SOFTWARE-DEFINED NETWORKING
SESSION II

Block Course – Winter 2016/17

Dr. David Koll

Organizational

Deadline for *all* Exercises: April 30th,

But: please submit all exercises together in a single file together with the presentation slides and paper reviews

Please make sure that you have completed exercise 5 before Friday (there might be some problems on different OSes, so please try out in advance)

Organizational

Hint: exercises are good practice for quiz -> best to do them now (and not in March/April)

Please don't use facilities of software-engineering group on the other side of floor -> you can use those on floor 3

Clarifications from Lecture 1

Production networks: production means that this is a network that is in active use (not a network for producing goods)

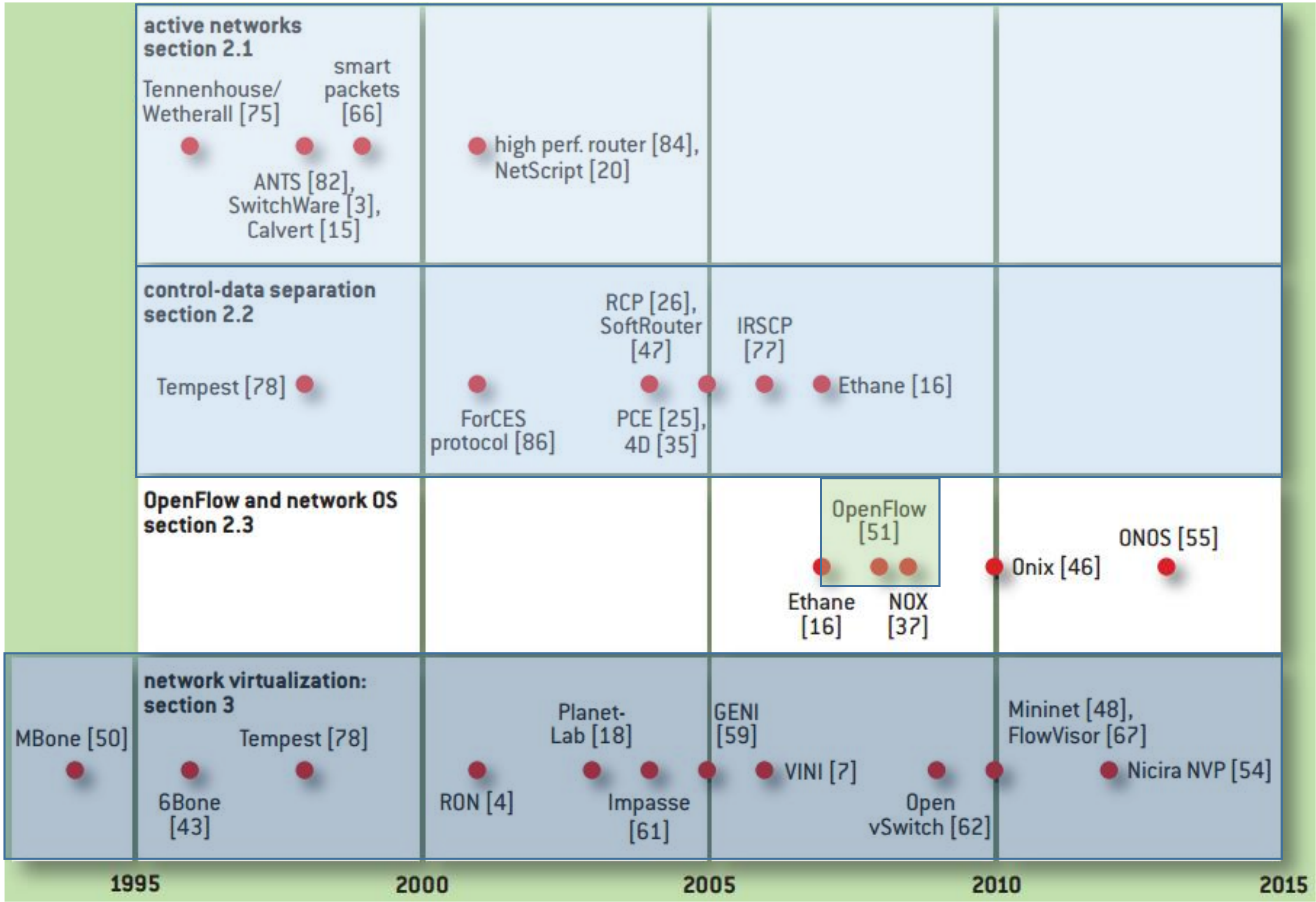
Exercise 5 Issues

MacOS Host Only Adapter -> see mailing list

sudo dhclient <interface without IP> -> you need to specify which interface you want to run dhclient on (usually eth1 or eth0, check with ifconfig –a)

Partly based on slides of Nick McKeown, Scott Shenker, Nick Feamster, and Jennifer Rexford

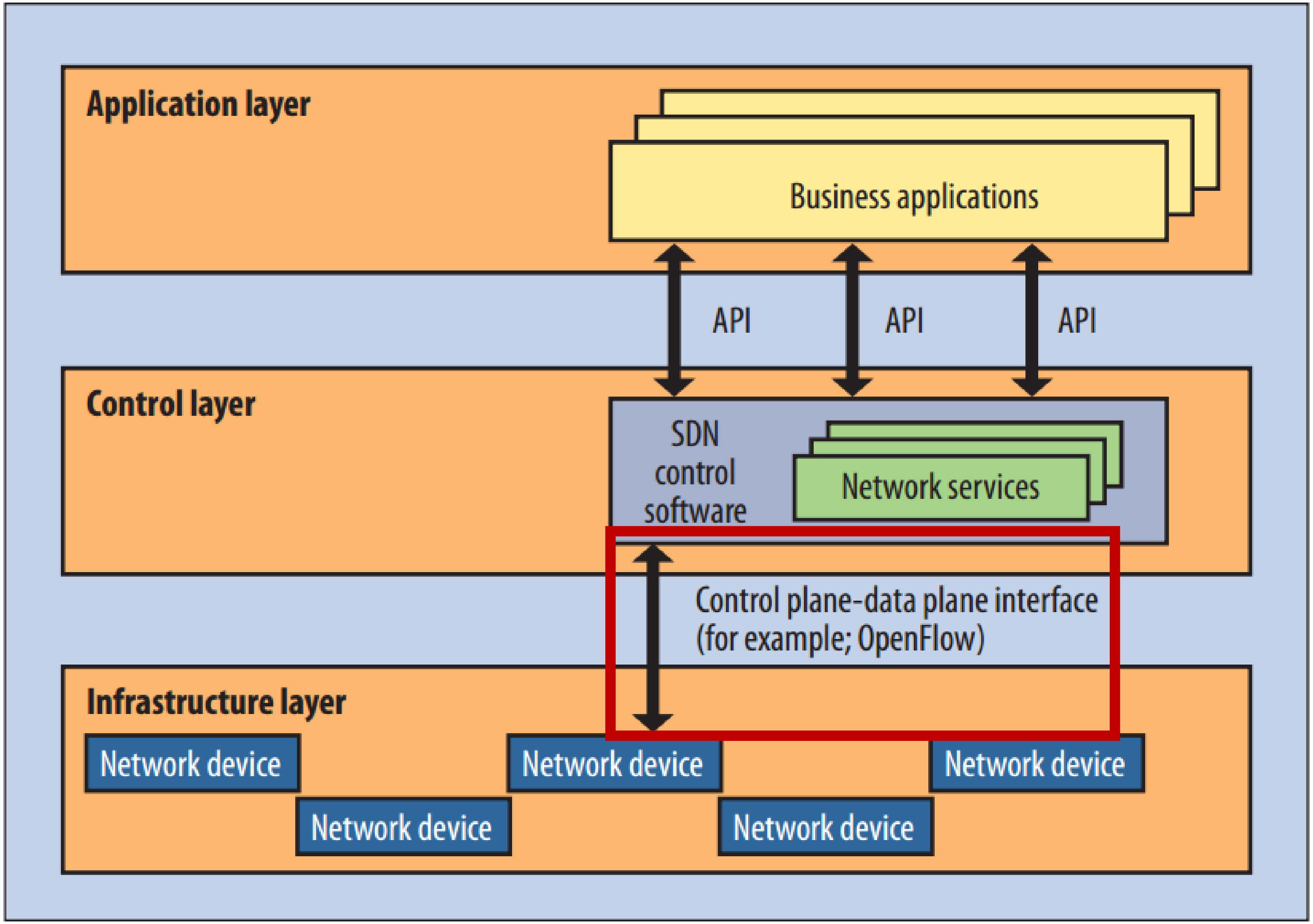
Recap



N. Feamster et al.: „The Road to SDN – An intellectural history of programmable networks" *ACM SIGCOMM Computer Communication Review* 44.2 (2014): 87-98.

OpenFlow – The de-facto standard Southbound interface

This lecture



What is OpenFlow

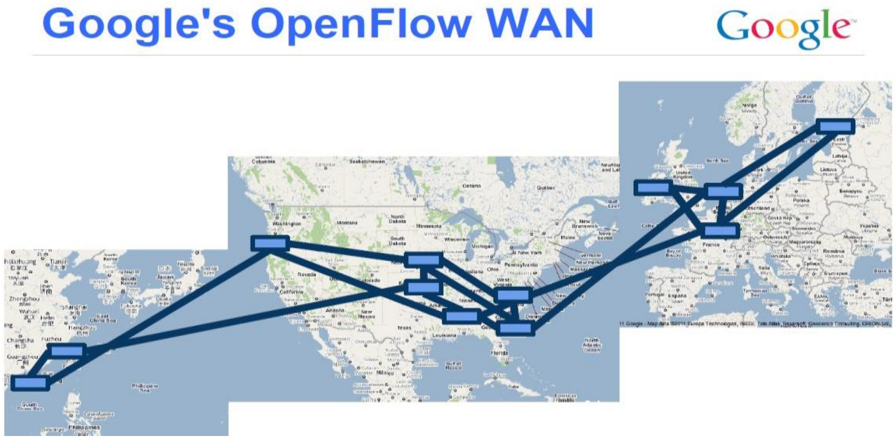
OpenFlow is one implementation of the Southbound interface in SDN

OpenFlow is NOT a synonym for SDN

OpenFlow is NOT THE ONLY Southbound interface

OpenFlow Background

- Rapid Development of OpenFlow Technologies
  - 2012 ONF meeting, Google announced that…
    - Google's G-Scale network is operating using OpenFlow
    - Developed for 2 years (2010~2012.1)
    - Saved CAPEX and OPEX



  - OpenFlow was known as an open standard to test experimental protocols in campus networks
  - OpenFlow → now evolving to Enterprise and Carrier grade SDN technologies
    - Commercial OpenFlow switches and controllers
    - NEC, NTT Data, Nicira , HP, IBM, BigSwitch, Brocade……

Jain, S., et al. "*B4: Experience with a globally-deployed software defined WAN.*" *ACM SIGCOMM CCR*. Vol. 43. No. 4. ACM, 2013.

OpenFlow Version History

Version	Date	Characteristics	Organization
OpenFlow 1.0	2009.12	MAC, IPv4, single flow table	OpenFlow Consortium
OpenFlow 1.1	2011.2	MPLS/tunnel, multiple flow tables, group table	OpenFlow Consortium
OpenFlow 1.2	2011.12	IPv6, Config., extensible match support	ONF
OpenFlow 1.3	2012.9	QoS (meter table)…	ONF
OpenFlow 1.4	2013.10	Optical port monitoring and config (frequency, power)	ONF
OpenFlow 1.5	2014.12	Egress table, pkt. type aware pipeline, flow entry stat trigger	ONF
OpenFlow 2.0	?	?	ONF

Open Networking Foundation

http://opennetworking.org

The founding Consortium

T

Google

facebook

YAHOO!

verizon

NTT Communications

Microsoft

Promoter Members:

