Chair for Network Architectures and Services – Prof. Carle
Department for Computer Science
TU München

# Master Course Computer Networks IN2097

**Prof. Dr.-Ing. Georg Carle**
**Christian Grothoff, Ph.D.**

**Chair for Network Architectures and Services**
**Institut für Informatik**
**Technische Universität München**
**http://www.net.in.tum.de**

Technische Universität München

## Chapter 6: Node Architectures

Our goals:

- understand main components of node architectures
- understand differences of packet-oriented and circuit-oriented protocol architectures
- understand principles for switching and virtualisation
- understand instantiations and implementations



## 6: Node Architectures

- 6.1 Virtual Circuit and Datagram Networks
- 6.2 What's inside a Router
- 6.3 Link virtualization: ATM, MPLS



## Classification of Switches

- Packet vs. circuit switches
  - packets have headers and samples don't
- Connectionless vs. connection oriented
  - connection oriented switches need a call setup
  - setup is handled in *control plane* by *switch controller*
  - connectionless switches deal with *self-contained* datagrams

| | *Connectionless (router)* | *Connection-oriented (switching system)* |
|---|---|---|
| Packet switch | Internet router | ATM switching system |
| Circuit switch | | Telephone switching system |

## Requirements

- Capacity of a switch is the maximum rate at which it can move information, assuming all data paths are simultaneously active
- Primary goal: maximize capacity
  - subject to cost and reliability constraints
- Circuit switch must reject call if can't find a path for samples from input to output
  - goal: minimize call blocking
- Packet switch must reject a packet if it can't find a buffer to store it awaiting access to output trunk
  - goal: minimize packet loss
- Don't reorder packets



## Connection setup

- In addition to routing and forwarding, 3rd important function in some network architectures:
  - ATM, frame relay, X.25
- before datagrams flow, two end hosts and intervening switches/routers establish virtual connection
  - switches/routers get involved
- network vs transport layer connection service:
  - network: between two hosts (may also involve intervening switches/routers in case of VCs)
  - transport: between two processes



## Network service model

Q: What *service model* for "channel" transporting datagrams from sender to receiver?

Example services for individual datagrams:

- guaranteed delivery
- guaranteed delivery with less than 40 msec delay

Example services for a flow of datagrams:

- in-order datagram delivery
- guaranteed minimum bandwidth to flow
- restrictions on changes in inter-packet spacing



## Network layer service models

| Network Architecture | Service Model | Guarantees ? Bandwidth | Loss | Order | Timing | Congestion feedback |
|---|---|---|---|---|---|---|
| Internet | best effort | none | no | no | no | no (inferred via loss) |
| ATM | CBR | constant rate | yes | yes | yes | no congestion |
| ATM | VBR | guaranteed rate | yes | yes | yes | no congestion |
| ATM | ABR | guaranteed minimum | no | yes | no | yes |
| ATM | UBR | none | no | yes | no | no |

## Network layer connection and connection-less service

- datagram network provides network-layer connectionless service
- VC network provides network-layer connection service
- analogous to the transport-layer services, but:
  - service: host-to-host
  - no choice: network provides one or the other
  - implementation: in network core



## 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 address)
- *every* router on source-to-destination path maintains "state" for each passing connection
- link, router resources (bandwidth, buffers) may be *allocated* to VC (dedicated resources = predictable service)



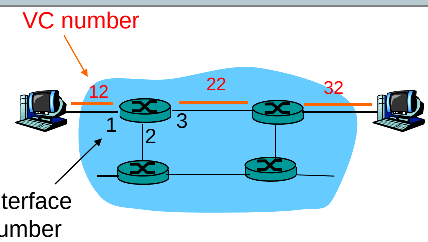
## VC implementation

a VC consists of:
1. path from source to destination
2. VC numbers, one number for each link along path
3. entries in forwarding tables in routers along path

- packet belonging to VC carries VC number (rather than destination address)
- VC number can be changed on each link.
  - New VC number comes from forwarding table



