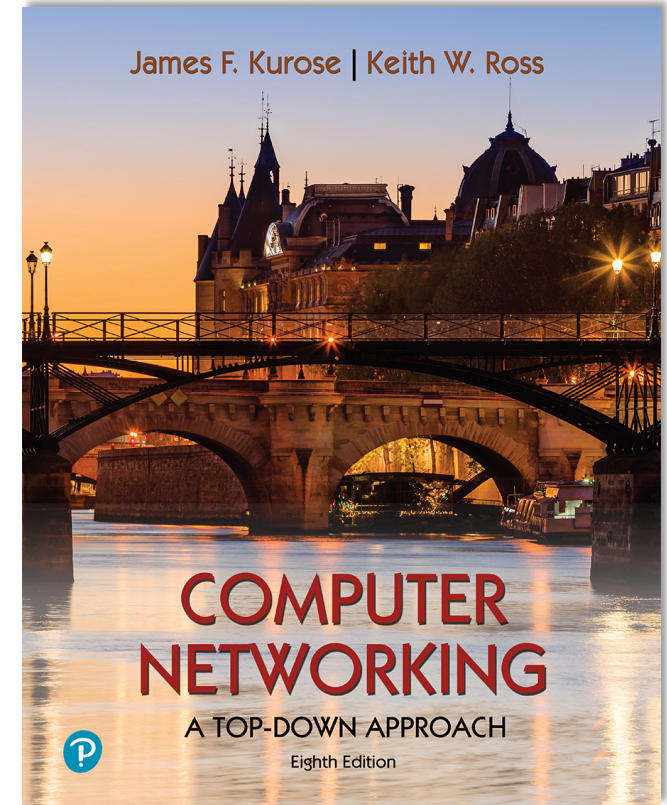


Chapter 6

The Link Layer and LANs



Computer Networking: A Top-Down Approach

8th edition

Jim Kurose, Keith Ross
Pearson, 2020

Acknowledgement: Based on the textbook's website:
https://gaia.cs.umass.edu/kurose_ross/index.php

Link layer, LANs: roadmap

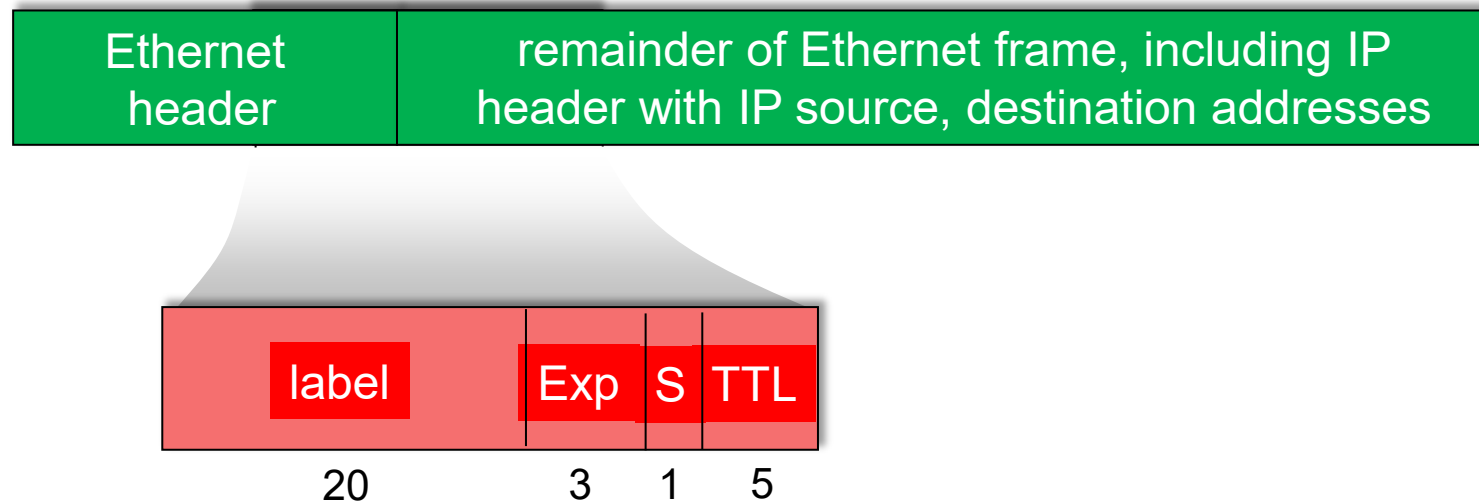
- introduction
- error detection, correction
- multiple access protocols
- LANs
 - addressing, ARP
 - Ethernet
 - switches
 - VLANs
- **link virtualization: MPLS**
- data center networking



- a day in the life of a web request

Multiprotocol label switching (MPLS)

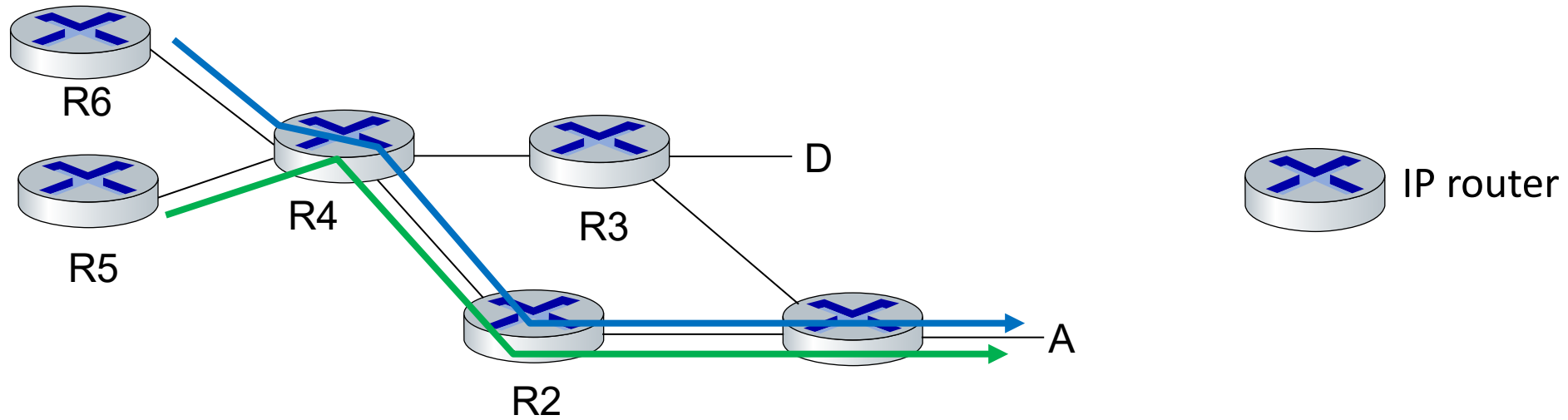
- **goal:** high-speed IP forwarding among network of MPLS-capable routers, using fixed length label (instead of shortest prefix matching)
 - faster lookup using fixed length identifier
 - borrowing ideas from Virtual Circuit (VC) approach
 - but IP datagram still keeps IP address!



