Networking Refresher

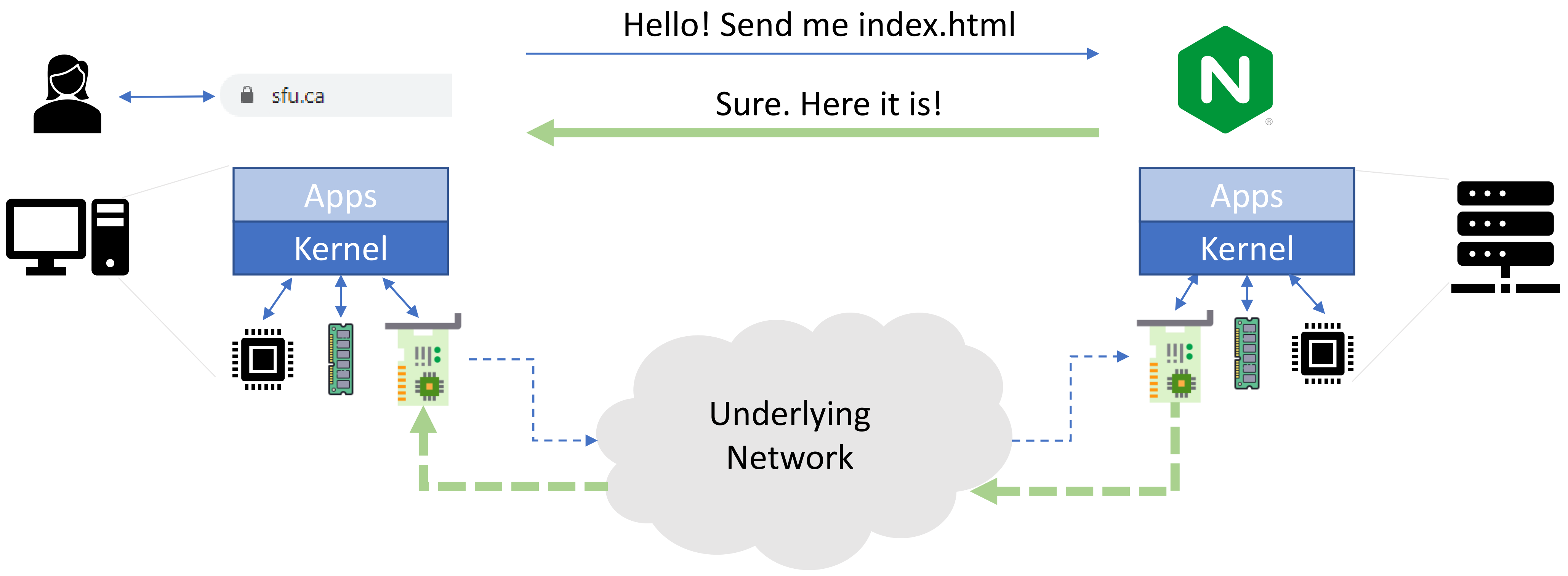
Outline

- Network Architecture
  - Components
  - Functionalities
  - Packet switching
- Network Layers
- Basics of Routing

Network Architecture

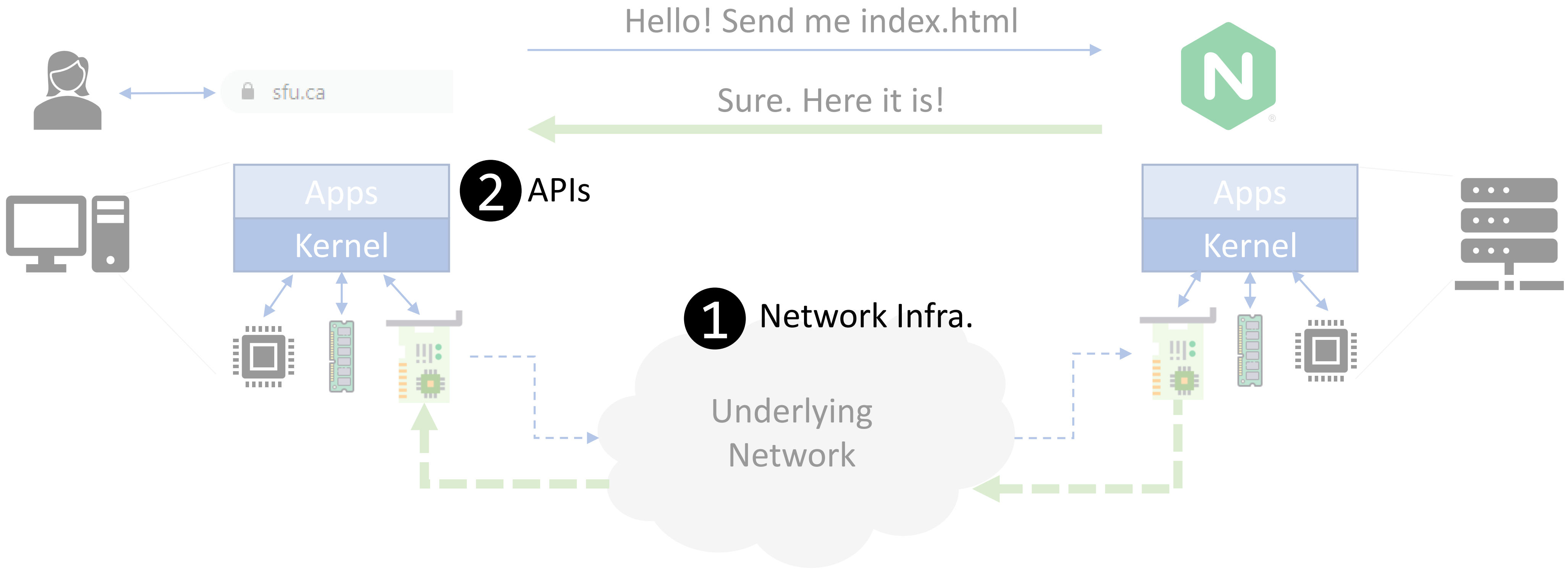
Main Goal

Remote processes communicating with each other.



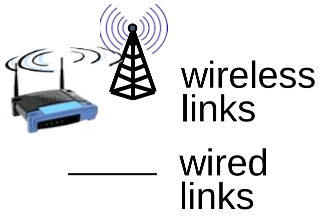
Main Goal: Two Requirements

Remote processes communicating with each other.

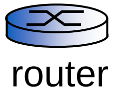


What is the Internet? A Component View

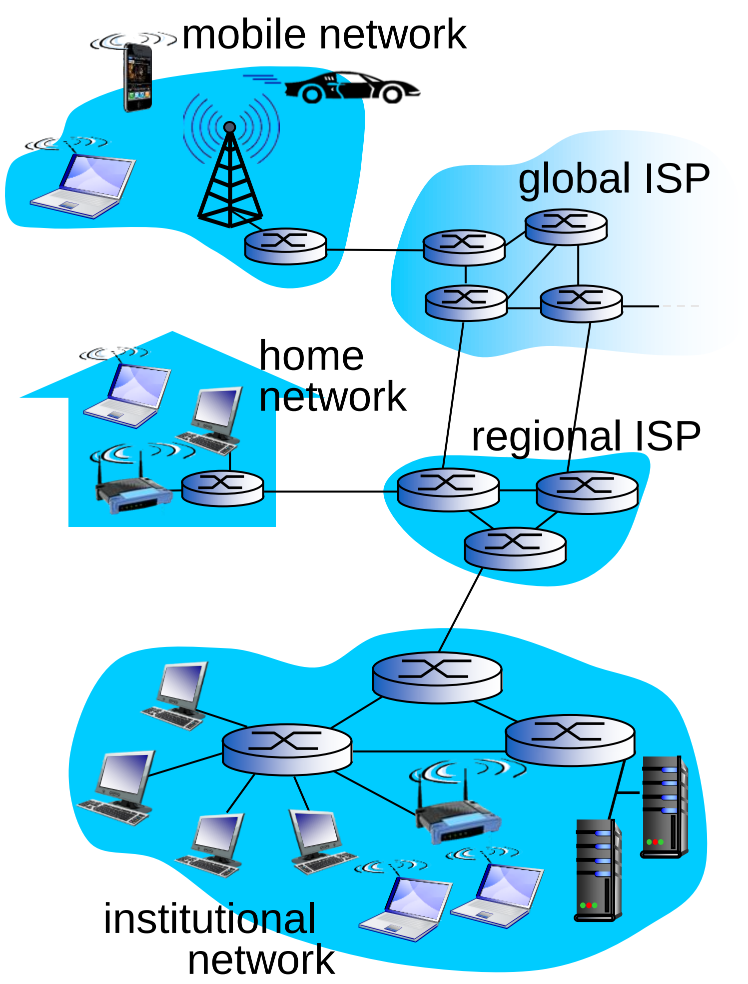
- **Millions of connected computing devices**:
  - *hosts* = *end systems*
  - running *network apps*



- **Communication links**
  - fiber, copper, radio, satellite
  - transmission rate: *bandwidth*

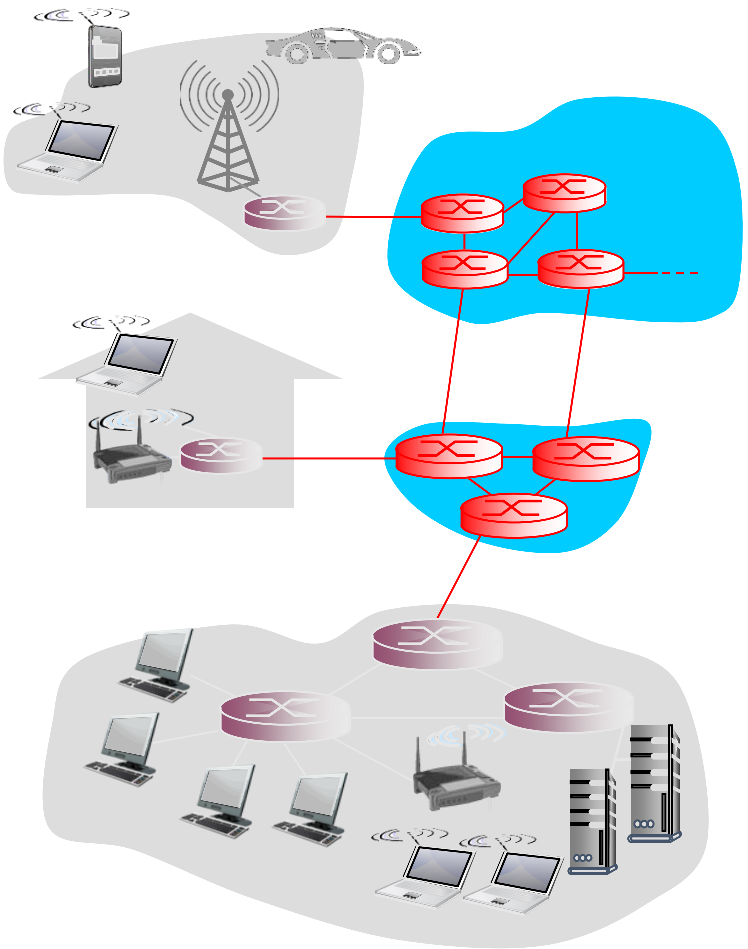


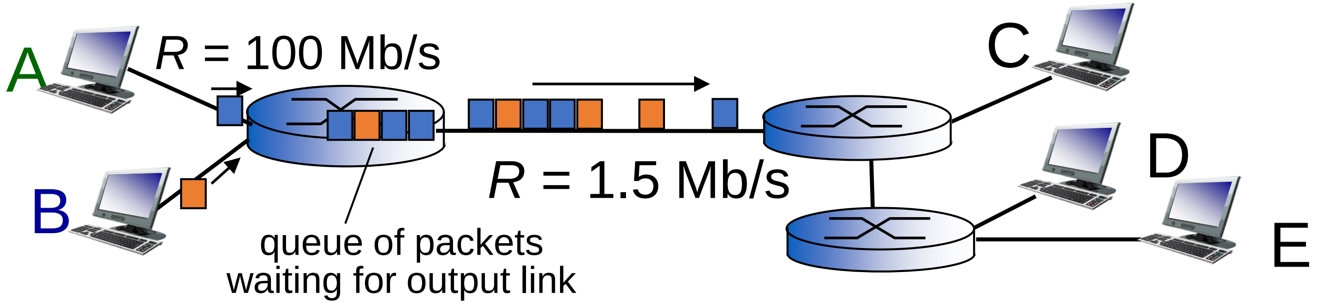
- **Packet switches:** forward packets (chunks of data)
  - *routers* and *switches*



The Network Core

- Mesh of interconnected routers
- Packet switching: Lines are not reserved by connections
- *Store-and-forward*: Routers only forward packets when whole packet is received
- What happens if a router receives/stores too much data?
  - Initially, packets are delayed (buffered)
  - Eventually, packets must be dropped





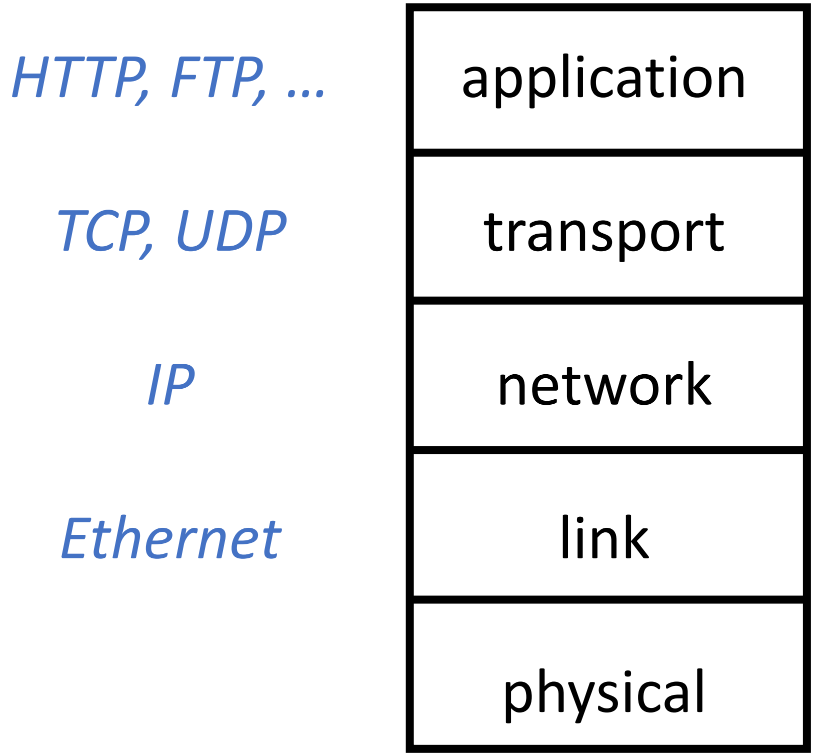
Internet Structure: Network of networks!

- End systems connect to Internet via access ISPs (Internet Service Providers)
  - Residential, company and university ISPs
- Access ISPs in turn must be interconnected
  - So that any two hosts can send packets to each other
- Resulting network of networks is complex
  - Evolution was driven by economics and national policies
- CDNs can use their own networks outside of the system

Network Layers

Protocol Layers

- Every packet has a series of headers, one for each layer
- Headers are read by intermediate devices for routing/filtering decisions

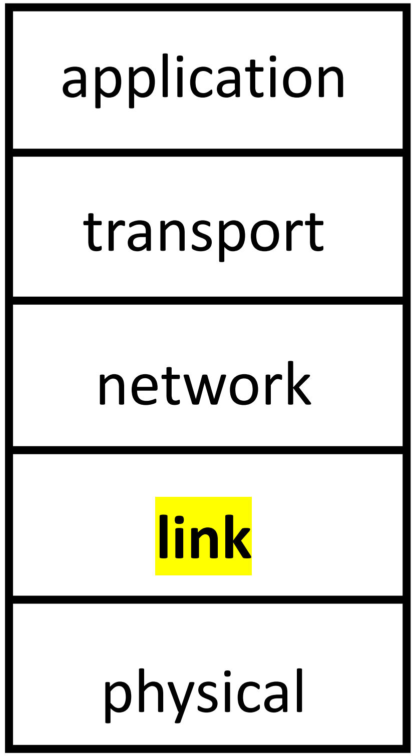


Data Link Layer

- Two devices that are not directly connected want to talk to each other
- The devices are identified by MAC addresses
- Instead, they are connected by *switches*
- Switches know MAC addresses and will forward packets to the right devices through the right port

Problems:

- Scaling: Switches can’t know every MAC address
- MAC addresses do not convey logical information about the network hierarchy

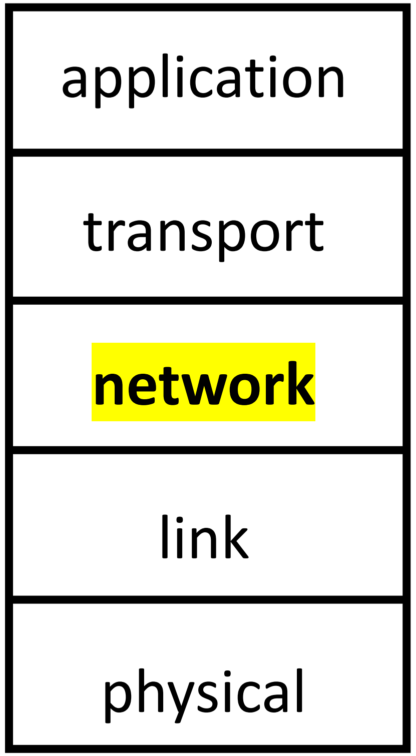


Network Layer (This lecture)

- IP addresses instead of MAC addresses for the wider Internet
  - Given destination IP, any router should be able to forward the packet towards the destination without knowing the whole path
- CIDR rules give logic to IP addresses to minimize routing table size
- Interior/exterior gateway protocols to route messages (more later)

Problem:

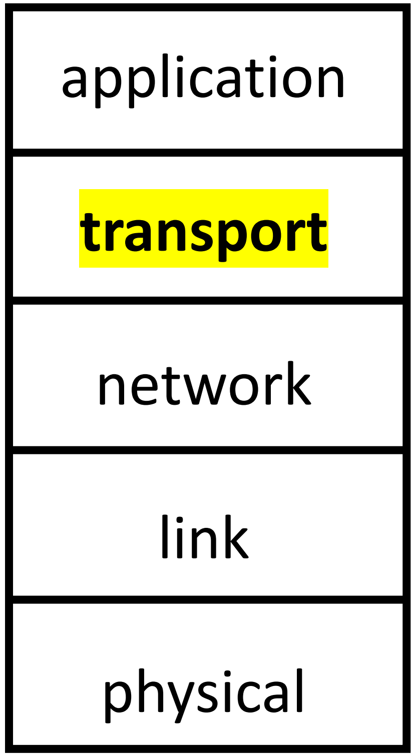
- No guarantee that packets arrive, no guarantee of order
- Congestion is an issue
- No distinction between different services on one end device
- No concept of “connections”



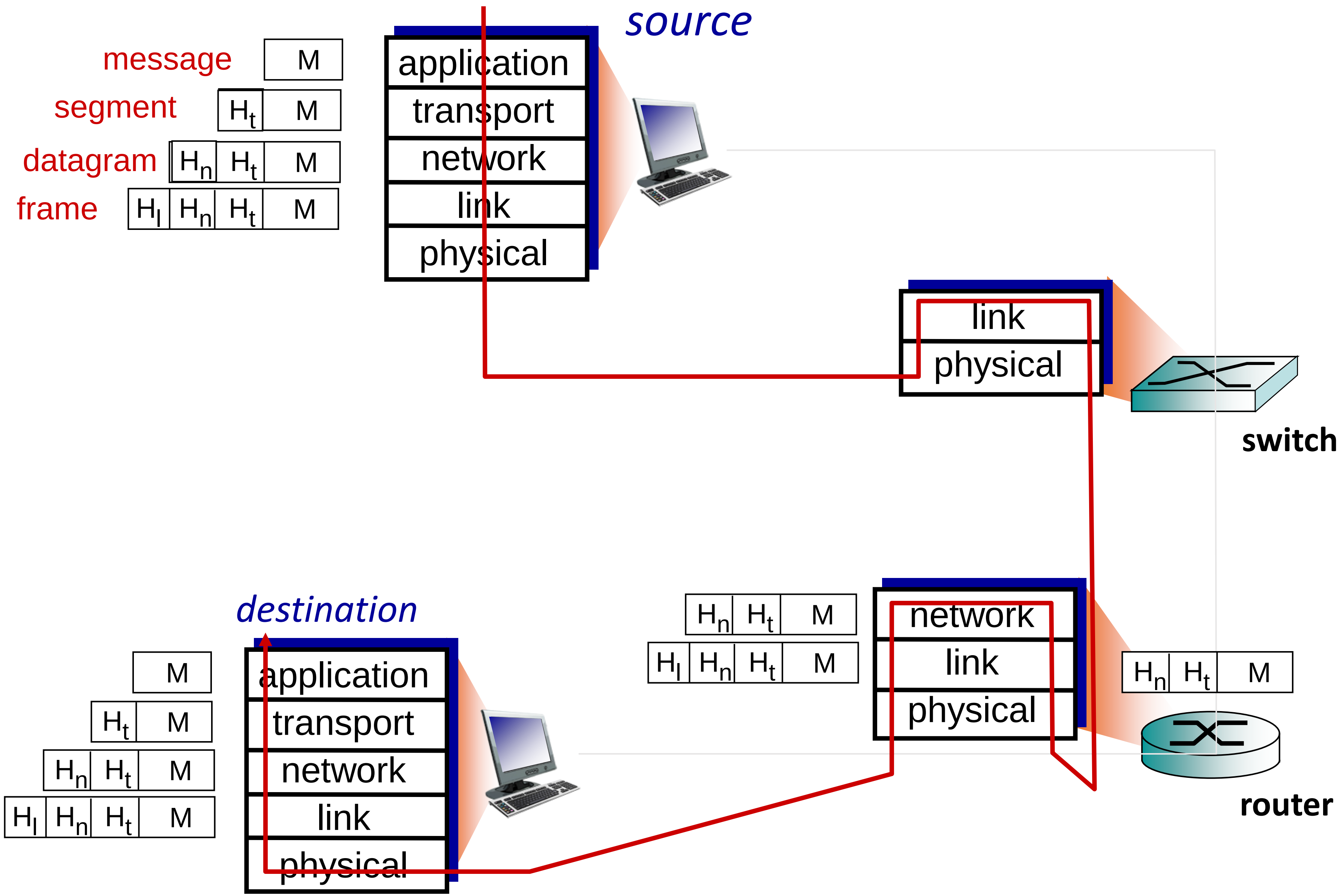
Transport Layer

- TCP, UDP, QUIC
- Enables (some of) the following features:
  - Port number to distinguish between applications, or multiple connections for the same application
  - Connection establishment
  - Flow control
  - Congestion control
  - Correct order of packets
  - Guarantee delivery of packets

