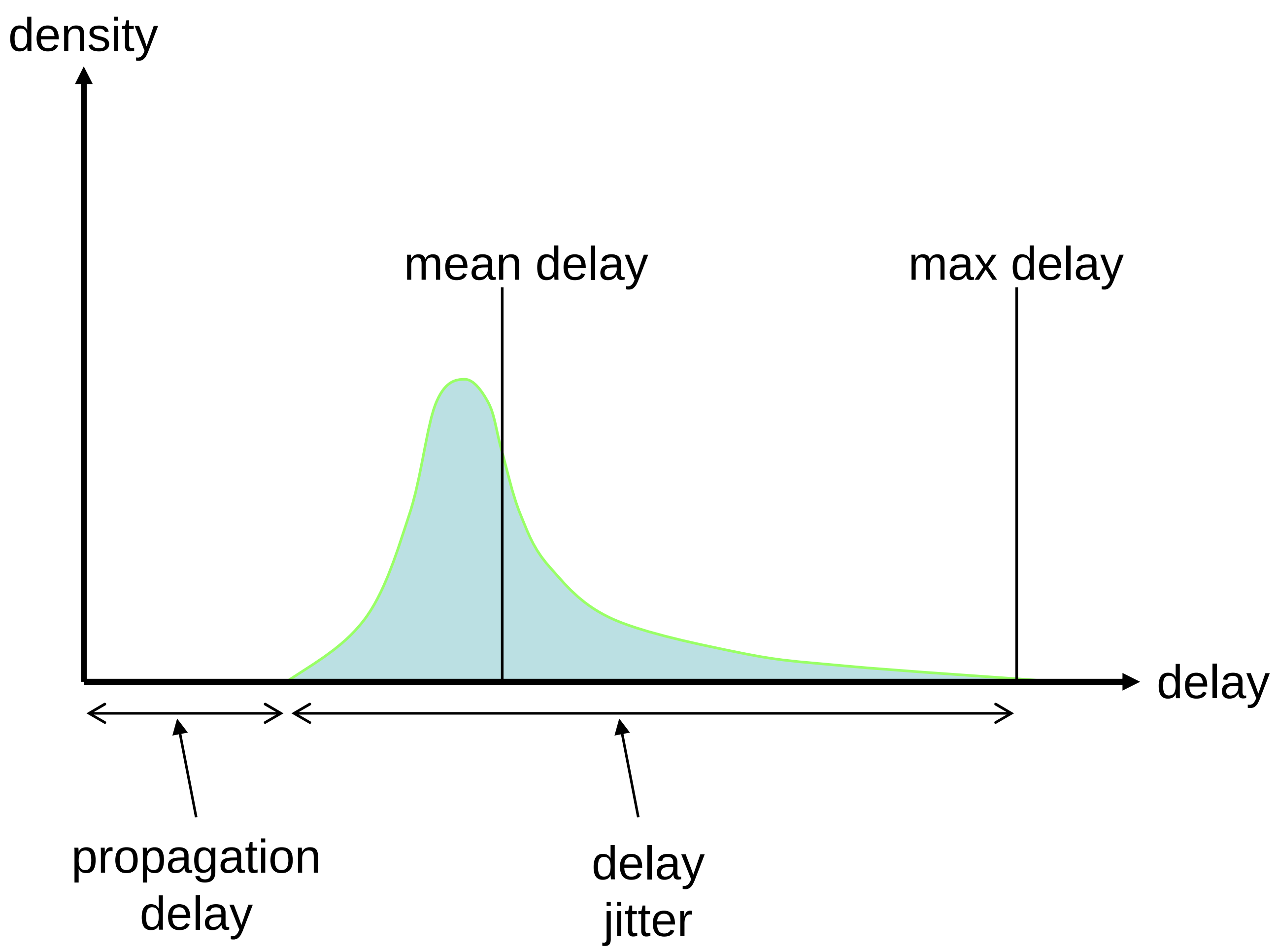
- R=link bandwidth (bps)
- L=packet length (bits)
- a=average packet arrival rate (1/s)



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!

Delay Distributions



Datagram or VC network: why?