MPLS capable routers

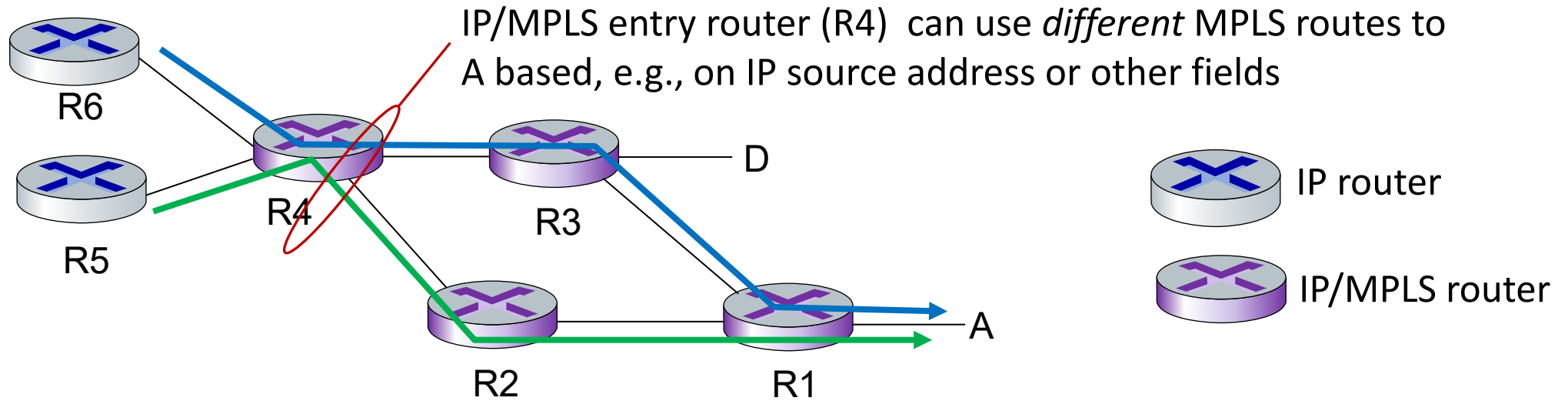
- a.k.a. label-switched router
- forward packets to outgoing interface based only on label value (*don't inspect IP address*)
 - MPLS forwarding table distinct from IP forwarding tables
- *flexibility*: MPLS forwarding decisions can *differ* from those of IP
 - use destination *and* source addresses to route flows to same destination differently (traffic engineering)
 - re-route flows quickly if link fails: pre-computed backup paths

MPLS versus IP paths



- **IP routing:** path to destination determined by destination address alone

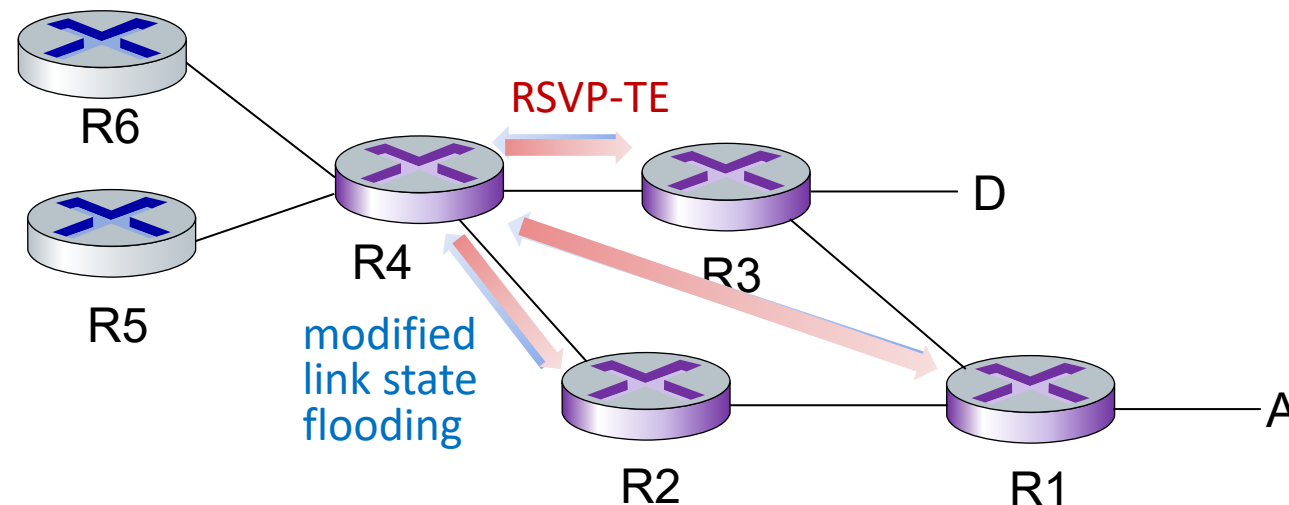
MPLS versus IP paths



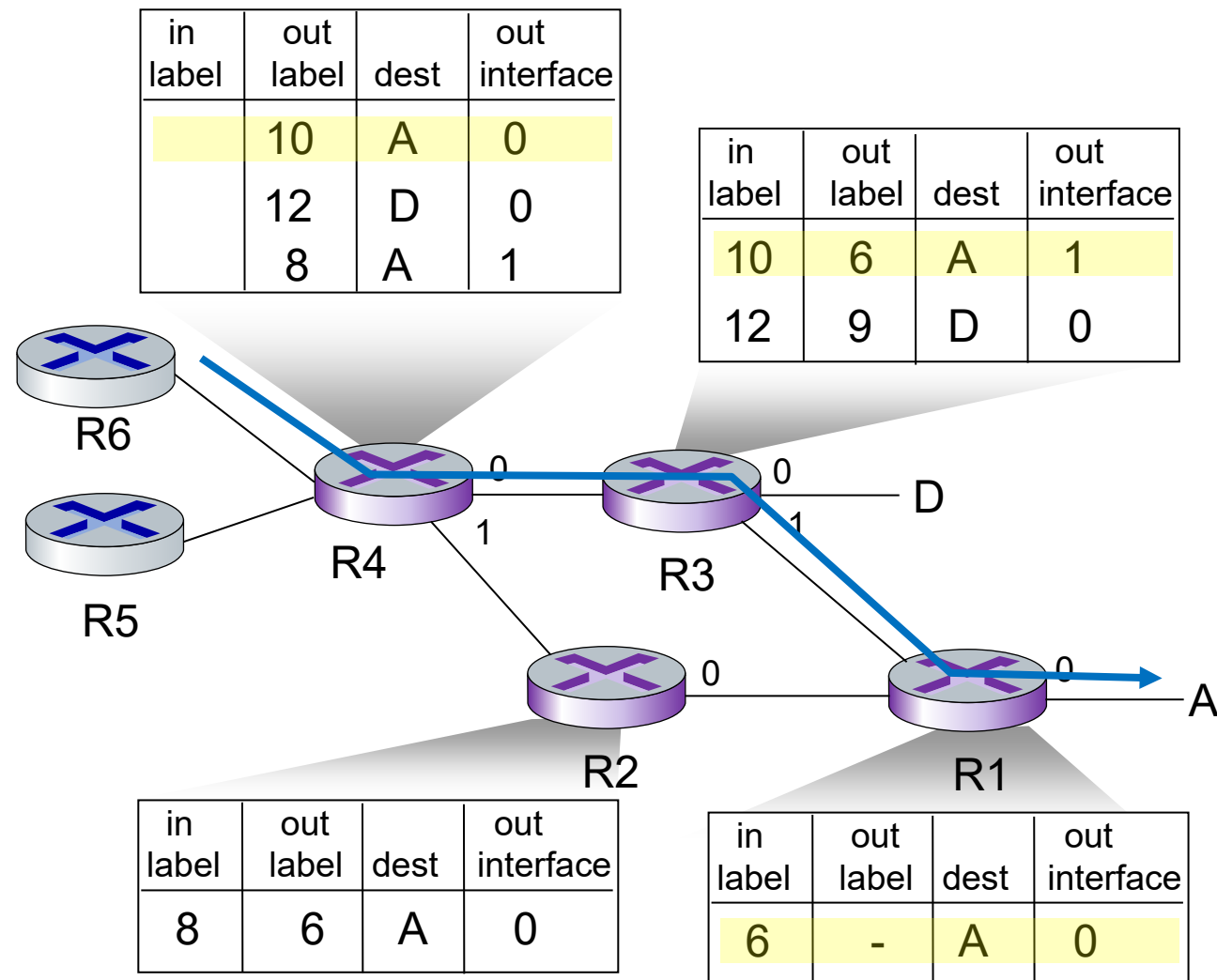
- **IP routing:** path to destination determined by destination address alone
- **MPLS routing:** path to destination can be based on source *and* destination address
 - flavor of generalized forwarding (MPLS 10 years earlier)
 - *fast reroute*: precompute backup routes in case of link failure