Not in UDP (applies to: Connection establishment, Flow control, Congestion control, Correct order of packets, Guarantee delivery of packets)

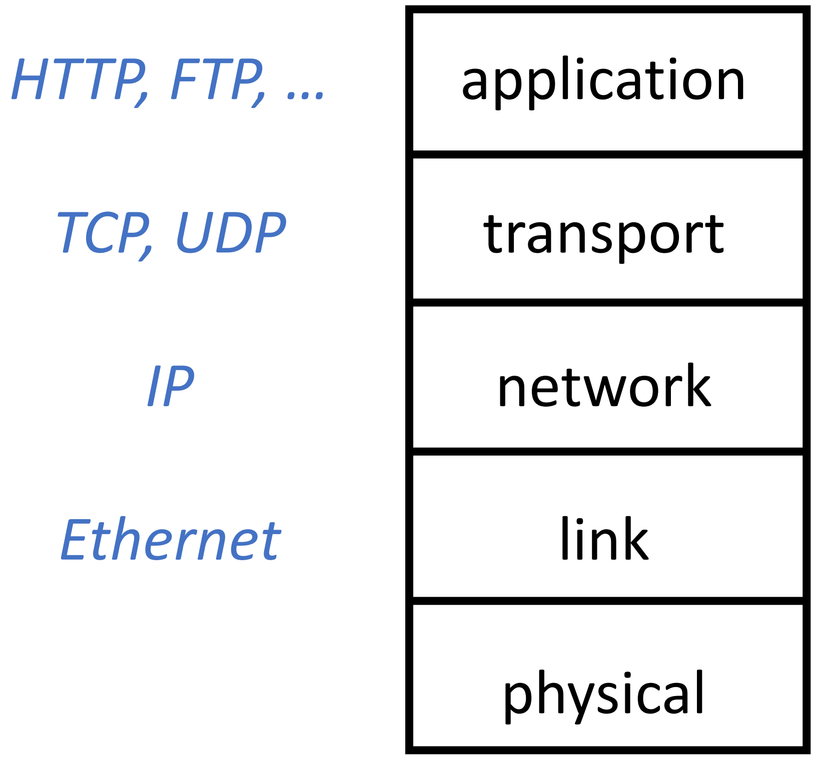


Encapsulation



TCP/IP Protocol Suite Summary

- *application:* supporting network applications
  - FTP, SMTP, HTTP
- *transport:* process-to-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"



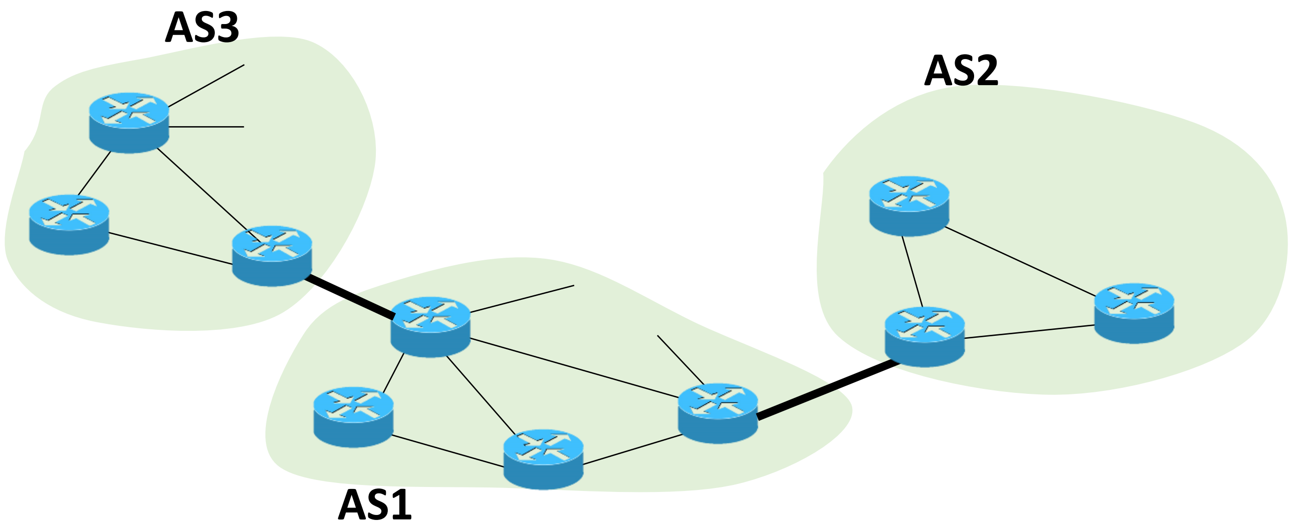
Basics of Routing

Internet Routing

- “Flat” routing not suited for the Internet
  - Scalability (as the network size increases)
    - Space complexity → Each node cannot be expected to store routes to every destination (or destination network)
    - Convergence times increase
    - Communication → Total message count increases
  - Administrative autonomy
    - Each internetwork may want to run its network independently
      - E.g., hide topology information from competitors
- Solution: Hierarchy via autonomous systems (AS’s)

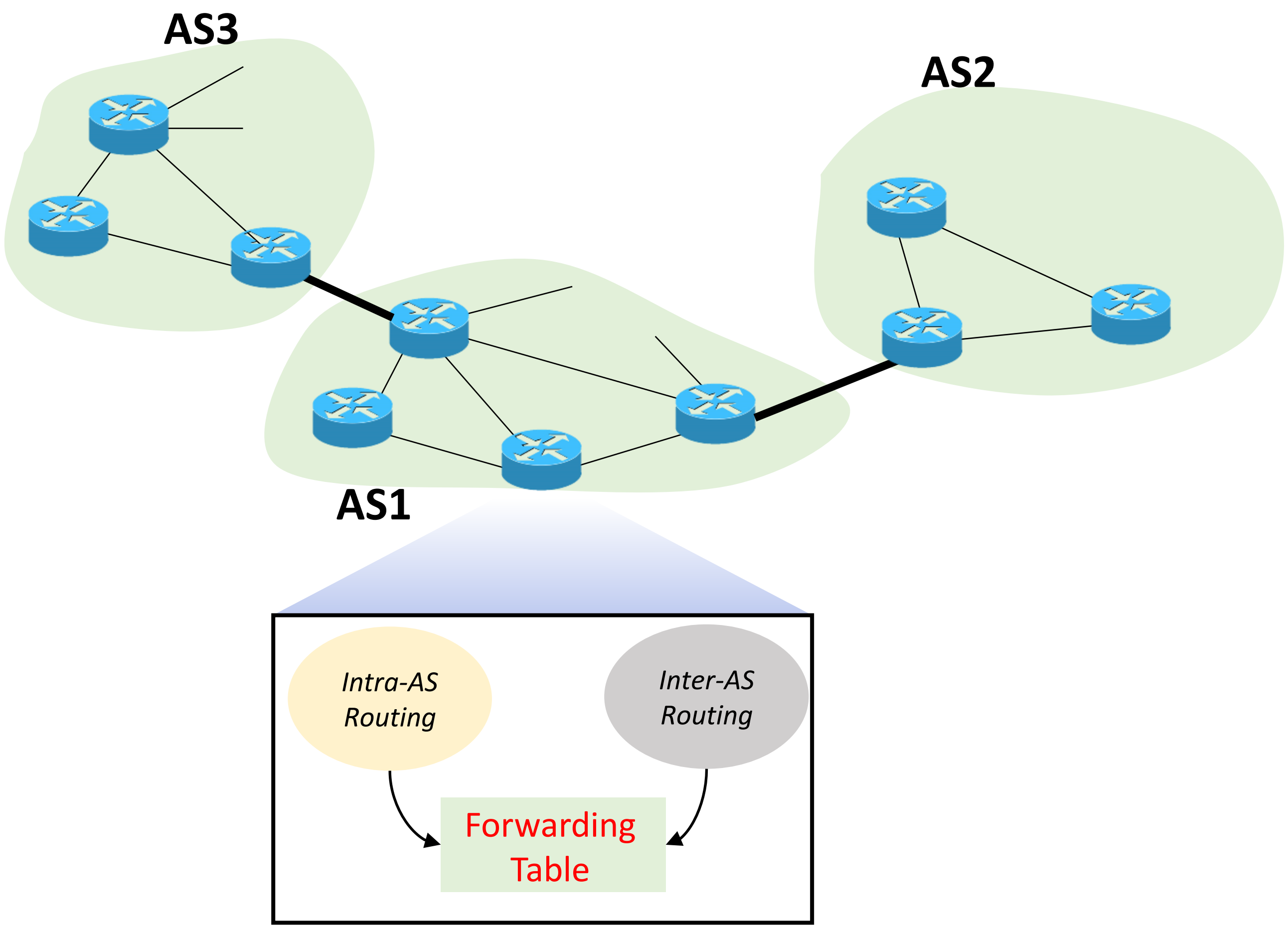
Today's Internet

- Uses hierarchy of AS's
- Each AS:
  - A set of routers under a single technical administration
  - **Intra-domain Routing:** Use an *interior gateway protocol (IGP)* and common metrics to route packets within the AS
  - **Inter-domain Routing:** Use an *exterior gateway protocol (EGP)* to route packets to other AS's



- IGP: OSPF, RIP
- EGP: BGP

Interconnected AS's



- Forwarding table is populated by IGPs and EGPs
  - **Interior gateway protocols (IGPs)** determine entries for destinations within AS
  - **IGPs and exterior gateway protocols (EGPs)** determine entries for external destinations

Interior Gateway Protocol: OSPF

- Link-state algorithm

- Router floods OSPF link-state advertisements to all other routers in entire AS
  - carried in OSPF messages directly over IP
  - includes neighbors, and bandwidth information (link cost)

- Each node independently computes a topology map
  - route computation using Dijkstra’s algorithm

Exterior Gateway Protocol

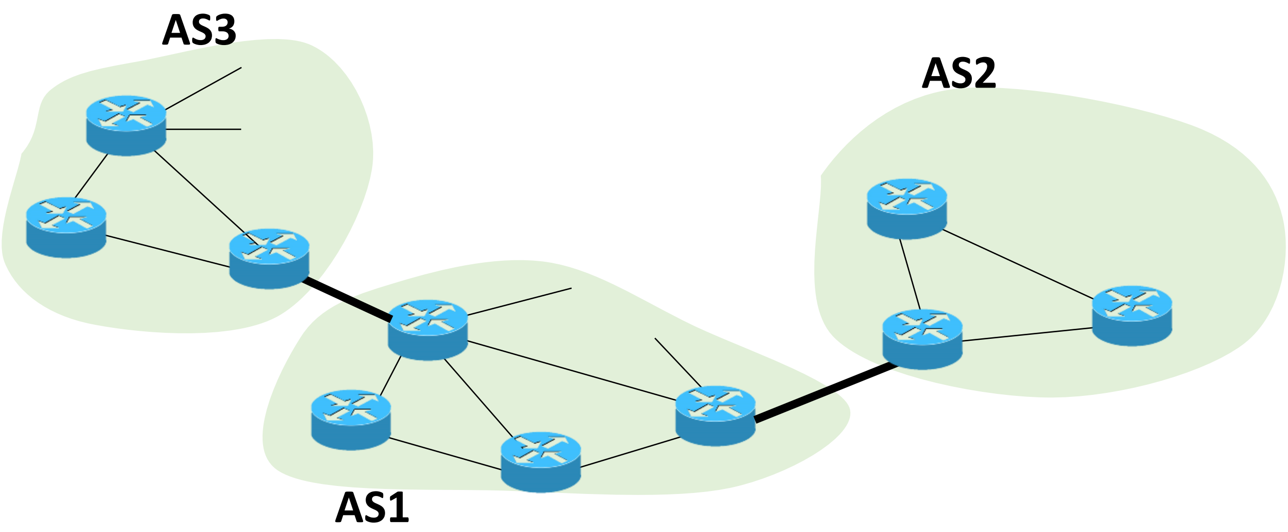
Suppose router in AS1 receives datagram destined **outside** of AS1:

- router should forward packet to gateway router, but which one?

AS1 must:

1. learn which destinations are reachable through AS2, which through AS3
2. propagate this reachability info to all routers in AS1

Job of EGPs!

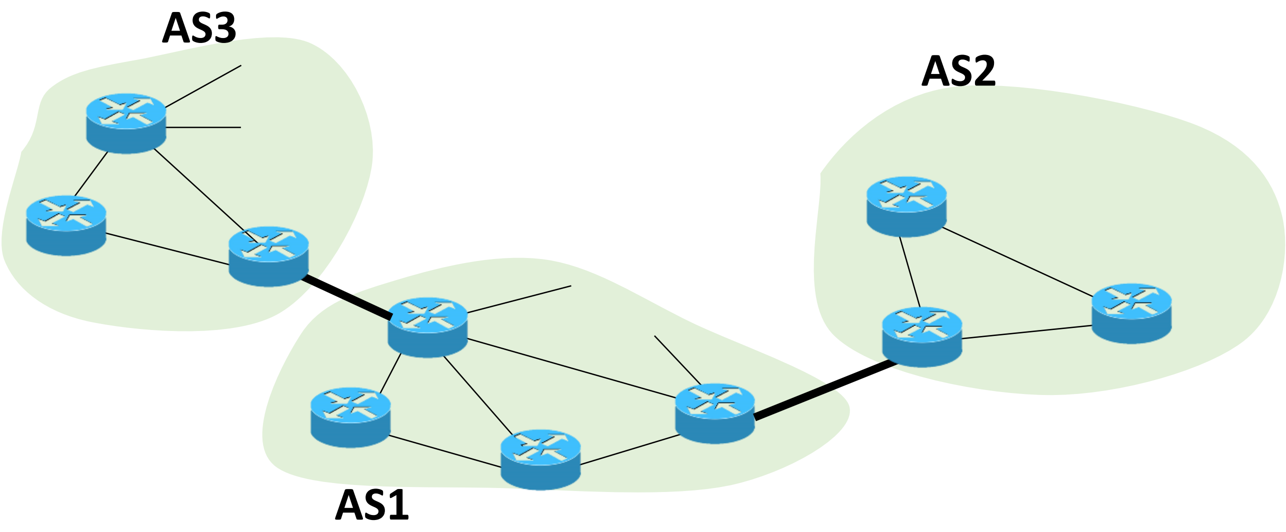


Exterior Gateway Protocol: BGP

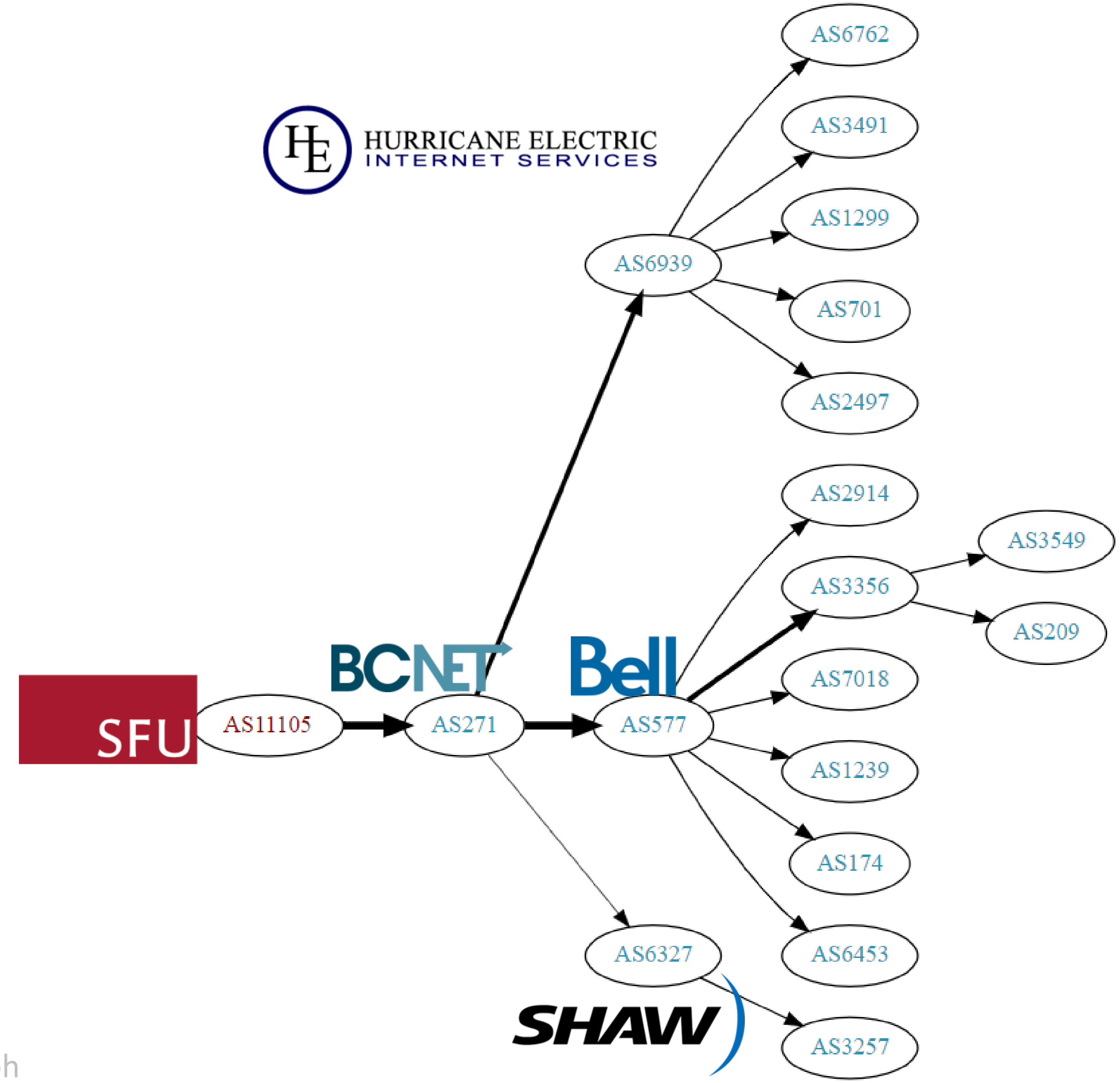
- BGP (Border Gateway Protocol):
  - the de-facto EGP
- Allows a subnet to advertise its existence to rest of Internet

- BGP provides each AS a means to:
  - eBGP: obtain subnet reachability information from neighboring ASes
  - iBGP: propagate reachability information to all AS-internal routers.

- Determines “good” routes to other networks based on reachability information and policy



An Example



https://bgpview.io/asn/11105#graph

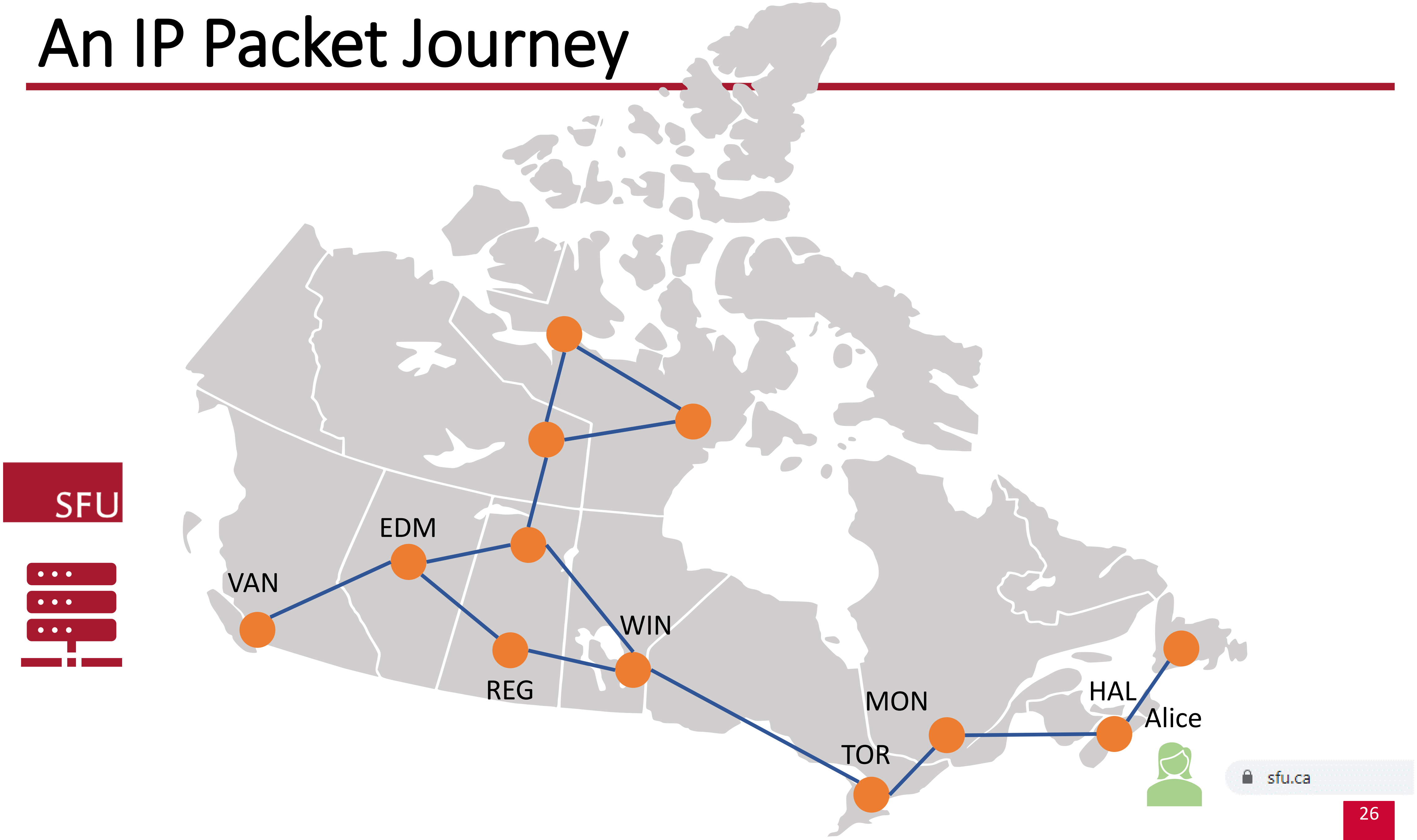
Recall: Security Goals

- Confidentiality: what can routers (and wiretappers) see?
- Integrity: what can MITMs change? What can other end devices spoof?
- Availability: is end device connectivity ensured? Can someone be "knocked off" the Internet?