Internet

- data exchange among computers
  - “elastic” service, no strict timing requirements
- “smart” end systems (computers)
  - can adapt, perform control, error recovery
  - simple inside network, complexity at “edge”
- many link types
  - different characteristics
  - uniform service difficult

ATM

- evolved from telephony
- human conversation:
  - strict timing, reliability requirements
  - need for guaranteed service
- “dumb” end systems
  - telephones
  - complexity inside network

Virtual circuits

"source-to-destination path behaves much like telephone circuit"

- performance-wise
- network actions along source-to-destination path

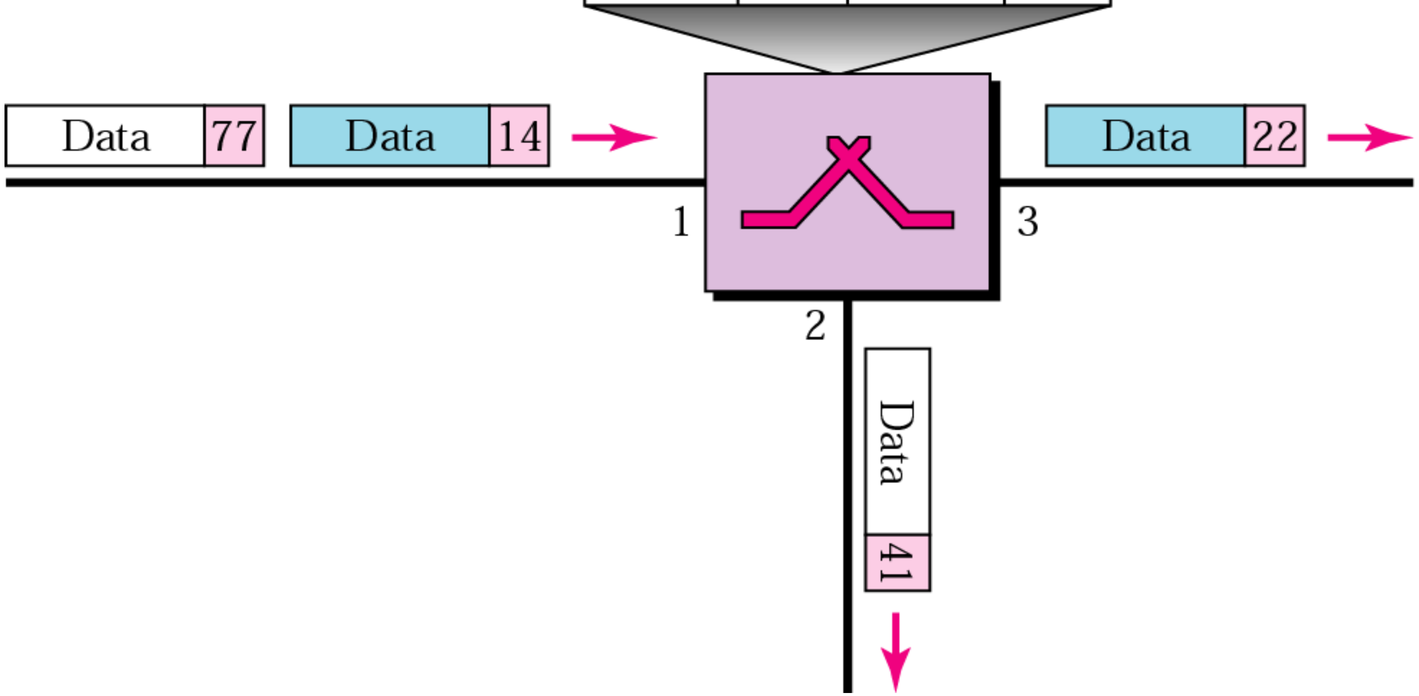
- call setup, teardown for each call *before* data can flow
- each packet carries VC identifier (not destination host ID)
- *every* router on source-dest path maintains "state" for each passing connection
  - transport-layer connection only involved two end systems
- link, router resources (bandwidth, buffers) may be *allocated* to VC
  - to get circuit-like performance

Virtual Circuits and Label Swapping

- Virtual Circuit Switching
- Multiplexing of Variable vs. Fixed Size Packets
- ATM Cell
- Virtual Path Identifiers and Virtual Channel Identifiers
- ATM Virtual Connections

