Network Layer: IPv6 & SDN

A/PROF. DUY NGO

Learning Objectives

4.3 IP: Internet Protocol

- **IPv6**

4.4 Generalized Forward and SDN

- match
- action
- OpenFlow examples of match-plus-action in action

IPv6: Motivation

- **initial motivation:** 32-bit address space has been exhausted.
- additional motivation:
  - header format helps speed processing/forwarding
  - header changes to facilitate QoS

IPv6 datagram format:

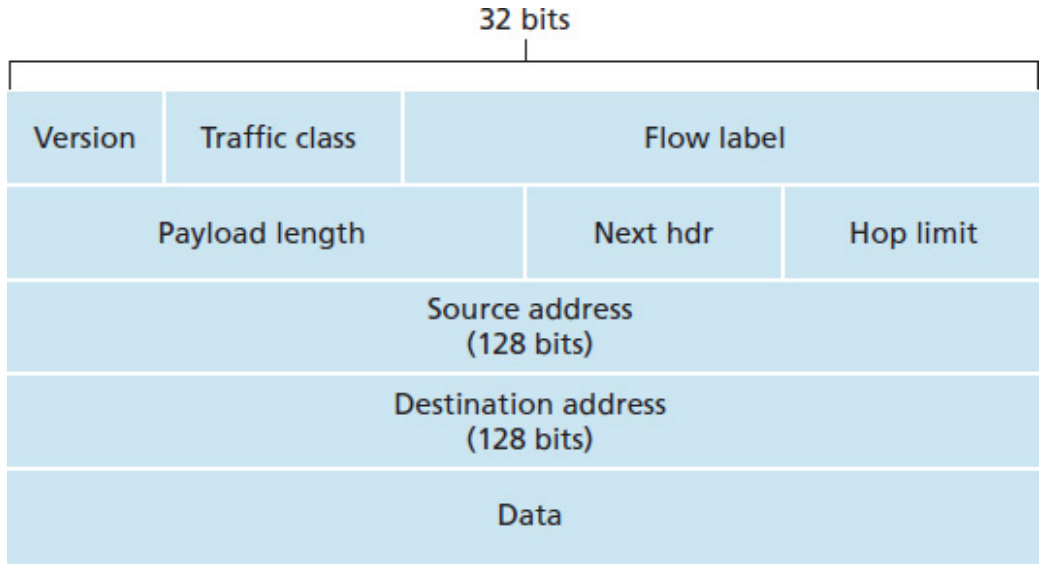
- fixed-length 40 byte header
- no fragmentation allowed

IPv6 Datagram Format

priority: identify priority among datagrams in flow

flow Label: identify datagrams in same “flow.” (concept of ”flow” not well defined).

next header: identify upper layer protocol for data



Other Changes from IPv4

checksum: removed entirely to reduce processing time at each hop

options: allowed, but outside of header, indicated by “Next Header” field

ICMPv6: new version of ICMP

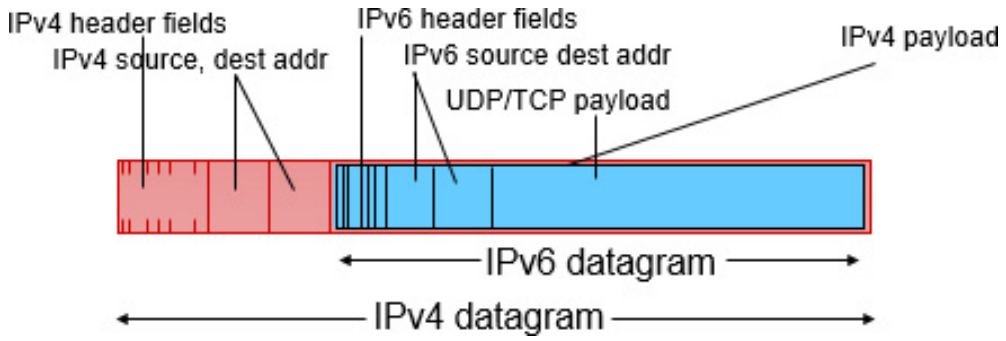
- additional message types, e.g. “Packet Too Big”
- multicast group management functions