## Forwarding table



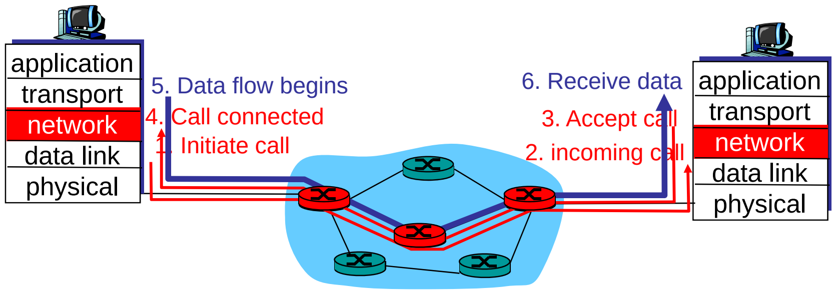

Forwarding table in northwest router:

| Incoming interface | Incoming VC # | Outgoing interface | Outgoing VC # |
|---|---|---|---|
| 1 | 12 | 3 | 22 |
| 2 | 63 | 1 | 18 |
| 3 | 7 | 2 | 17 |
| 1 | 97 | 3 | 87 |
| … | … | … | … |

Routers maintain connection state information!

## Virtual circuits: signaling protocols

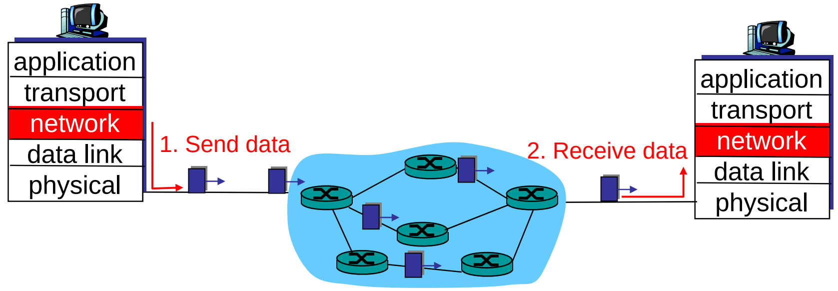
- used to setup, maintain  teardown VC
- used in ATM, frame-relay, X.25
- not used in today's Internet






## Datagram networks

- no call setup at network layer
- routers: no state about end-to-end connections
  - no network-level concept of "connection"
- packets forwarded using destination host address
  - packets between same source-dest pair may take different paths






## Datagram or VC network: why?

**Internet (datagram)**
- data exchange among computers
  - "elastic" service, no strict timing req.
- "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 (VC)**
- evolved from telephony
- human conversation:
  - strict timing, reliability requirements
  - need for guaranteed service
- "dumb" end systems
  - telephones
  - complexity inside network



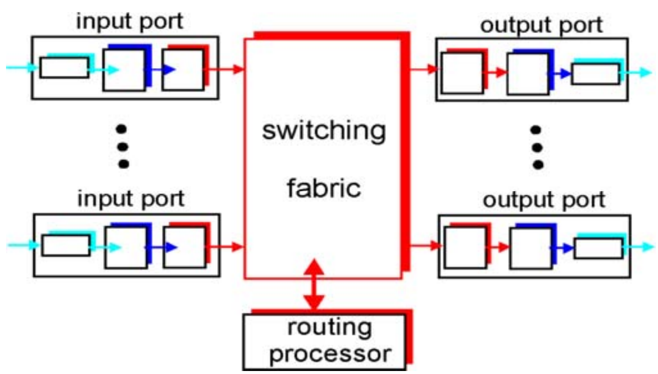
## 6: Node Architectures

- 6.1 Virtual Circuit and Datagram Networks
- 6.2 What's inside a Router
- 6.3 Link virtualization: ATM, MPLS

## Router Architecture Overview

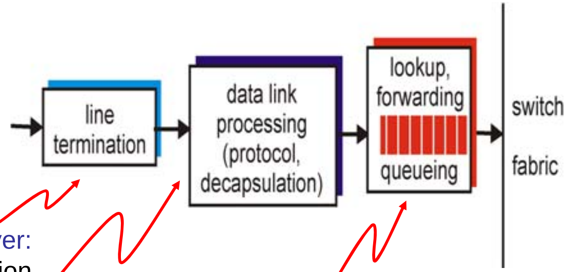
Two key router functions:

- run routing algorithms/protocol (RIP, OSPF, BGP)
- *forwarding* datagrams from incoming to outgoing link