MPLS signaling

- modify OSPF, IS-IS link-state flooding protocols to carry info used by MPLS routing:
 - e.g., link bandwidth, amount of “reserved” link bandwidth
- entry MPLS router uses RSVP-TE signaling protocol to set up MPLS forwarding at downstream routers



MPLS forwarding tables



Link layer, LANs: roadmap

- introduction
- error detection, correction
- multiple access protocols
- LANs
 - addressing, ARP
 - Ethernet
 - switches
 - VLANs
- link virtualization: MPLS
- data center networking



- a day in the life of a web request

Datacenter networks

10's to 100's of thousands of hosts, often closely coupled, in close proximity:

- e-business (e.g. Amazon)
- content-servers (e.g., YouTube, Akamai, Apple, Microsoft)
- search engines, data mining (e.g., Google)

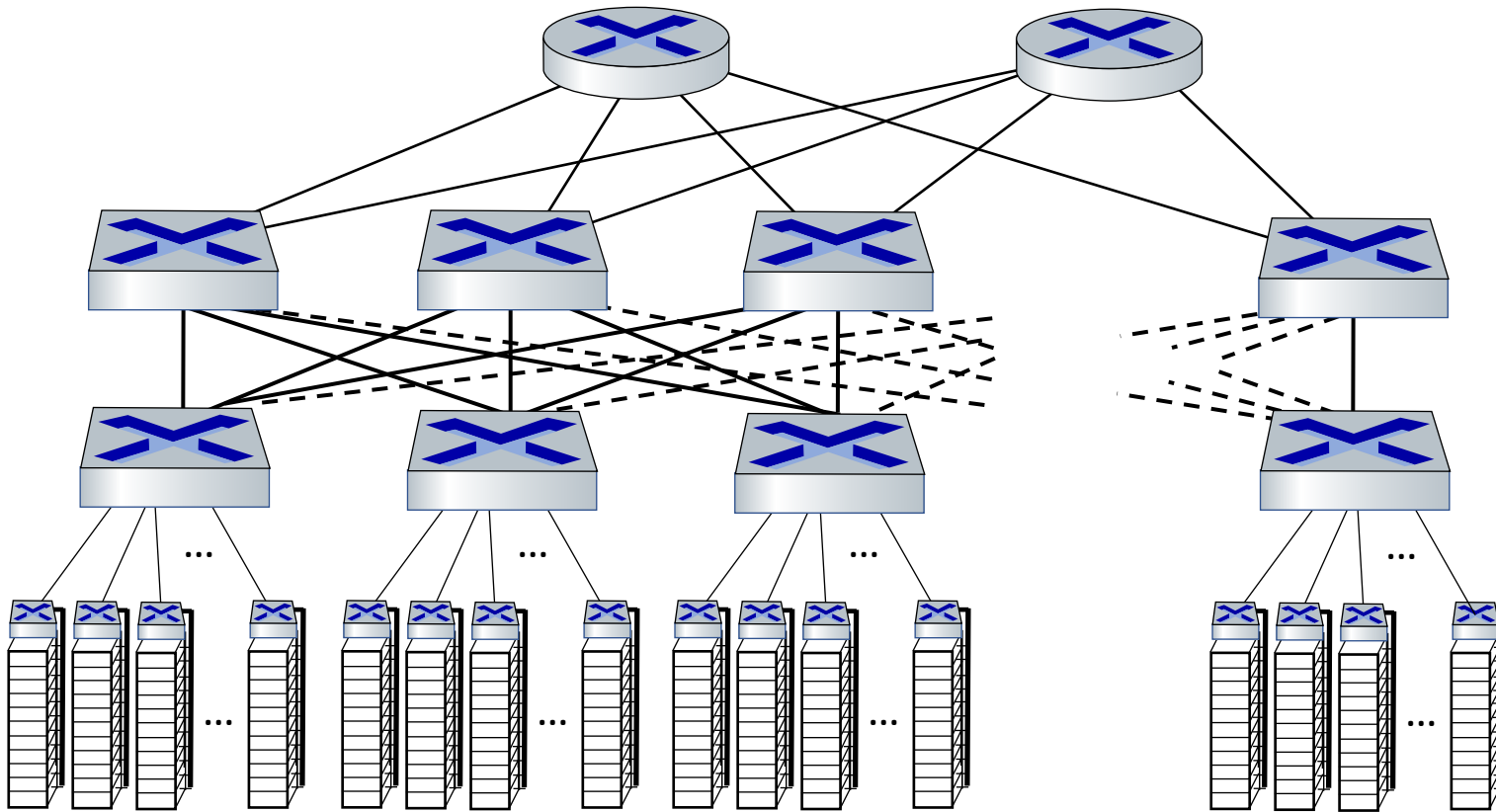
challenges:

- multiple applications, each serving massive numbers of clients
- reliability
- managing/balancing load, avoiding processing, networking, data bottlenecks