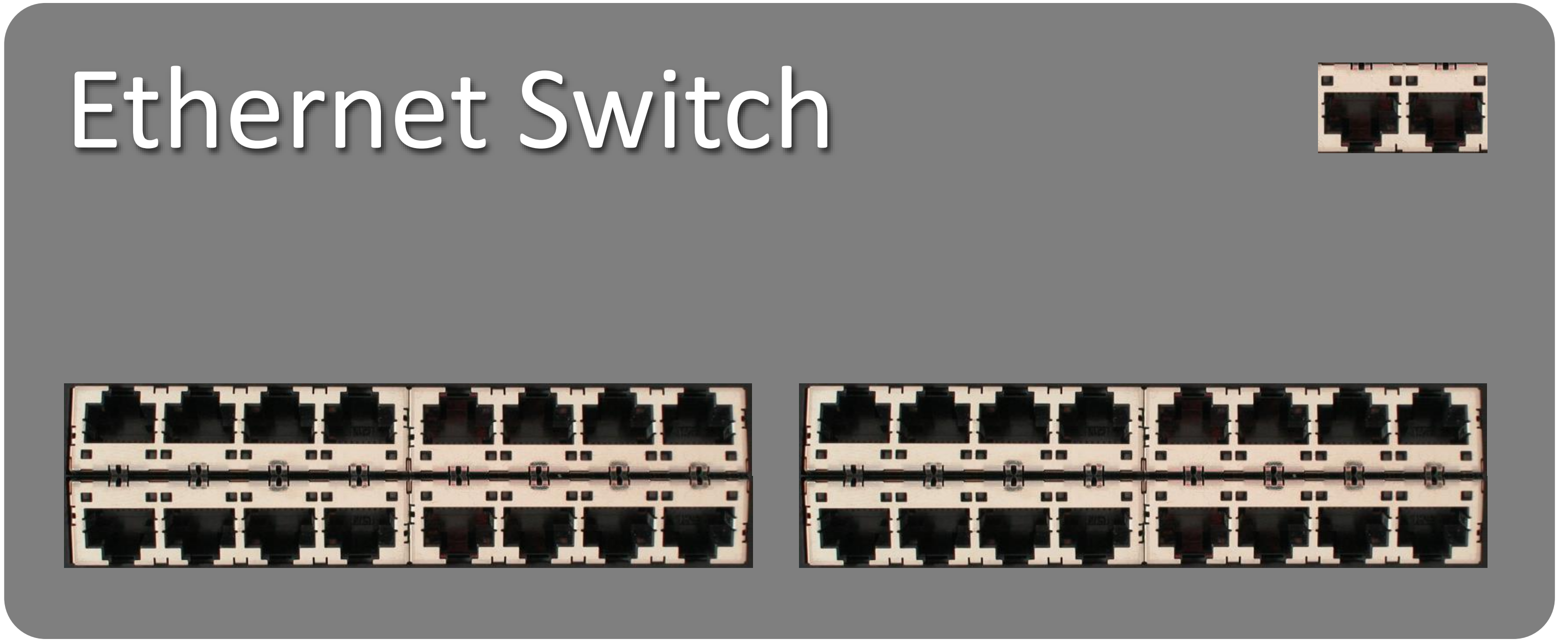
- Operators and service providers
- Make up the board of directors
- Have voting rights

Adopter Members

List of Members:

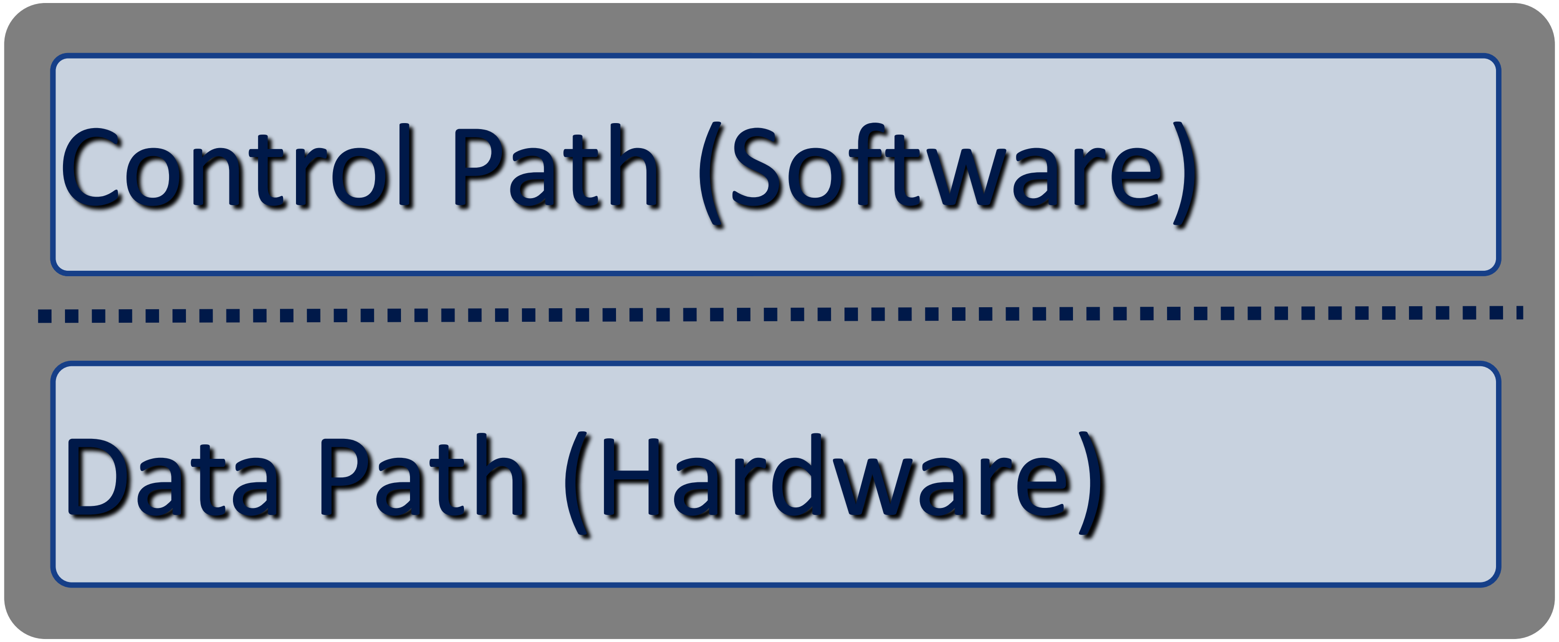
- Big Switch Networks
- Broadcom
- Brocade
- Ciena
- Cisco
- Citrix
- Comcast
- CompTIA
- Cyan
- Dell
- Elbrys
- Ericsson
- ETRI
- Extreme Networks
- EZchip
- Force10Networks
- Fujitsu
- Hitachi
- HP
- Huawei
- IBM
- Infoblox
- Intel
- IP Infusion
- Ixia
- Juniper Networks
- Korea Telecom
- LineRate Systems
- LSI
- Marvell
- Mellanox
- Metaswitch Networks
- Midokura
- NEC
- Netgear
- Netronome
- Nicira Networks
- Nokia Siemens Networks
- Plexxi Inc.
- Pronto Systems
- Radware
- Riverbed Technology
- Samsung
- Spirent
- Tencent
- Texas Instruments
- Vello Systems
- VMware
- ZTE Corporation

Recall: What is SDN?



Ethernet Switch

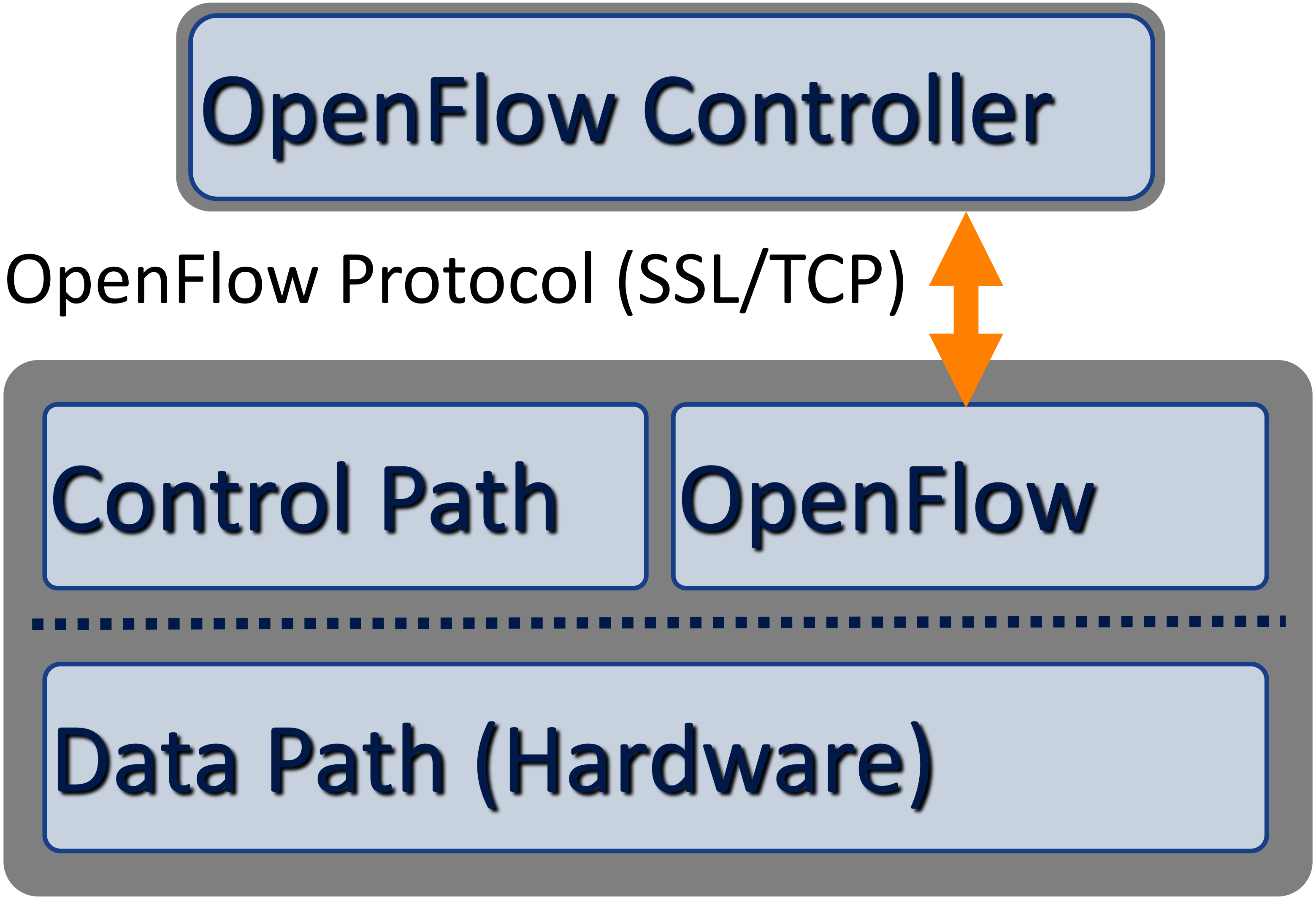
Separation of Control and Data Plane



Control Path (Software)

Data Path (Hardware)

How do we control a switch then?



OpenFlow Controller

OpenFlow Protocol (SSL/TCP)