## Input Port Functions




Physical layer:
bit-level reception

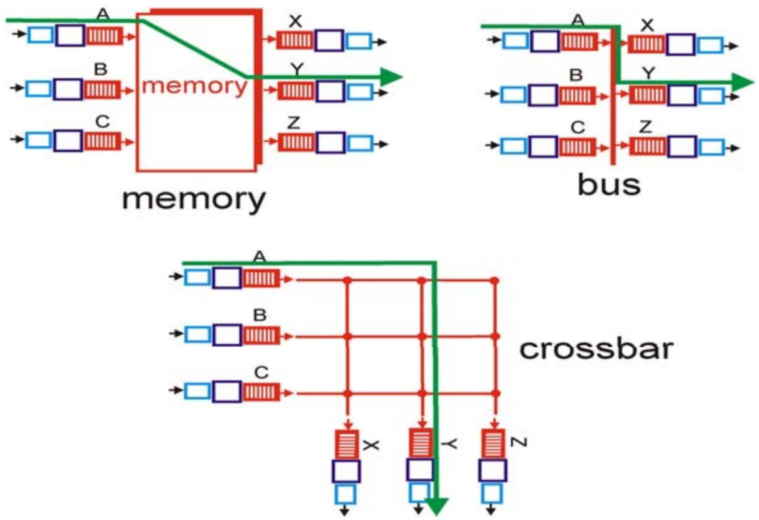
Data link layer:
e.g., Ethernet

**Decentralized switching**:

- given datagram destination, lookup output port using forwarding table in input port memory
- goal: complete input port processing at 'line speed'
- queuing: if datagrams arrive faster than forwarding rate into switch fabric



## Three types of switching fabrics

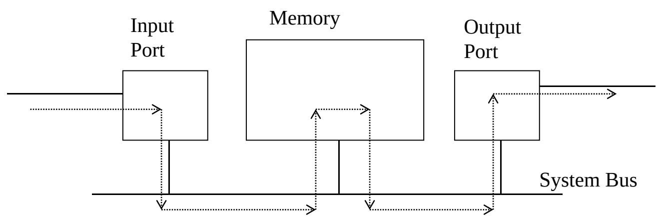





## Switching Via Memory

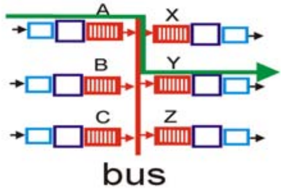
First generation routers:

- traditional computers with switching under direct control of CPU
- packet copied to system's memory
- speed limited by memory bandwidth (2 bus crossings per datagram)



## Switching Via a Bus

- datagram from input port memory to output port memory via a shared bus
- bus contention: switching speed limited by bus bandwidth
- 32 Gbps bus, Cisco 5600: sufficient speed for access and enterprise routers



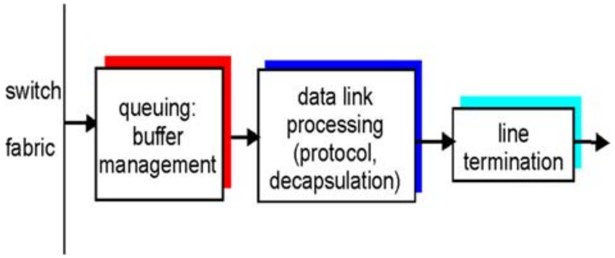



## Switching Via An Interconnection Network

- overcome bus bandwidth limitations
- Banyan networks, other interconnection nets initially developed to connect processors in multiprocessor
- advanced design: fragmenting datagram into fixed length cells, switch cells through the fabric.
- Cisco 12000: switches 60 Gbps through the interconnection network



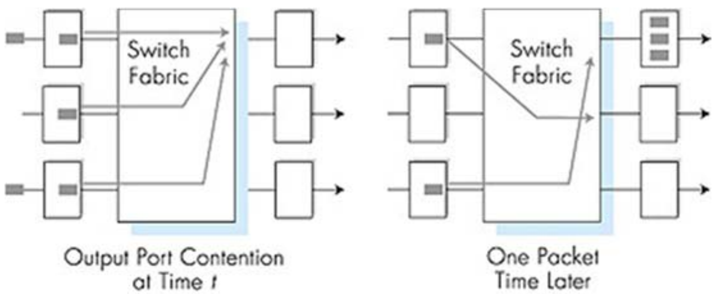
## Output Ports




- *Buffering* required when datagrams arrive from fabric faster than the transmission rate
- *Scheduling discipline* chooses among queued datagrams for transmission