Transition from IPv4 to IPv6

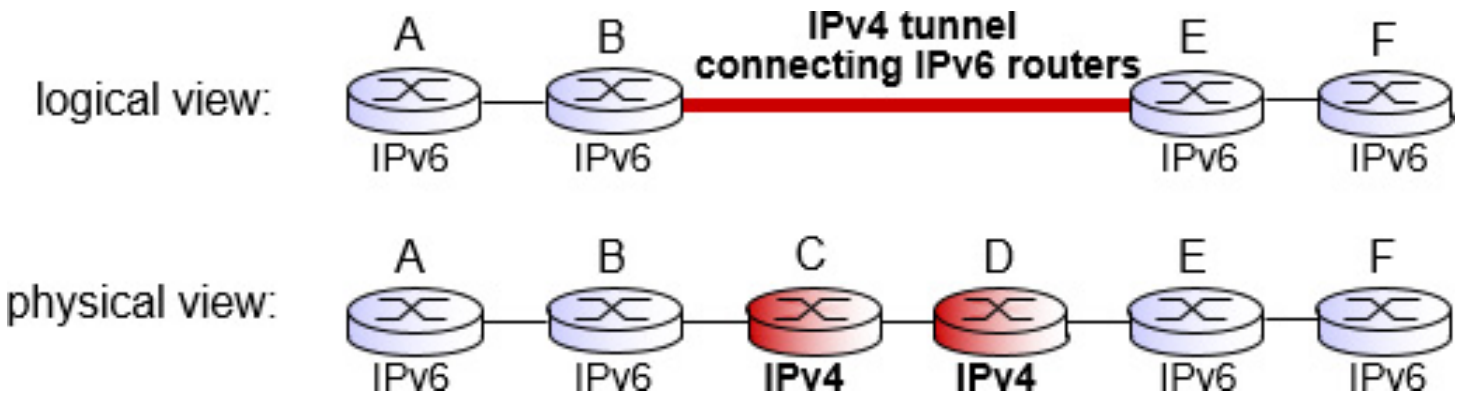
not all routers can be upgraded simultaneously

- no “flag days”
- how will network operate with mixed IPv4 and IPv6 routers?

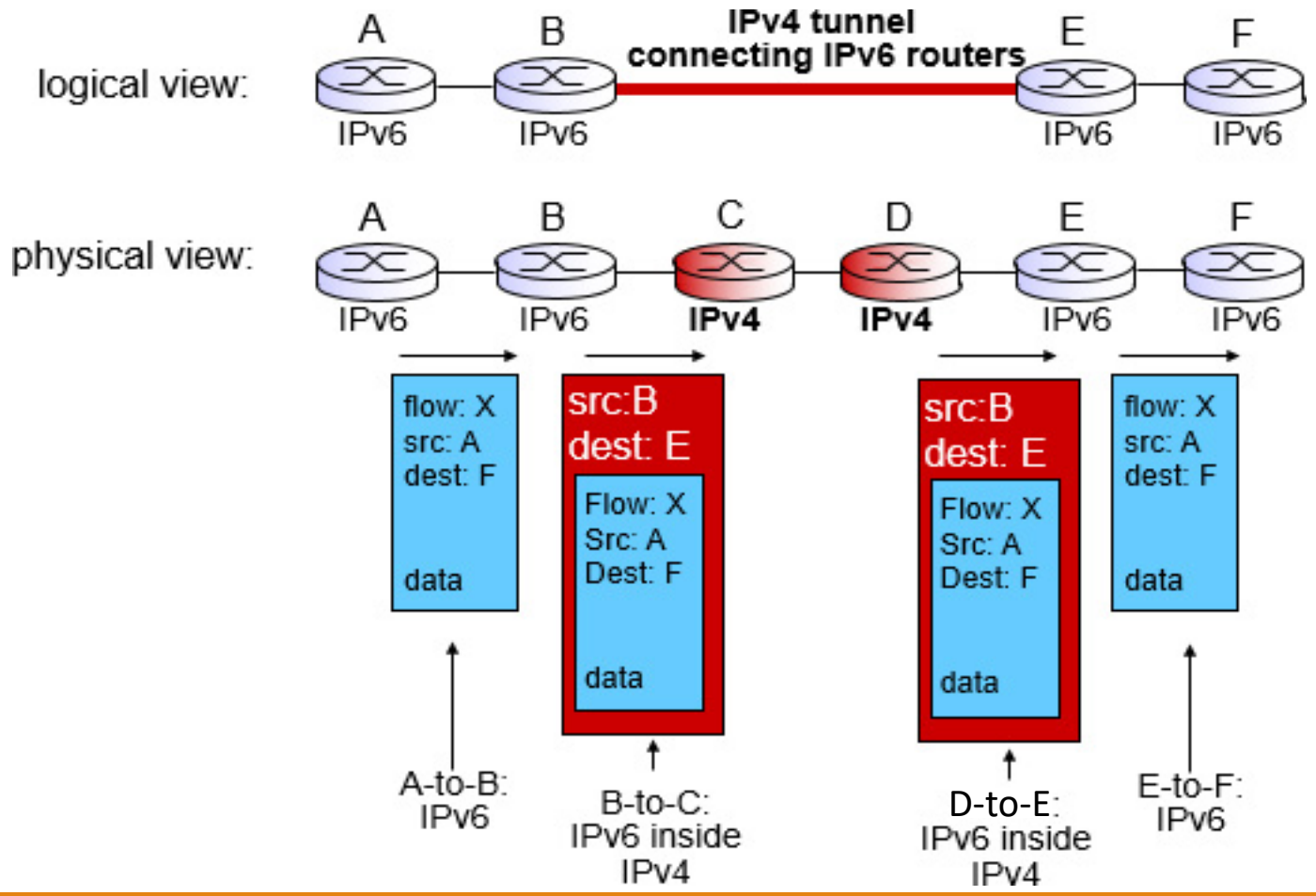
tunneling: IPv6 datagram carried as **payload** in IPv4 datagram among IPv4 routers