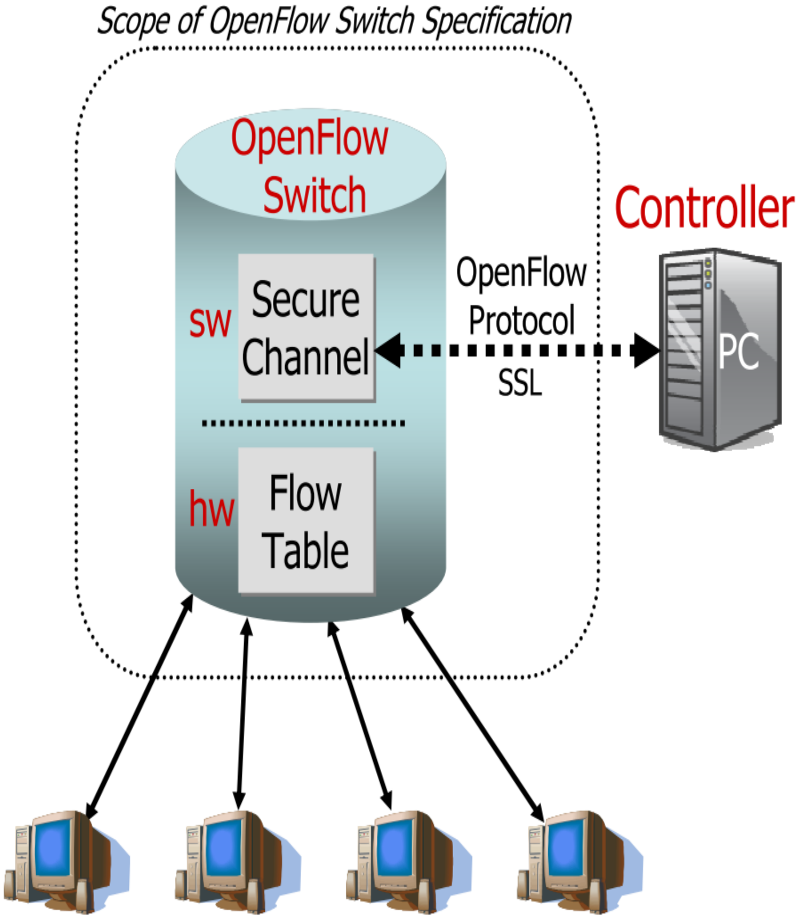
Control Path

OpenFlow

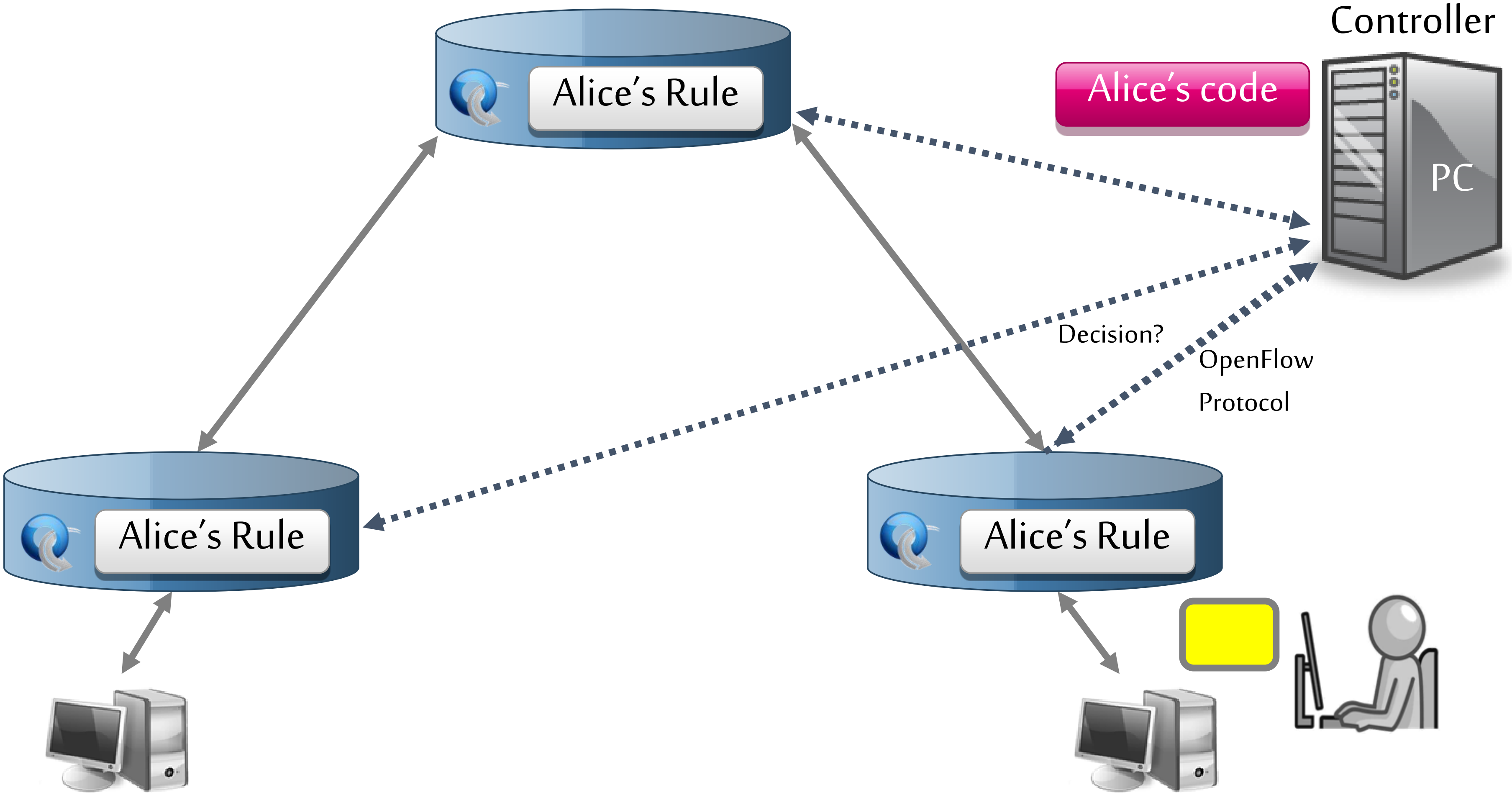
Data Path (Hardware)

Components of OpenFlow Network

- Controller
  - OpenFlow protocol messages
  - Controlled channel
  - Processing
    - Pipeline Processing
    - Packet Matching
    - Instructions & Action Set
- OpenFlow switch
  - Secure Channel (SC)
  - Flow Table
    - Flow entry



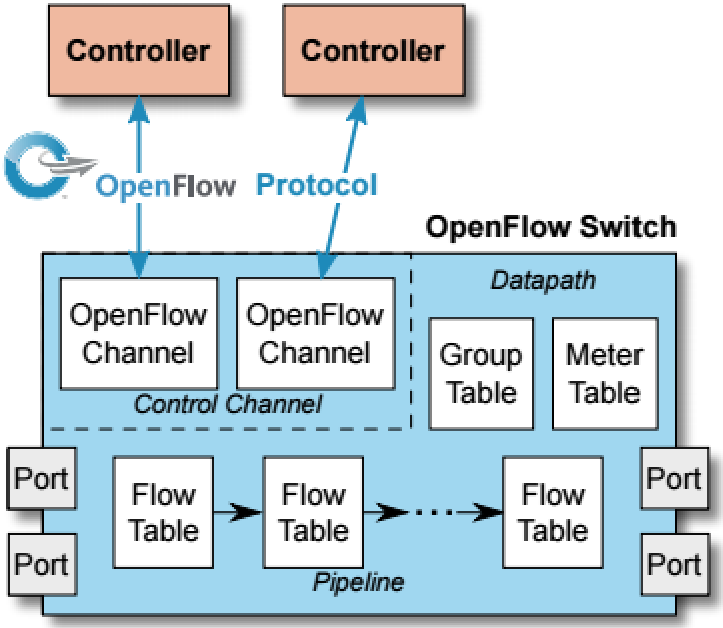
OpenFlow Usage



OpenFlow offloads control intelligence to a remote software

OpenFlow

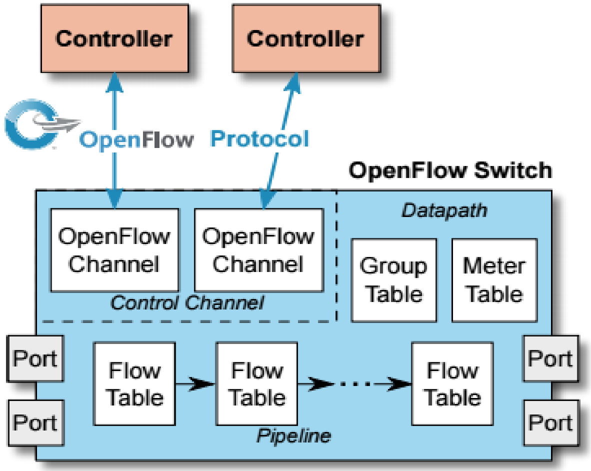
- Communication between the controller and the network devices (i.e., switches)



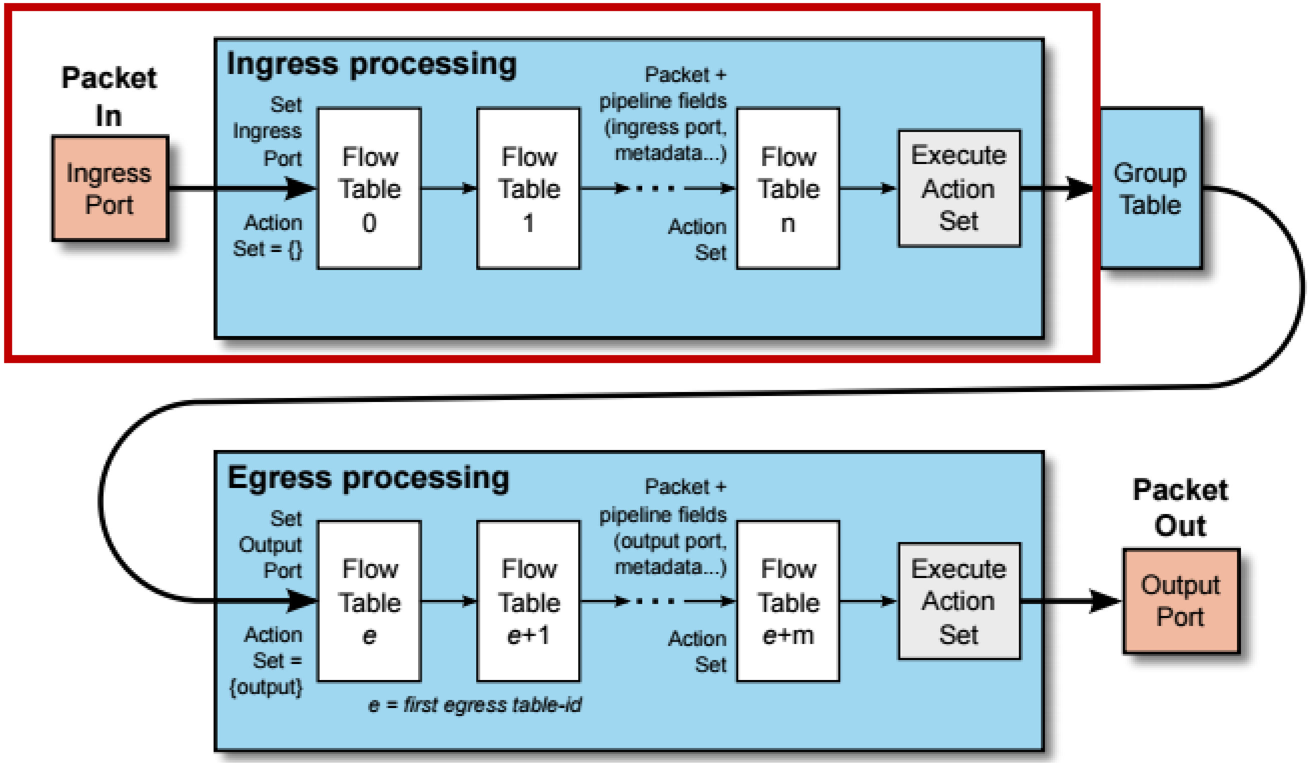
All figures extracted from the specification: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.5.1.pdf (April 2015)

OpenFlow

- Main components: *Flow, Group and Meter Tables*
  - Controller can manipulate these tables via the OpenFlow protocol (*add, update, delete*)
  - Flow Table: reactively or proactively defines how incoming/outgoing packets are forwarded
  - Group Table: additional processing (e.g., multicast)
  - Meter Table: QoS implementation

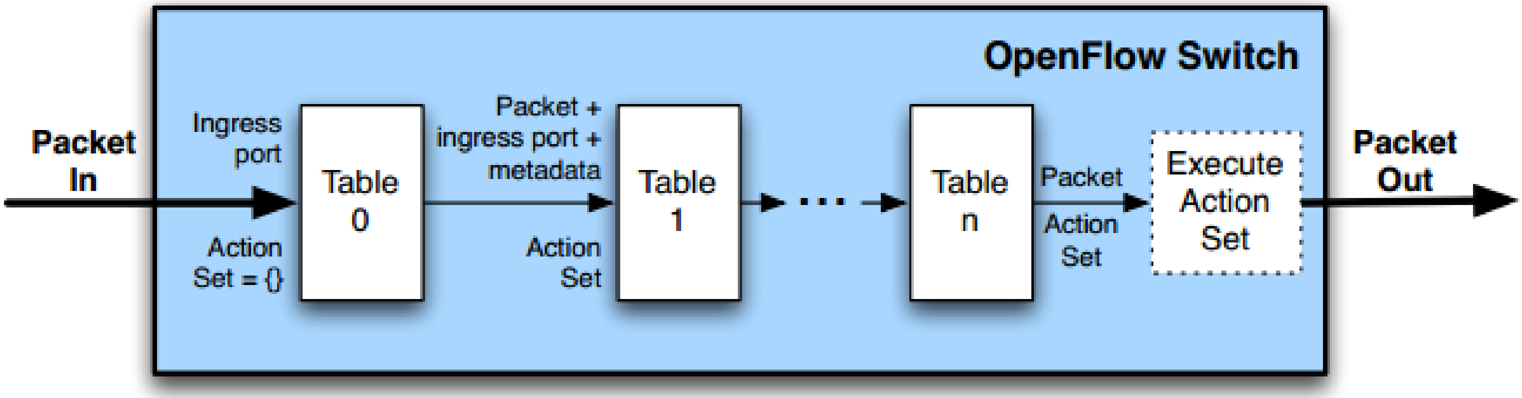


OpenFlow – Packet Processing



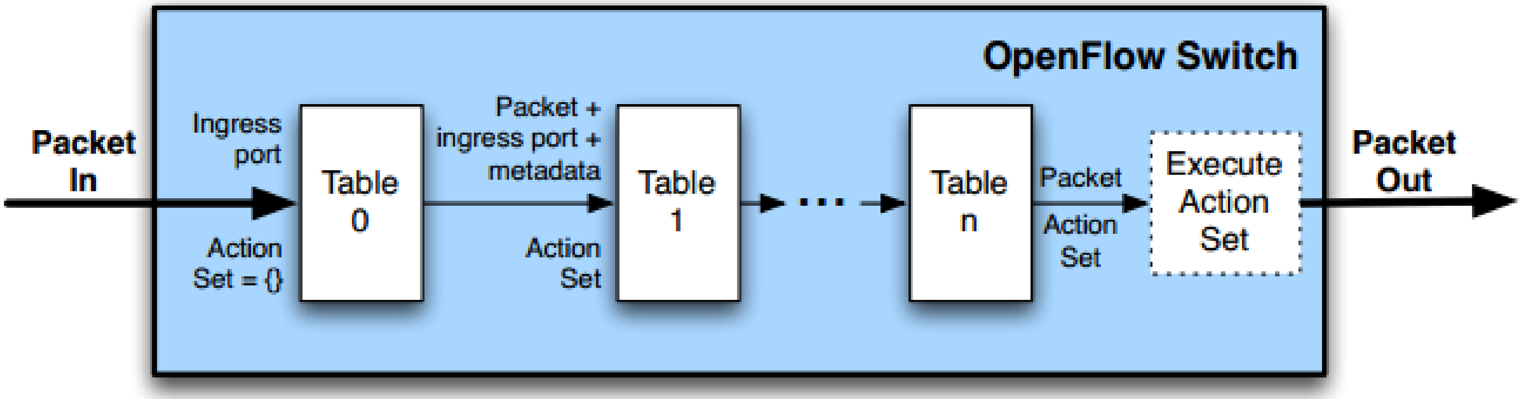
OpenFlow – Switches

- Two different versions of an OpenFlow Switch
  - *OF-only* (packets can only be processed by OF tables) and *OF-hybrid* (allow optional normal Ethernet handling (see CN lecture))
- OF-only: all packets go through a *pipeline*
  - Each pipeline contains one or multiple flow tables with each containing one or multiple *flow entries*



