CHAPTER 4

The Network Layer

Roadmap:

4.1 Introduction

4.2 Virtual circuit and datagram networks

4.3 What's inside a router

4.4 IP: Internet Protocol
- datagram format
- IPv4 addressing
- ICMP
- IPv6

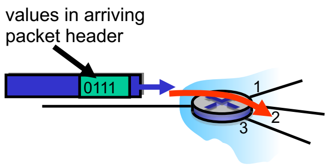
4.5 Routing algorithms
- link state
- distance vector
- hierarchical routing

4.6 Routing in the Internet (intra-AS routing)

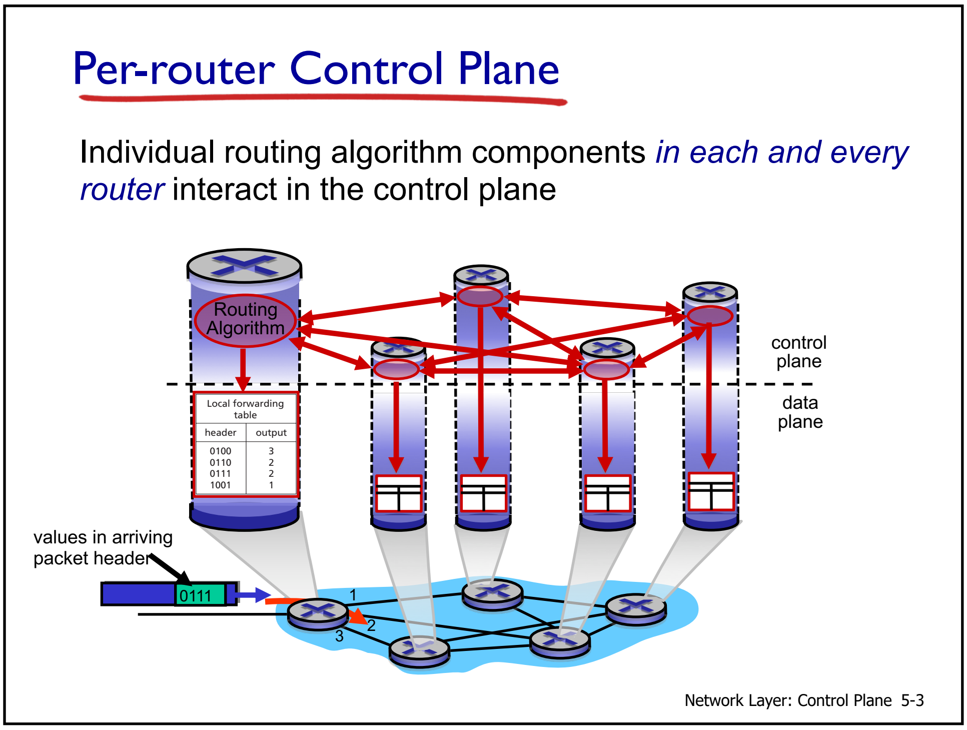
4.7 Generalized Forward and SDN
- match
- action
- OpenFlow (examples of match-plus-action in action)

Network layer: data plane, control plane

Data Plane
- local, per-router function
- determines how datagram arriving on router input port is forwarded to router output port
- forwarding function