Tunneling (1 of 2)



Tunneling (2 of 2)



IPv6: Adoption

Google: 28.2% of clients access services via IPv6

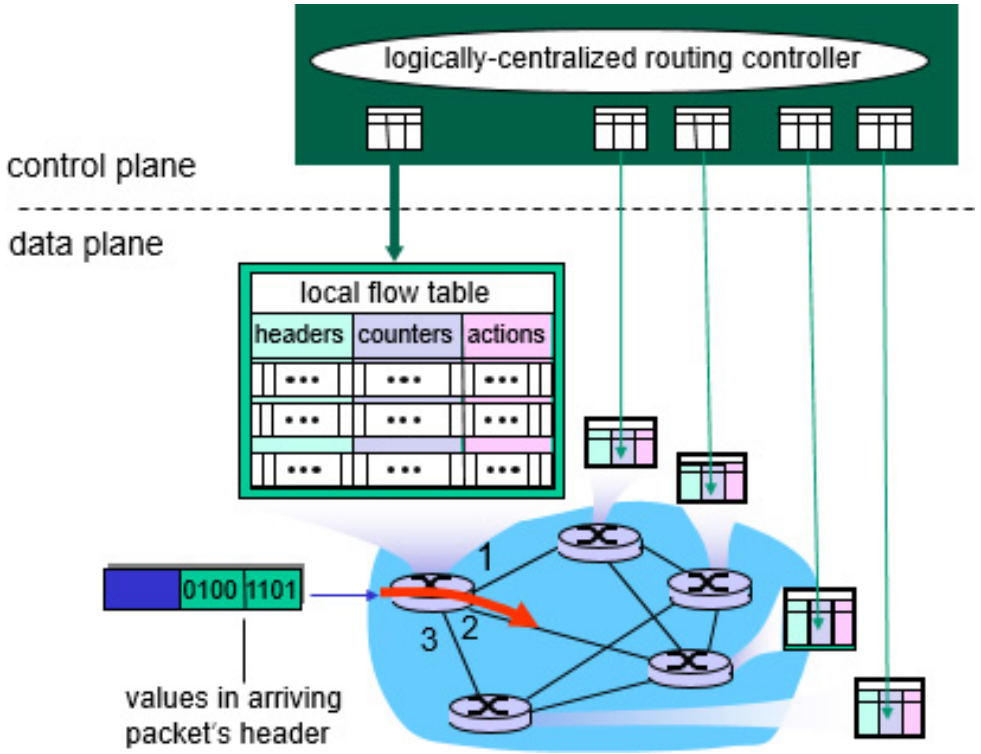
NIST (National Institute of Standards and Technology): 1/3 of all US government domains are IPv6 capable

Long (long!) time for deployment, use

- 20 years and counting!
- think of application-level changes in last 20 years: WWW, Facebook, streaming media, Skype, …
- **Why?**

Generalized Forwarding and SDN

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



OpenFlow Data Plane Abstraction (1 of 2)

- **flow: defined by header fields**
- **generalized forwarding: simple packet-handling rules**
  - **Pattern: match** values in packet header fields
  - **Actions: for matched packet:** drop, forward, modify, matched packet or send matched packet to controller
  - **Priority:** disambiguate overlapping patterns
  - **Counters:** #bytes and #packets