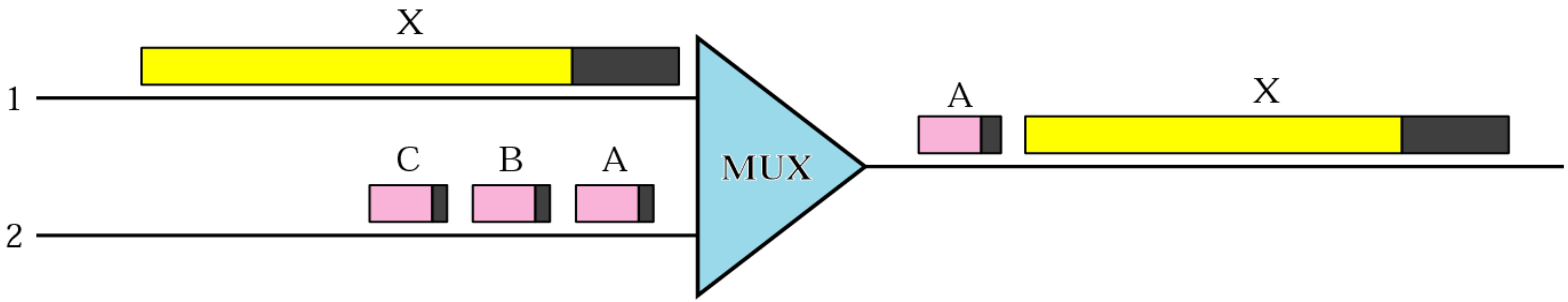
Virtual Circuit Switching

Incoming		Outgoing	
Port	VCI	Port	VCI
1	14	3	22
1	77	2	41

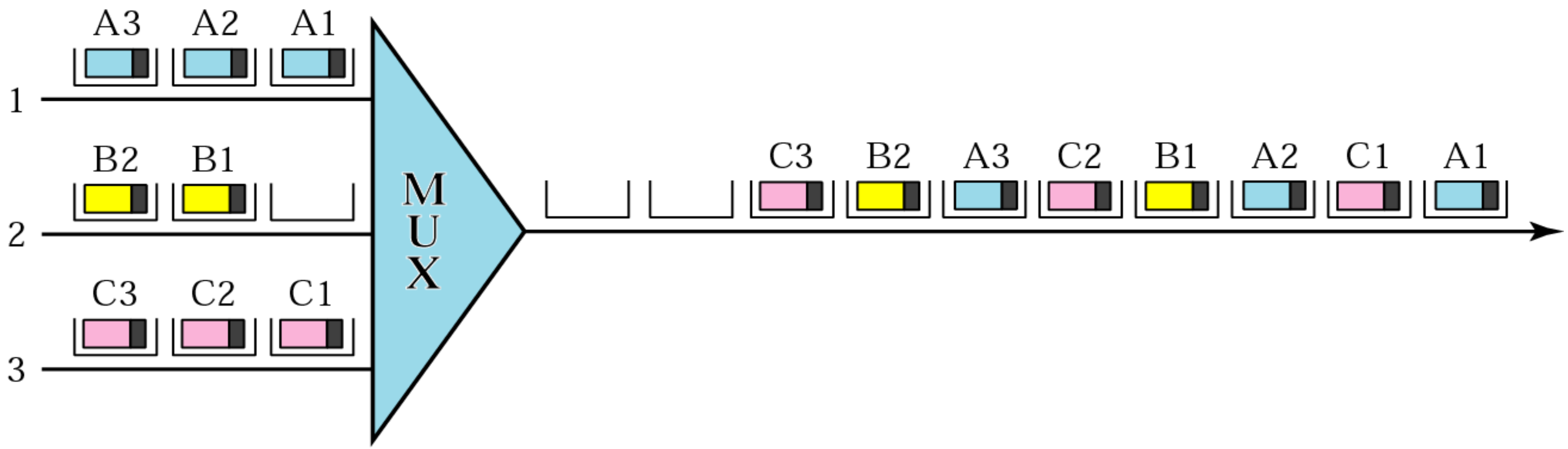


Multiplexing of Variable vs. Fixed Size Packets

- Multiplexing of variable size packets

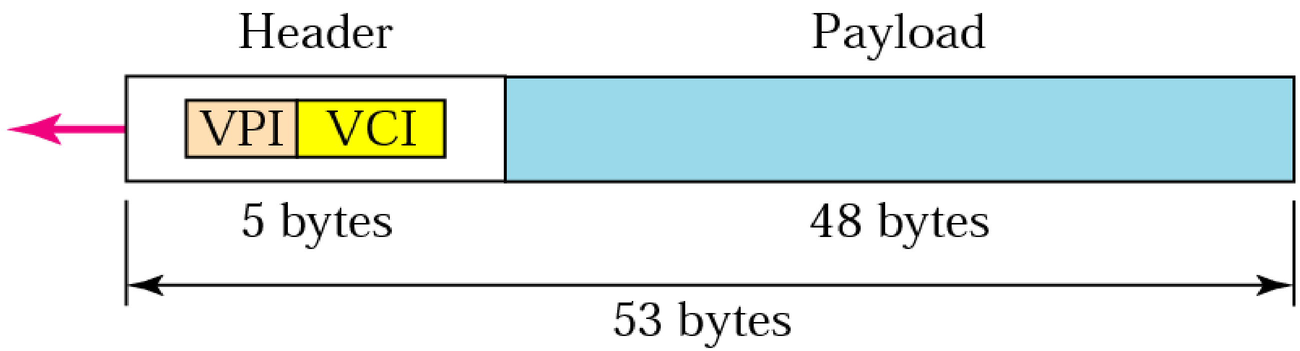


- ATM Multiplexing

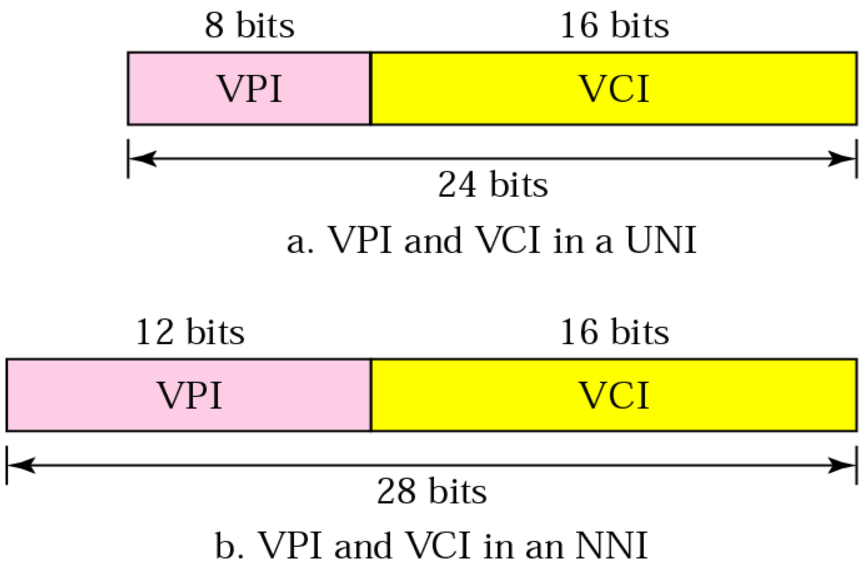