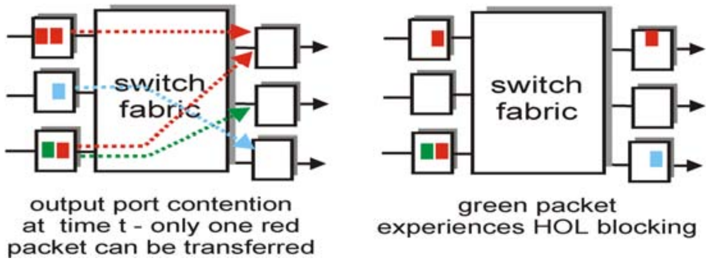
## Output port queueing




- buffering when arrival rate via switch exceeds output line speed
- *queueing (delay) and loss due to output port buffer overflow!*

## Input Port Queuing

- Fabric slower than input ports combined -> queueing may occur at input queues
- Head-of-the-Line (HOL) blocking: queued datagram at front of queue prevents others in queue from moving forward
- *queueing delay and loss due to input buffer overflow!*






## 6: Node Architectures

- 6.1 Virtual Circuit and Datagram Networks
- 6.2 What's inside a Router
- 6.3 Link virtualization: ATM, MPLS



## Virtualization of networks

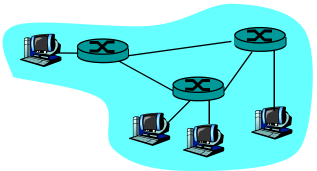
- Virtualization of resources: powerful abstraction in systems engineering:
- computing examples: virtual memory, virtual devices
  - Virtual machines: e.g., java
  - IBM VM operation system from 1960's/70's
- layering of abstractions: don't sweat the details of the lower layer, only deal with lower layers abstractly

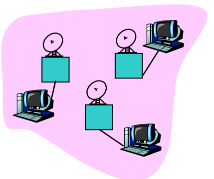


## The Internet: virtualizing networks

- 1974: multiple unconnected nets
  - ARPAnet
  - data-over-cable networks
  - packet satellite network (Aloha protocol)
  - packet radio network

… differing in:
- addressing conventions
- packet formats
- error recovery
- routing


ARPAnet


satellite net

"A Protocol for Packet Network Intercommunication", V. Cerf, R. Kahn, IEEE Transactions on Communications, May, 1974, pp. 637-648.

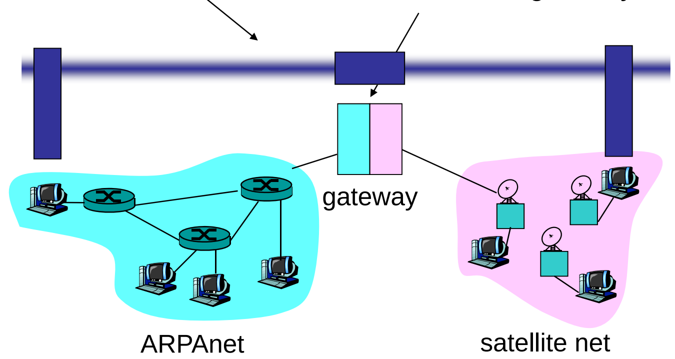
## The Internet: virtualizing networks

Internetwork layer (IP):

- addressing: internetwork appears as single, uniform entity, despite underlying local network heterogeneity
- network of networks

Gateway:

- "embed internetwork packets in local packet format or extract them"
- route (at internetwork level) to next gateway






## Cerf & Kahn's Internetwork Architecture

- What is virtualized?
- two layers of addressing: internetwork and local network
- new layer (IP) makes everything homogeneous at internetwork layer
- underlying local network technology
  - cable
  - satellite
  - 56K telephone modem
  - today: ATM, MPLS
- … "invisible" at internetwork layer. Looks like a link layer technology to IP!



## ATM and MPLS

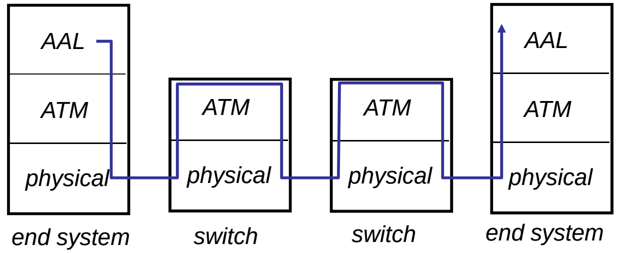
- ATM, MPLS separate networks in their own right
  - different service models, addressing, routing from Internet
- viewed by Internet as logical link connecting IP routers
  - just like dialup link is really part of separate network (telephone network)
- ATM, MPLS: of technical interest in their own right