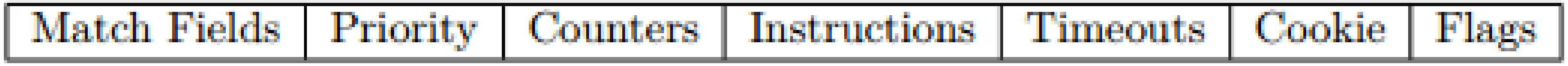
OpenFlow – Switches

- Incoming packets are matched against Table 0 first
- Find highest priority match and execute instructions (might be a Goto-Table instruction)
- Goto: Only possible forward



OpenFlow – Flow Table Entries

- Flow Table entry structure:

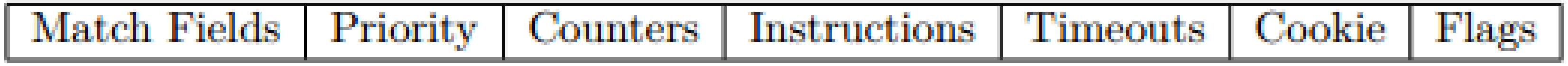


Match Fields	Priority	Counters	Instructions	Timeouts	Cookie	Flags

- Match fields: where matching applies (i.e., ingress port, packet headers, etc.)
- Priority: matching precedence of flow entry
- Counters: update on packet match with entry
- Instructions: what to do with the packet
- Timeout: max idle time of flow before ending

OpenFlow – Flow Table Entries

- Flow Table entry structure:



Match Fields	Priority	Counters	Instructions	Timeouts	Cookie	Flags

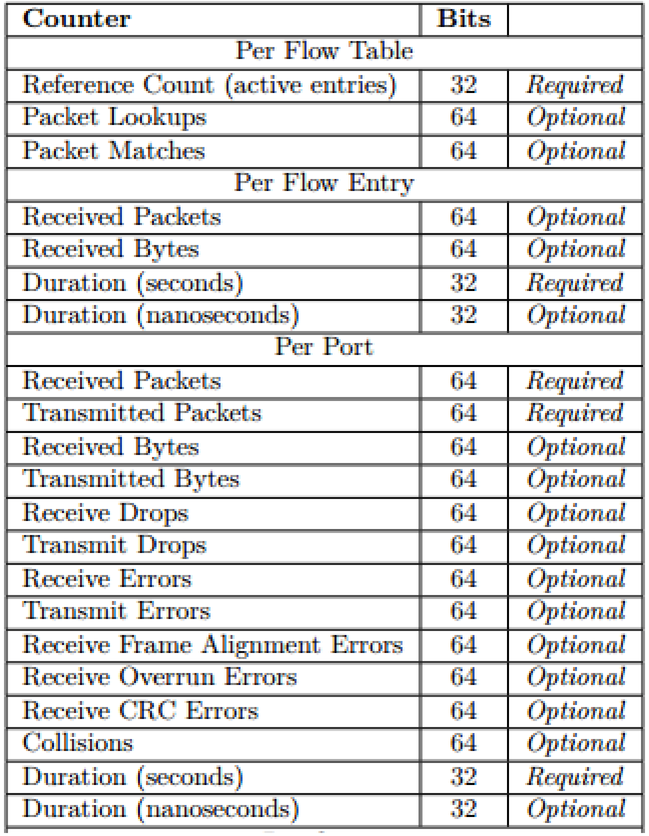
- Match fields: where matching applies (i.e., ingress port, packet (IP, eth) headers, etc.)
    - A flow entry with all match fields as wildcard and priority 0: *table miss* entry

- Priority: matching precedence of flow entry

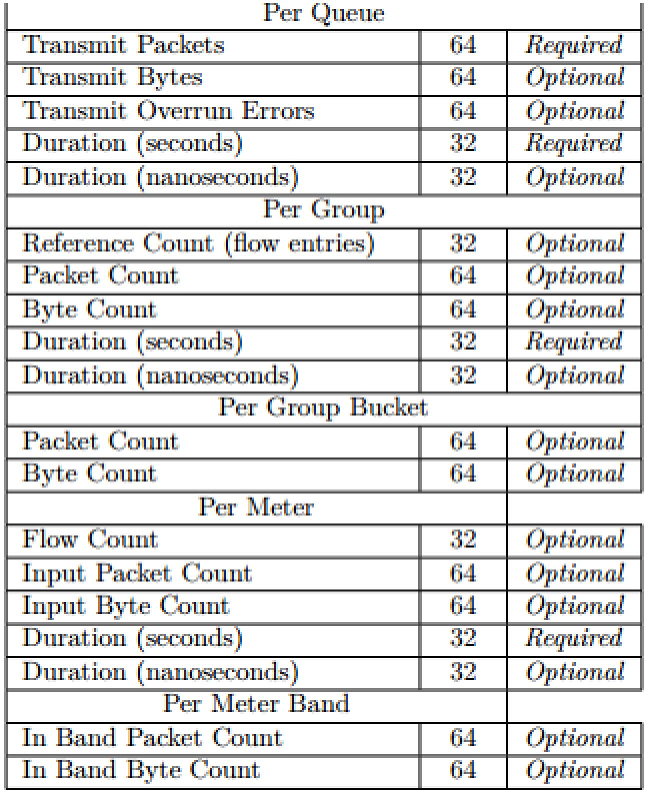
OpenFlow – Flow Table with no match

- If no match in table: *table miss*
- Handling: depends on table configuration – might be *drop packet, forward to other table, forward to controller*
- Forward to controller allows to set up a flow entry (i.e., at the beginning of a flow)

OpenFlow Flow Entries – Counters

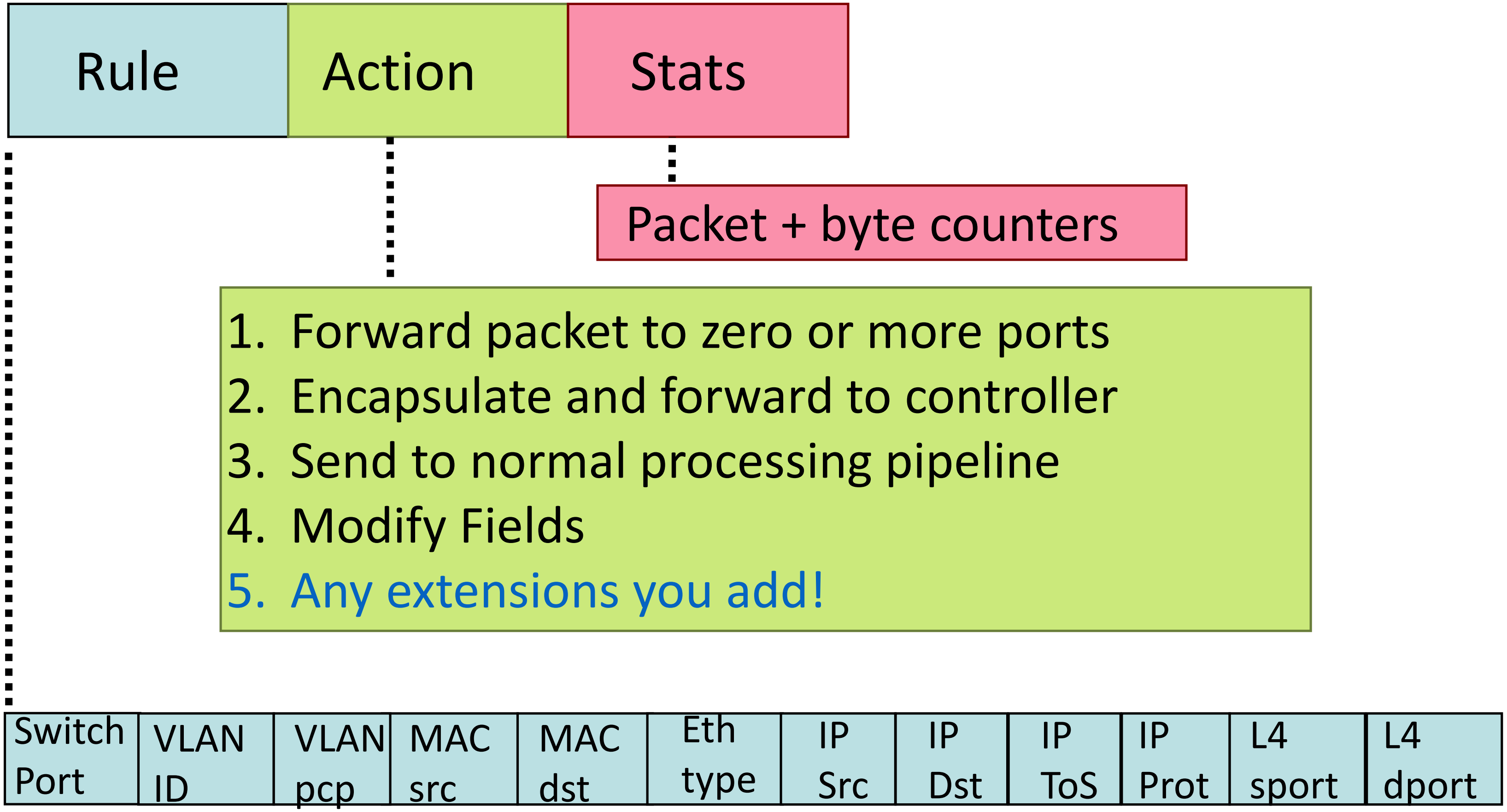


Counter	Bits	
Per Flow Table		
Reference Count (active entries)	32	*Required*
Packet Lookups	64	*Optional*
Packet Matches	64	*Optional*
Per Flow Entry		
Received Packets	64	*Optional*
Received Bytes	64	*Optional*
Duration (seconds)	32	*Required*
Duration (nanoseconds)	32	*Optional*
Per Port		
Received Packets	64	*Required*
Transmitted Packets	64	*Required*
Received Bytes	64	*Optional*
Transmitted Bytes	64	*Optional*
Receive Drops	64	*Optional*
Transmit Drops	64	*Optional*
Receive Errors	64	*Optional*
Transmit Errors	64	*Optional*
Receive Frame Alignment Errors	64	*Optional*
Receive Overrun Errors	64	*Optional*
Receive CRC Errors	64	*Optional*
Collisions	64	*Optional*
Duration (seconds)	32	*Required*
Duration (nanoseconds)	32	*Optional*



Per Queue		
Transmit Packets	64	*Required*
Transmit Bytes	64	*Optional*
Transmit Overrun Errors	64	*Optional*
Duration (seconds)	32	*Required*
Duration (nanoseconds)	32	*Optional*
Per Group		
Reference Count (flow entries)	32	*Optional*
Packet Count	64	*Optional*
Byte Count	64	*Optional*
Duration (seconds)	32	*Required*
Duration (nanoseconds)	32	*Optional*
Per Group Bucket		
Packet Count	64	*Optional*
Byte Count	64	*Optional*
Per Meter		
Flow Count	32	*Optional*
Input Packet Count	64	*Optional*
Input Byte Count	64	*Optional*
Duration (seconds)	32	*Required*
Duration (nanoseconds)	32	*Optional*
Per Meter Band		
In Band Packet Count	64	*Optional*
In Band Byte Count	64	*Optional*

OpenFlow - Instructions



Rule

Action

Stats

Packet + byte counters

1. Forward packet to zero or more ports
2. Encapsulate and forward to controller
3. Send to normal processing pipeline
4. Modify Fields
5. Any extensions you add!