ATM Identifiers

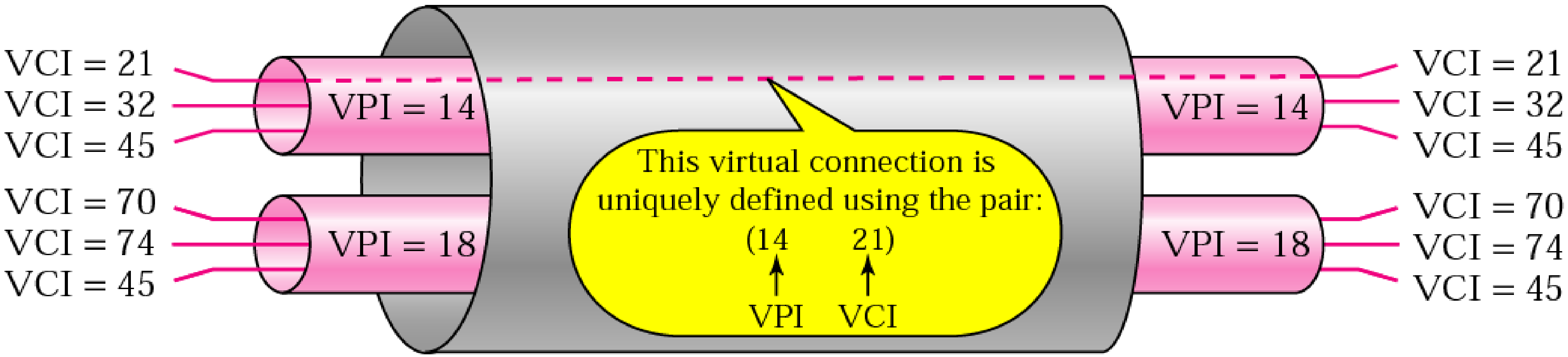
- ATM Cell



- Virtual Path Identifiers and Virtual Channel Identifiers



ATM Virtual Connections



Common network/protocol functions

Goals:

- Identify and study common architectural components, protocol mechanisms
- *synthesis:* big picture
- *depth:* important topics not covered in an introductory course