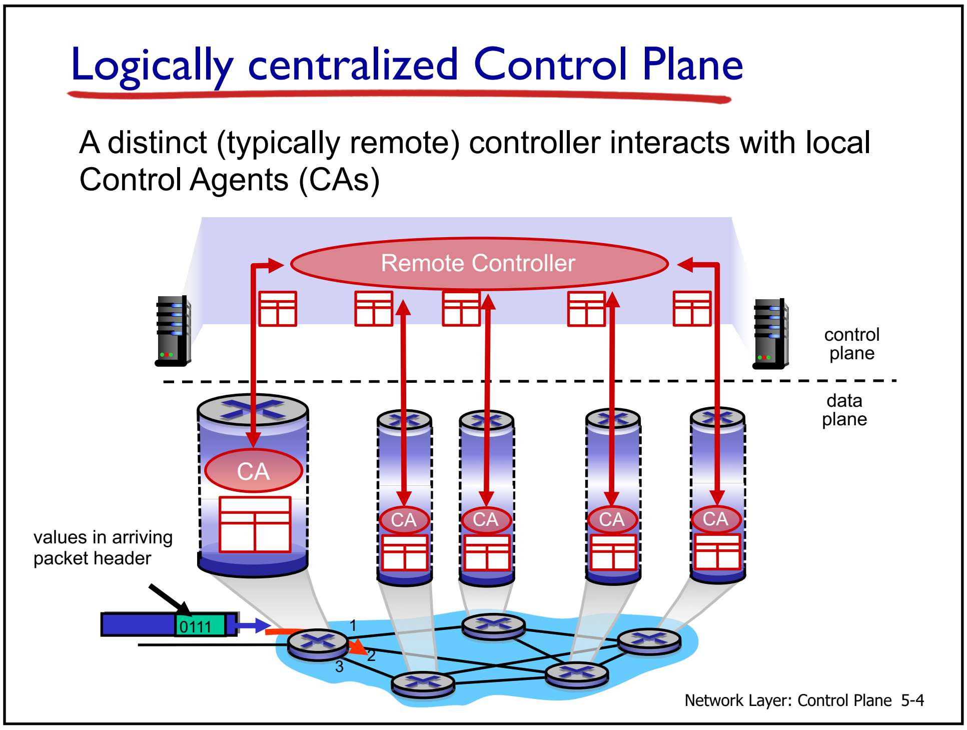
Control Plane
- network-wide logic
- determines how datagram is routed among routers along end-end path from source host to destination host
- two control-plane approaches:
  - *traditional routing algorithms:* implemented in routers
  - *Software-Defined Networking (SDN)*: implemented in (remote) servers



Per-router Control Plane

Individual routing algorithm components *in each and every router* interact in the control plane

header	output
0100	3
0110	2
0111	2
1001	1

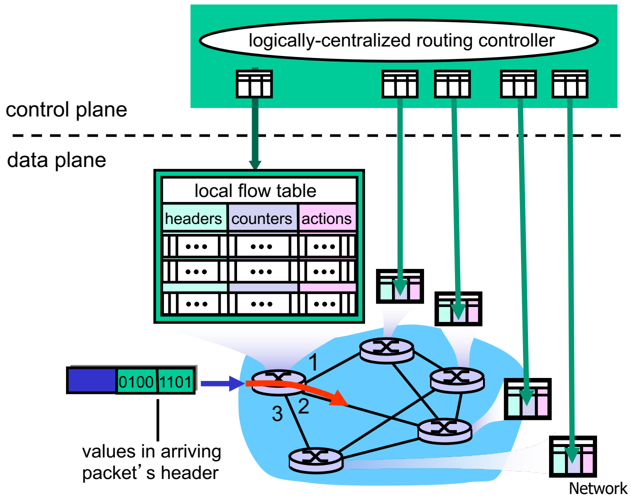


Logically centralized Control Plane

A distinct (typically remote) controller interacts with local Control Agents (CAs)