## Asynchronous Transfer Mode: ATM

- **1990's/00 standard for high-speed** (155Mbps to 622 Mbps and higher) *Broadband Integrated Service Digital Network* architecture
- Goal: *integrated, end-end transport of carry voice, video, data*
  - meeting timing/QoS requirements of voice, video (versus Internet best-effort model)
  - "next generation" telephony: technical roots in telephone world
  - packet-switching (fixed length packets, called "cells") using virtual circuits

## ATM architecture



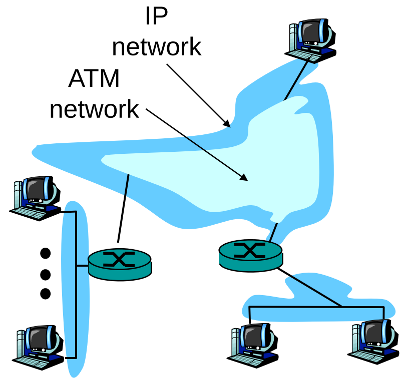

- adaptation layer: only at edge of ATM network
  - data segmentation/reassembly
  - roughly analagous to Internet transport layer
- ATM layer: "network" layer
  - cell switching, routing
- physical layer



## ATM: network or link layer?

Vision: end-to-end transport: "ATM from desktop to desktop"
- ATM *is* a network technology

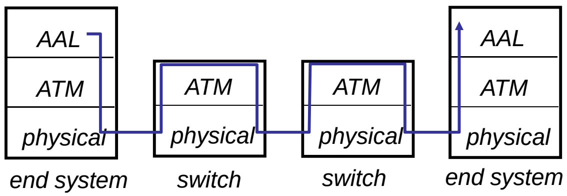
Reality: used to connect IP backbone routers
- "IP over ATM"
- ATM as switched link layer, connecting IP routers






## ATM Adaptation Layer (AAL)

- ATM **Adaptation Layer** (AAL): "adapts" upper layers (IP or native ATM applications) to ATM layer below
- AAL present **only in end systems**, not in switches
- AAL layer segment (header/trailer fields, data) fragmented across multiple ATM cells
  - analogy: TCP segment in many IP packets

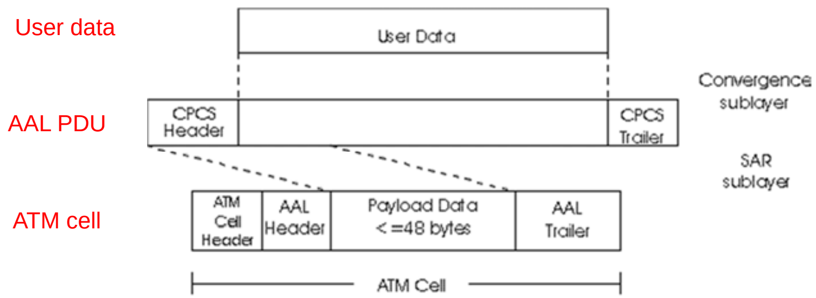





## ATM Adaptation Layer (AAL) [more]

Different versions of AAL layers, depending on ATM service class:

- AAL1: for CBR (Constant Bit Rate) services, e.g. circuit emulation
- AAL2: for VBR (Variable Bit Rate) services, e.g., MPEG video
- AAL5: for data (eg, IP datagrams)



## ATM Layer

Service: transport cells across ATM network

- analogous to IP network layer
- very different services than IP network layer

| Network Architecture | Service Model | Guarantees ? Bandwidth | Loss | Order | Timing | Congestion feedback |
|---|---|---|---|---|---|---|
| Internet | best effort | none | no | no | no | no (inferred via loss) |
| ATM | CBR | constant rate | yes | yes | yes | no congestion |
| ATM | VBR | guaranteed rate | yes | yes | yes | no congestion |
| ATM | ABR | guaranteed minimum | no | yes | no | yes |
| ATM | UBR | none | no | yes | no | no |



## ATM Layer: Virtual Circuits

- VC transport: cells carried on VC from source to dest
  - call setup, teardown for each call *before* data can flow
  - each packet carries VC identifier (not destination ID)
  - *every* switch on source-dest path maintain "state" for each passing connection
  - link,switch resources (bandwidth, buffers) may be *allocated* to VC: to get circuit-like perf.
- Permanent VCs (PVCs)
  - long lasting connections
  - typically: "permanent" route between to IP routers
- Switched VCs (SVC):
  - dynamically set up on per-call basis