Switch Port	VLAN ID	VLAN pcp	MAC src	MAC dst	Eth type	IP Src	IP Dst	IP ToS	IP Prot	L4 sport	L4 dport

+ mask what fields to match

Examples

Switching

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	00:1f:..	*	*	*	*	*	*	*	port6

Flow Switching

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
port3	00:20..	00:1f..	0800	vlan1	1.2.3.4	5.6.7.8	4	17264	80	port6

Firewall

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

Examples

Routing

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

VLAN Switching

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	00:1f..	*	vlan1	*	*	*	*	*	port6, port7, port9

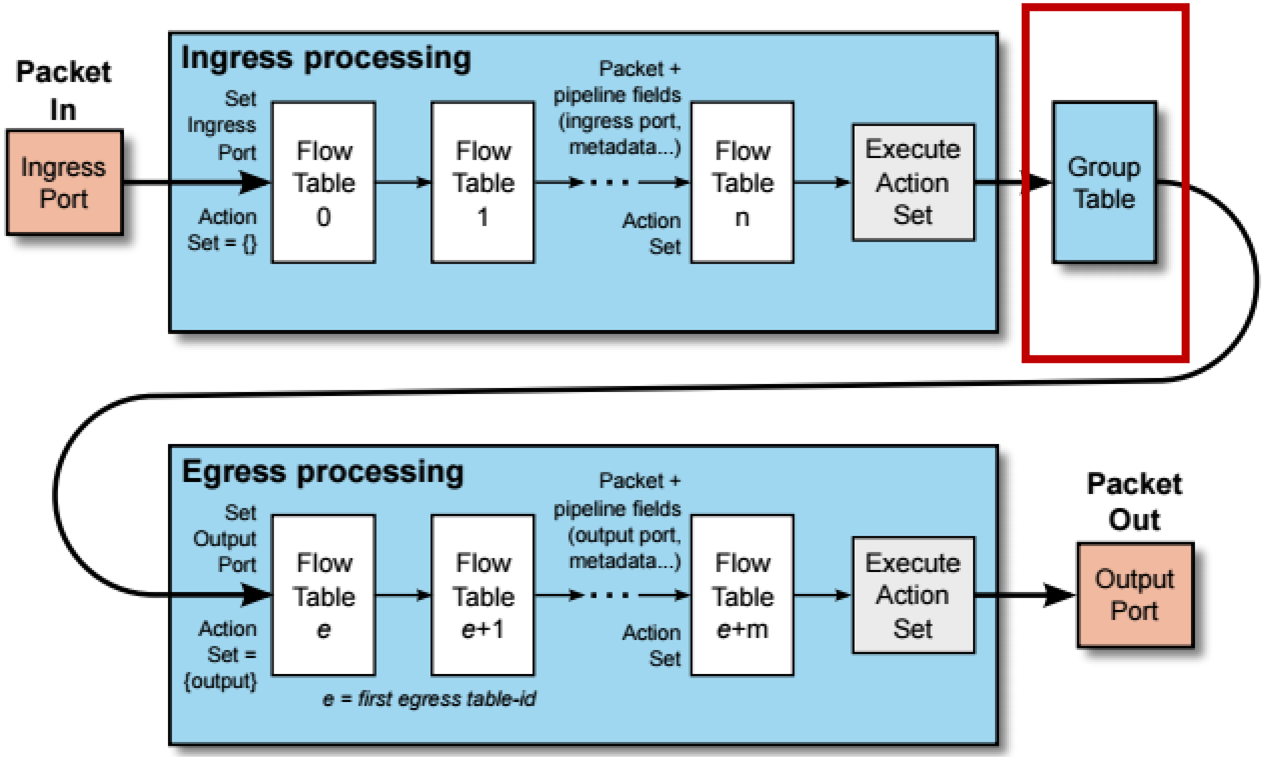
Multicast

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	9.8.7.4.	*	*	443	*	Group 1

OpenFlow – Flow Entry Removal

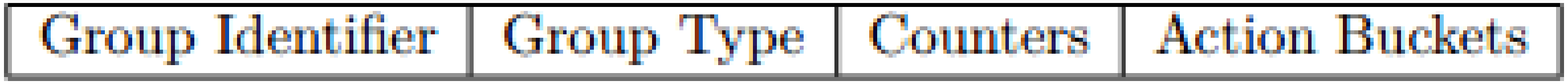
- *Request of controller*
    - Active issueing of a OF delete command
    - e.g., change in routing
- *Idle timeout*
    - flow entry expires if it is not matched for a specified period of time (usually seconds)
- *Hard timeout*
    - flow entry has a pre-determined maximum TTL
    - Hard timeout > idle timeout

OpenFlow – Packet Processing



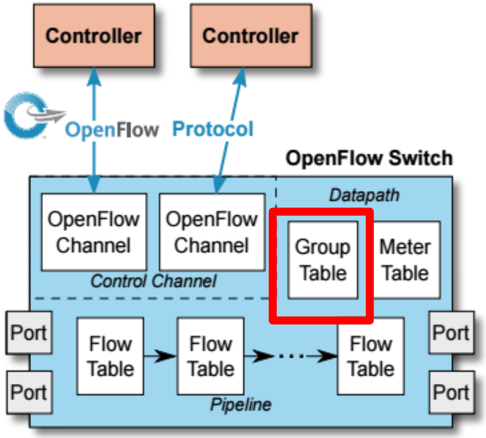
OpenFlow – Switches

- Group Table entry structure:



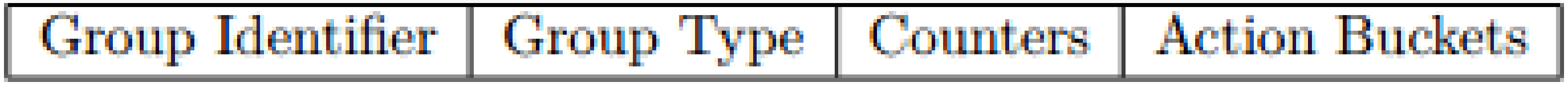
Group Identifier	Group Type	Counters	Action Buckets

- Group Identifier: 32-bit ID to uniquely define group on the switch (locally)
- Group Type: *indirect/all/fast failover/select*
  - Specifies which *action bucket* is executed
- Counters: update on packet processed
- Action Buckets: ordered list of buckets,
- each containing a *set* of instructions



OpenFlow – Switches

- Group Table entry structure:



Group Identifier	Group Type	Counters	Action Buckets

- Group Tables allow for more complex forwarding
  - E.g., multicast: use *all* group type to execute all action buckets (packet will be cloned for each bucket, and then forwarded through the instruction set)

OpenFlow Group Table

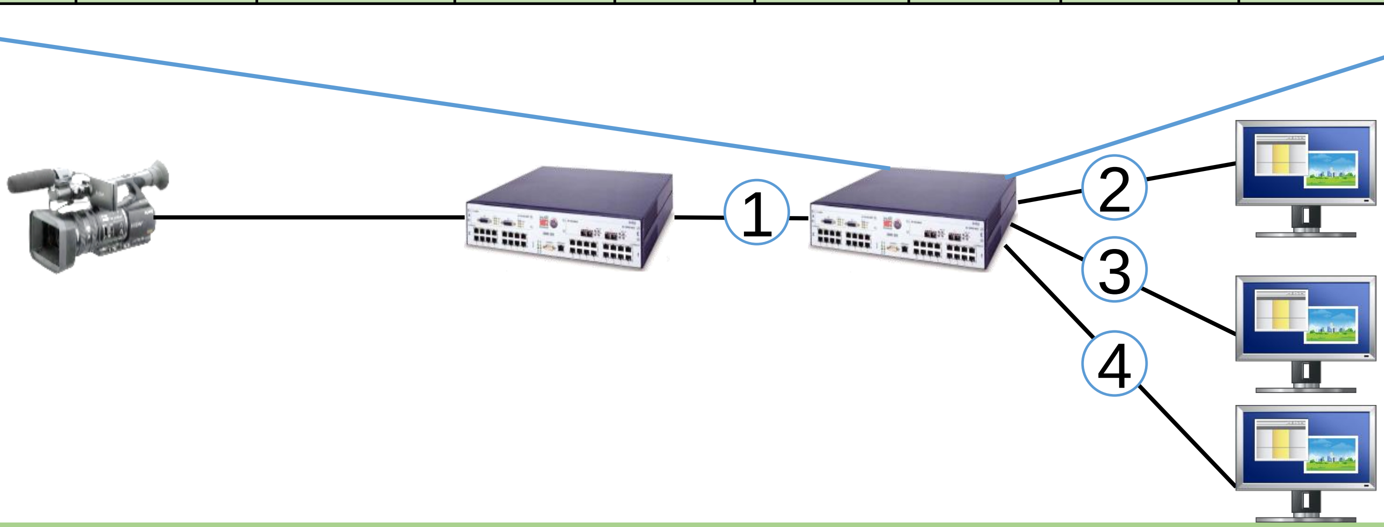
- Multicast
  - Type=all

Group Table

Group ID	Group Type	Counter	Action Buckets
100	All	999	Port2, Port3, Port4

Flow Table

Switch Port	MAC src	MAC dst	Ether Type	VLAN ID	Src IP	Dst IP	Proto No.	TCP S Port	TCP D Port	Action
*	*	00:FF:..	*	*	*	*	*	*	*	Port 6
Port 1	*	*	0800	*	224…	224…	4	4566	6633	Group 100



OpenFlow Group Table

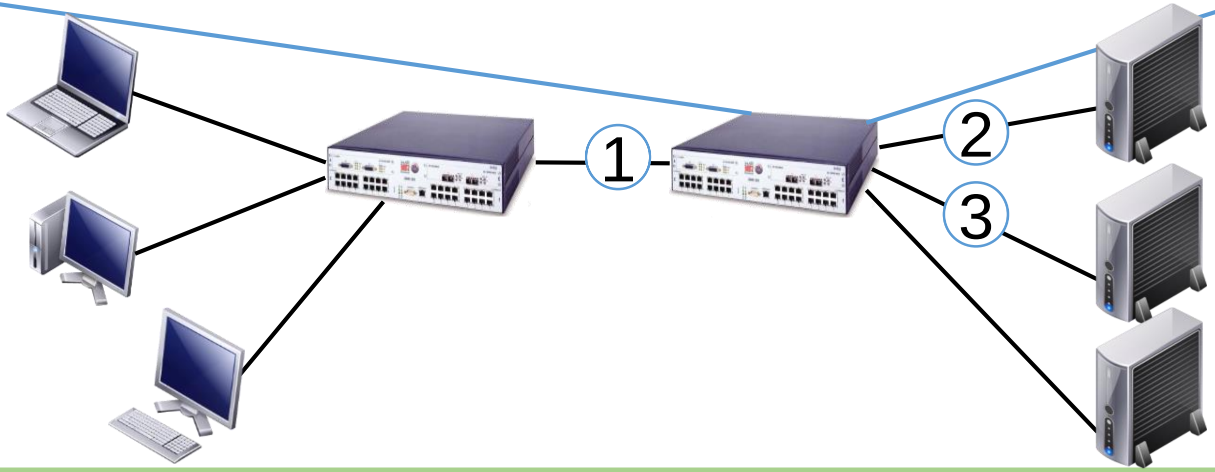
- Load Balancing
  - Type=select

Group Table

Group ID	Group Type	Counter	Action Buckets
100	Select	999	Port2, Port3

Flow Table

Switch Port	MAC src	MAC dst	Ether Type	VLAN ID	Src IP	Dst IP	Proto No.	TCP S Port	TCP D Port	Action
*	*	00:FF:..	*	*	*	*	*	*	*	Port 1
Port 1	*	*	0800	*	1.2.3 …	*	4	*	80	Group 100



OpenFlow Group Table

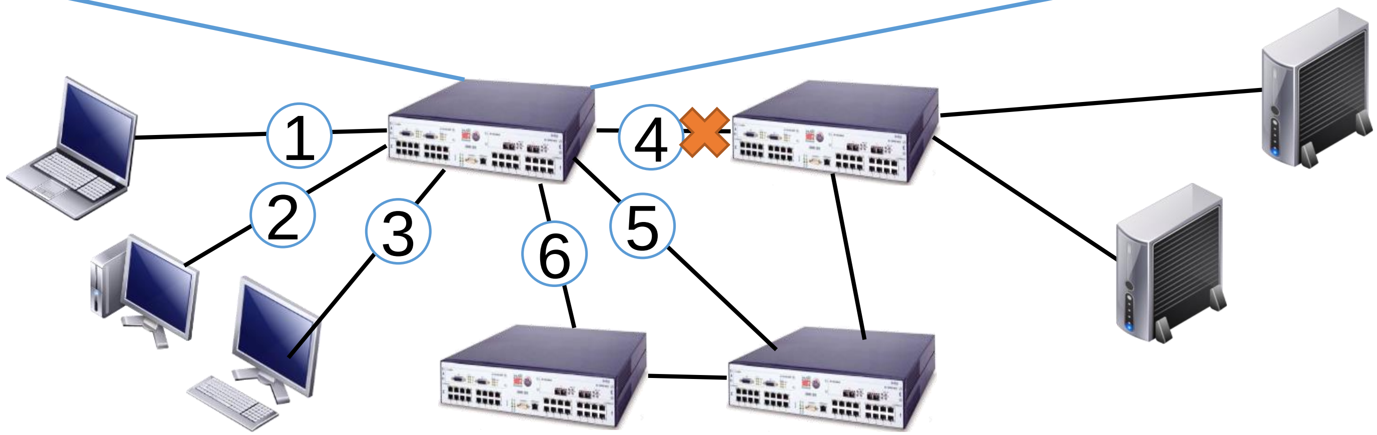
- Fast Failover
  - Type=fast-failover (ff)

Group Table

Group ID	Group Type	Counter	Action Buckets
100	Fast-failover	777	Port4, Port5, Port6

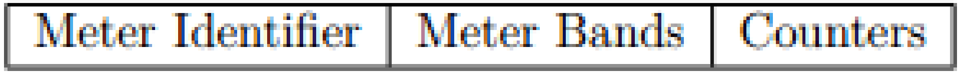
Flow Table

Switch Port	MAC src	MAC dst	Ether Type	VLAN ID	Src IP	Dst IP	Proto No.	TCP S Port	TCP D Port	Action
Port 1	*	*	*	*	1.2.2	*	*	*	*	Port 7
Port 1	00:FF …	*	0800	*	1.2.3 …	11.1…	*	*	*	Group 100