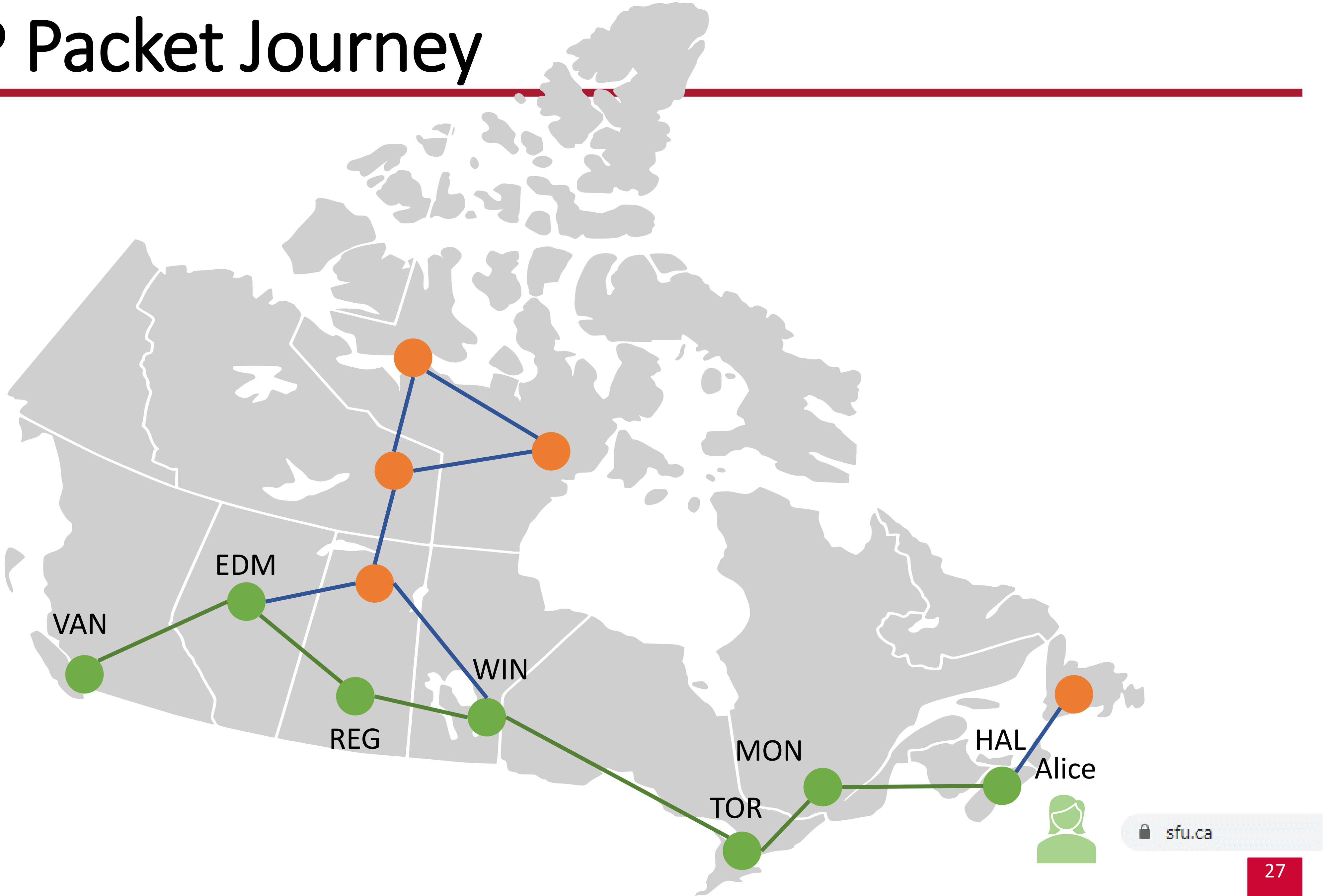
Sources of Network Vulnerabilities

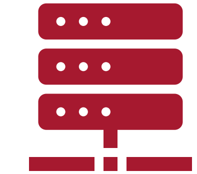
- Protocol-level vulnerabilities
  - Implicit trust assumptions in design
- Implementation vulnerabilities
  - Both on routers and end-hosts
- Incomplete specifications
  - Often left to the programmers