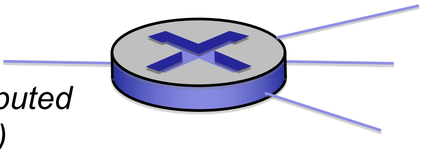
Generalized Forwarding and SDN

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

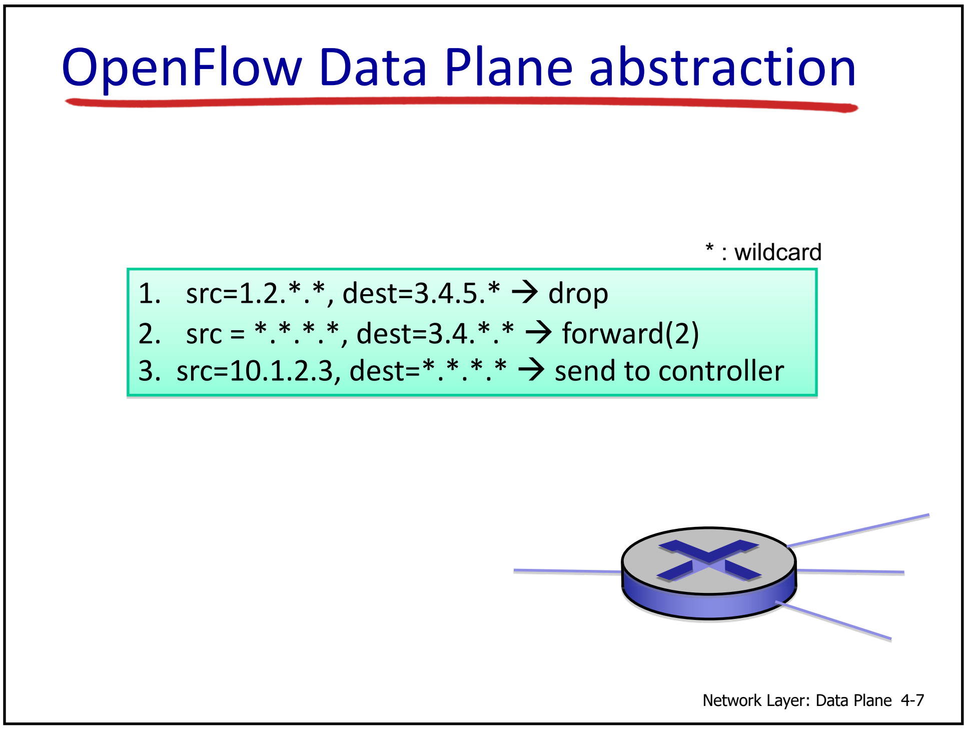

logically-centralized routing controller
control plane
data plane
local flow table
headers
counters
actions
•••
0100
1101
1
2
3
values in arriving packet' s header

OpenFlow Data Plane abstraction

- *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



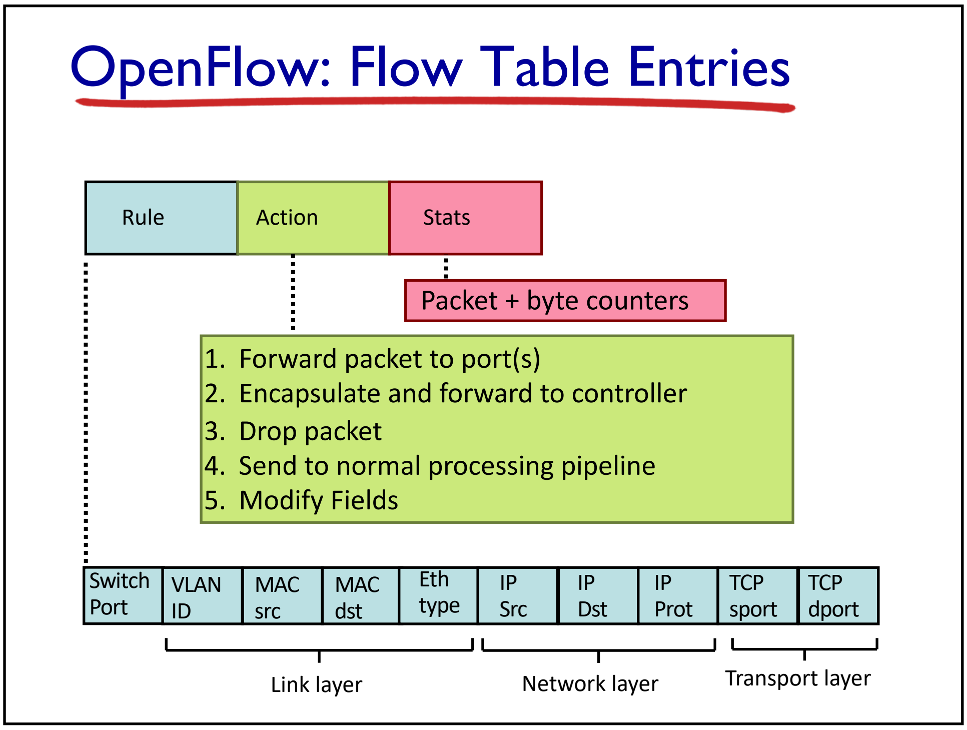
***Flow table** in a router (computed and distributed by controller) define router's match+action rules*