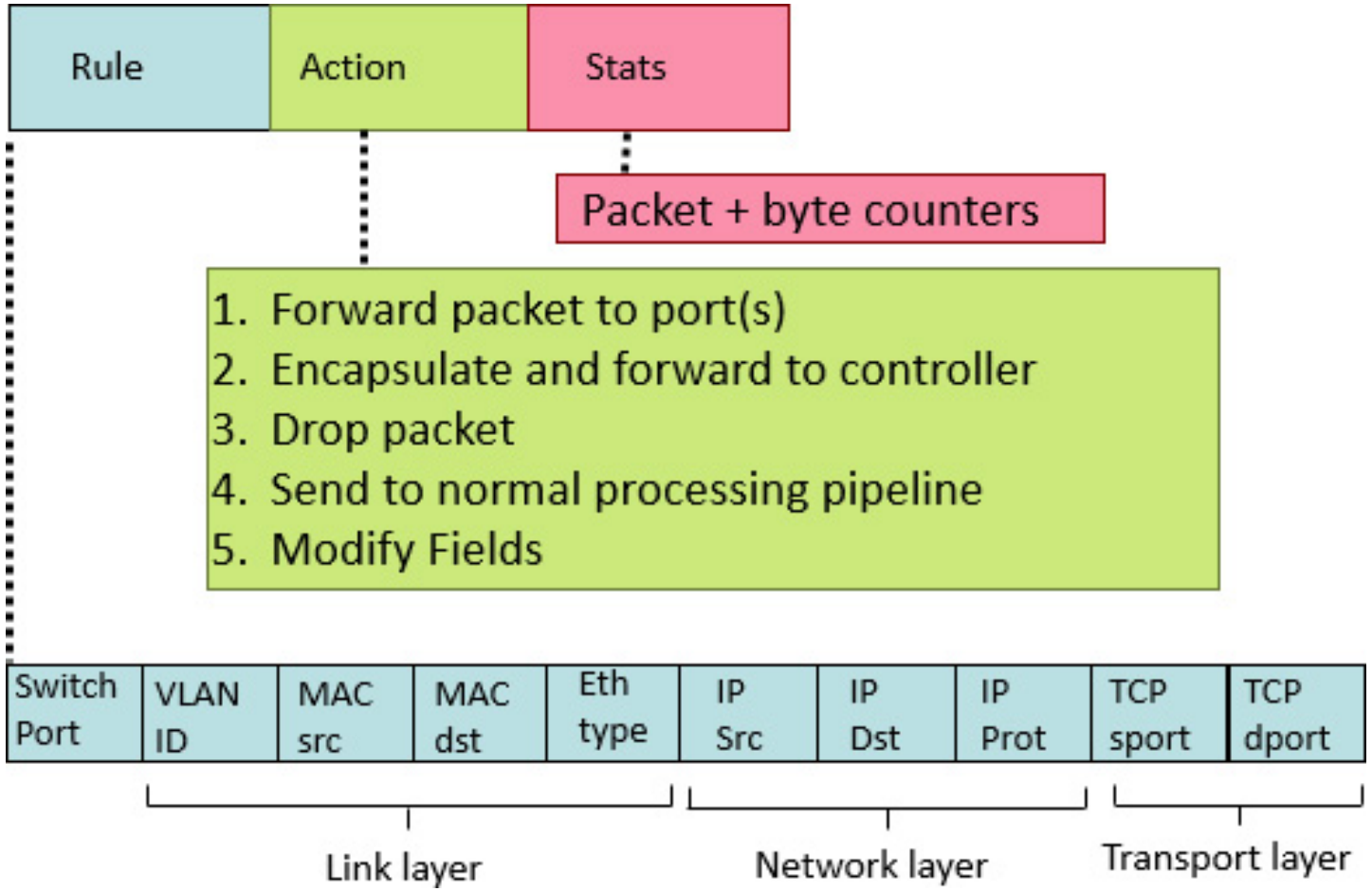
Flow table in a router (computed and distributed by controller) define router's match+action rules

OpenFlow Data Plane Abstraction (2 of 2)

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

* : wildcard

OpenFlow: Flow Table Entries



Example (1 of 2)

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

Example (2 of 2)

Destination-based layer 2 (switch) forwarding:

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	22:A7:23:11:E1:02	*	*	*	*	*	*	*	*	port3

layer 2 frames from MAC address 22:A7:23:11:E1:02 should be forwarded to output port 6

OpenFlow Abstraction (1 of 2)

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

OpenFlow Abstraction (2 of 2)

Firewall

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

NAT

- **match:** IP address and port
- **action:** rewrite address and port

OpenFlow Example

Example: datagrams from hosts h5 and h6 should be sent to h3 or h4, via s1 and from there to s2