## ATM VCs

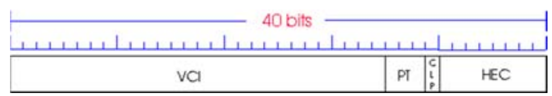
- Advantages of ATM VC approach:
  - QoS performance guarantee for connection mapped to VC (bandwidth, delay, delay jitter)
- Drawbacks of ATM VC approach:
  - Inefficient support of datagram traffic
  - one PVC between each source/dest pair) does not scale (N*2 connections needed)
  - SVC introduces call setup latency, processing overhead for short lived connections



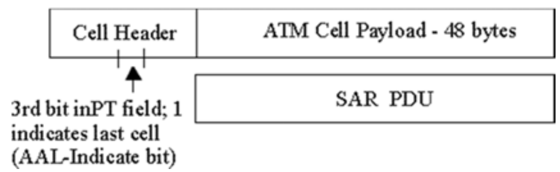
## ATM Layer: ATM cell

- 5-byte ATM cell header
- 48-byte payload
  - Why?: small payload -> short cell-creation delay for digitized voice
  - halfway between 32 and 64 (compromise!)

Cell header

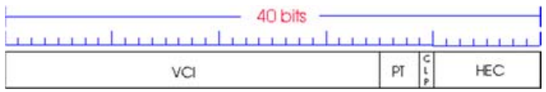



Cell format



## ATM cell header

- **VCI:** virtual channel ID
  - will *change* from link to link thru net
- **PT:** Payload type (e.g. RM cell versus data cell)
- **CLP:** Cell Loss Priority bit
  - CLP = 1 implies low priority cell, can be discarded if congestion
- **HEC:** Header Error Checksum
  - cyclic redundancy check






## ATM Physical Layer (more)

*Two* pieces (sublayers) of physical layer:

- Transmission Convergence Sublayer (TCS): adapts ATM layer above to PMD sublayer below
- Physical Medium Dependent: depends on physical medium being used

TCS Functions:

- Header **checksum** generation: 8 bits CRC
- Cell **delineation**
- With “unstructured” PMD sublayer, transmission of **idle cells** when no data cells to send



## ATM Physical Layer

Physical Medium Dependent (PMD) sublayer

- **SONET/SDH**: transmission frame structure (like a container carrying bits);
  - bit synchronization;
  - bandwidth partitions (TDM);
  - several speeds: OC3 = 155.52 Mbps; OC12 = 622.08 Mbps; OC48 = 2.45 Gbps, OC192 = 9.6 Gbps
- **TI/T3**: transmission frame structure (old telephone hierarchy): 1.5 Mbps/ 45 Mbps
- **unstructured**: just cells (busy/idle)