OpenFlow Data Plane abstraction

* : wildcard

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



OpenFlow: Flow Table Entries

Rule	Action	Stats

Packet + byte counters

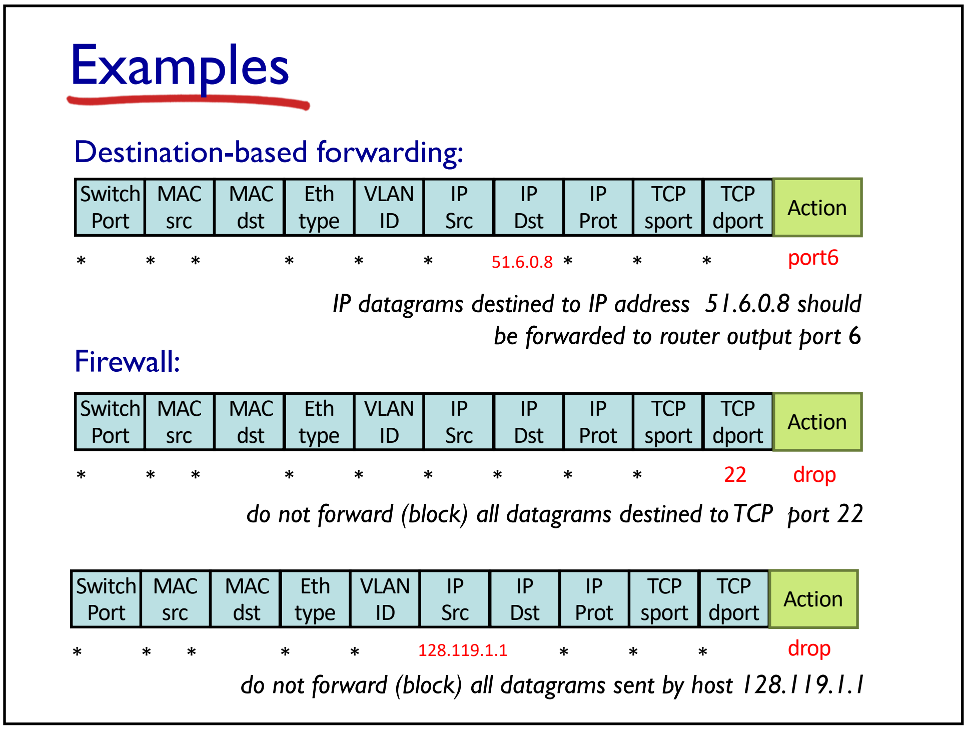
1. Forward packet to port(s)
2. Encapsulate and forward to controller
3. Drop packet
4. Send to normal processing pipeline
5. Modify Fields