Inside a 40-ft Microsoft container, Chicago data center

Datacenter networks: network elements



Border routers

- connections outside datacenter

Tier-1 switches

- connecting to ~16 T-2s below

Tier-2 switches

- connecting to ~16 TORs below

Top of Rack (TOR) switch

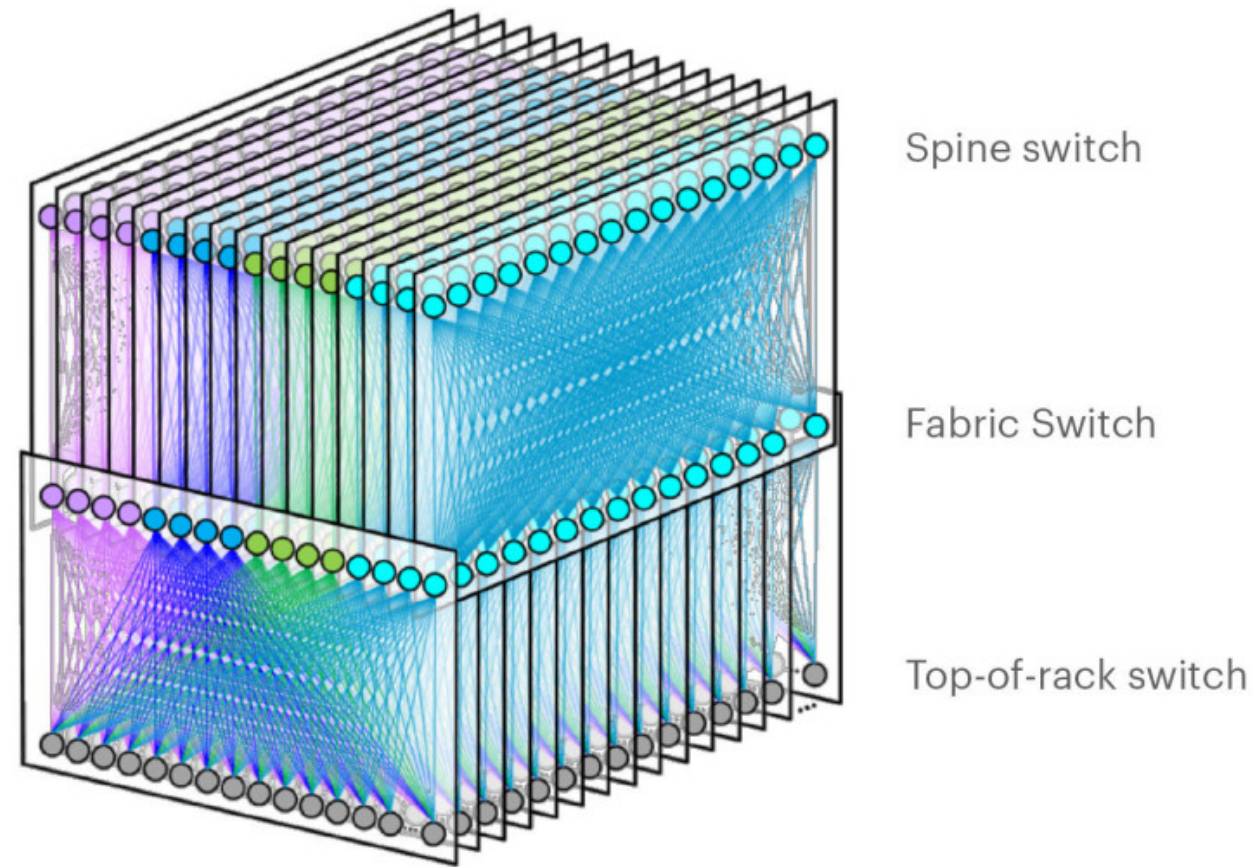
- one per rack
- 100G-400G Ethernet to blades

Server racks

- 20- 40 server blades: hosts

Datacenter networks: network elements

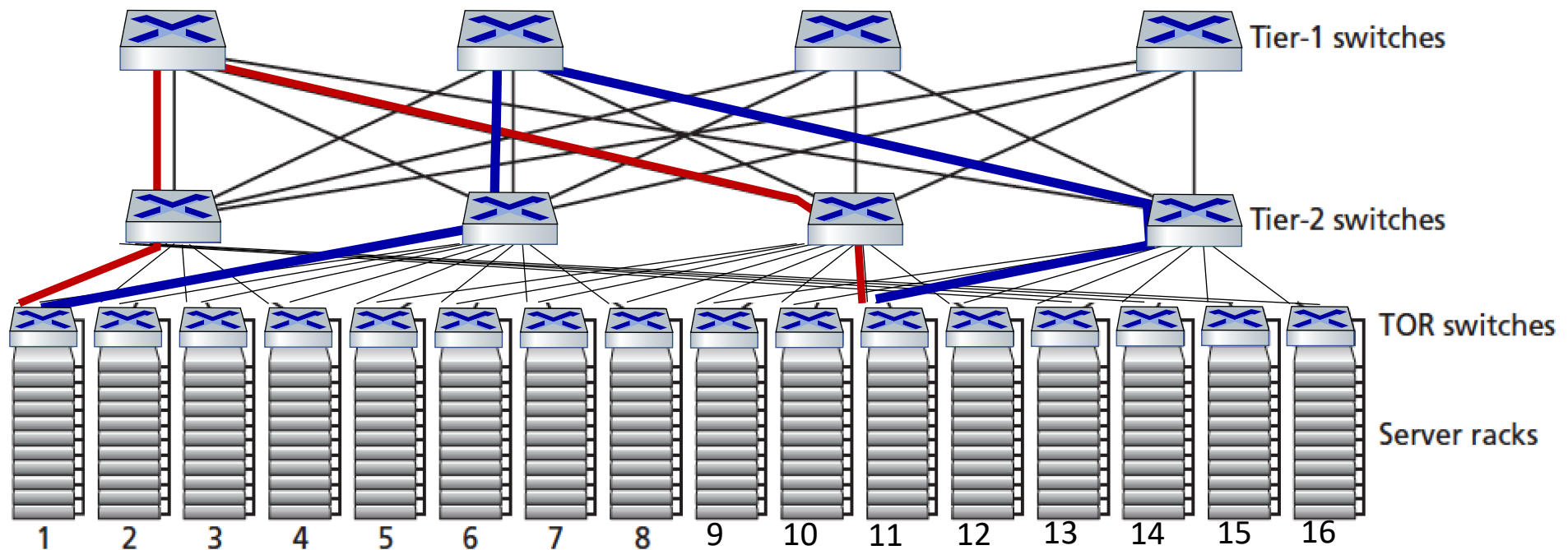
Facebook F16 data center network topology:



<https://engineering.fb.com/data-center-engineering/f16-minipack/> (posted 3/2019)