Overview:

- signaling: telephone networks, Internet, ATM networks
- state management (signaling)
- randomization
- indirection
- multiplexing
- virtualization
- design for scale

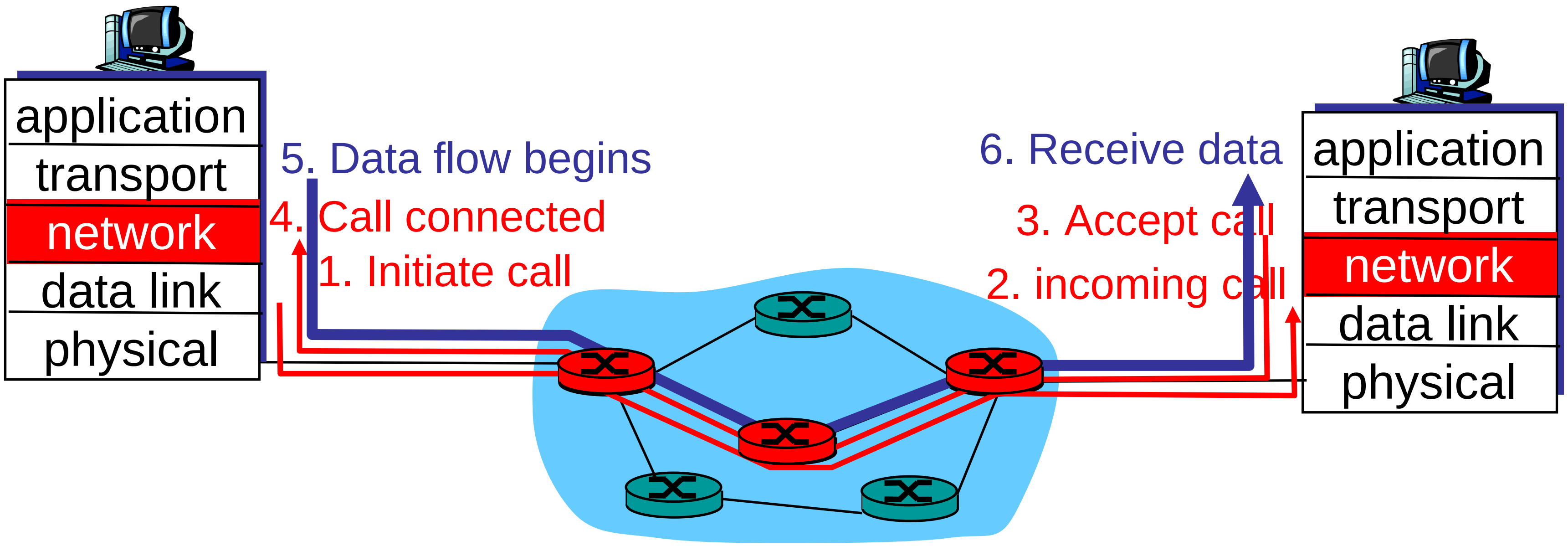
1. Signaling

signaling: exchange of messages among network entities to enable (provide service) to connection/call