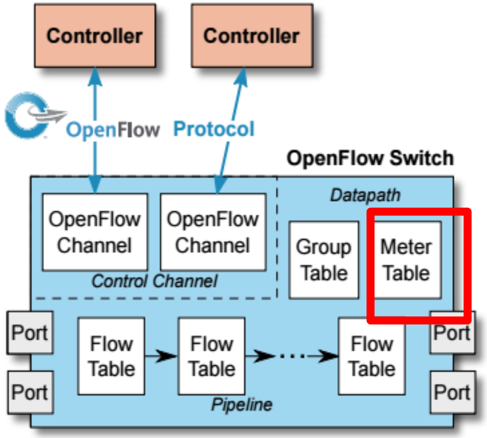
OpenFlow – Switches

- Meter Table entry structure:



Meter Identifier	Meter Bands	Counters

- Meter Identifier: 32-bit ID to uniquely define meter on the switch (locally)
- Meter Bands: an unordered list of meter bands, each specifying rate of band
- Counters: update on packet processed



OpenFlow Meter Table

- Meter Table (ver 1.3)
  - Counts packet rate of a matched flow
  - QoS control → Rate-limit, DiffServ …

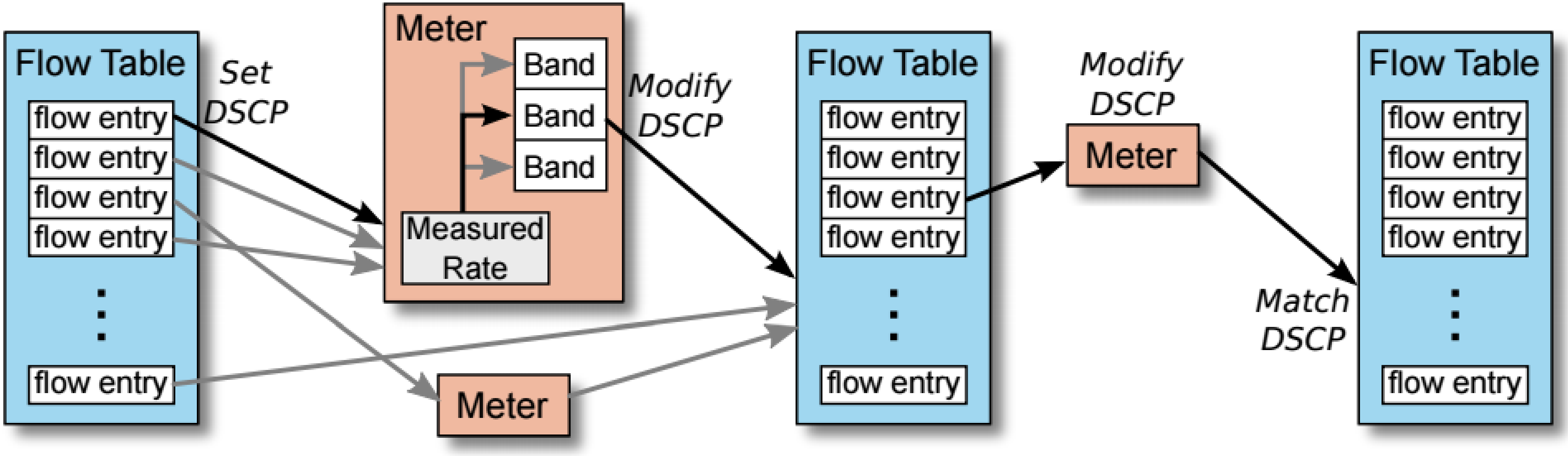
Meter Table

Meter ID	Band Type	Rate	Counter
100	Drop	1000 kbps	1000

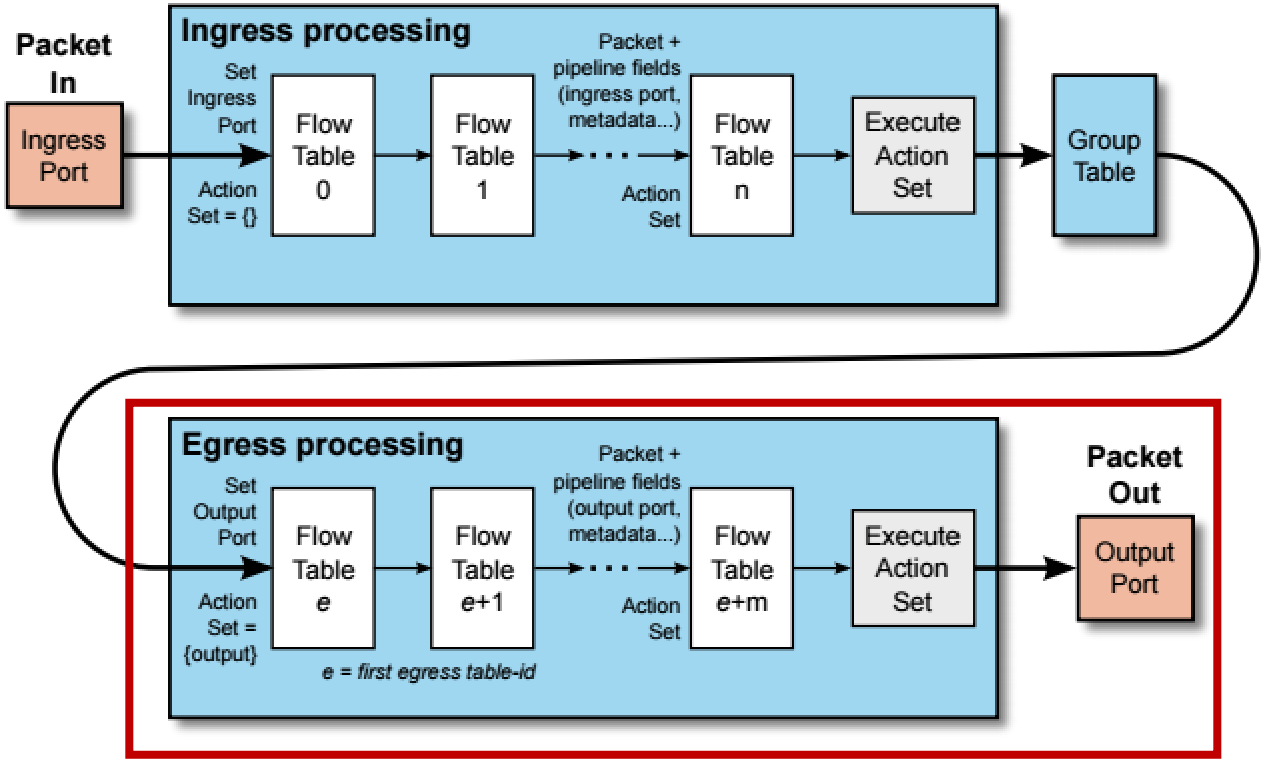
Flow Table

Switch Port	MAC src	MAC dst	Ether Type	Src IP	Dst IP	Proto No.	TCP S Port	TCP D Port	Inst. Meter	Action
Port 1	*	*	*	1.2.2	*	*	*	*	N/A	Port 7
Port 1	00:FF …	*	0800	1.2.3 …	11.1…	*	*	*	Meter 100	Port 2

OpenFlow – Switches



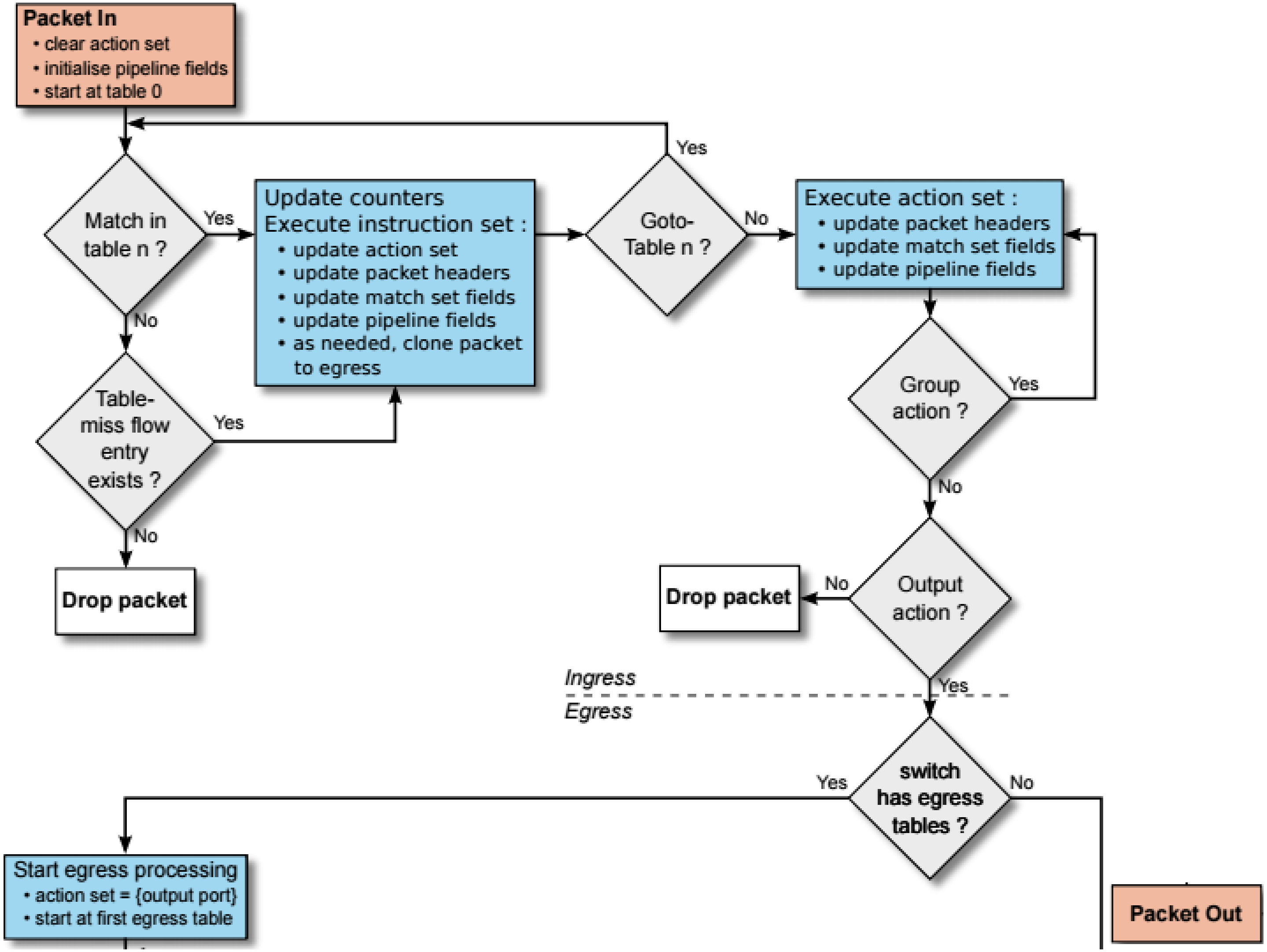
OpenFlow – Packet Processing



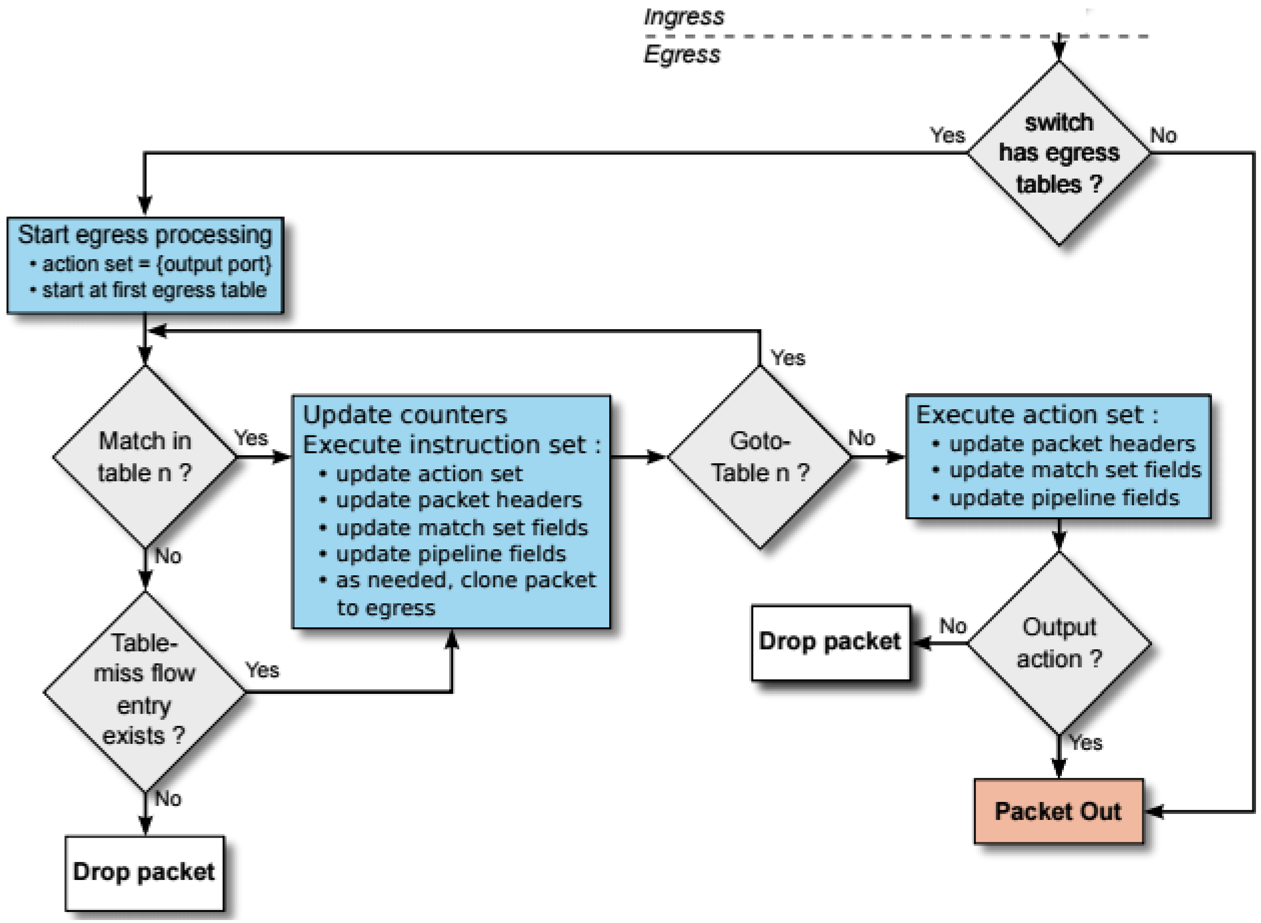
OpenFlow – Egress Processing

- In general: same as ingress processing
  - Matching, instruction execution, table miss, etc.
  - Note: ingress tables can not „goto“ to egress tables
- Differences:
  - Beginning of ingress: action set empty
  - Beginning of egress: output port set, not changeable!
  - Use of group tables not specified
- Egress processing:
  - Used to do processing based on output port
  - E.g.: link connected to that port may require encapsulation
  - Packet cloning for further use

OpenFlow - Matching



OpenFlow - Matching



switch has egress tables ?