An IP Packet Journey

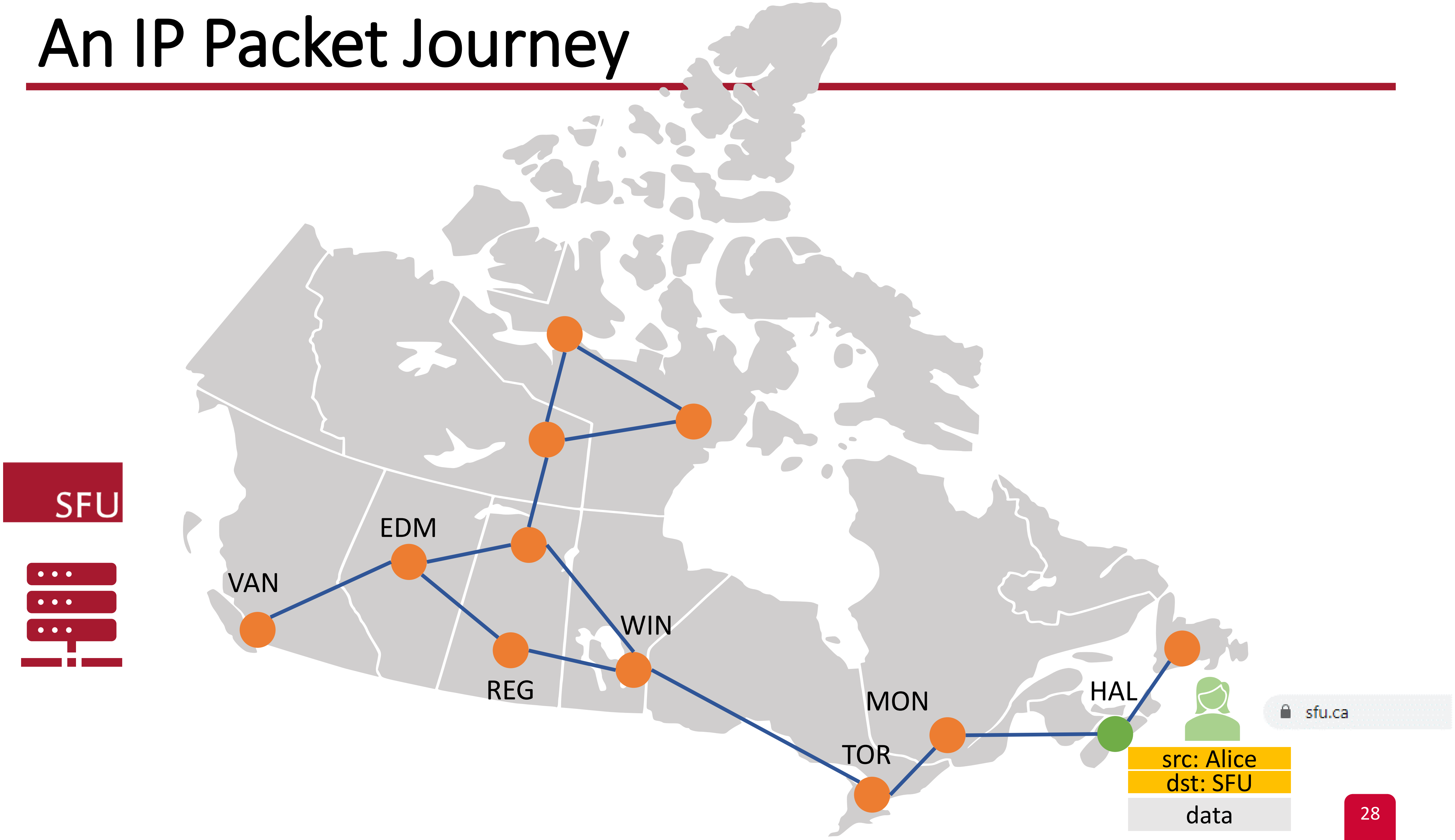


An IP Packet Journey



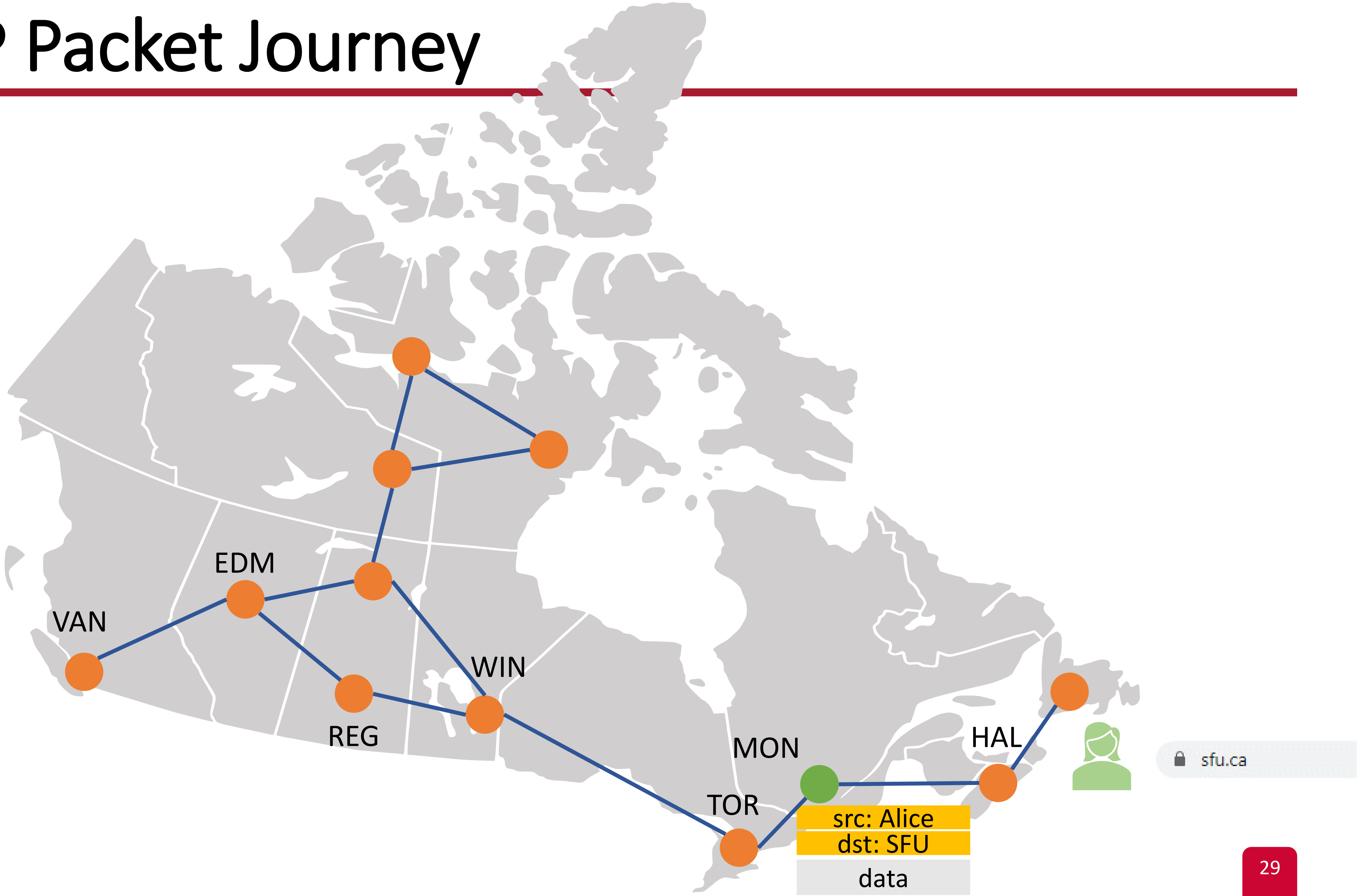


An IP Packet Journey

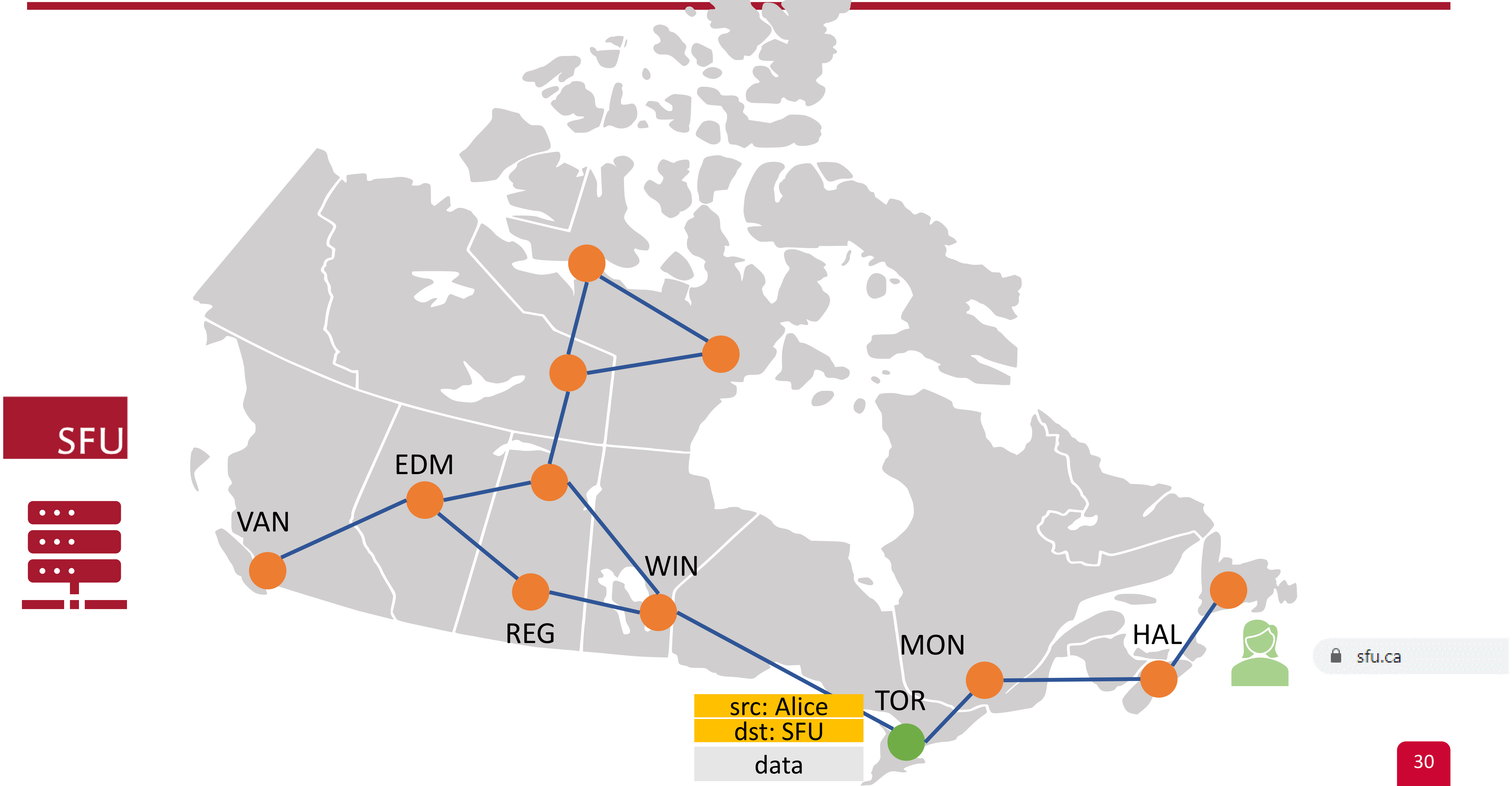


An IP Packet Journey

SFU

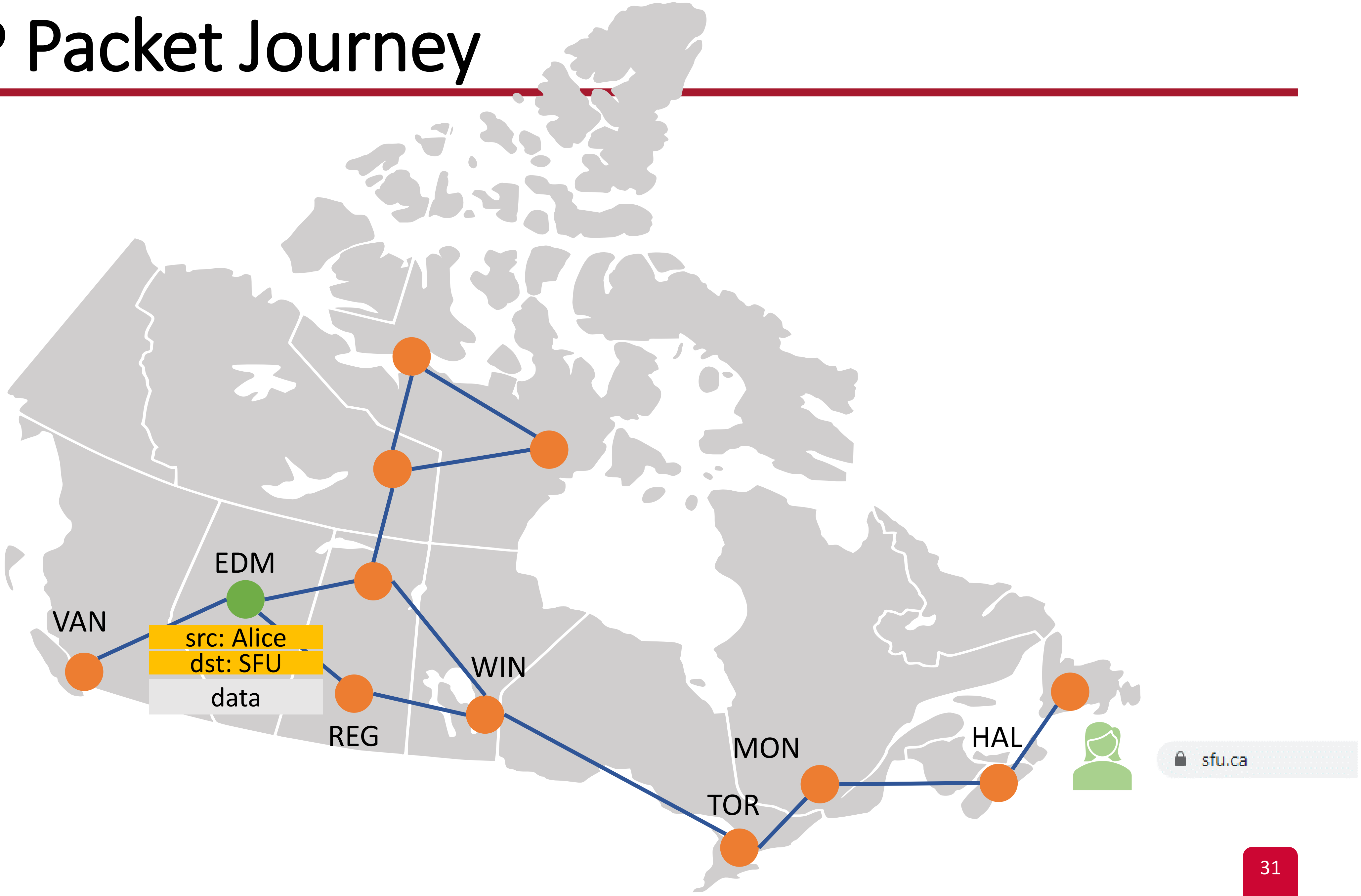


An IP Packet Journey

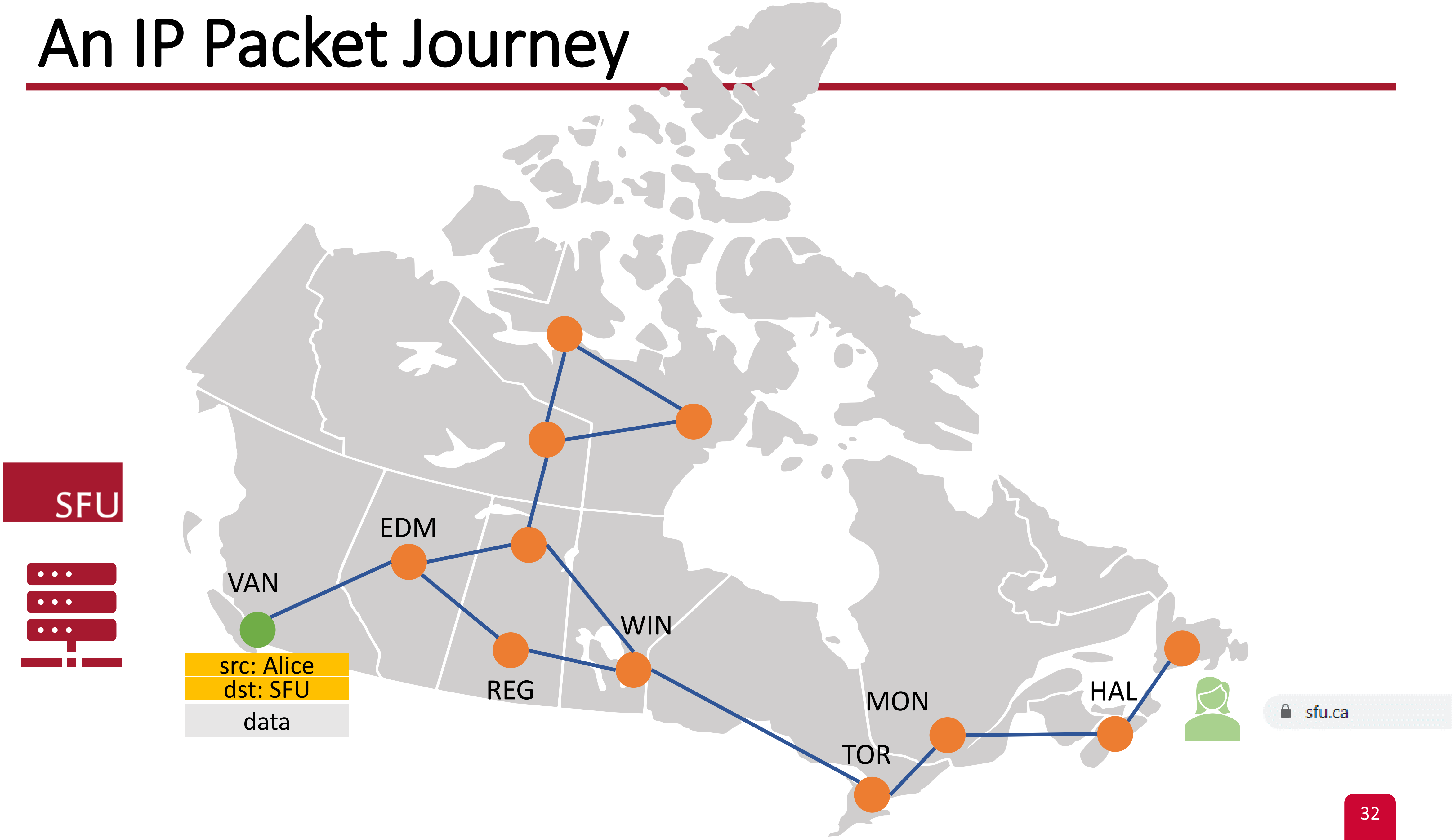


An IP Packet Journey

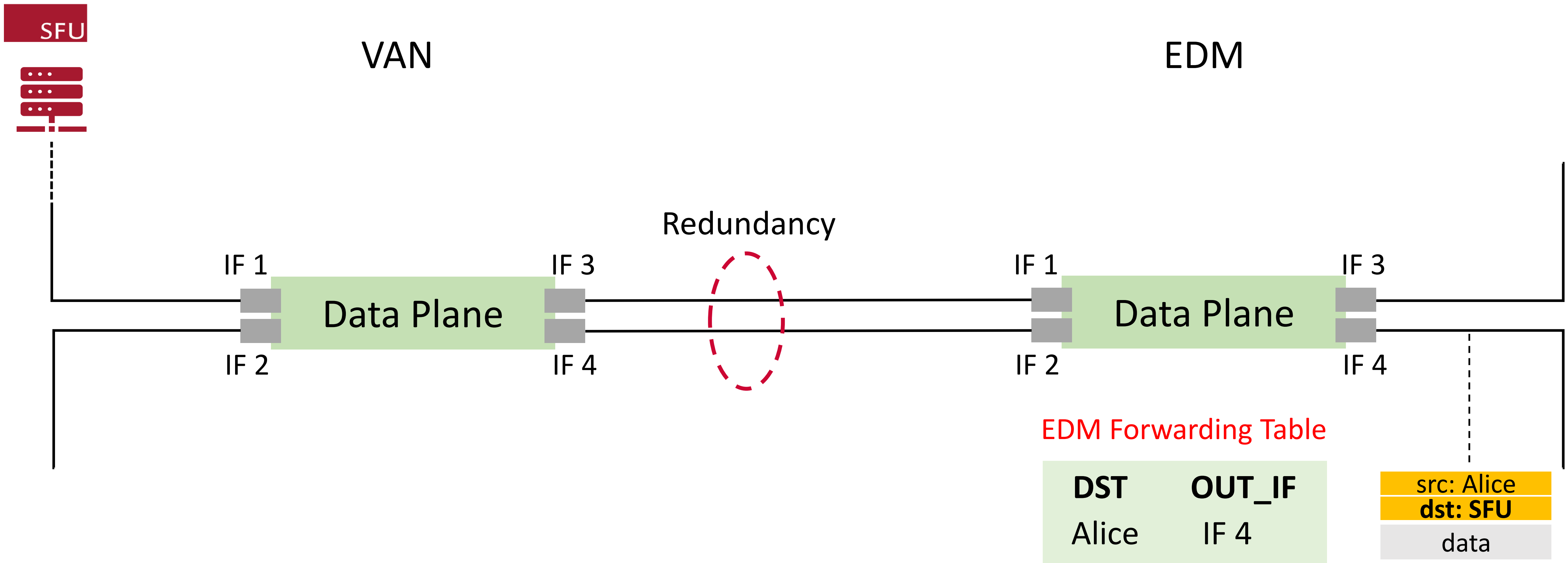
SFU



An IP Packet Journey



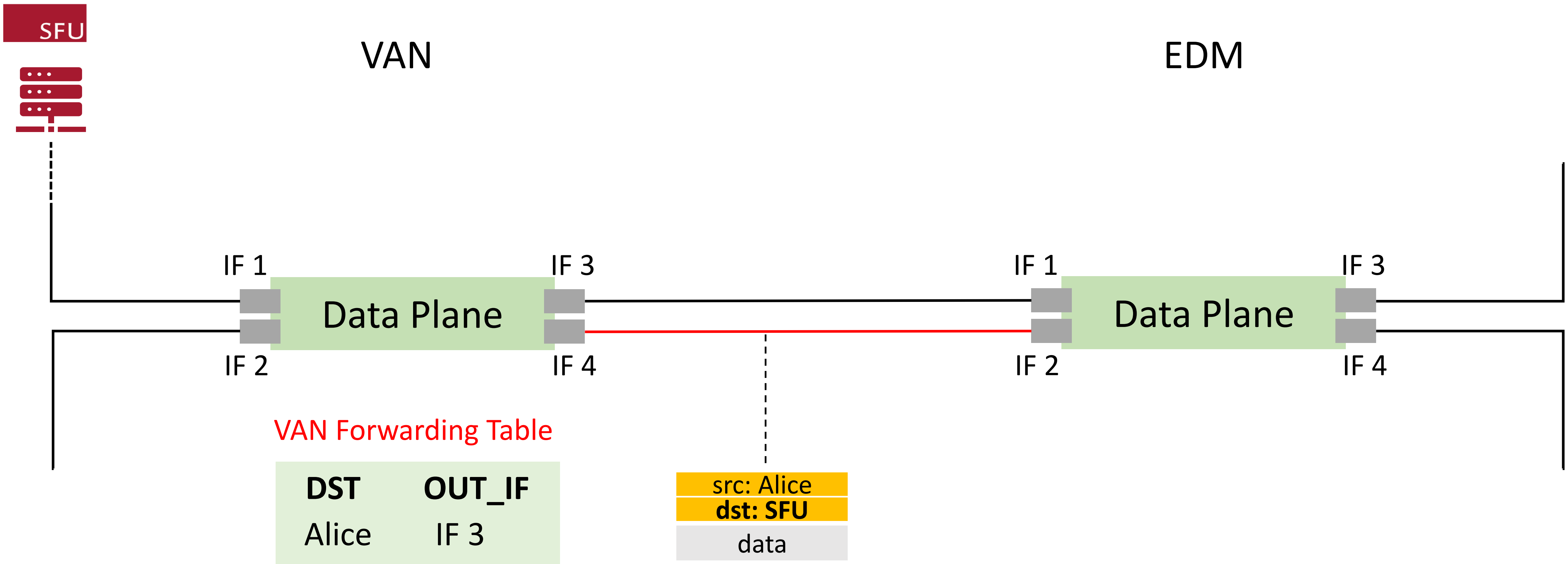
What happens between two routers



EDM Forwarding Table

DST	OUT_IF
Alice	IF 4
SFU	IF 2

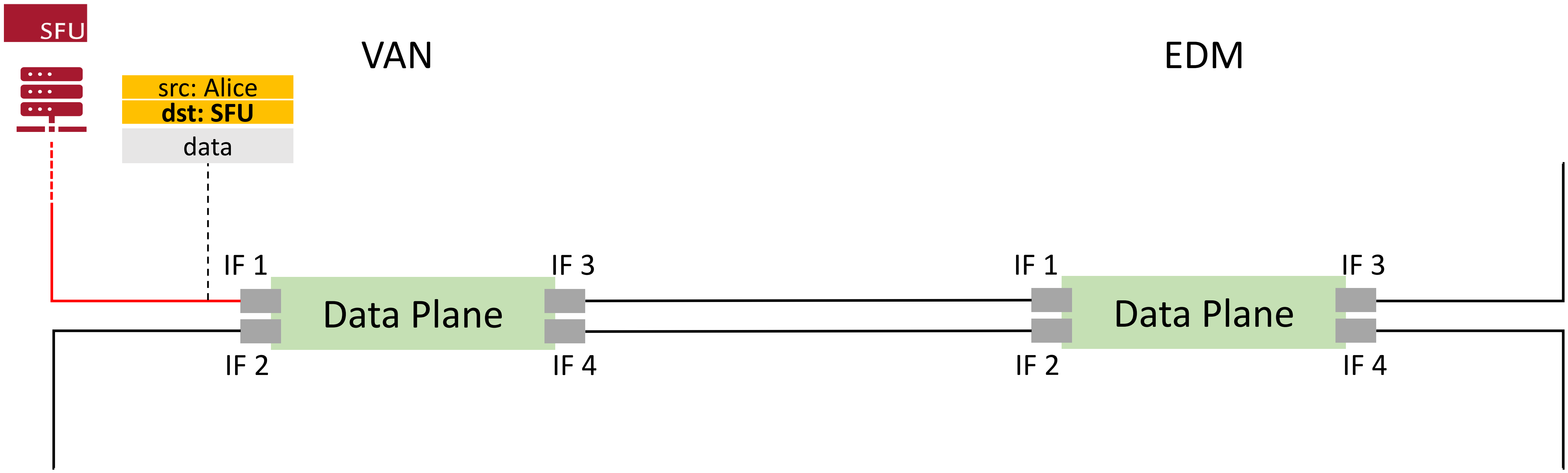
What happens between two routers



VAN Forwarding Table

DST	OUT_IF
Alice	IF 3
SFU	IF 1

What happens between two routers



What happens between two routers

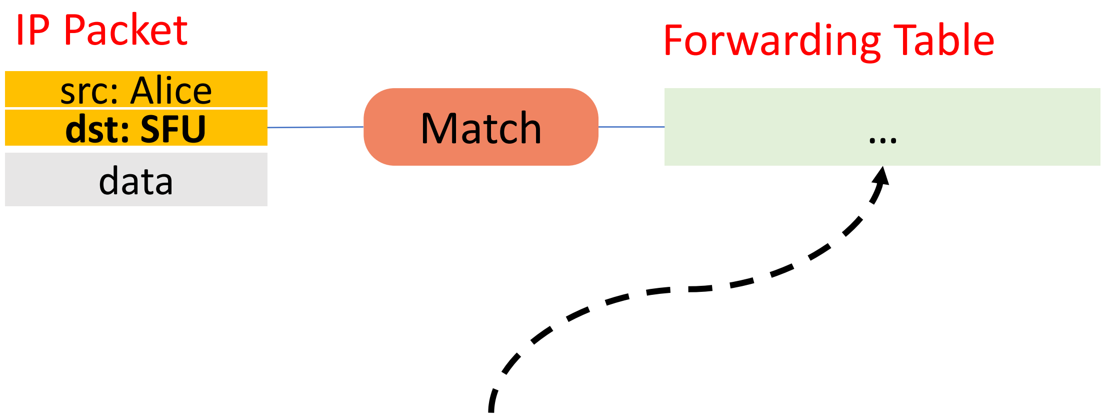
This is called **Packet Forwarding**

- moving packets from router's input to appropriate router output
- done by the data-plane component

Forwarding happens by:

- examining the **destination address**, and
- matching it with a local forwarding table

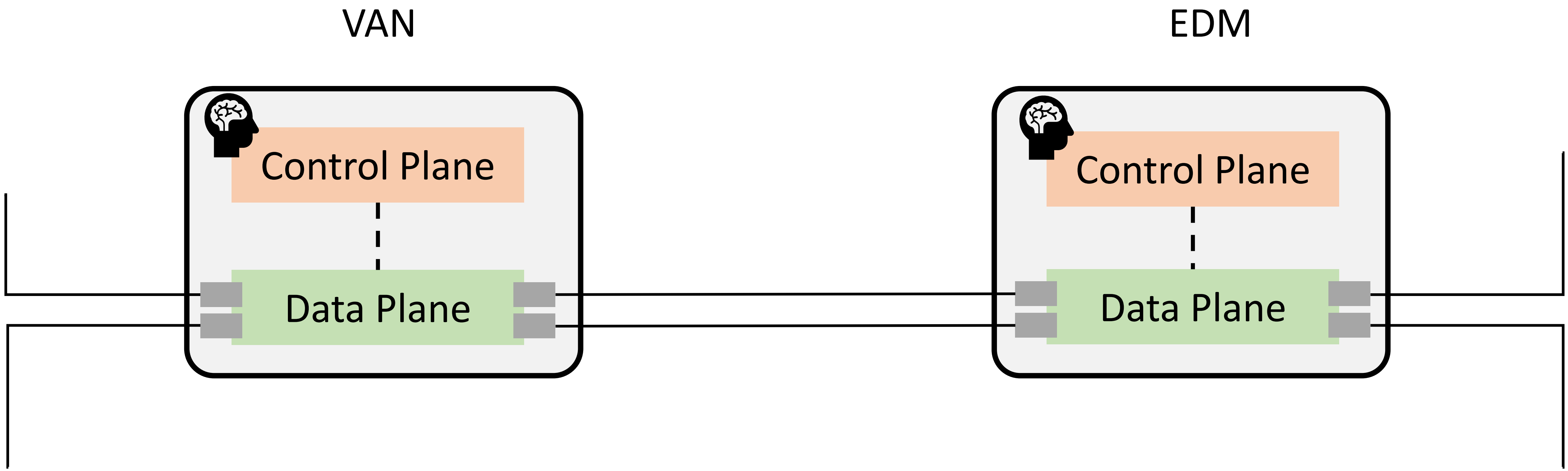
Using other fields?



But, who calculates the forwarding tables?

Routers Have “Brains”

This brain is called the Control Plane



Routers Have “Brains”

The control plane runs a routing algorithm to:

- find routes, and
- fill the tables

VAN

Control Plane

Routing algorithm

VAN Forwarding Table

DST	OUT_IF
Alice	IF 3
SFU	IF 1

EDM

Control Plane

Routing algorithm

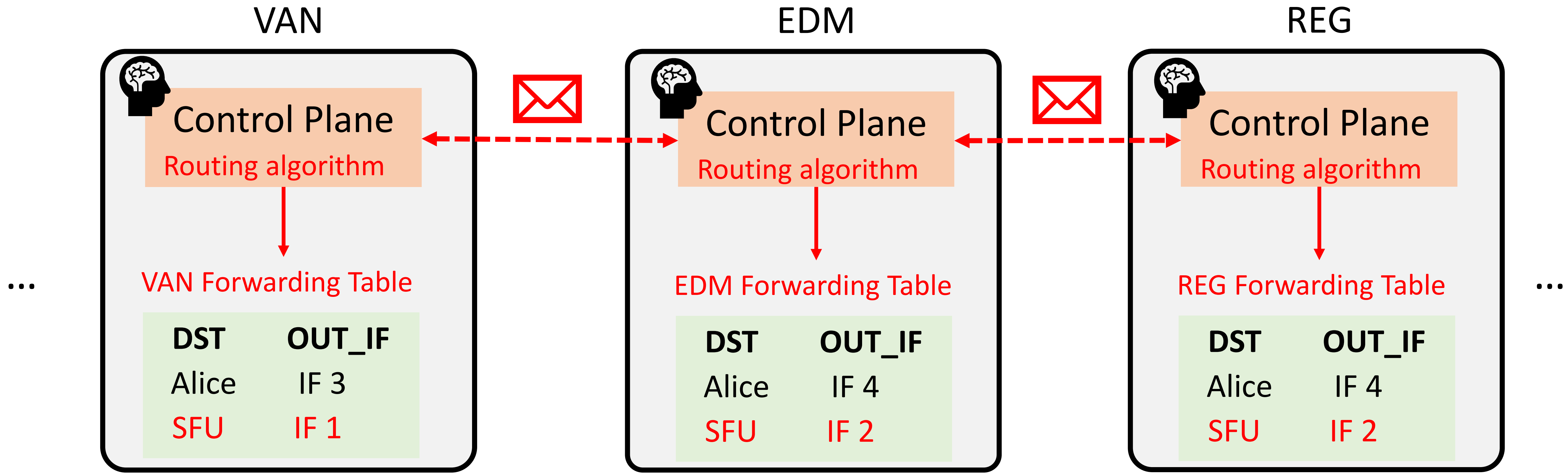
EDM Forwarding Table

DST	OUT_IF
Alice	IF 4
SFU	IF 2

Control Plane: Two Approaches

Distributed Approach: routers exchange messages with each other to calculate the tables