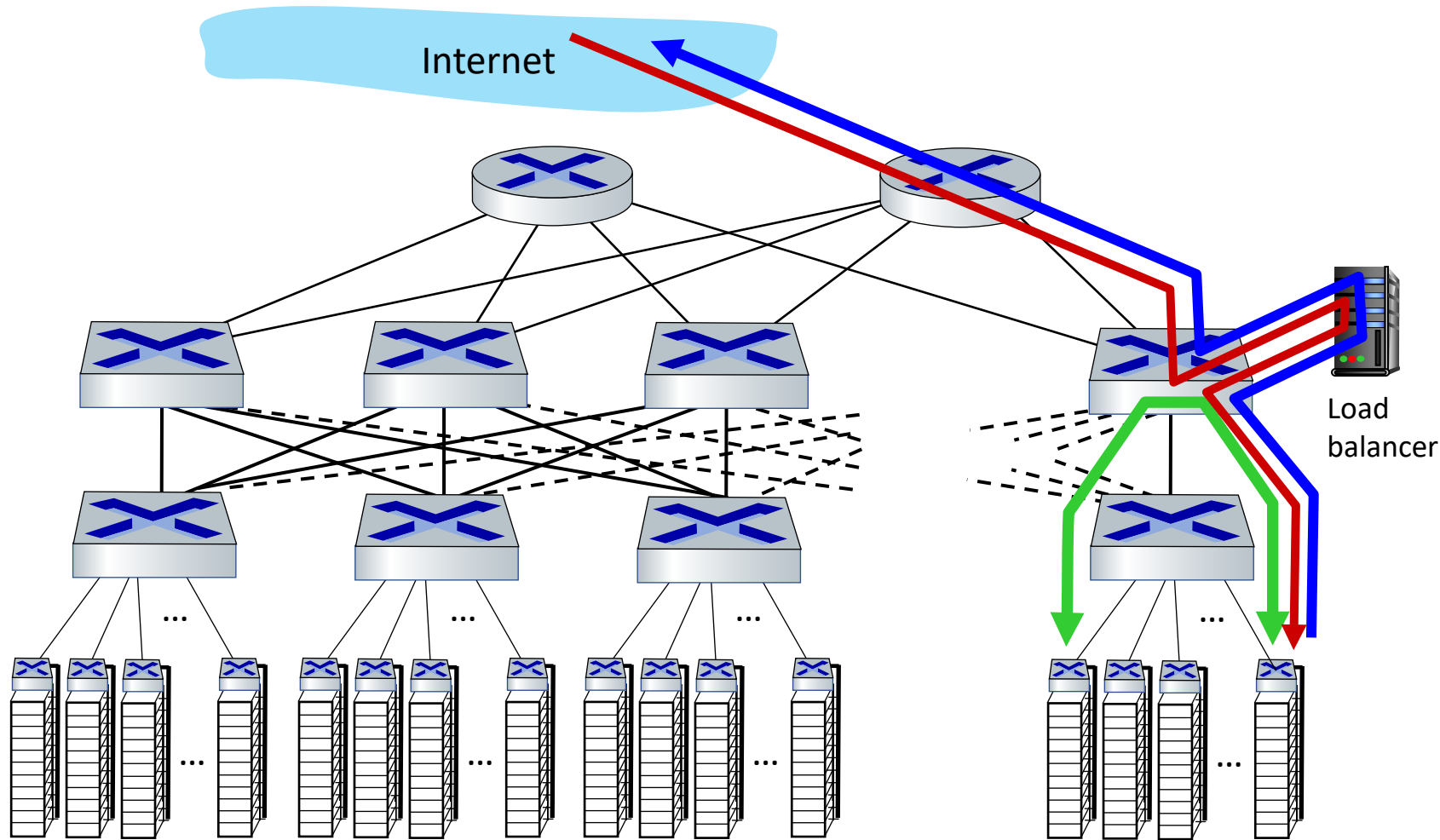
Datacenter networks: multipath

- rich interconnection among switches, racks:
 - increased throughput between racks (multiple routing paths possible)
 - increased reliability via redundancy



two **disjoint** paths highlighted between racks 1 and 11

Datacenter networks: application-layer routing



load balancer:
application-layer
routing

- receives external client requests
- directs workload within data center
- returns results to external client (hiding data center internals from client)

Datacenter networks: protocol innovations

- link layer:

- RoCE: remote DMA (RDMA) over Converged Ethernet

- transport layer:

- ECN (explicit congestion notification) used in transport-layer congestion control (DCTCP, DCQCN)
- experimentation with hop-by-hop (backpressure) congestion control

- routing, management:

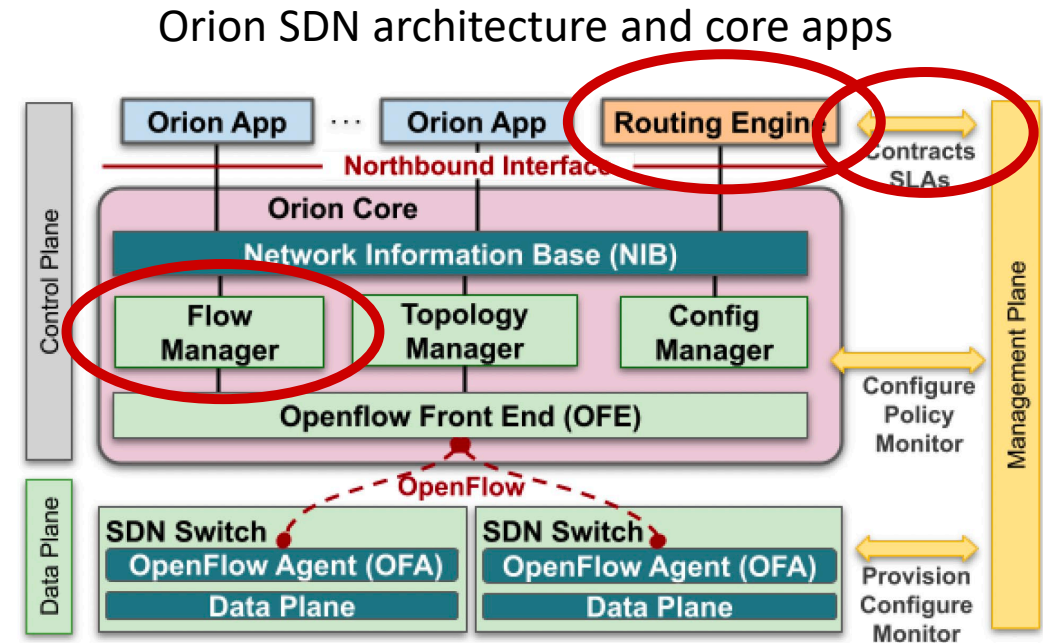
- SDN widely used within/among organizations' datacenters
- place related services, data as close as possible (e.g., in same rack or nearby rack) to minimize tier-2, tier-1 communication

ORION: Google's new SDN control plane for internal datacenter (Jupiter) + wide area (B4) network

- **routing** (intradomain, iBGP), traffic engineering: implemented in *applications* on top of ORION core
- **edge-edge flow-based** controls (e.g., CoFlow scheduling) to meet contract SLAs
- **management**: pub-sub distributed microservices in Orion core, OpenFlow for switch signaling/monitoring

Note:

- no routing protocols, congestion control (partially) also managed by SDN rather than by protocol
- are protocols dying?



Link layer, LANs: roadmap

- introduction
- error detection, correction
- multiple access protocols
- LANs
 - addressing, ARP
 - Ethernet
 - switches
 - VLANs
- link virtualization: MPLS
- data center networking