Start egress processing
- action set = {output port}
- start at first egress table

Match in table n ?

Update counters
Execute instruction set :
- update action set
- update packet headers
- update match set fields
- update pipeline fields
- as needed, clone packet to egress

Goto-Table n ?

Execute action set :
- update packet headers
- update match set fields
- update pipeline fields

Table-miss flow entry exists ?

Drop packet

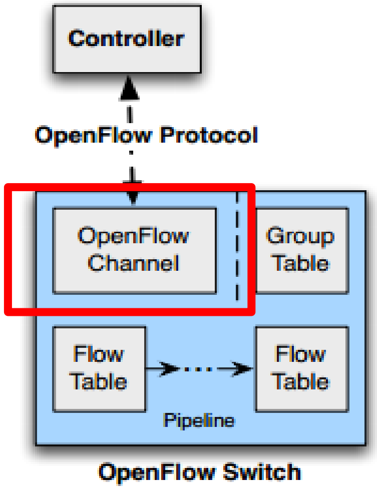
Output action ?

Drop packet

Packet Out

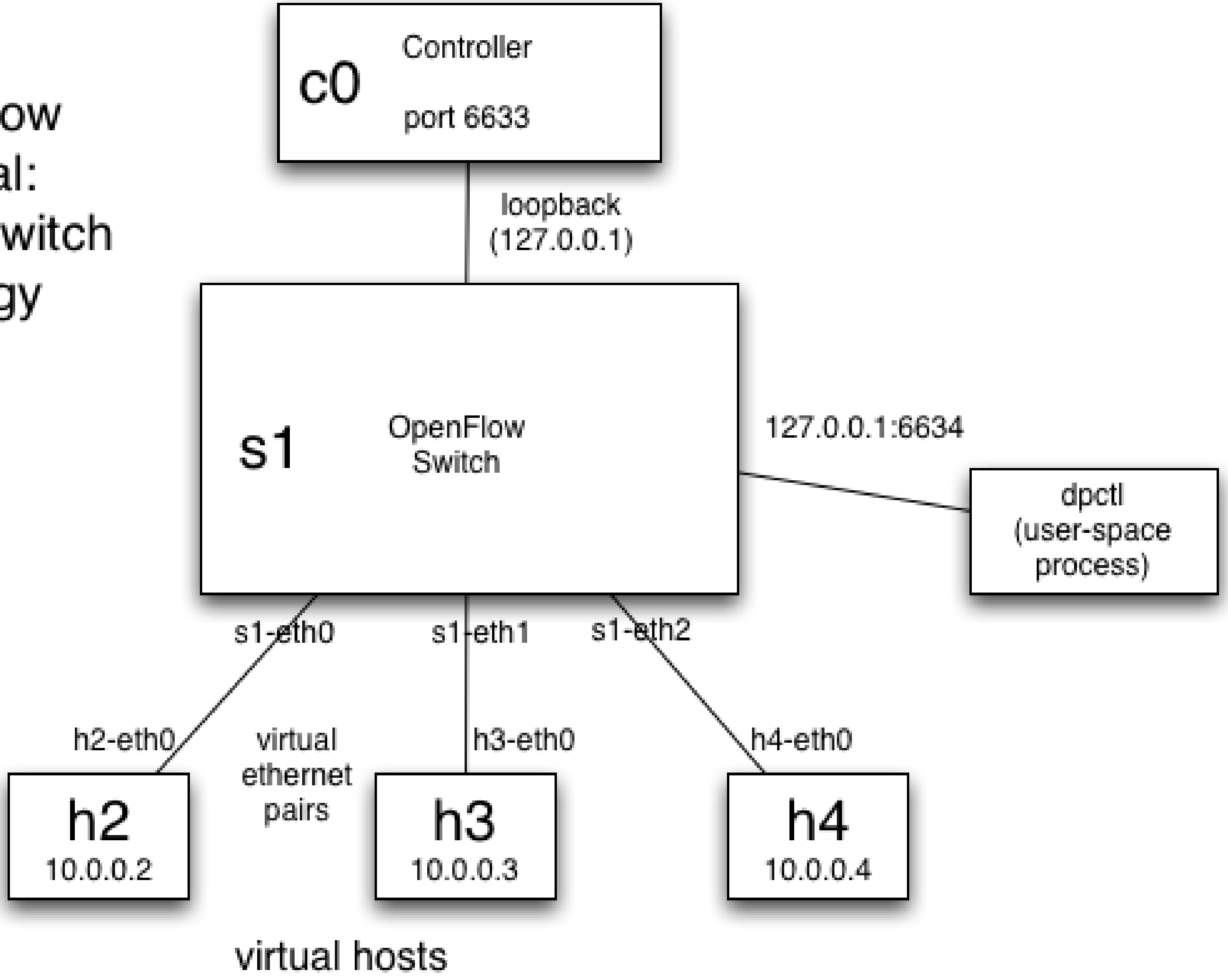
OpenFlow – OpenFlow Channel

- Different message types available:
  - *Controller-to-Switch*, *Asynchronous* or *Symmetric*
- Controller-to-Switch:
  - Lets the controller control the switch
  - E.g., *Modify-State* command to manipulate flow tables
- Asynchronous:
  - Switch-to-controller requests (e.g., at table miss)
- Symmetric:
  - May be sent from both ends (e.g., echo command)



OpenFlow - Example

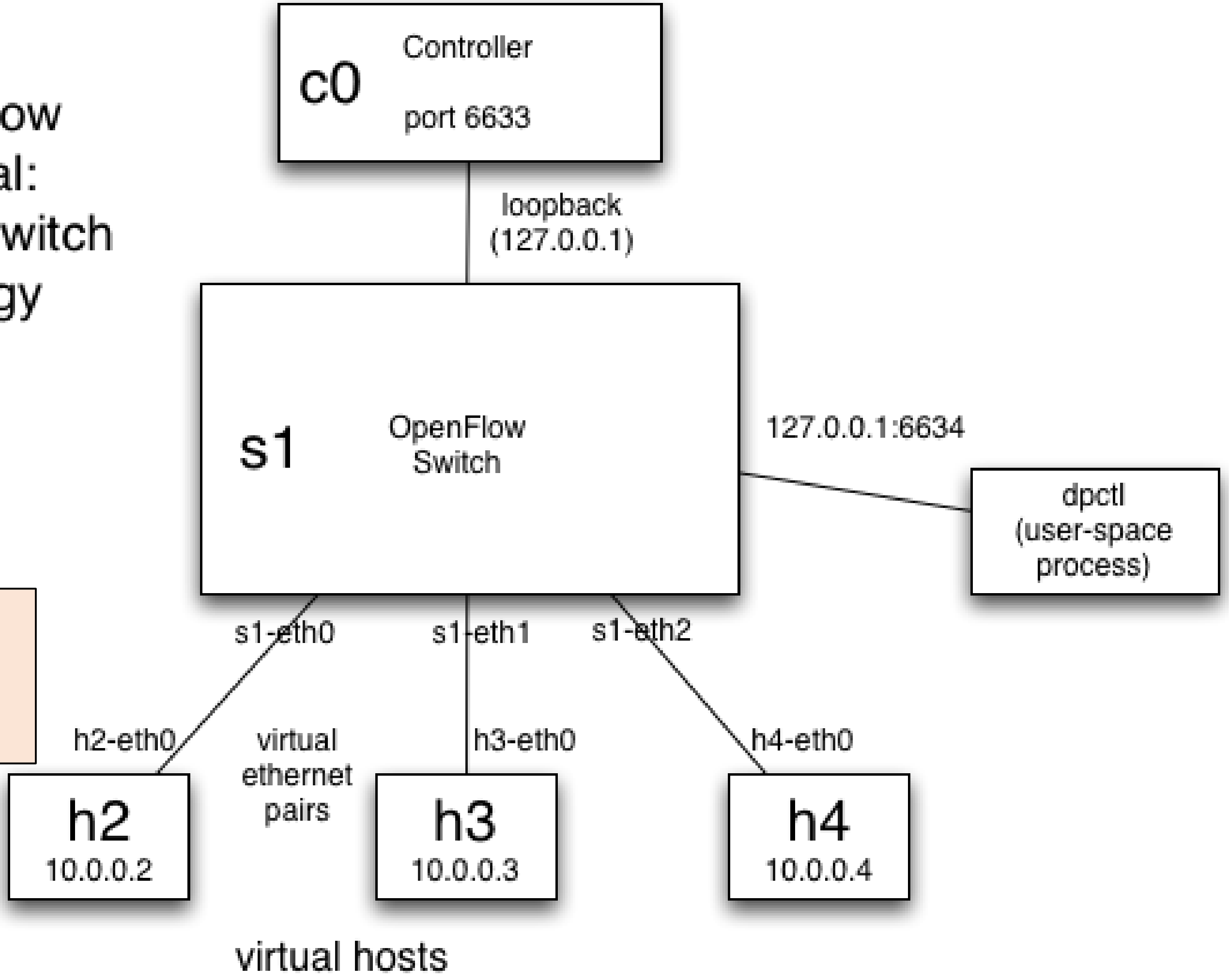
OpenFlow Tutorial: 3hosts-1switch topology