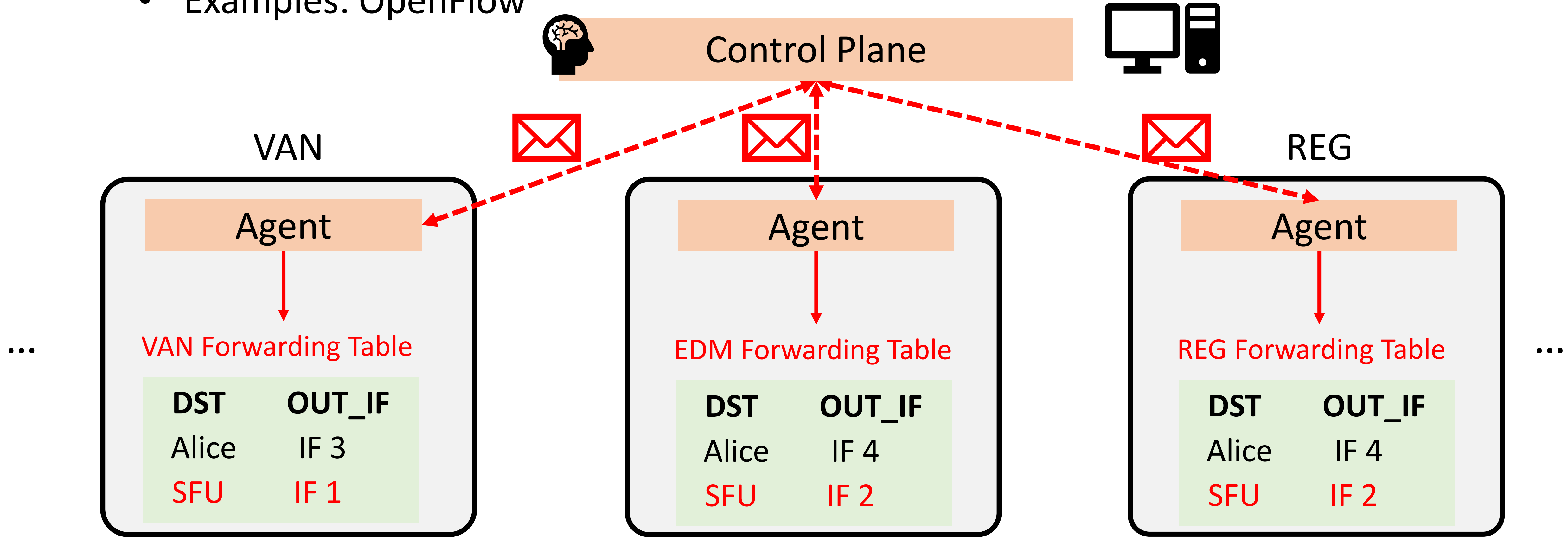
- Examples: OSPF, IS-IS



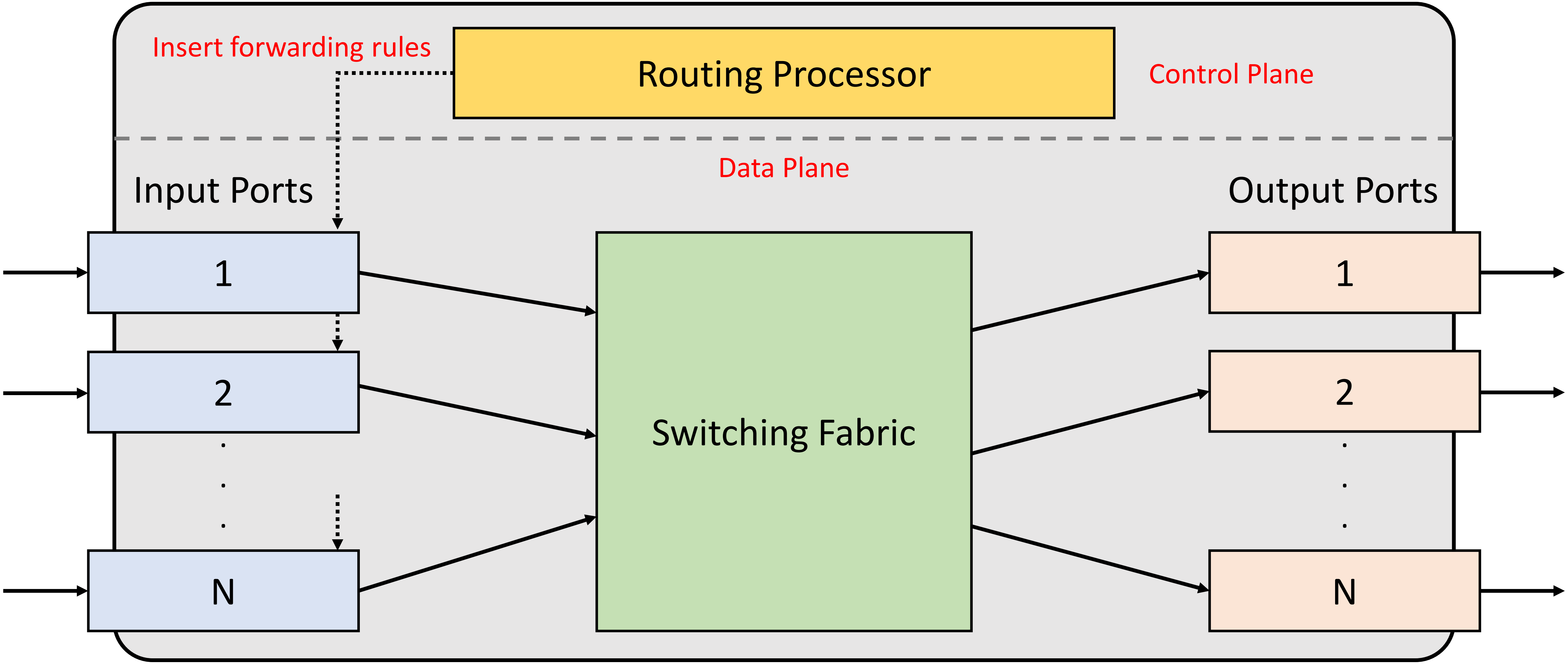
Control Plane: Two Approaches

Centralized Approach: routers exchange messages with an external software

- Software-defined networking (SDN)
- Examples: OpenFlow



Router Architecture Overview

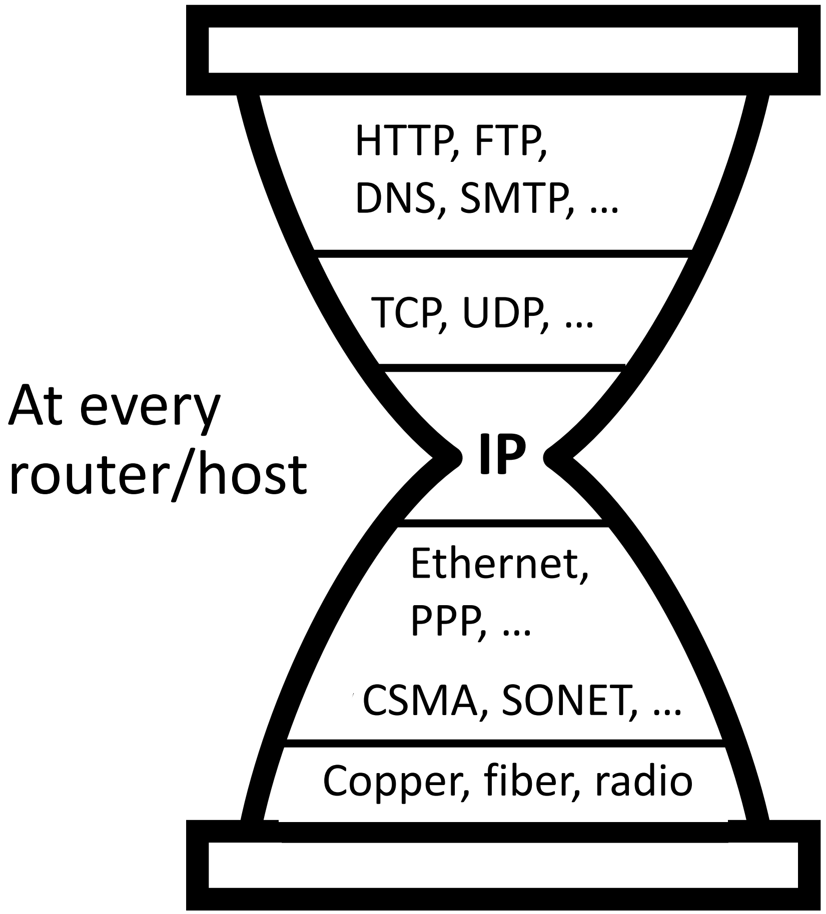


IP Overview

IP is the waist of the “hourglass”

- Multiple higher-layer protocols
  - Transport and Application
- Multiple lower-layer protocols
  - Link and Physical
- Single Internet protocol

  → No need to update routers and hosts every time we have a new service!



IPv4 Datagram Format

Internet Protocol Version 4 (IPv4)							
Offsets	Octet	0		1	2		3
Octet	Bit	0–3	4–7	8–15	16–18	19–23	24–31
0	0	Version	Header Length				
4	32						
8	64						
12	96						
16	128						
20	160						
24+	192+						

Header & data

Fragmentation

Addressing

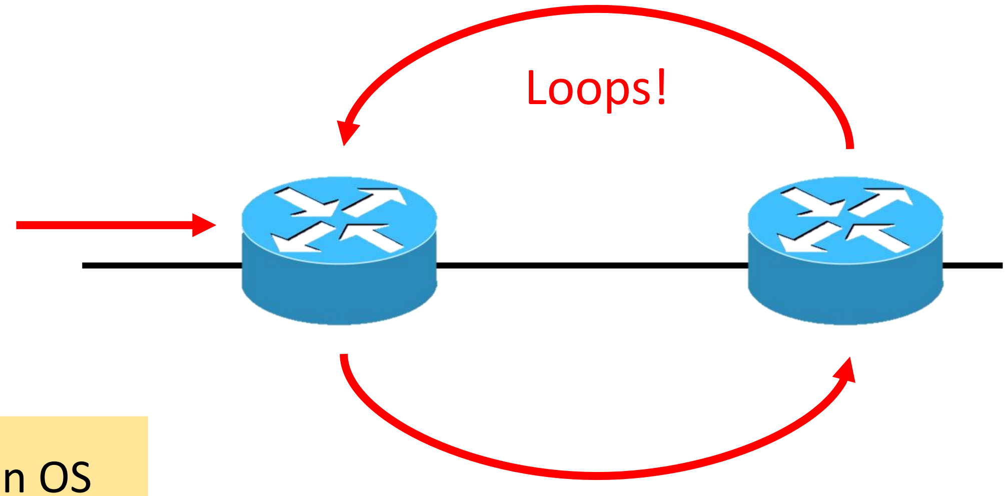
E.g., TCP segment

```
ICMP 0x01
TCP  0x06
UDP  0x11
IPv6 0x29
```

Min. header size is 20 bytes

Time-to-live (TTL)

- Max. number of traversed hops
  - Before a datagram is dropped **(Why?)**



- TTL value is set by the source
  - Linux/Mac 64
  - Windows 128
  - Solaris, Cisco IOS 255

Often used in OS Fingerprinting tools

Time-to-live (TTL)

- When a router receives an IP datagram:
  - If TTL is 0 → drop pkt
  - Decrement TTL by 1
- Does the router need to recalculate checksum?

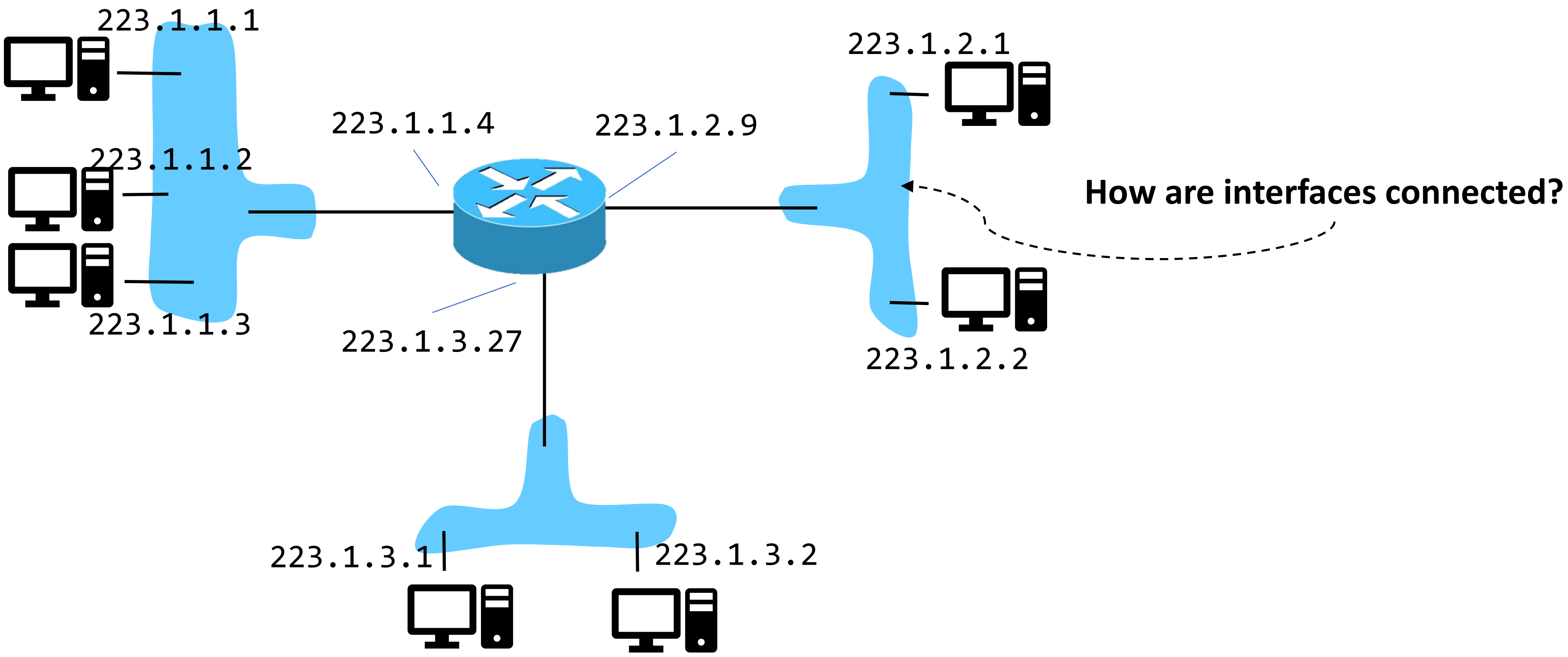
IPv4 Addressing

- **IP address:** 32-bit identifier for host, router interface
- **Interface:** connection between host/router and physical link

- A router typically has multiple interfaces
- A host typically has one or two interfaces

IP addresses/subnets are associated with each interface

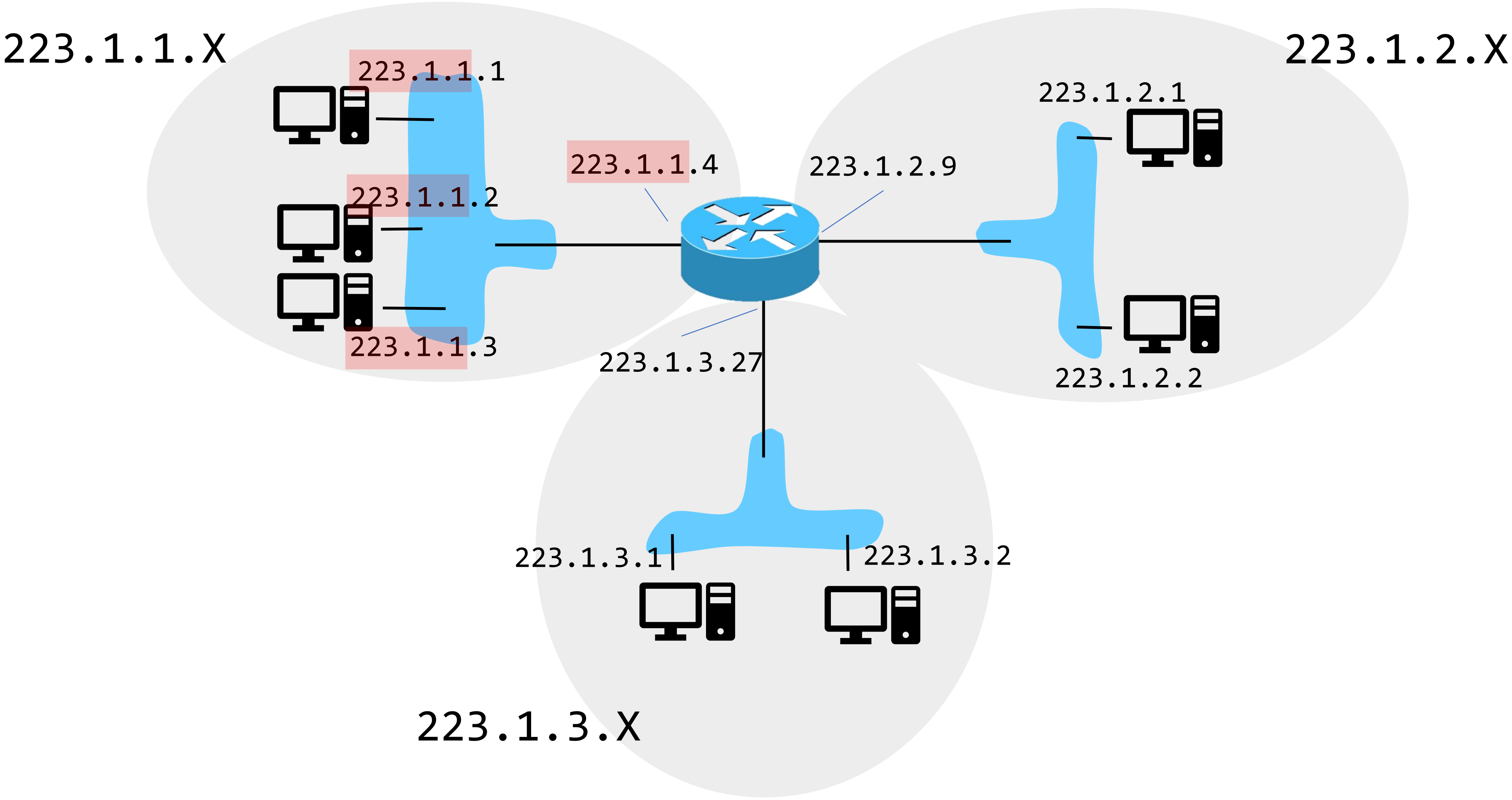
IPv4 Addressing



223.1.1.1 = 11011111 00000001 00000001 00000001

223 . 1 . 1 . 1

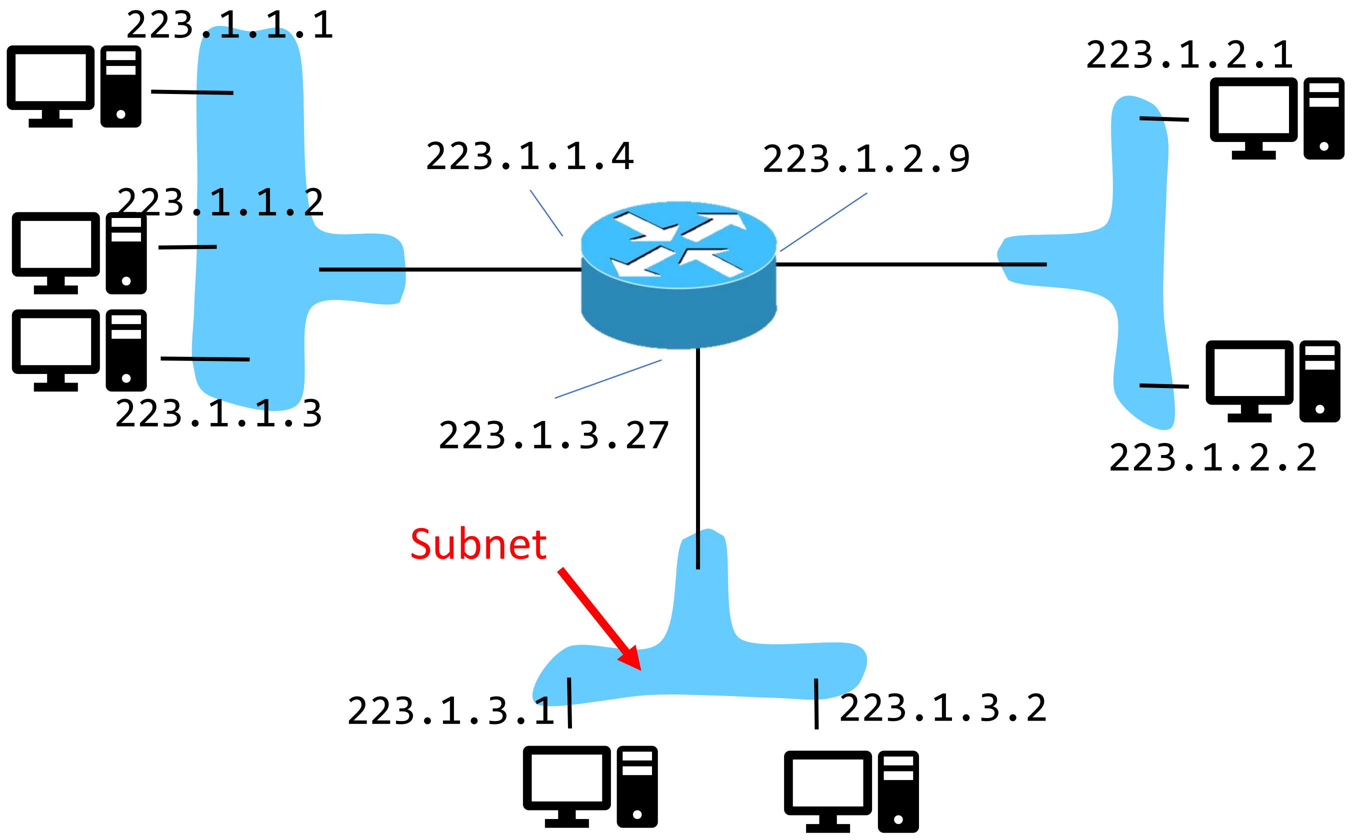
IPv4 Addressing



Subnets

- IP address:
  - subnet part: high order bits
  - host part: low order bits

- What's a subnet ?
  - device *interfaces* with same subnet part of IP address
  - can physically reach each other *without* intervening router

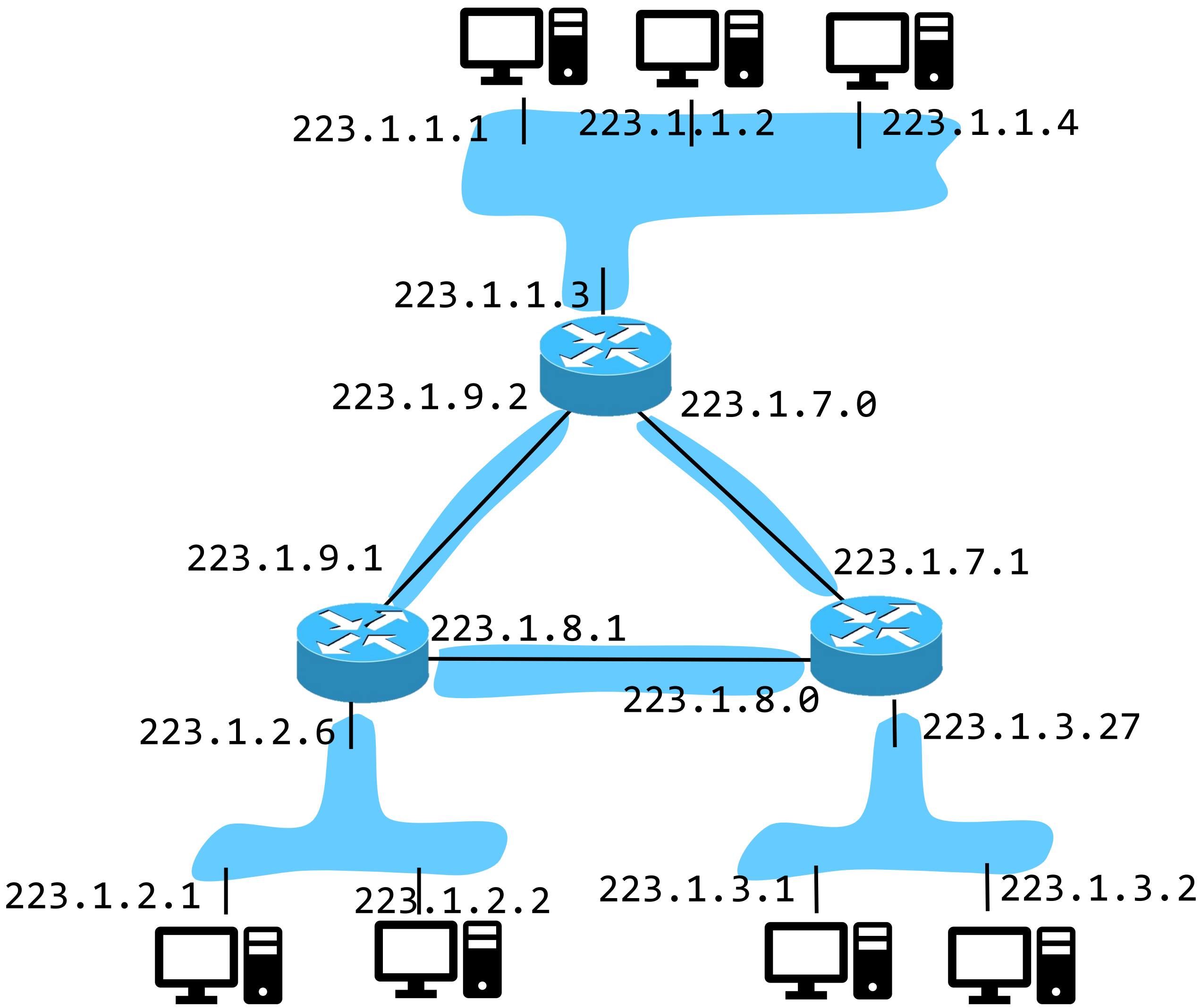


This network consists of three subnets

Subnets

- How many subnets?
  - 6

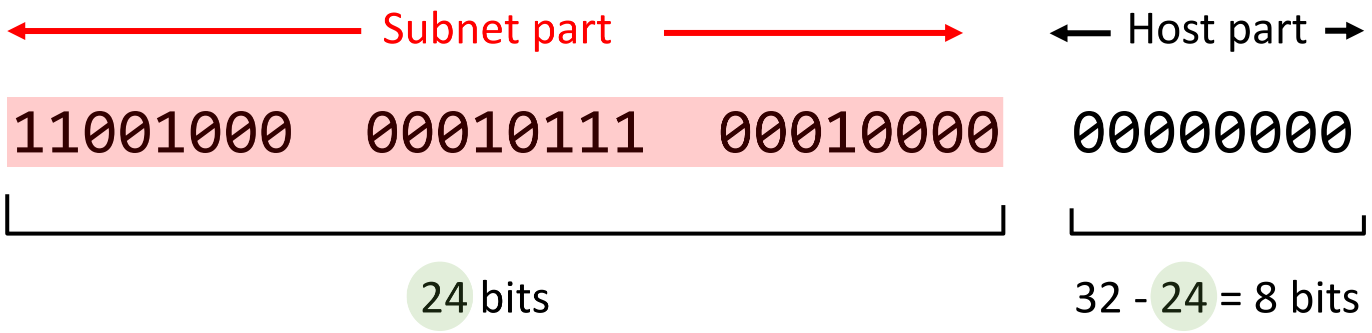
- Recipe
  - to determine the subnets, detach each interface from its host or router, creating islands of isolated networks
  - each isolated network is called a subnet



IPv4 Addressing: CIDR

- CIDR: Classless Inter Domain Routing
- IP address is composed of
  - a subnet part (or prefix)
  - a host part (or suffix)
- Address format: `a.b.c.d/x`, where x is # bits in subnet portion of address (called mask)

`200.23.16.0/24`



IPv4 Addressing: CIDR

200.23.16.0/24

/24 bits means that we have 8 bits to address up to 256 hosts

Subnet part (Prefix)	Host part (Suffix)	IP address
11001000.00010111.00010000.	00000000	200.23.16.0
11001000.00010111.00010000.	00000001	200.23.16.1
11001000.00010111.00010000.	00000010	200.23.16.2
	…	
11001000.00010111.00010000.	11111110	200.23.16.254
11001000.00010111.00010000.	11111111	200.23.16.255

IPv4 Addressing: CIDR

In practice, the first and last IP addresses of a prefix are reserved

→ /24 can support up to 254 (=256-2) hosts

Subnet part (Prefix)	Host part (Suffix)	IP address
11001000.00010111.00010000.	00000000	200.23.16.0
11001000.00010111.00010000.	11111111	200.23.16.255

Identifies the network (host part is all 0's) → 200.23.16.0

Identifies the broadcast address (host part is all 1's) → 200.23.16.255

How to get an IP address?