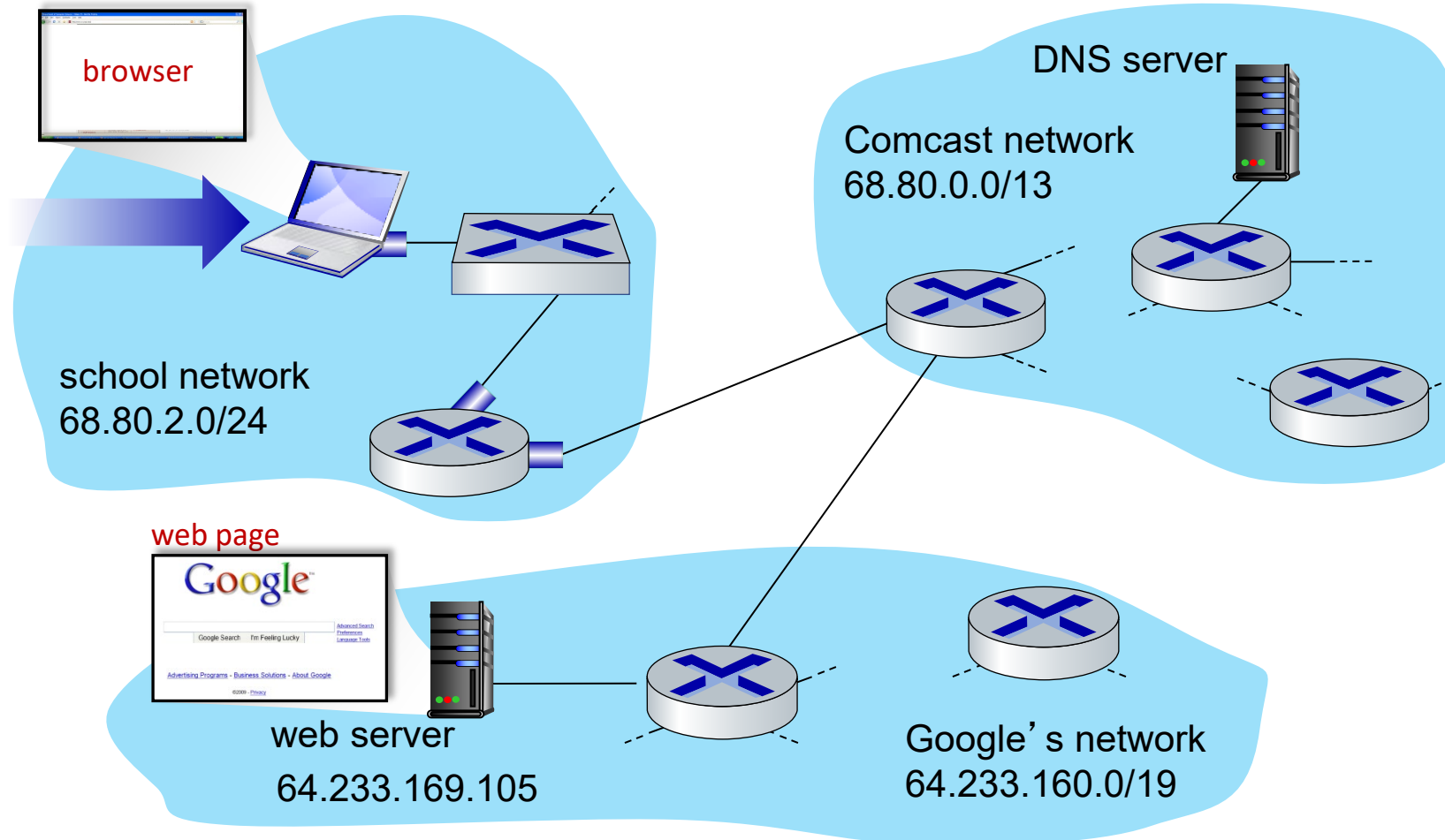


- a day in the life of a web request

Synthesis: a day in the life of a web request

- our journey down the protocol stack is now complete!
 - application, transport, network, link
- putting-it-all-together: synthesis!
 - *goal*: identify, review, understand protocols (at all layers) involved in seemingly simple scenario: requesting www page
 - *scenario*: student attaches laptop to campus network, requests/receives www.google.com

A day in the life: scenario

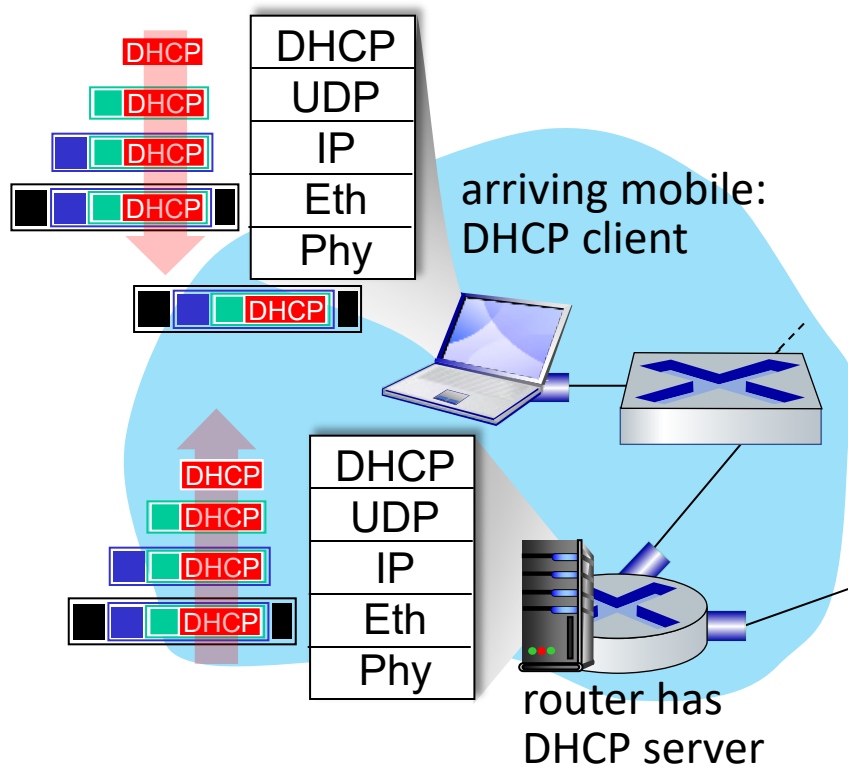


scenario:

- arriving mobile client attaches to network ...
- requests web page:
`www.google.com`

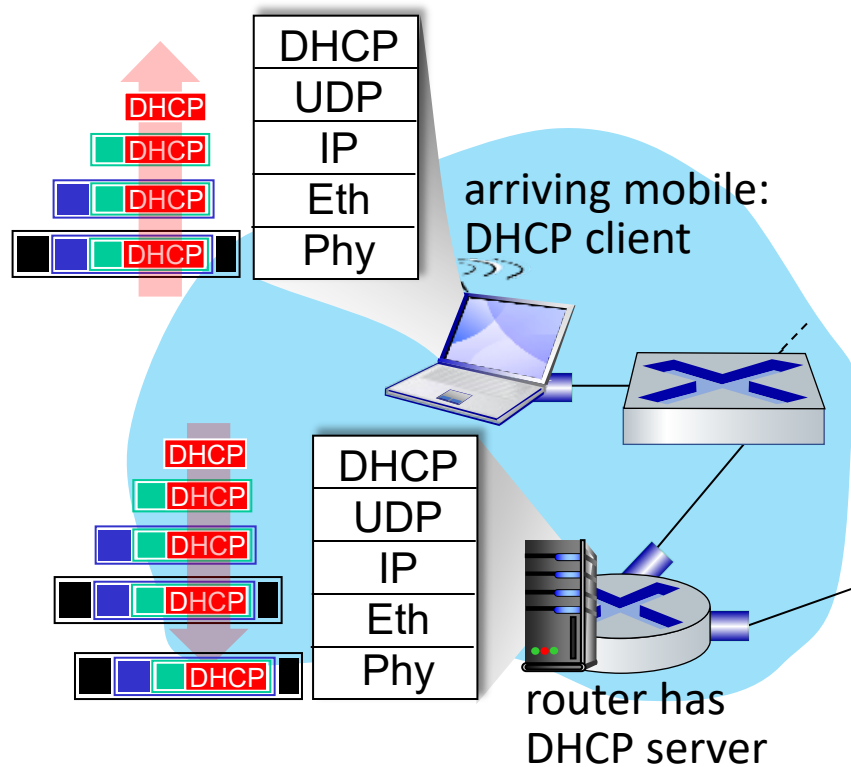
Sounds simple! 