- before, during, after connection/call
  - call setup and teardown (state)
  - call maintenance (state)
  - measurement, billing (state)
- between:
  - end-user <-> network
  - end-user <-> end-user
  - network element <-> network element

Virtual circuits: signaling protocols

- used to set up, maintain teardown VC
- used in (G)MPLS, ATM, frame-relay, X.25
- not used in today's Internet at L3 (network layer)



Examples

- Q.921, SS7 (Signaling System no. 7): telephone network
- Q.2931: ATM
- RSVP (Resource Reservation Protocol)
- SIP (Session Initiation Protocol): Internet

- Signalling between which entities?
    - end-user <-> network
    - end-user <-> end-user
    - network element <-> network element

SIP: Session Initiation Protocol [RFC 3261]

SIP long-term vision:

- all telephone calls, video conference calls take place over Internet
- people are identified by names or e-mail addresses, rather than by phone numbers
- you can reach callee, no matter where callee roams, no matter what IP device callee is currently using

SIP key person:

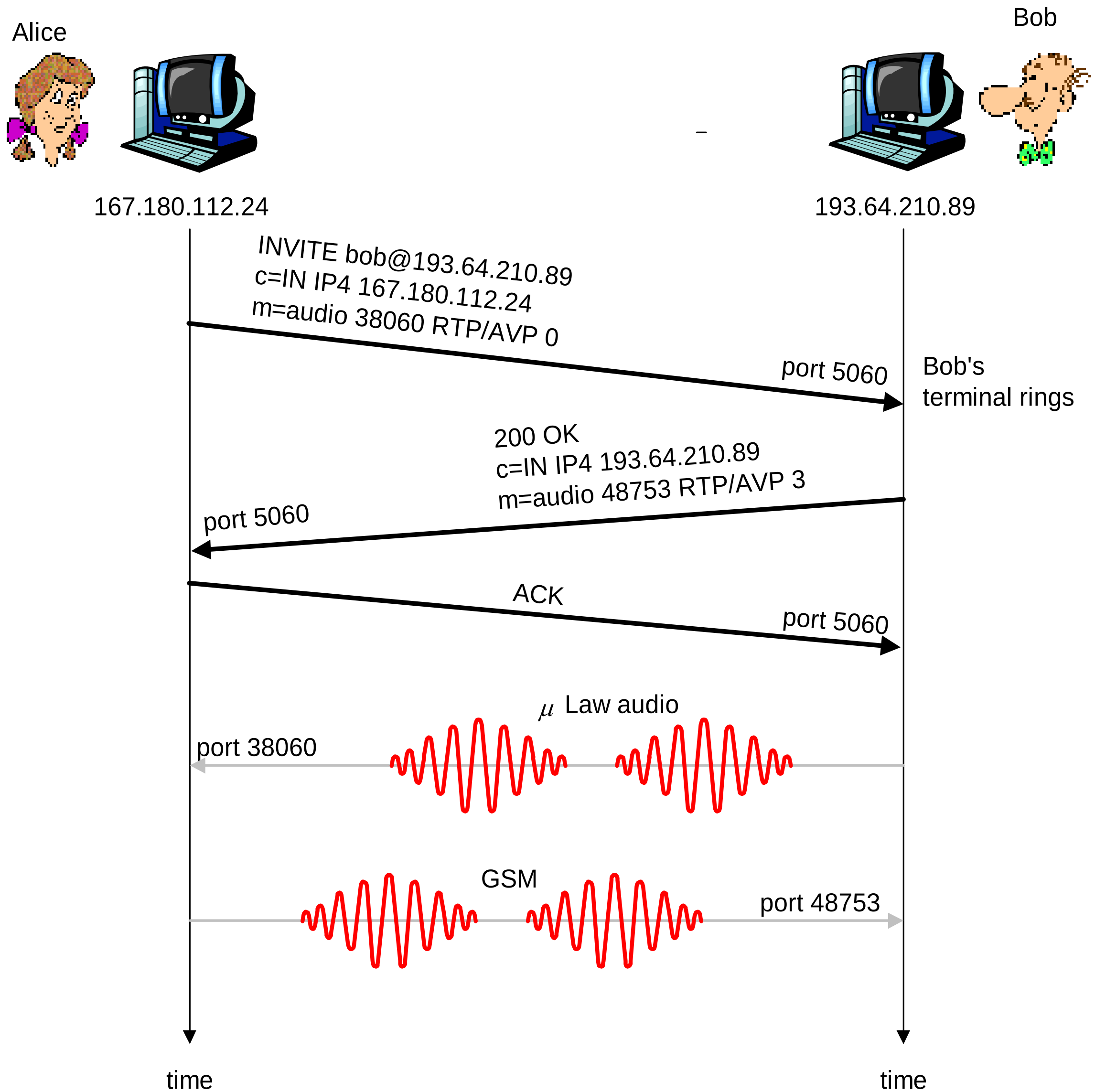
Henning Schulzrinne, Columbia University