How does a *host* get IP address?

- Hard-coded by system admin in a file
- DHCP: Dynamic Host Configuration Protocol
  - dynamically get address from a server

How to get an IP address?

How does a *network* get IP address?

- Gets allocated portion of its provider ISP's address space

How to get an IP address?

Example: Given an ISP network called 733 with address `200.23.16.0/20`.

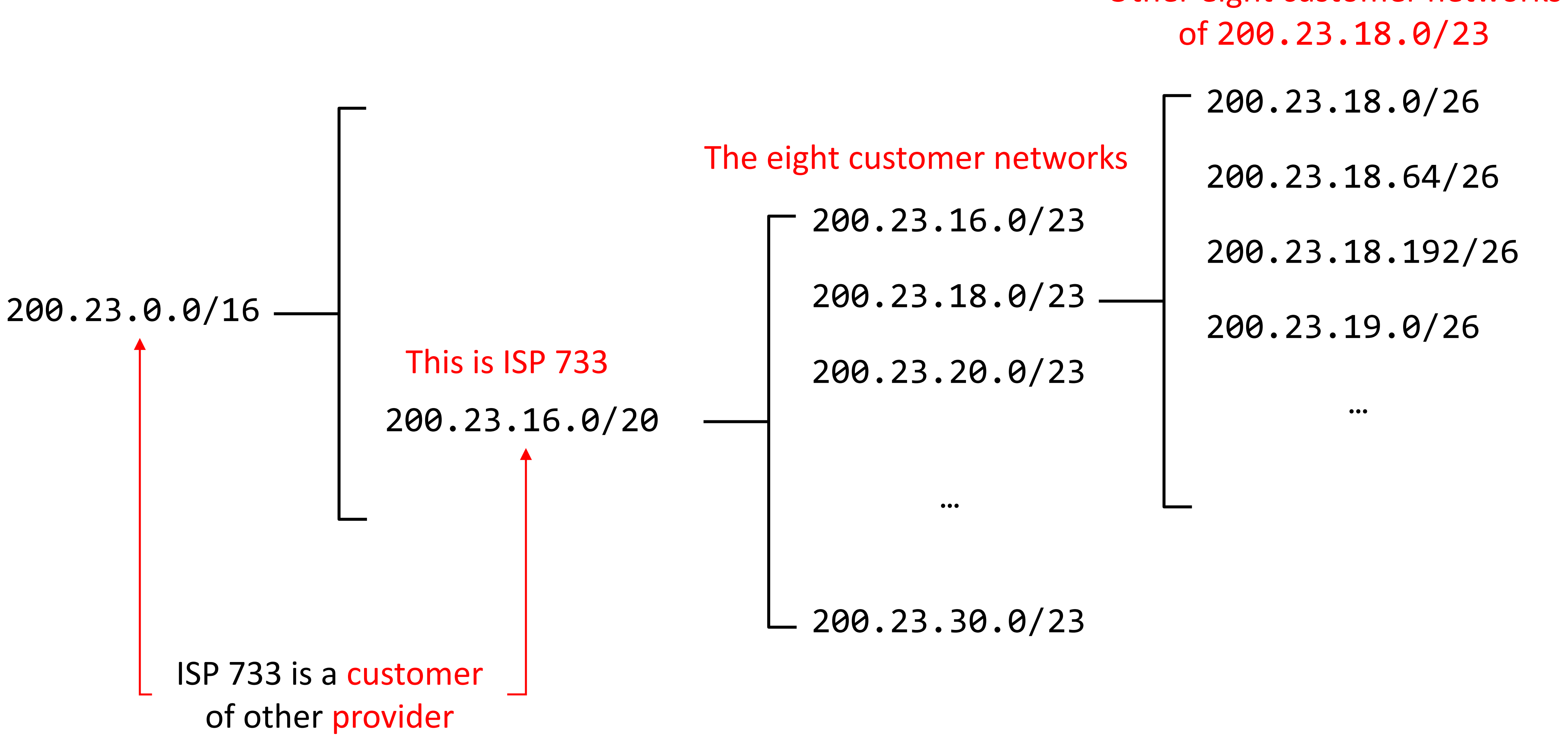
How can it allocate IP addresses for 8 customer networks?

Use additional 3 bits to allocate addresses for the 8 customer networks.

ISP 733 block	11001000	00010111	00010000	00000000	200.23.16.0/20
Organization 0	11001000	00010111	00010000	00000000	200.23.16.0/23
Organization 1	11001000	00010111	00010010	00000000	200.23.18.0/23
Organization 2	11001000	00010111	00010100	00000000	200.23.20.0/23
		…			
Organization 7	11001000	00010111	00011110	00000000	200.23.30.0/23

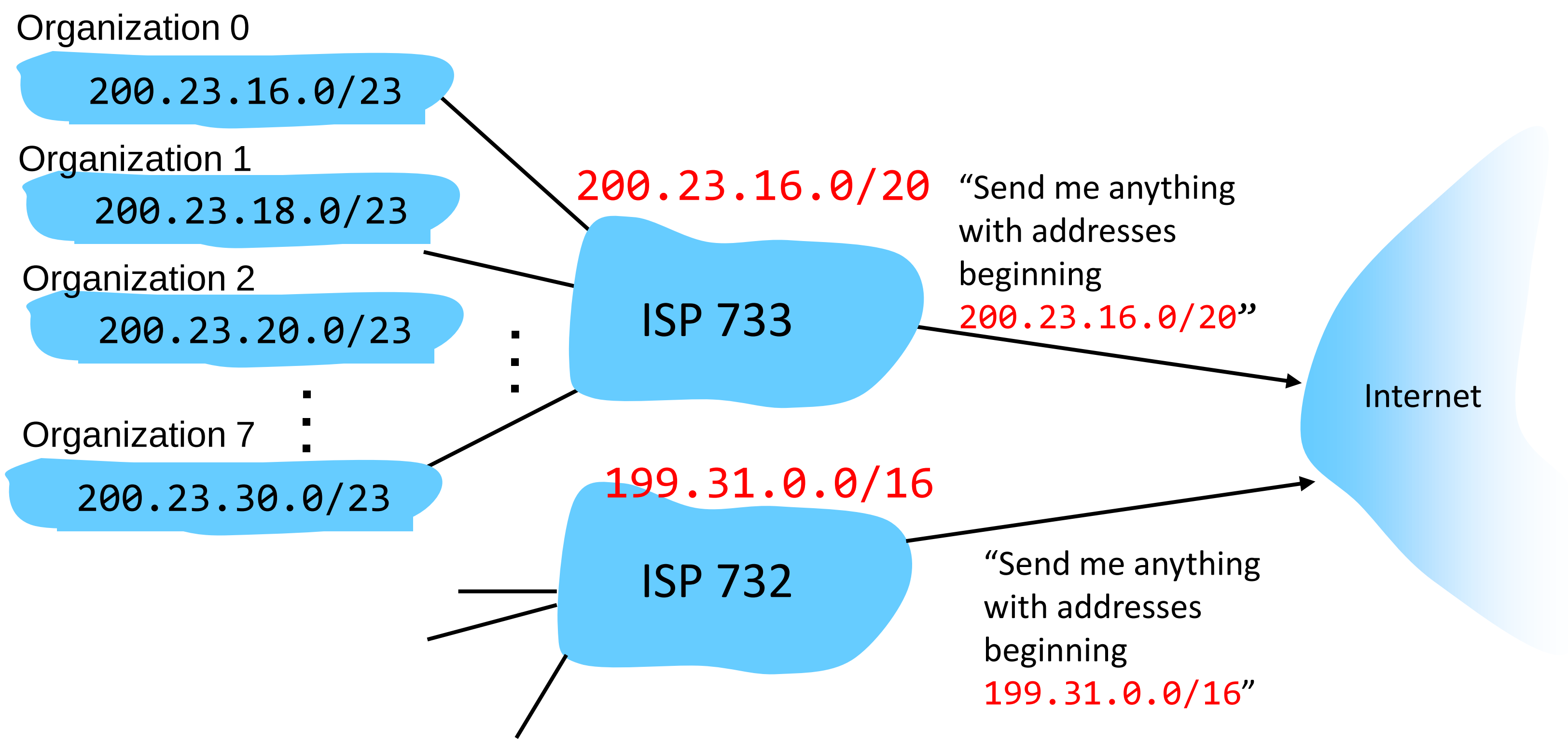
Hierarchical IP Addressing

- IP addresses are hierarchical



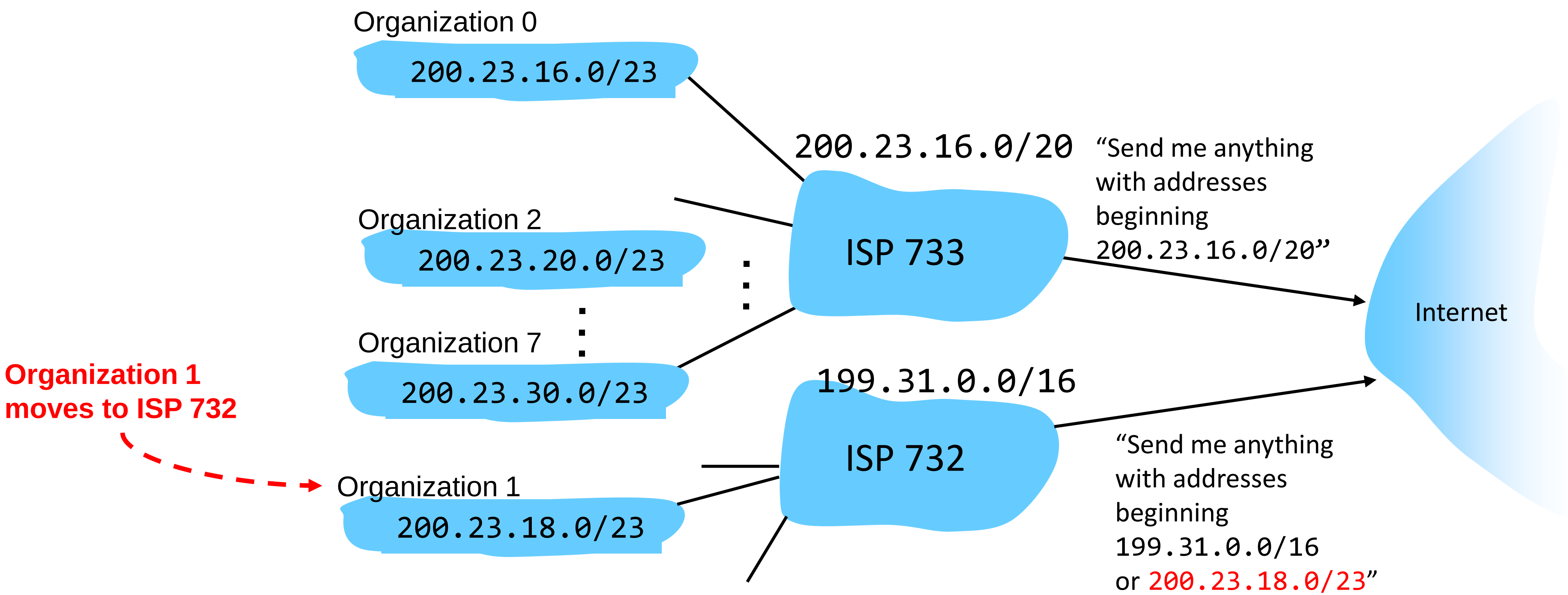
Hierarchical IP Addressing

- Hierarchical addressing allows efficient advertisement of routing information:



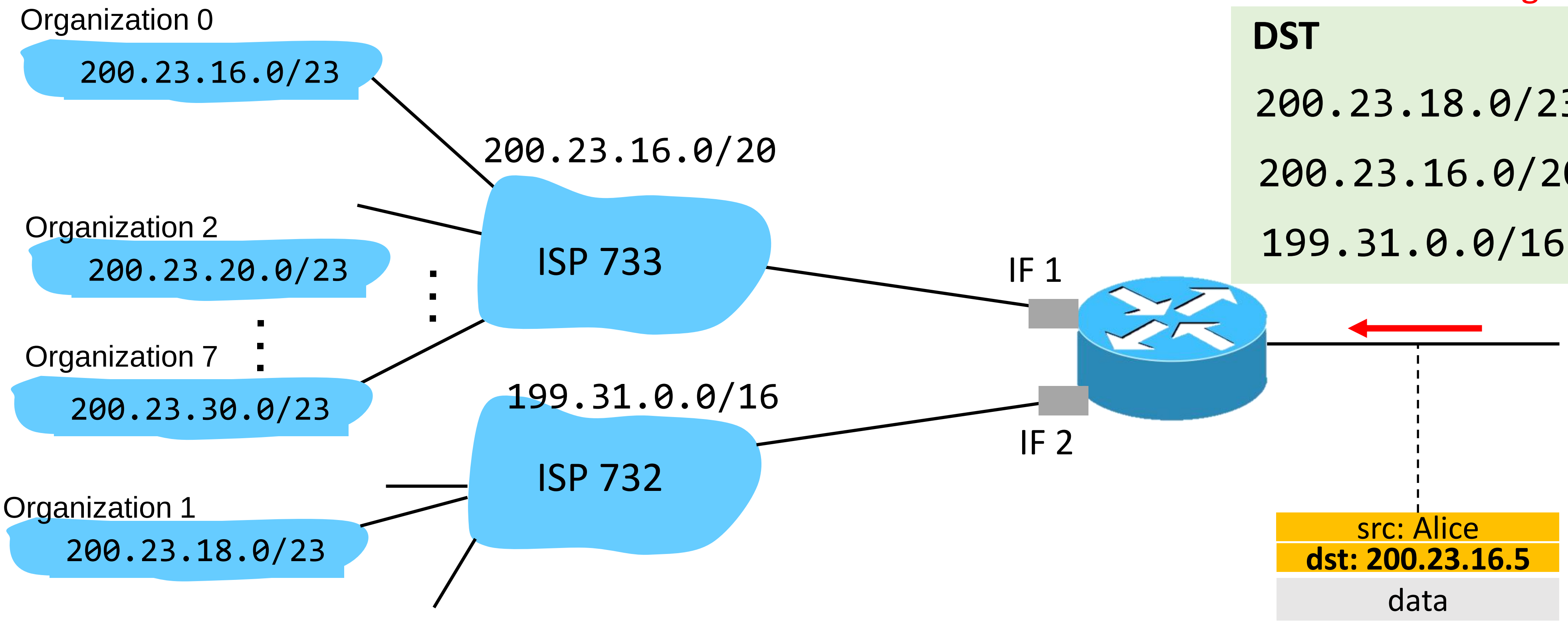
Hierarchical IP Addressing

- Hierarchical addressing allows efficient advertisement of routing information:



Hierarchical IP Addressing

- Routers forward a packet to its destination based on the subnet part, not the host part
  - use longest address prefix that matches destination address
  - This is called the longest prefix matching



Forwarding Table

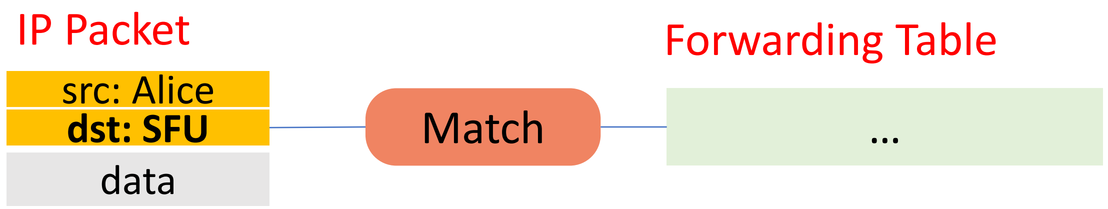
DST	OUT_IF
200.23.18.0/23	IF2
200.23.16.0/20	IF1
199.31.0.0/16	IF2

Hierarchical IP Addressing: Summary

- Scalable forwarding tables
- Adding/removing hosts without modifying forwarding table
- Small prefix advertisement overhead

Destination-based Forwarding

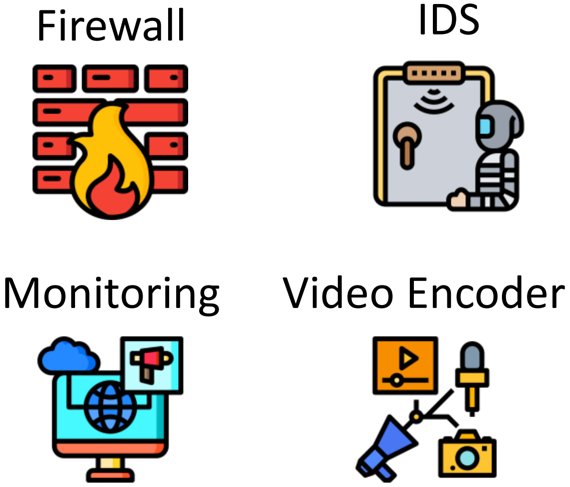
- Look-up is done at the input port
- IP routers forward packets by:
  - examining the **destination address**, and
  - matching it with a local forwarding table



They use the longest prefix matching algorithm

Generalized Forwarding

- Large-scale networks are complex
  - They don't have "routers" networks.
  - They include middleboxes and other devices



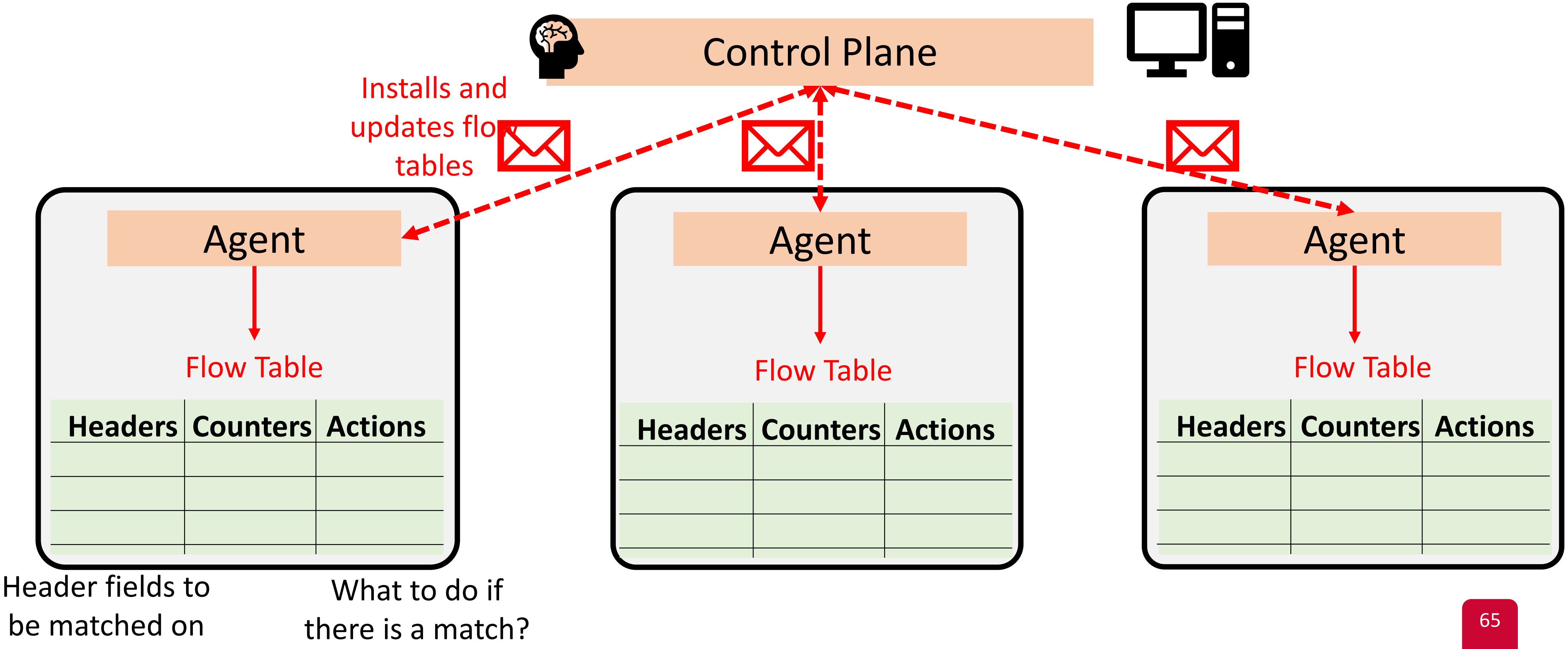
- Network management becomes a hard task

- Network operators need a unified way to manage all of their network devices!