A day in the life: connecting to the Internet



- connecting laptop needs to get its own IP address, addr of first-hop router, addr of DNS server: use **DHCP**
- DHCP request **encapsulated** in **UDP**, encapsulated in **IP**, encapsulated in **802.3 Ethernet**
- Ethernet frame **broadcast** (dest: FFFFFFFFFFFFFFFF) on LAN, received at router running **DHCP** server
- Ethernet **de-muxed** to IP de-muxed, UDP de-muxed to DHCP

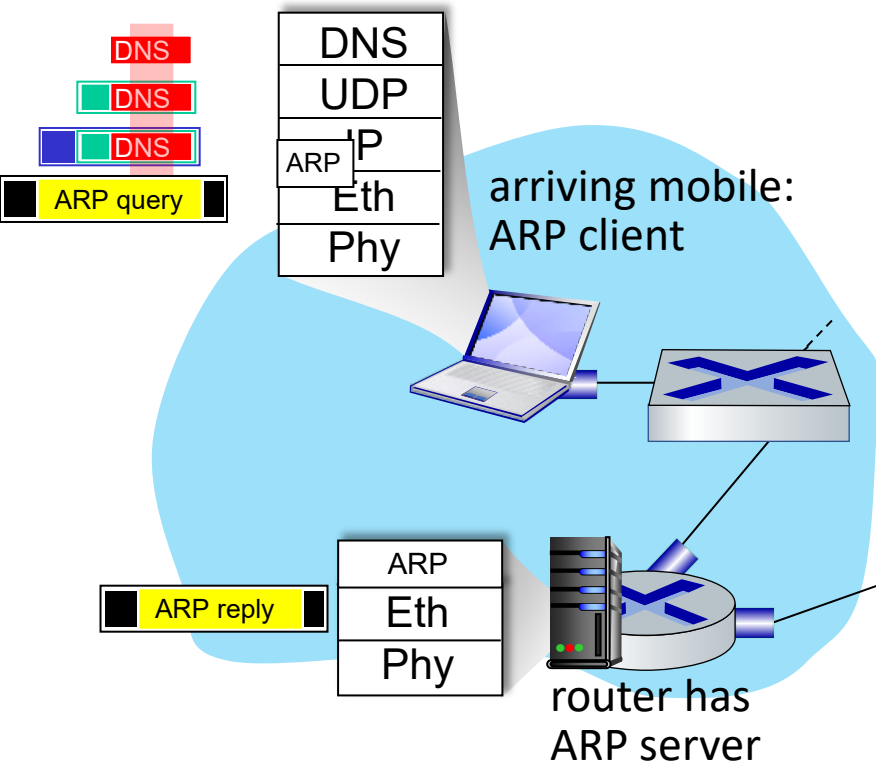
A day in the life: connecting to the Internet



- DHCP server formulates **DHCP ACK** containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server
- encapsulation at DHCP server, frame forwarded (**switch learning**) through LAN, demultiplexing at client
- DHCP client receives DHCP ACK reply

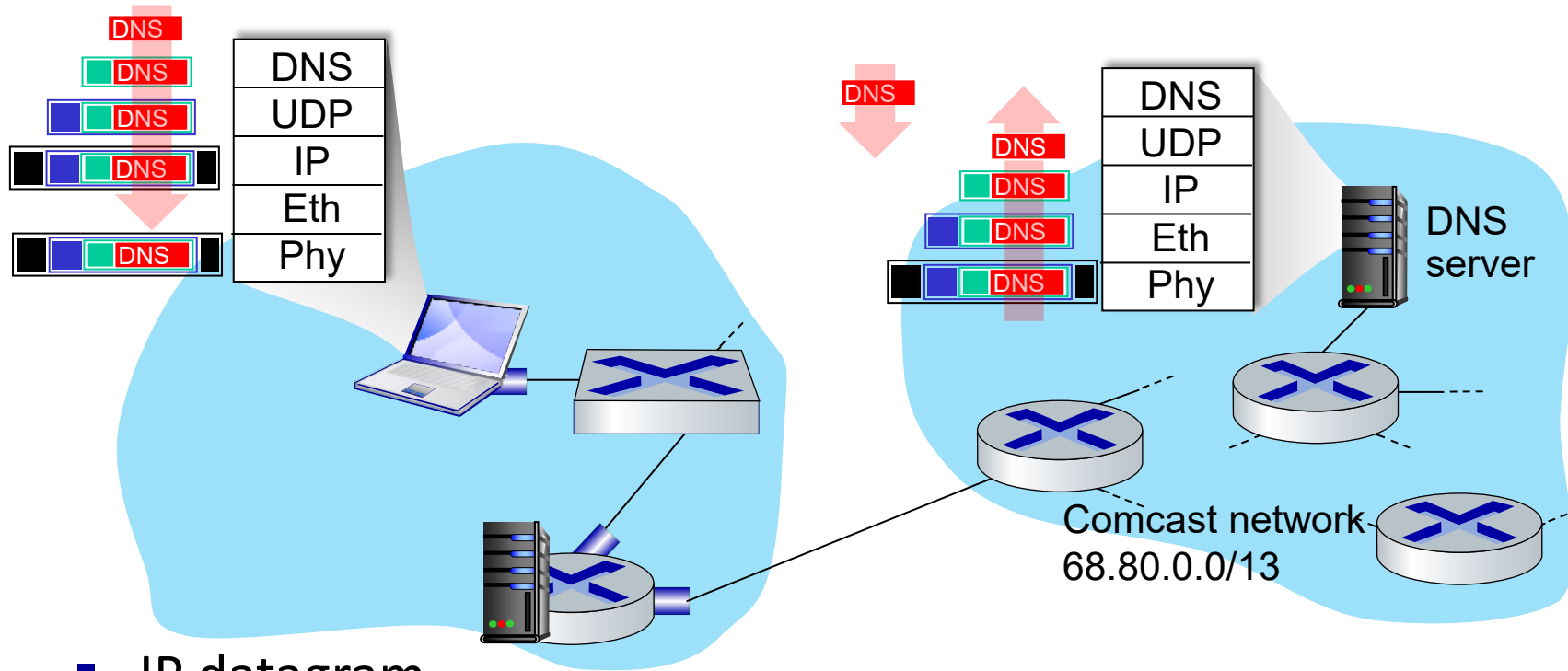
Client now has IP address, knows name & addr of DNS server, IP address of its first-hop router

A day in the life... ARP (before DNS, before HTTP)



- before sending **HTTP** request, need IP address of `www.google.com`: **DNS**
- DNS query created, encapsulated in UDP, encapsulated in IP, encapsulated in Eth. To send frame to router, need MAC address of router interface: **ARP**
- **ARP query** broadcast, received by router, which replies with **ARP reply** giving MAC address of router interface
- client now knows MAC address of first hop router, so can now send frame containing DNS query