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

Link layer

Network layer

Transport layer



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

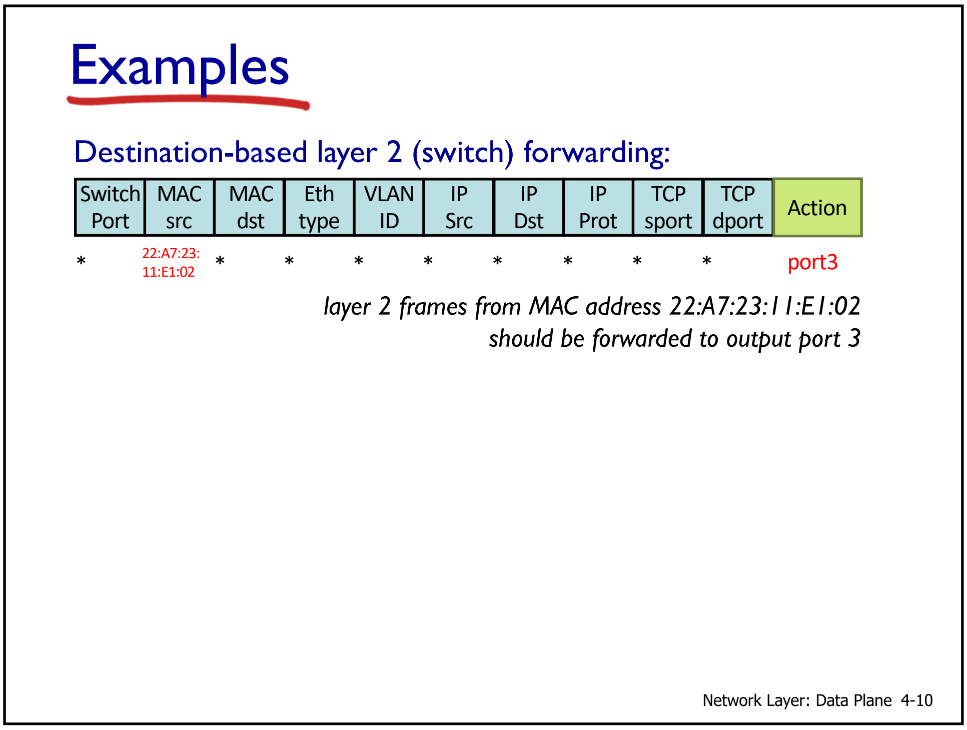
Firewall:

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

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

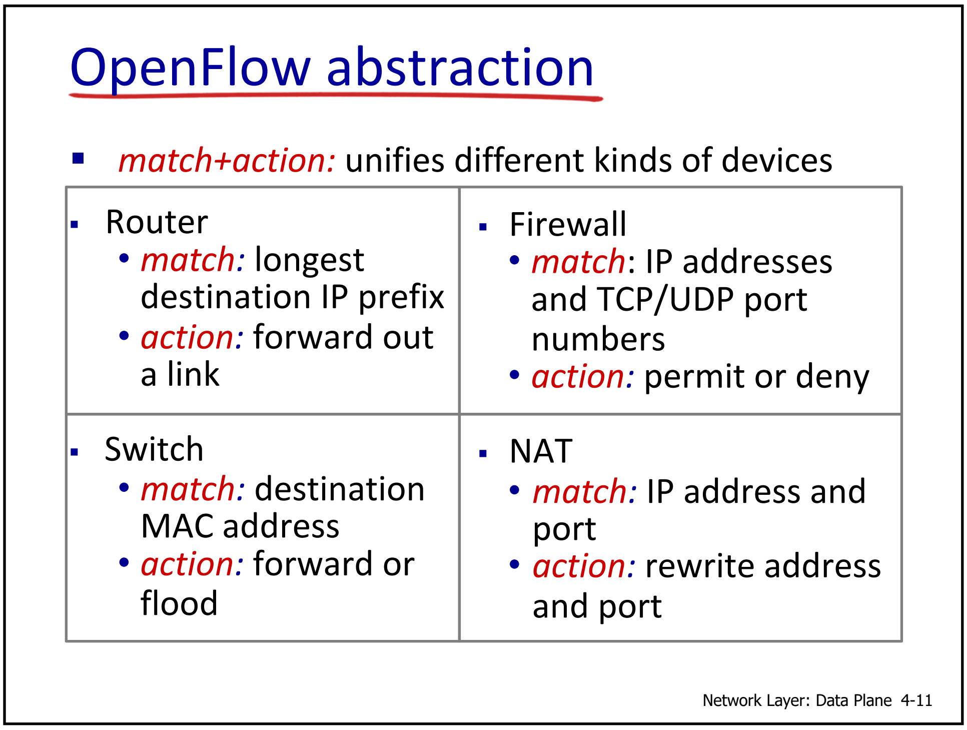


Examples

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

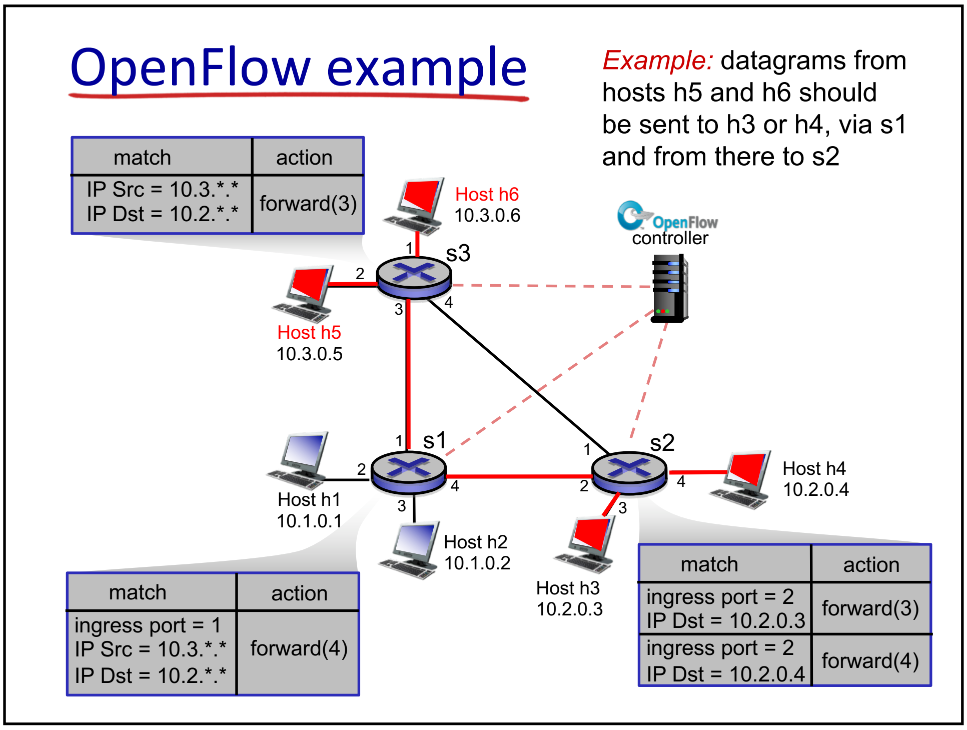
layer 2 frames from MAC address 22:A7:23:11:E1:02 should be forwarded to output port 3



OpenFlow abstraction

- *match+action:* unifies different kinds of devices

Router • *match:* longest destination IP prefix • *action:* forward out a link	Firewall • *match*: IP addresses and TCP/UDP port numbers • *action*: permit or deny
Switch • *match:* destination MAC address • *action:* forward or flood	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

match	action
IP Src = 10.3.*.* IP Dst = 10.2.*.*	forward(3)

match	action
ingress port = 1 IP Src = 10.3.*.* IP Dst = 10.2.*.*	forward(4)

match	action
ingress port = 2 IP Dst = 10.2.0.3	forward(3)
ingress port = 2 IP Dst = 10.2.0.4	forward(4)