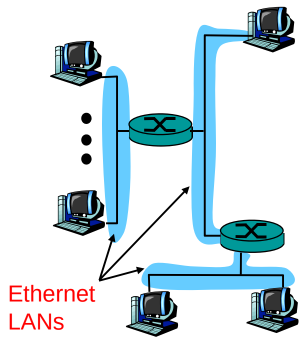
## IP-Over-ATM

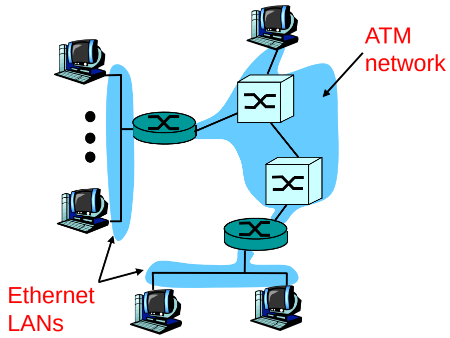
Classic IP only

- 3 “networks” (e.g., LAN segments)
- MAC (802.3) and IP addresses

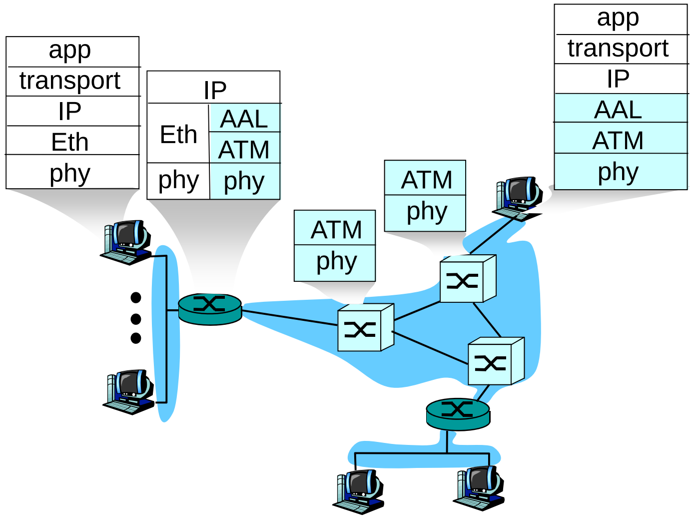
IP over ATM

- replace “network” (e.g., LAN segment) with ATM network
- ATM addresses, IP addresses



# IP-Over-ATM






# Datagram Journey in IP-over-ATM Network

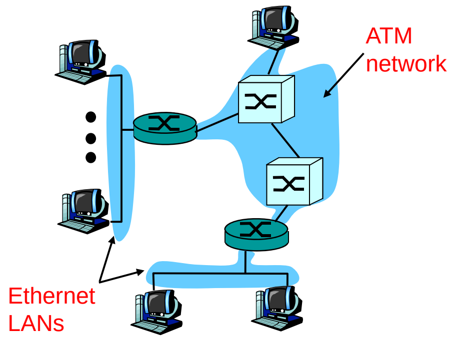
- at Source Host:
  - IP layer maps between IP, ATM dest address (using ARP)
  - passes datagram to AAL5
  - AAL5 encapsulates data, segments cells, passes to ATM layer
- ATM network: moves cell along VC to destination
- at Destination Host:
  - AAL5 reassembles cells into original datagram
  - if CRC OK, datagram is passed to IP



# IP-Over-ATM

Issues:

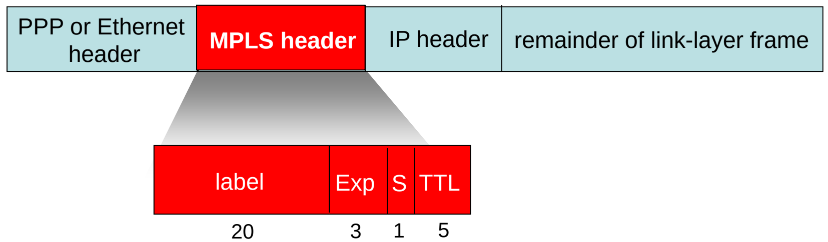
- IP datagrams into ATM AAL5 PDUs
- from IP addresses to ATM addresses
  - just like IP addresses to 802.3 MAC addresses!






# Multiprotocol label switching (MPLS)

- initial goal: speed up IP forwarding by using fixed length label (instead of IP address) to do forwarding
  - borrowing ideas from Virtual Circuit (VC) approach
  - but IP datagram still keeps IP address!

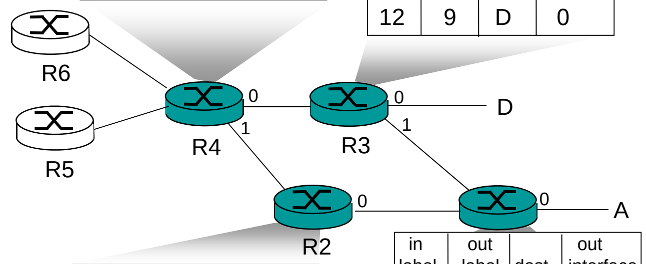


## MPLS capable routers

- a.k.a. label-switched router
- forwards packets to outgoing interface based only on label value (don't inspect IP address)
  - MPLS forwarding table distinct from IP forwarding tables
- signaling protocol needed to set up forwarding
  - RSVP-TE
  - forwarding possible along paths that IP alone would not allow (e.g., source-specific routing) !!
  - use MPLS for traffic engineering
- must co-exist with IP-only routers



## MPLS forwarding tables




| in label | out label | dest | out interface |
|---|---|---|---|
| | 10 | A | 0 |
| | 12 | D | 0 |
| | 8 | A | 1 |

| in label | out label | dest | out interface |
|---|---|---|---|
| 10 | 6 | A | 1 |
| 12 | 9 | D | 0 |

| in label | out label | dest | out interface |
|---|---|---|---|
| 8 | 6 | A | 0 |

| in label | out label | dest | out interface |
|---|---|---|---|
| 6 | - | A | 0 |