A day in the life... using DNS

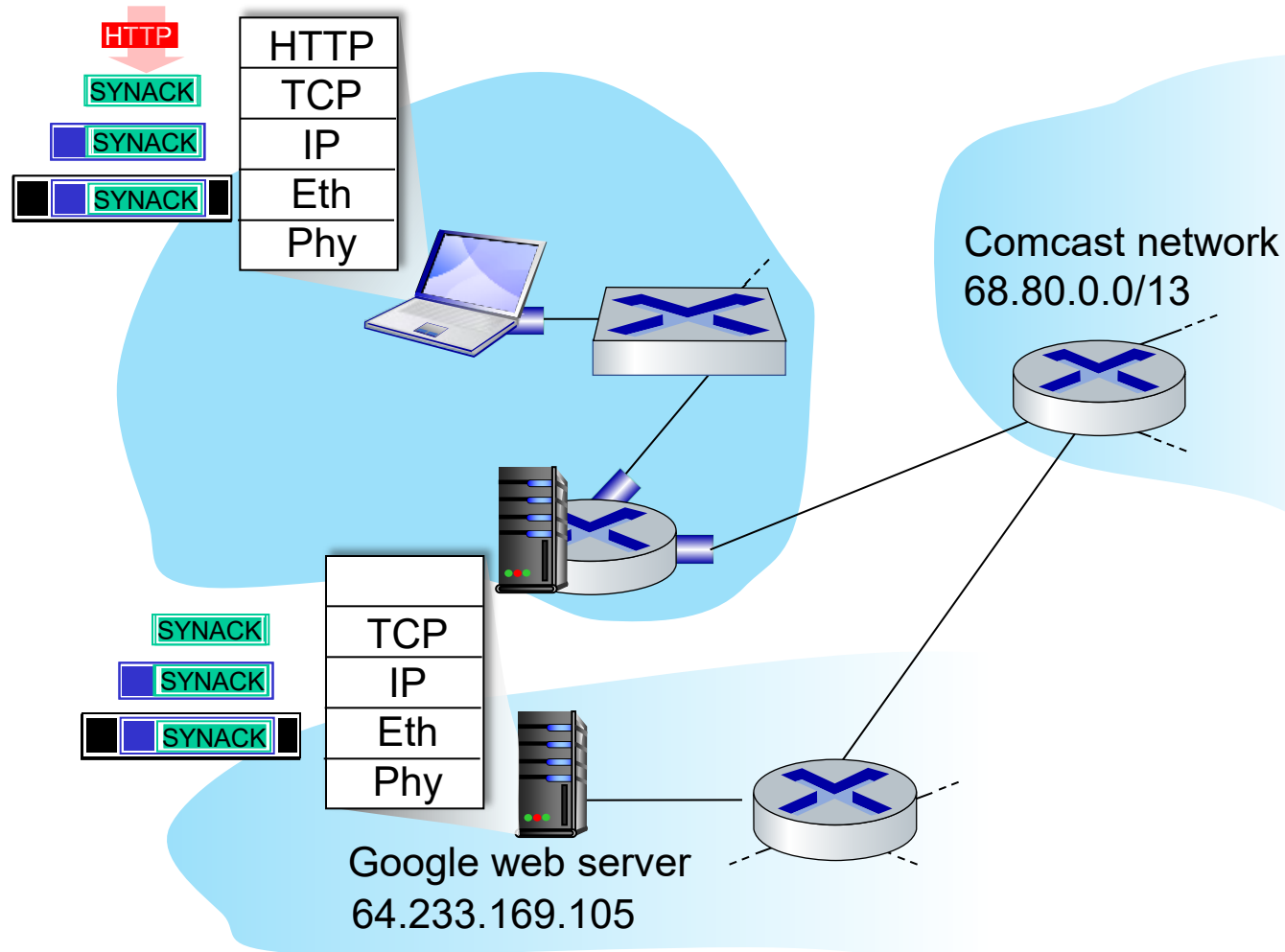


- IP datagram containing DNS query forwarded via LAN switch from client to 1st hop router

- IP datagram forwarded from campus network into Comcast network, routed (tables created by **RIP**, **OSPF**, **IS-IS** and/or **BGP** routing protocols) to DNS server

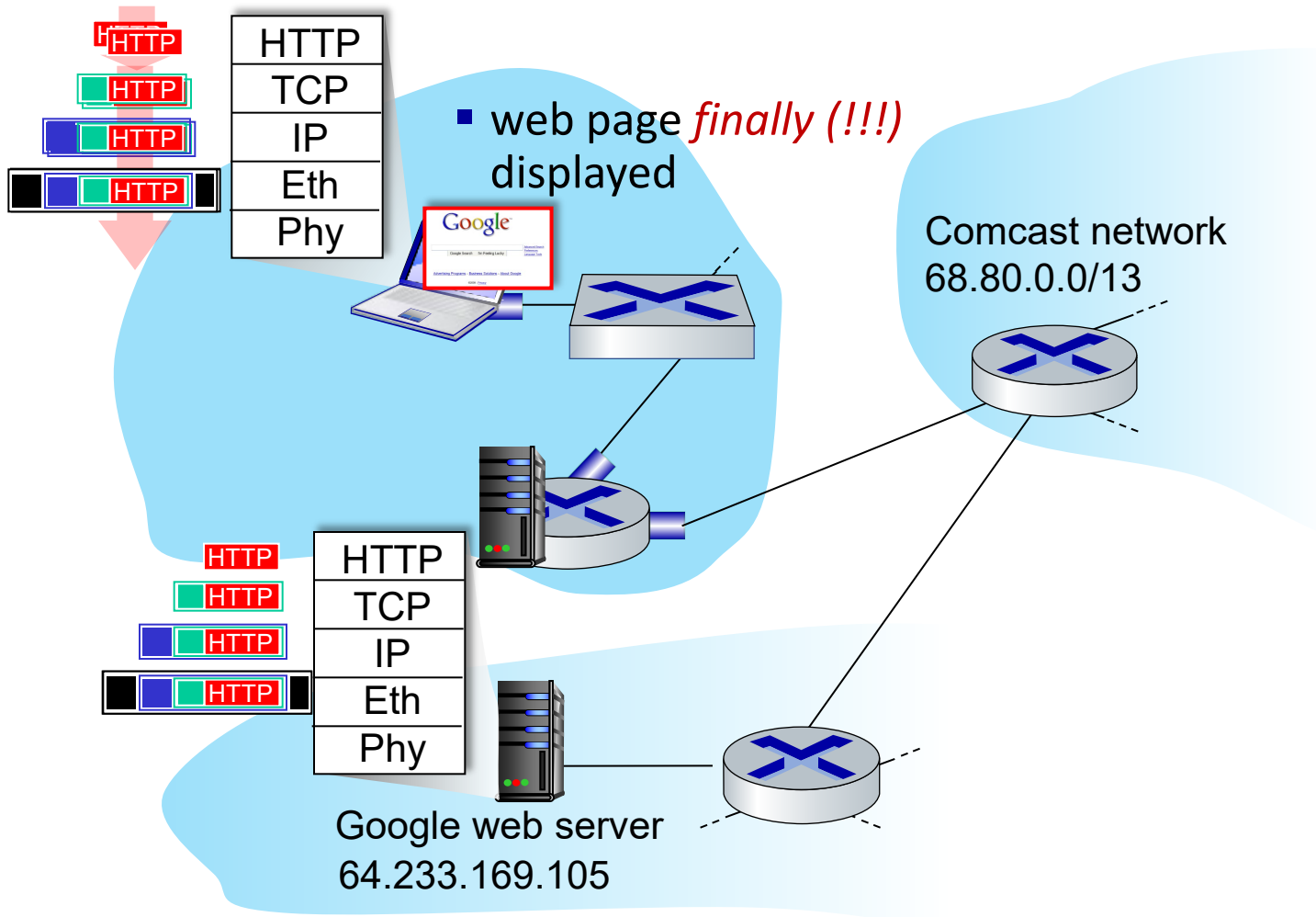
- de-muxed to DNS
- DNS replies to client with IP address of www.google.com

A day in the life...TCP connection carrying HTTP



- to send HTTP request, client first opens **TCP socket** to web server
- TCP **SYN segment** (step 1 in TCP 3-way handshake) inter-domain routed to web server
- web server responds with **TCP SYNACK** (step 2 in TCP 3-way handshake)
- TCP **connection established!**

A day in the life... HTTP request/reply



- **HTTP request** sent into TCP socket
- IP datagram containing HTTP request routed to `www.google.com`
- web server responds with **HTTP reply** (containing web page)
- IP datagram containing HTTP reply routed back to client

Chapter 6: Summary

- principles behind data link layer services:
 - error detection, correction
 - sharing a broadcast channel: multiple access
 - link layer addressing
- instantiation, implementation of various link layer technologies
 - Ethernet
 - switched LANS, VLANs
 - virtualized networks as a link layer: MPLS
- synthesis: a day in the life of a web request

Chapter 6: let's take a breath

- journey down protocol stack *complete* (except PHY)
- solid understanding of networking principles, practice!
- could stop here but *more* interesting topics!
 - wireless
 - security