- M. Handley, H. Schulzrinne, and E. Schooler, “SIP: session initiation protocol,” Internet Draft, Internet Engineering Task Force, Mar. 1997. Work in progress.
- H. Schulzrinne, A comprehensive multimedia control architecture for the Internet, 1997

SIP Services

- Setting up a call, SIP provides mechanisms ..
  - for caller to let callee know she wants to establish a call
  - so caller, callee can agree on media type, encoding
  - to end call
- determine current IP address of callee:
  - maps mnemonic identifier to current IP address
- call management:
  - add new media streams during call
  - change encoding during call
  - invite others
  - transfer, hold calls

Setting up a call to known IP address



- Alice's SIP invite message indicates her port number, IP address, encoding she prefers to receive (AVP 0: PCM ulaw)
- Bob's 200 OK message indicates his port number, IP address, preferred encoding (AVP 3: GSM)
- SIP is an out-of-band signalling protocol
- SIP messages can be sent over TCP or UDP. (All messages are ack'ed)
- default SIP port number is 5060.

Setting up a call (more)

- codec negotiation:
  - suppose Bob doesn't have PCM ulaw encoder.
  - Bob will instead reply with 606 Not Acceptable Reply, listing his encoders
  - Alice can then send new INVITE message, advertising different encoder
- rejecting a call
  - Bob can reject with replies "busy," "gone," "payment required," "forbidden"
- media can be sent over RTP or some other protocol

Example of SIP message

```
INVITE sip:bob@domain.com SIP/2.0
Via: SIP/2.0/UDP 167.180.112.24
From: sip:alice@hereway.com
To: sip:bob@domain.com
Call-ID: a2e3a@pigeon.hereway.com
Content-Type: application/sdp
Content-Length: 885

c=IN IP4 167.180.112.24
m=audio 38060 RTP/AVP 0
```

Notes:

- HTTP message syntax
- sdp = session description protocol
- Call-ID is unique for every call.

- Here we don't know Bob's IP address. Intermediate SIP servers needed.
- Alice sends, receives SIP messages using SIP default port 5060
- Alice specifies in Via: header that SIP client sends, receives SIP messages over UDP

Name translation and user location

- caller wants to call callee, but only has callee's name or e-mail address.
- need to get IP address of callee's current host:
  - user moves around
  - DHCP protocol
  - user has different IP devices (PC, PDA, car device)
- result can be based on:
  - time of day (work, home)
  - caller (don't want boss to call you at home)
  - status of callee (calls sent to voicemail when callee is already talking to someone)

<u>Service provided by SIP servers:</u>

- SIP registrar server
- SIP proxy server

SIP Registrar

- when Bob starts SIP client, client sends SIP REGISTER message to Bob's registrar server

  (similar function needed by Instant Messaging)
- registrar analogous to authoritative DNS server

Register Message:

```
REGISTER sip:domain.com SIP/2.0
Via: SIP/2.0/UDP 193.64.210.89
From: sip:bob@domain.com
To: sip:bob@domain.com
Expires: 3600
```