- One solution: Software-defined networks

Generalized Forwarding

- Each router contains a flow table that is computed and distributed by a logically centralized controller



OpenFlow Data Plane Abstraction

- Flow: defined by header fields
- Simple packet-handling rules
  - Pattern: match values in packet header fields
  - Actions: (for a matched pkt)
    - drop, forward, modify a matched packet or send matched packet to controller
  - Priority: disambiguate overlapping patterns
  - Counters: #bytes and #packets

```
1. src=1.2.*.*, dest=3.4.5.* → drop
2. src = *.*.*.*, dest=3.4.*.* → forward(2)
3. src=10.1.2.3, dest=*.*.*.* → send to controller
```

Questions?

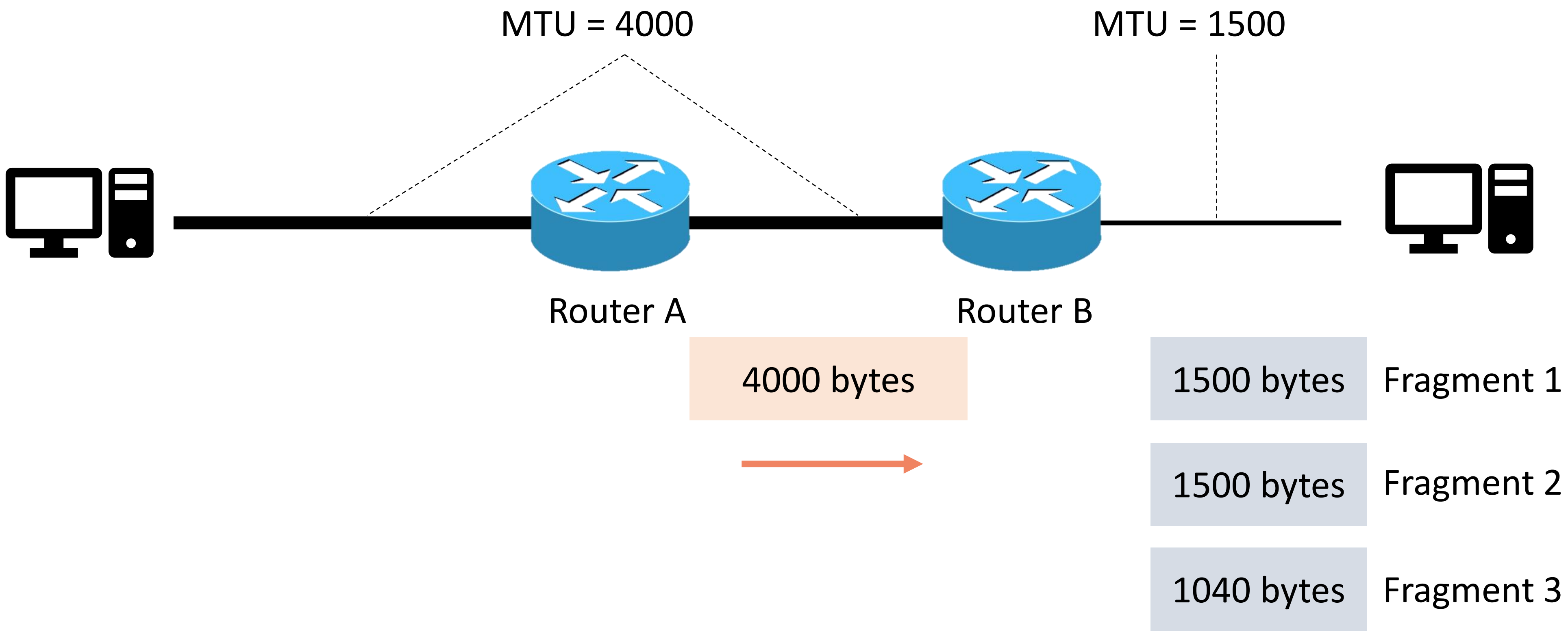
Extras

IPv4 Fragmentation

- Different link layer protocols have different MTU
  - Maximum transmission unit
- A router can break a datagram into fragments
  - If MTU of outgoing link is less than pkt size
- A destination reassembles IP fragments
  - To be delivered to transport layer
  - *Why is the reassembly done at destinations?*

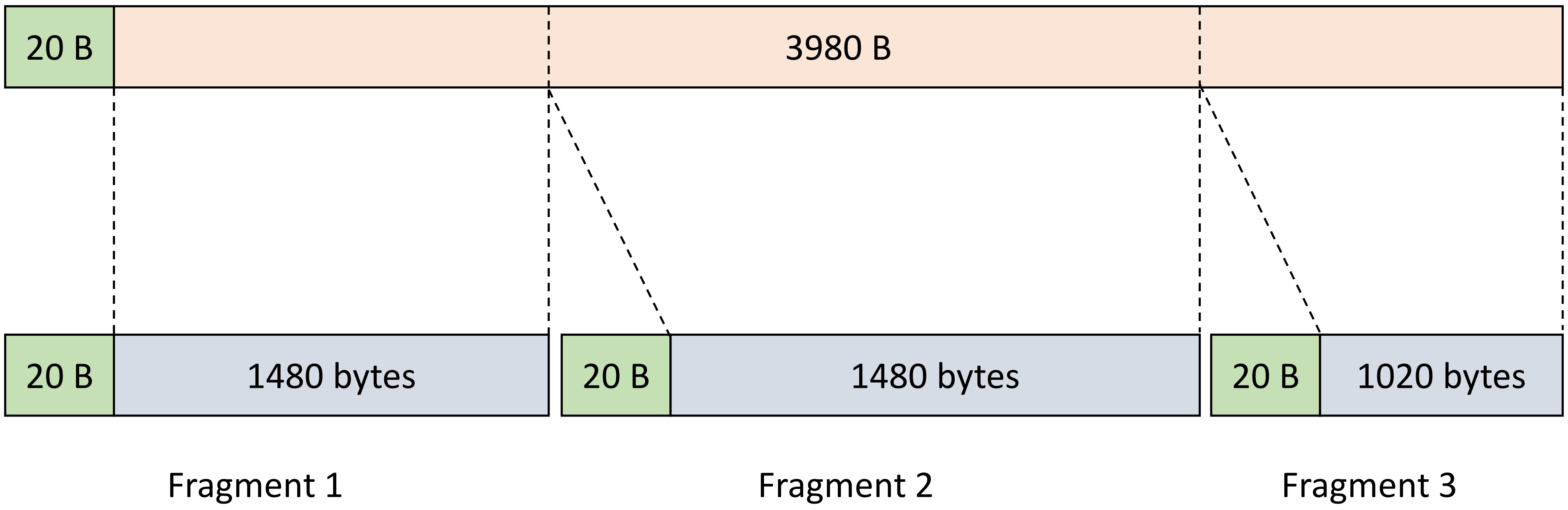
IPv4 Fragmentation

- Example:



IPv4 Fragmentation

- Example:



IPv4 Fragmentation

- Issues
  - All fragments must be delivered to the destination → not guaranteed!
  - Last fragment may have non-optimal size → wasting router resources
  - Destination needs to hold IP fragments in memory
  - Only first datagram contains TCP/UDP header
    - Firewalls and other network functions don't work well with IP fragments
- In the current Internet, fragmentation is not recommended
- IPv6 does not support fragmentation

Kent and Mogul https://www.hpl.hp.com/techreports/Compaq-DEC/WRL-87-3.pdf 1987

IPv6

- Initial motivation:
  - 32-bit address space soon to be completely allocated.

- Additional motivation:
  - header format helps speed processing/forwarding
  - header changes to facilitate QoS

- IPv6 datagram format:
  - fixed-length **40-byte** header
  - no fragmentation allowed

IPv6 Datagram Format

Internet Protocol Version 6 (IPv6)							
Offsets	Octet	0		1		2	3
Octet	Bit	0–3	4–7	8–11	12–15	16–23	24–31
0	0	Version	Traffic Class		Flow Label		
4	32	Payload Length				Next Header	Hop Limit
8	64	Source IP Address					
12	96						
16	128						
20	160						
24	192	Destination IP Address					
28	224						
32	256						
36	288						

Priority/Traffic Class: identify priority among datagrams in flow

Flow Label: identify datagrams in same "flow"

Next header: identify upper layer protocol for data

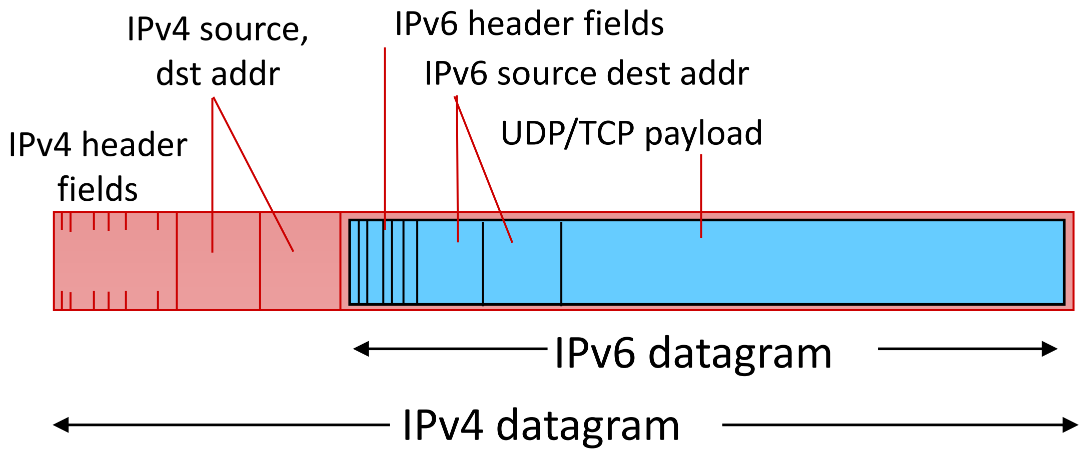
Other Changes

- Checksum: removed entirely to reduce processing time at each hop
- Options: allowed, but outside of header, indicated by “Next Header” field
- No Fragmentation:
  - Packet is dropped if its size is larger than outgoing link MTU
  - An error message is sent to the sender

IPv4 → IPv6

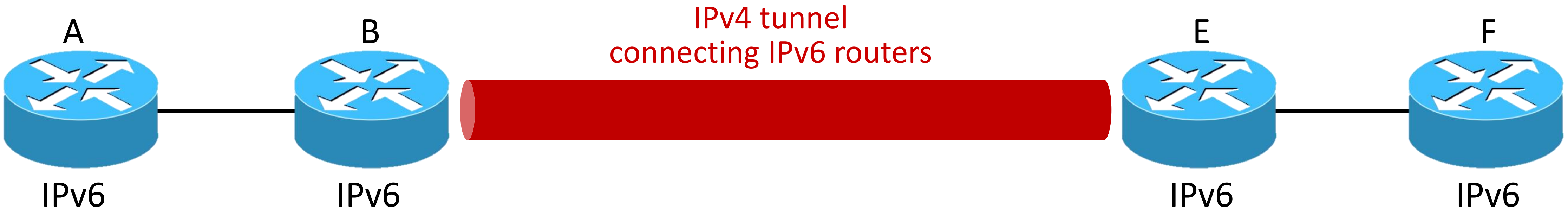
- Not all routers can be upgraded simultaneously
  - how will network operate with mixed IPv4 and IPv6 routers?

- **Tunneling:**
  - IPv6 datagram carried as payload in IPv4 datagram among IPv4 routers

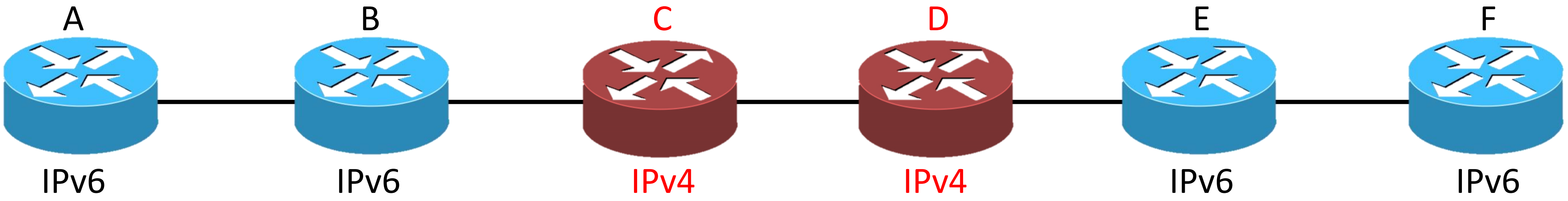


IPv4 → IPv6: Tunneling

Logical View



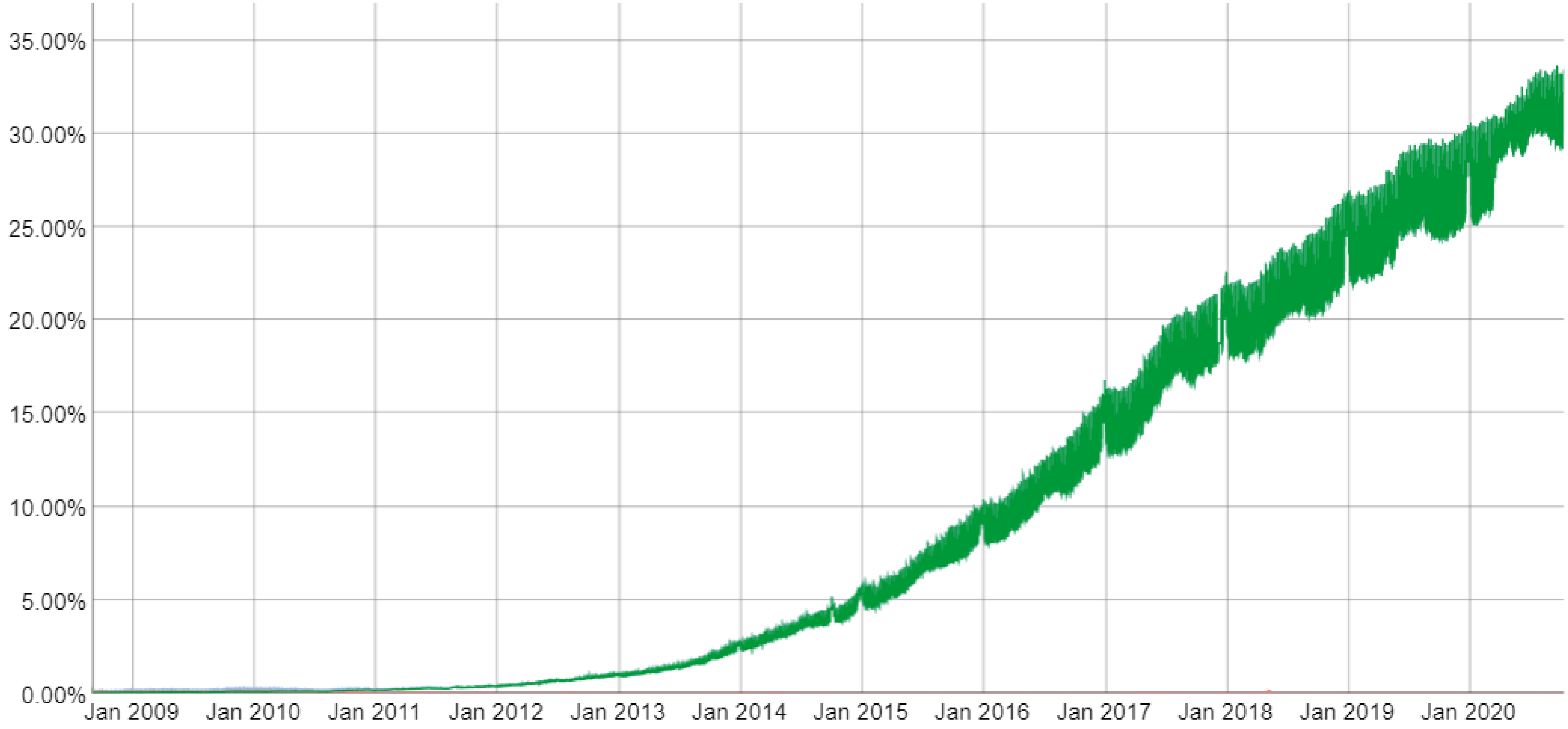
Physical View



IPv6 Deployment

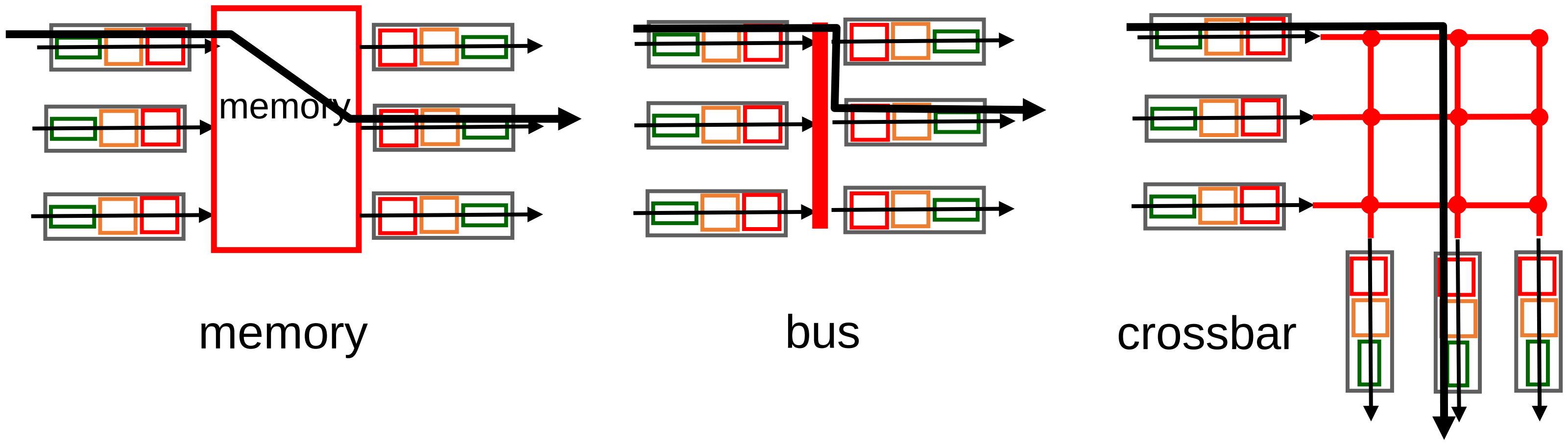
- It is hard to change the network-layer protocols!
- IPv6 was first introduced in 1995!

Percentage of users accessing Google using IPv6

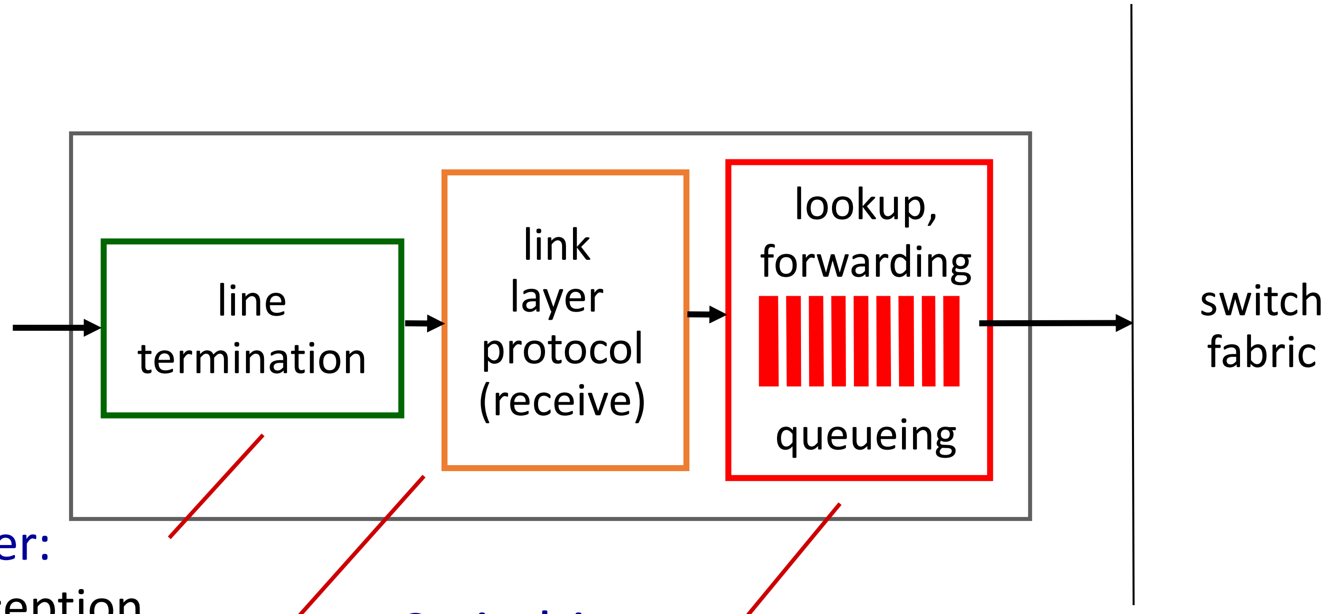


Switching Fabric

- Transfer packet from input buffer to appropriate output buffer
- Switching rate: rate at which packets can be transfer from inputs to outputs
- Three types of switching fabrics



Input Port Functions



Physical layer:
bit-level reception

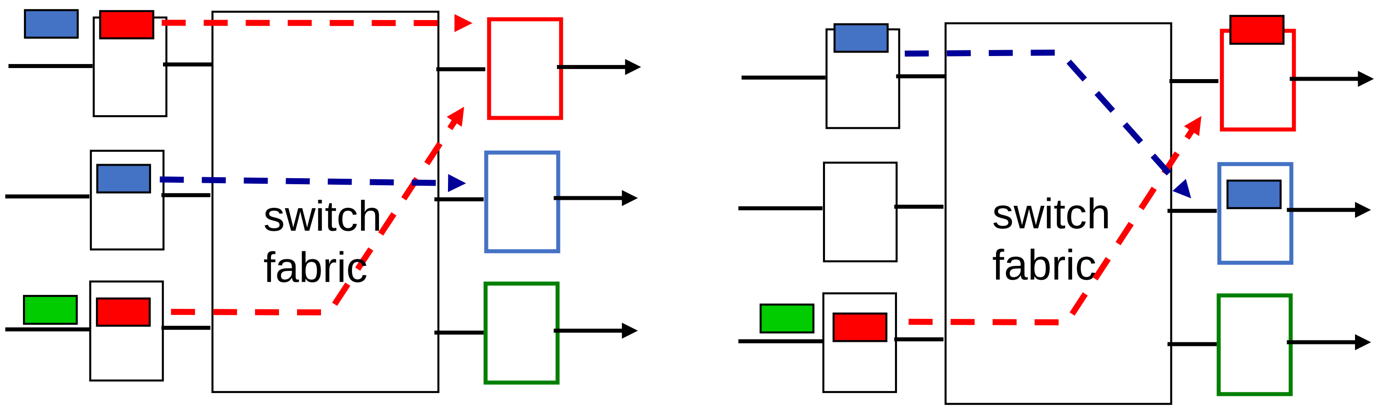
Data-link layer:
e.g., Ethernet

Switching:

- Using header field values, lookup output port using forwarding table in input port memory *("match plus action")*
- **Goal:** complete input port processing at 'line speed'
- **Queuing:** if datagrams arrive faster than forwarding rate into switch fabric

Input Port Queuing

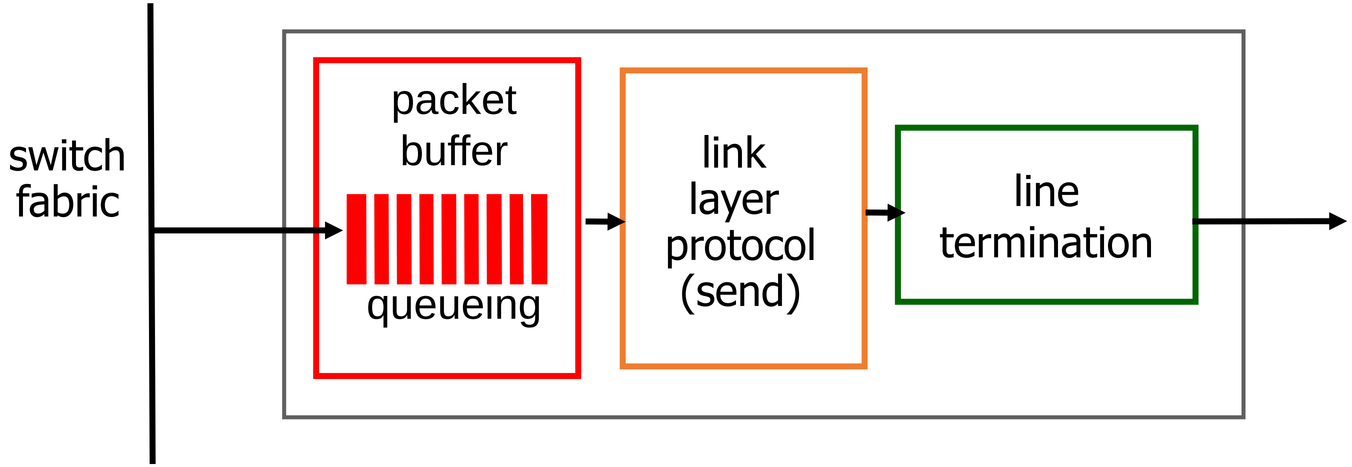
- Fabric slower than input ports combined → queuing may occur at input queues
  - queuing delay and loss due to input buffer overflow!
- Head-of-the-Line (HOL) blocking: queued datagram at front of queue prevents others in queue from moving forward



Output port contention: only one red datagram can be transferred. *lower red packet is blocked*

One packet time later: green packet experiences HOL blocking

Output Ports



- Buffering required when packets arrive from fabric faster than the transmission rate
  - Packets can be lost due to congestion, lack of buffers
- Scheduling discipline chooses among queued packets for transmission
  - Priority scheduling – who gets best performance, network neutrality

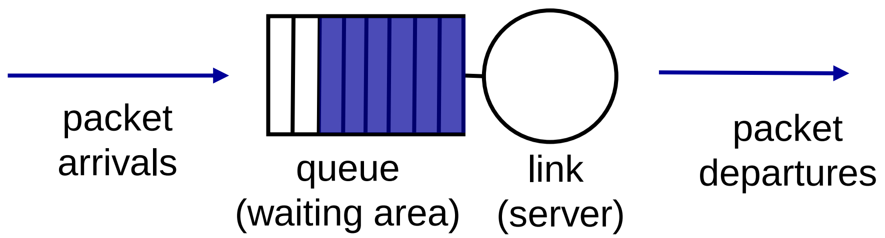
Output Port Queuing

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

Scheduling Policy

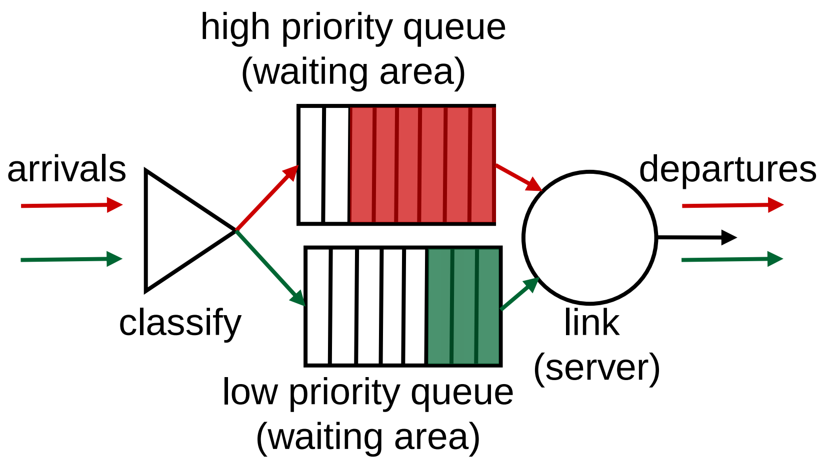
- Scheduling: choose next packet to send on link

- FIFO (first in first out) scheduling: send in order of arrival to queue
  - discard policy: if packet arrives to full queue: who to discard?
    - tail drop: drop arriving packet
    - priority: drop/remove on priority basis
    - random: drop/remove randomly



Scheduling Policy: Priority

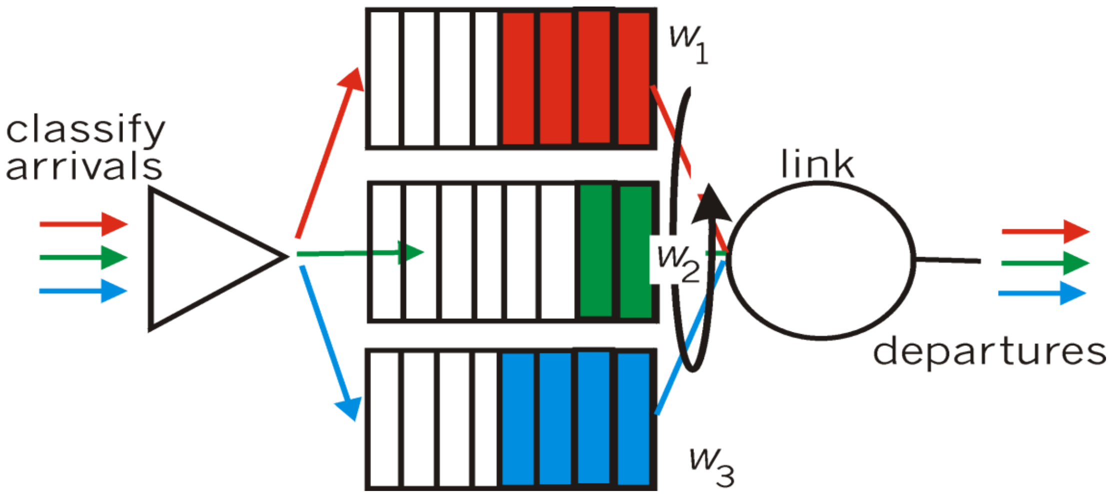
- Priority scheduling: send highest priority queued packet
  - multiple classes, with different priorities
  - class may depend on marking or other header info, e.g. IP source/dest, port numbers, etc.



Scheduling Policy: Other Policies

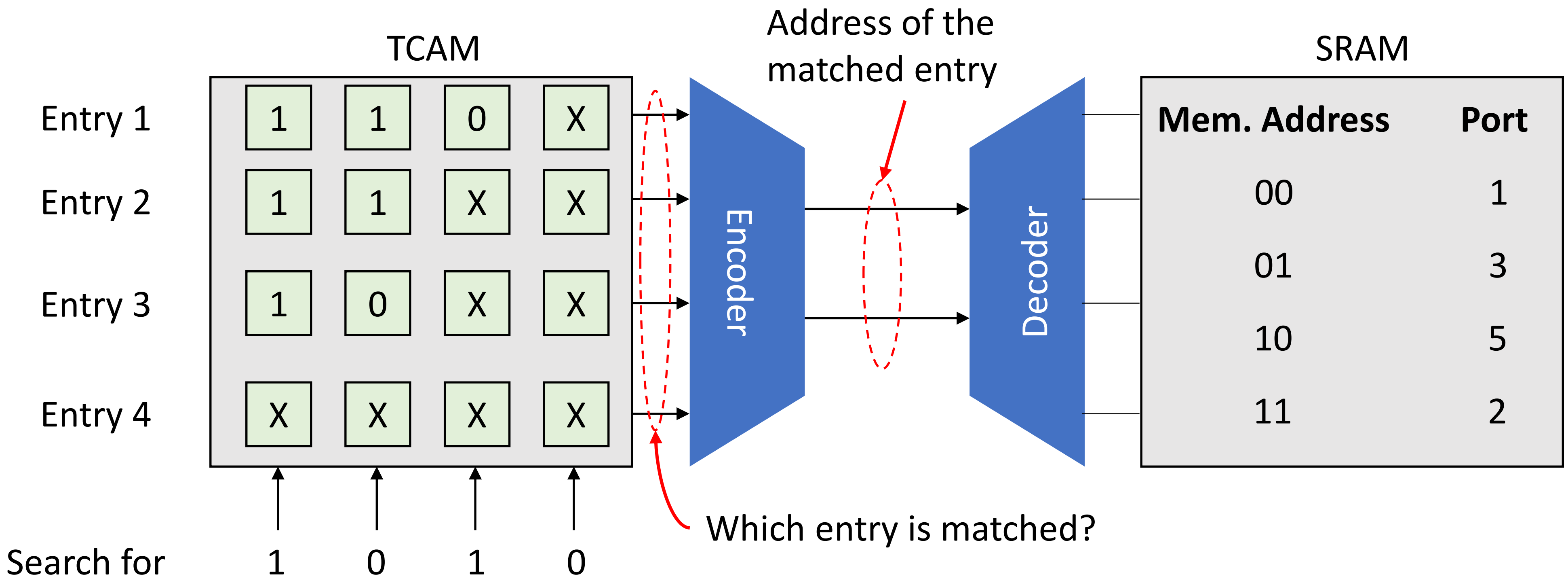
- Round Robin (RR):
  - multiple classes
  - cyclically scan class queues, sending one complete packet from each class (if available)

- Weighted Fair Queuing (WFQ):
  - generalized Round Robin
  - each class gets weighted amount of service in each cycle



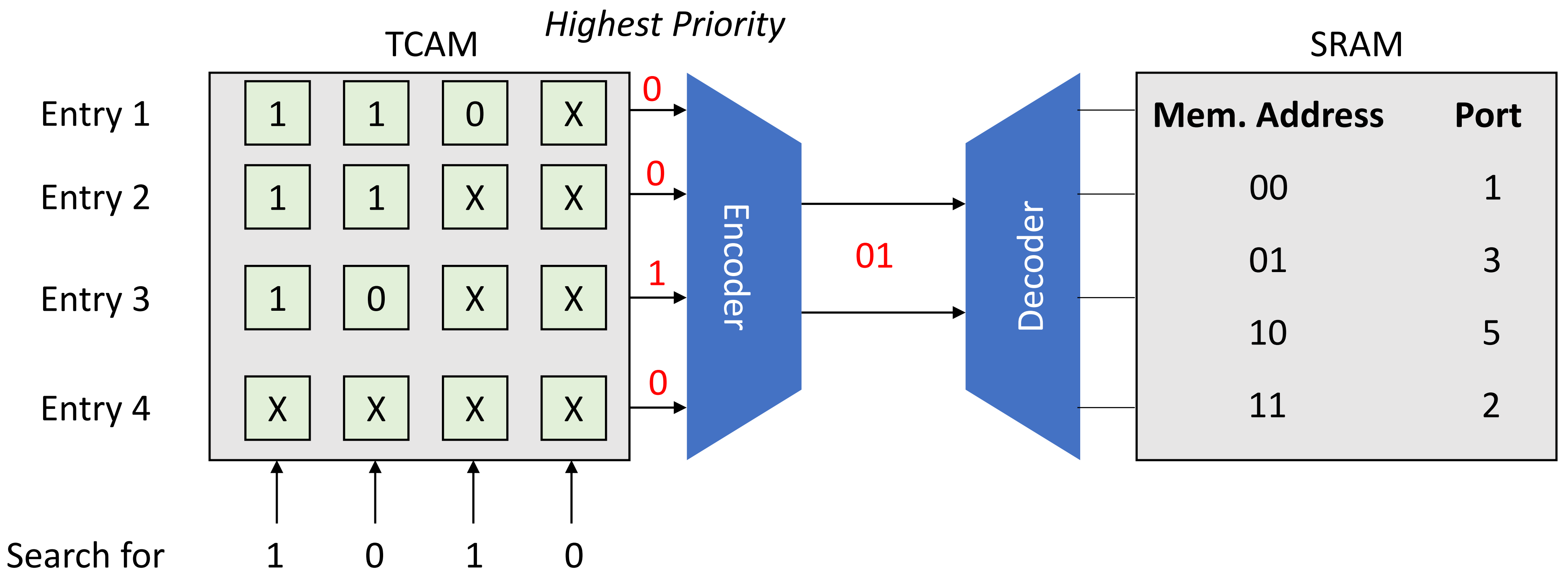
Longest Prefix Matching

- Longest prefix matching: often performed using ternary content addressable memories (TCAMs)
- Content addressable :
  - present address to TCAM
  - retrieve address in one clock cycle, regardless of table size



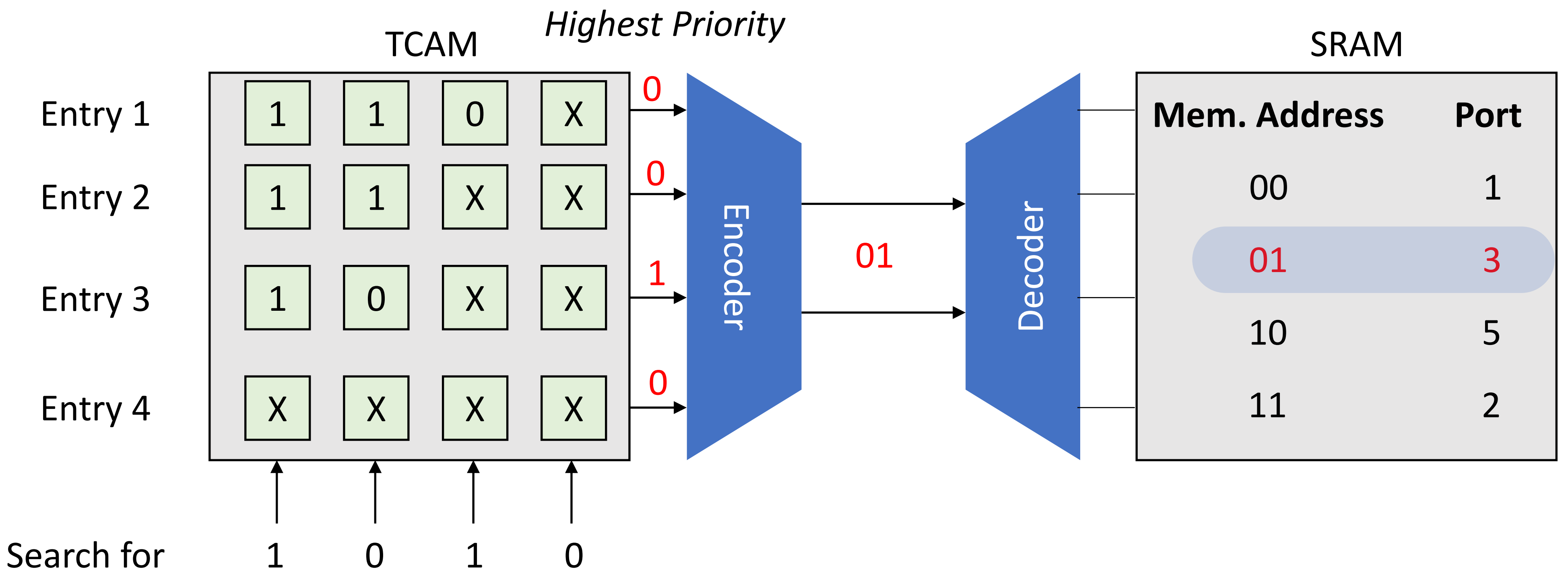
Longest Prefix Matching

- Longest prefix matching: often performed using ternary content addressable memories (TCAMs)
- Content addressable :
  - present address to TCAM
  - retrieve address in one clock cycle, regardless of table size



Longest Prefix Matching

- Longest prefix matching: often performed using ternary content addressable memories (TCAMs)
- Content addressable :
  - present address to TCAM
  - retrieve address in one clock cycle, regardless of table size

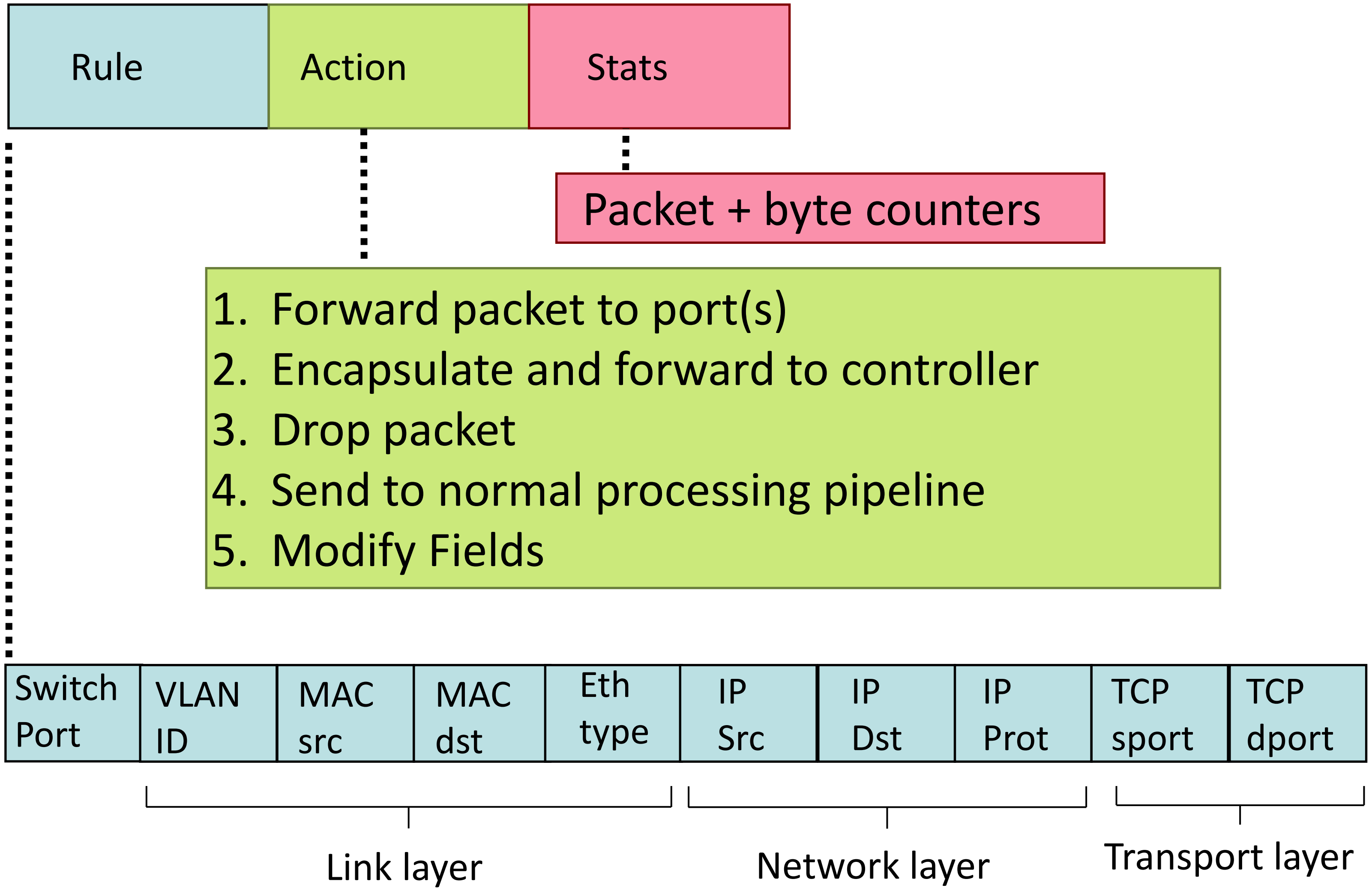


TCAM Advantages and Disadvantages

- Advantages:
  - Simpler that other (trie-based) algorithms
  - Read operation is done in one clock cycle

- Disadvantages:
  - Requires larger chip area
    - E.g., a typical SRAM cell contains 6T, while a TCAM cell contains 16T!
  - High power consumption

OpenFlow: Flow Table Entries



OpenFlow: Examples

Destination-based forwarding:

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	51.6.0.8	*	*	*	port6

IP datagrams destined to IP address 51.6.0.8 should be forwarded to router output port 6

Firewall:

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Forward
*	*	*	*	*	*	*	*	*	22	drop

do not forward (block) all datagrams destined to TCP port 22

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Forward
*	*	*	*	*	128.119.1.1	*	*	*	*	drop

do not forward (block) all datagrams sent by host 128.119.1.1

OpenFlow Abstraction

- Match+action: unifies different kinds of devices

- Router
  - *match:* longest destination IP prefix
  - *action:* forward out a link
- Switch
  - *match:* destination MAC address
  - *action:* forward or flood

- Firewall
  - *match*: IP addresses and TCP/UDP port numbers
  - *action:* permit or deny
- NAT
  - *match:* IP address and port
  - *action:* rewrite address and port