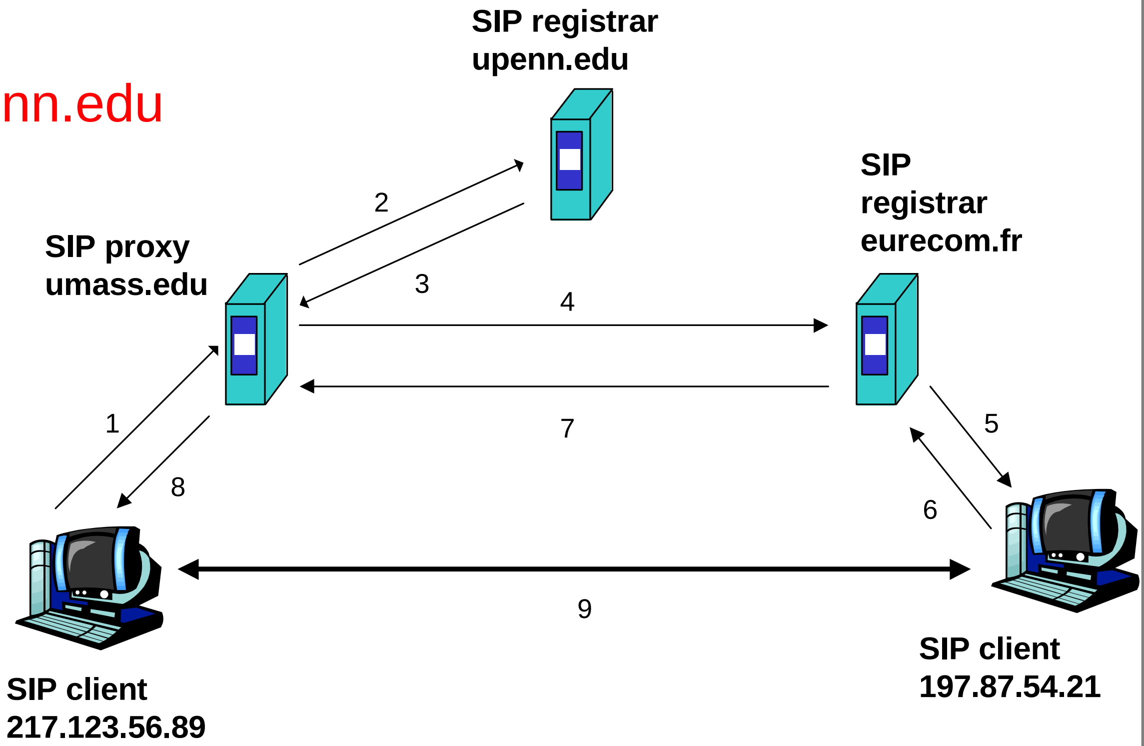
SIP Proxy

- Alice sends invite message to her proxy server
  - contains address sip:bob@domain.com
- proxy responsible for routing SIP messages to callee
  - possibly through multiple proxies.
- callee sends response back through the same set of proxies.
- proxy returns SIP response message to Alice
  - contains Bob's IP address
- proxy analogous to local DNS server

Example

Caller jim@umass.edu
places a call to keith@upenn.edu

(1) Jim sends INVITE message to umass SIP proxy.
(2) Proxy forwards request to upenn registrar server.
(3) upenn server returns redirect response, indicating that it should try keith@eurecom.fr



(4) umass proxy sends INVITE to eurecom registrar.
(5) eurecom registrar forwards INVITE to 197.87.54.21, which is running keith's SIP client.
(6-8) SIP response sent back
(9) media sent directly between clients.
Note: SIP ack messages not shown.

Comparison with H.323

- H.323 is another signaling protocol for real-time, interactive services
- H.323 is a complete, vertically integrated suite of protocols for multimedia conferencing: signaling, registration, admission control, transport, codecs
- SIP is a single component. Works with RTP, but does not mandate it. Can be combined with other protocols, services
- H.323 comes from the ITU (telephony).
- SIP comes from IETF: Borrows much of its concepts from HTTP
  - SIP has Web flavor, whereas H.323 has telephony flavor.
- SIP was based on the KISS principle: Keep it simple stupid. (Remark: after all SIP extensions, this is not any more the case.)