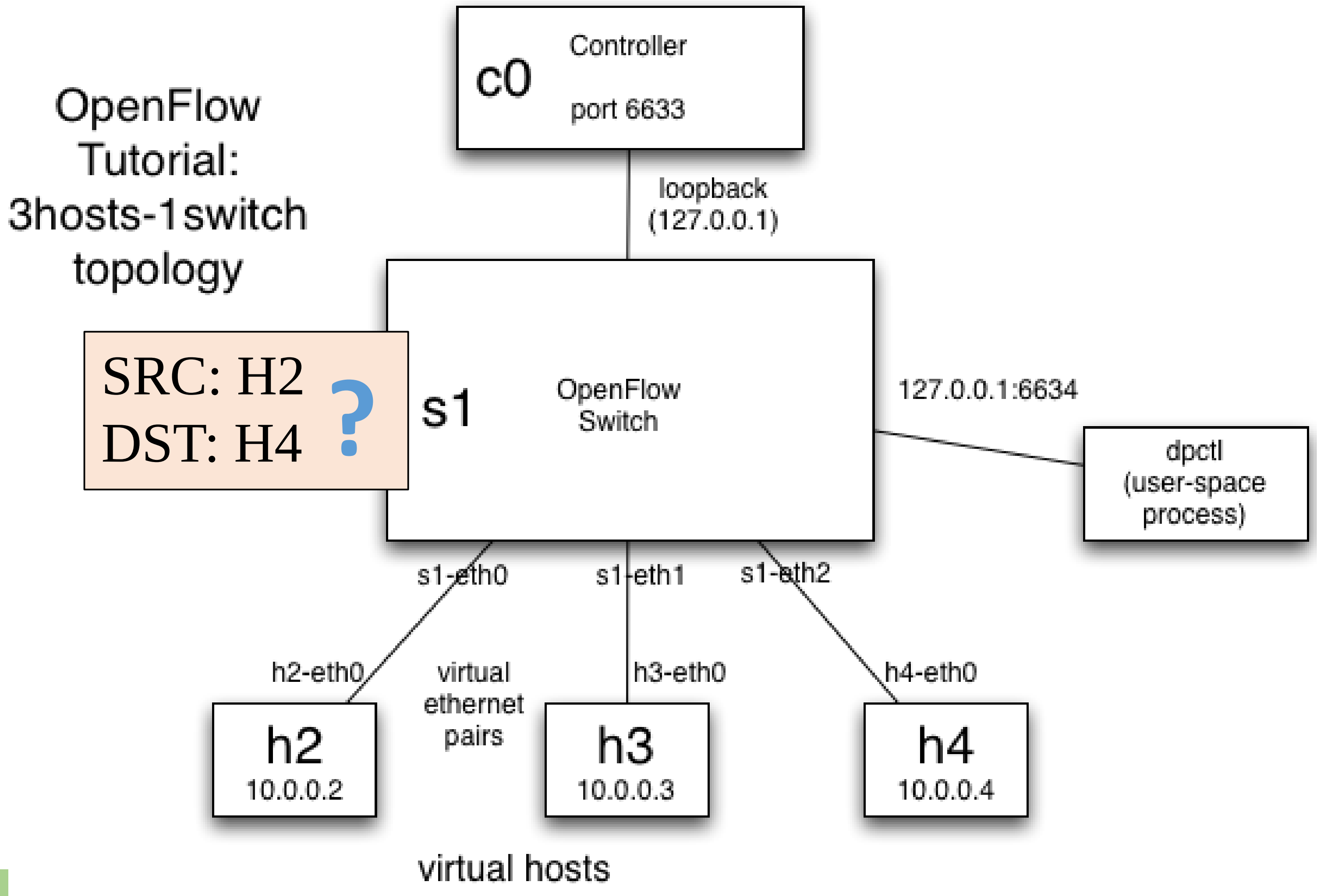
OpenFlow - Example

OpenFlow
Tutorial:
3hosts-1switch
topology

SRC: H2
DST: H4

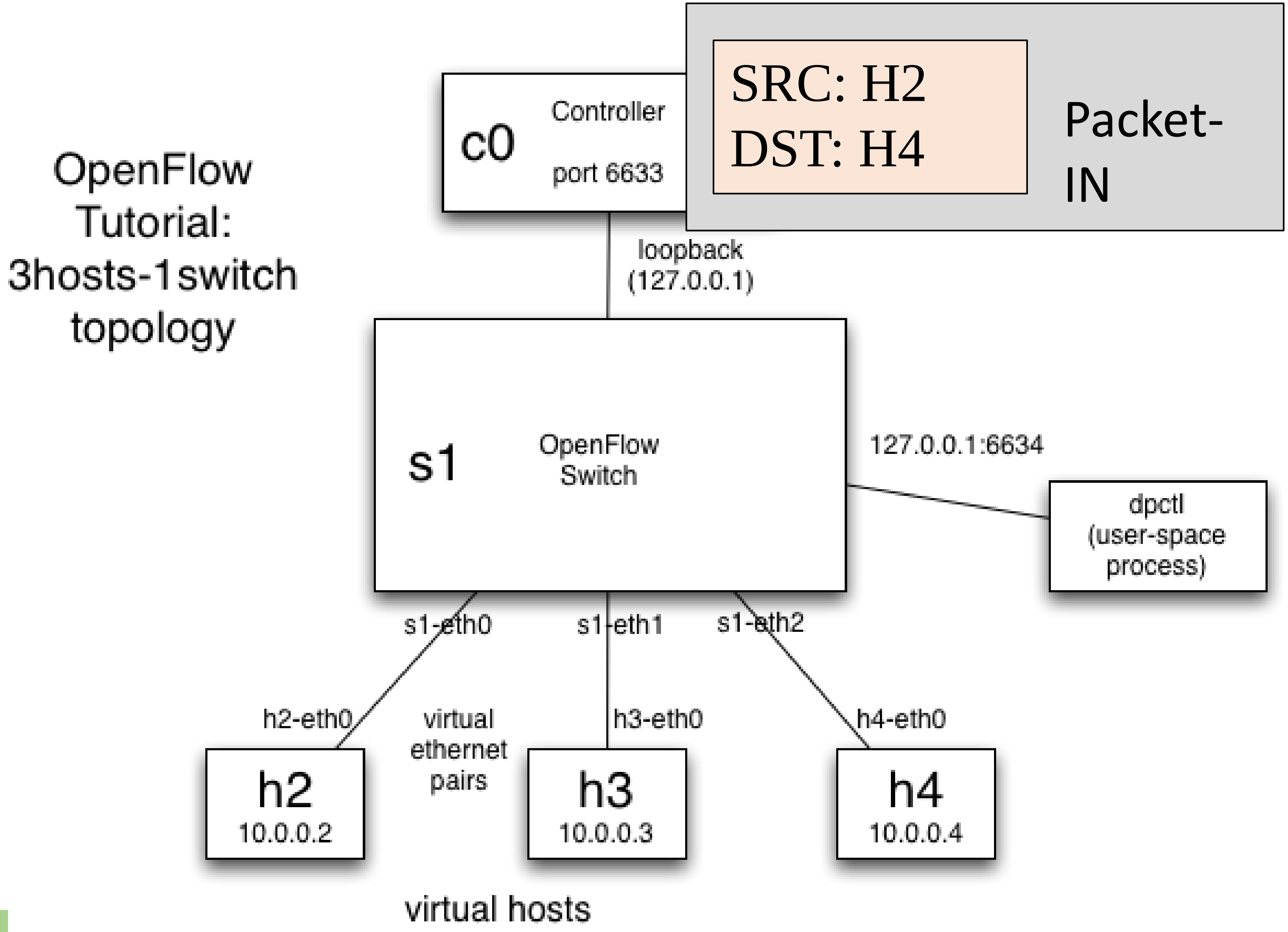


OpenFlow - Example



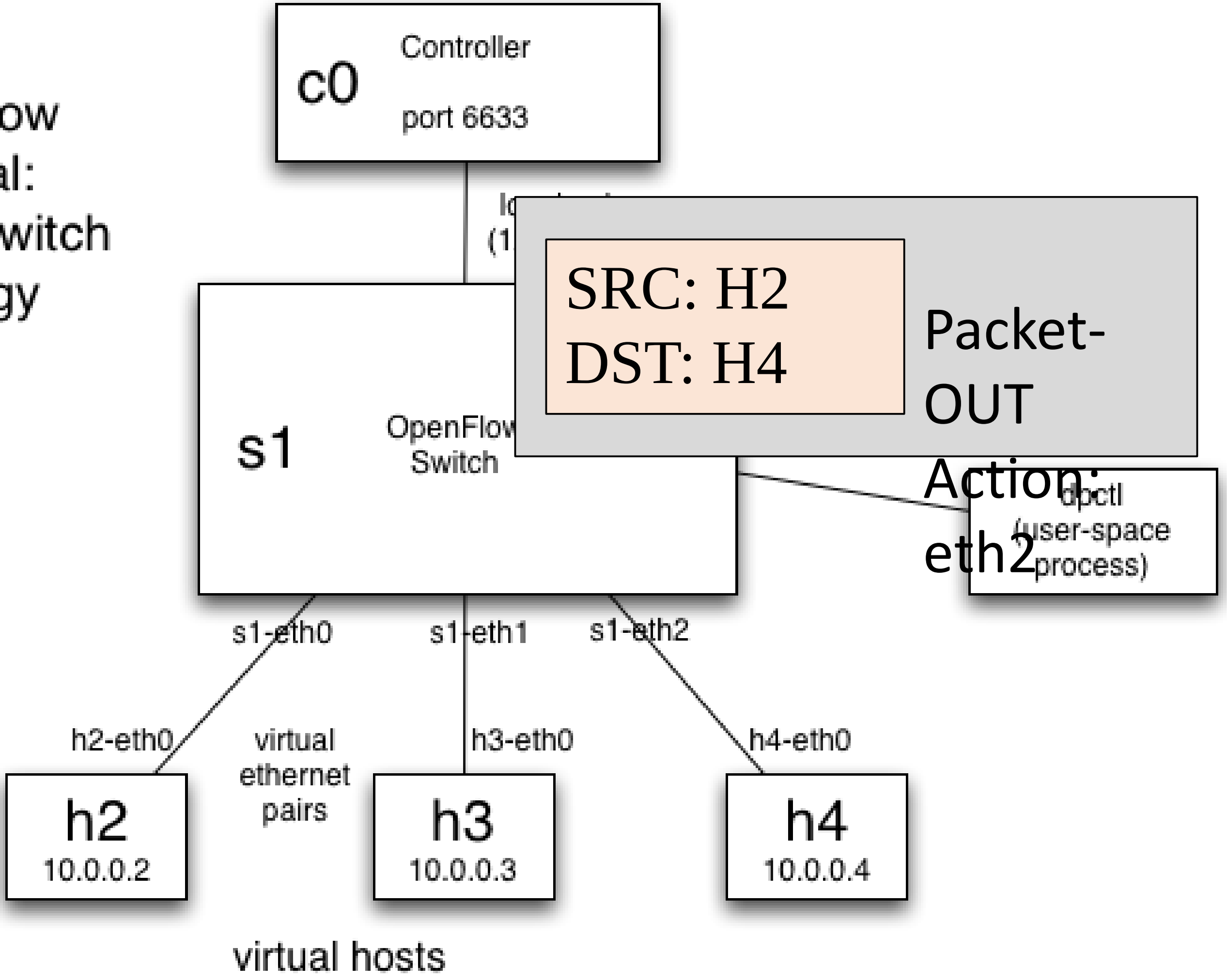
SRC: H2
DST: H4 ?

OpenFlow - Example



OpenFlow - Example

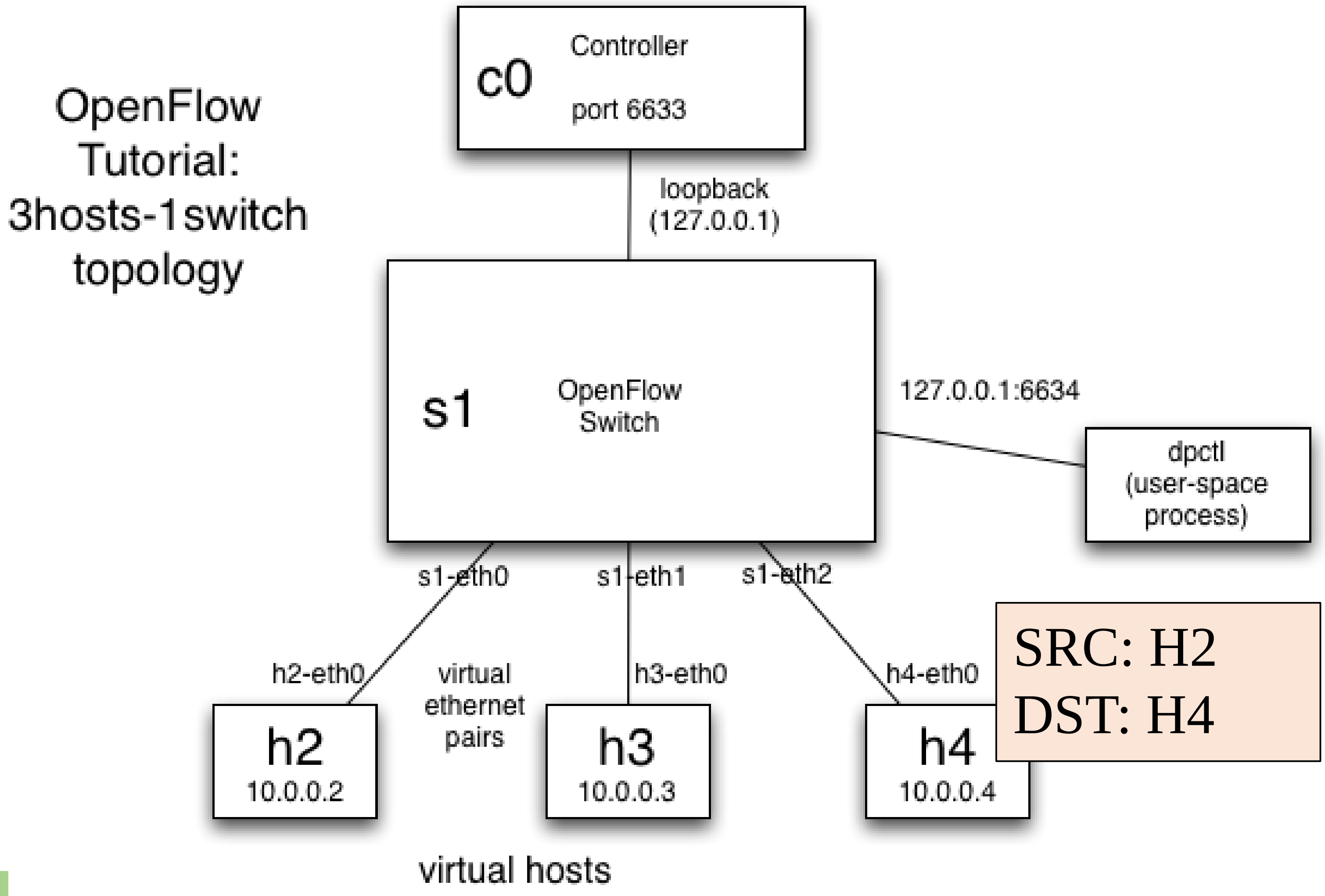
OpenFlow Tutorial: 3hosts-1switch topology



SRC: H2
DST: H4

Packet-OUT Action: eth2

OpenFlow - Example



OpenFlow Tutorial: 3hosts-1switch topology

c0 Controller port 6633

loopback (127.0.0.1)

s1 OpenFlow Switch

127.0.0.1:6634

dpctl (user-space process)

s1-eth0

s1-eth1

s1-eth2

h2-eth0

virtual ethernet pairs

h3-eth0

h4-eth0

h2 10.0.0.2

h3 10.0.0.3

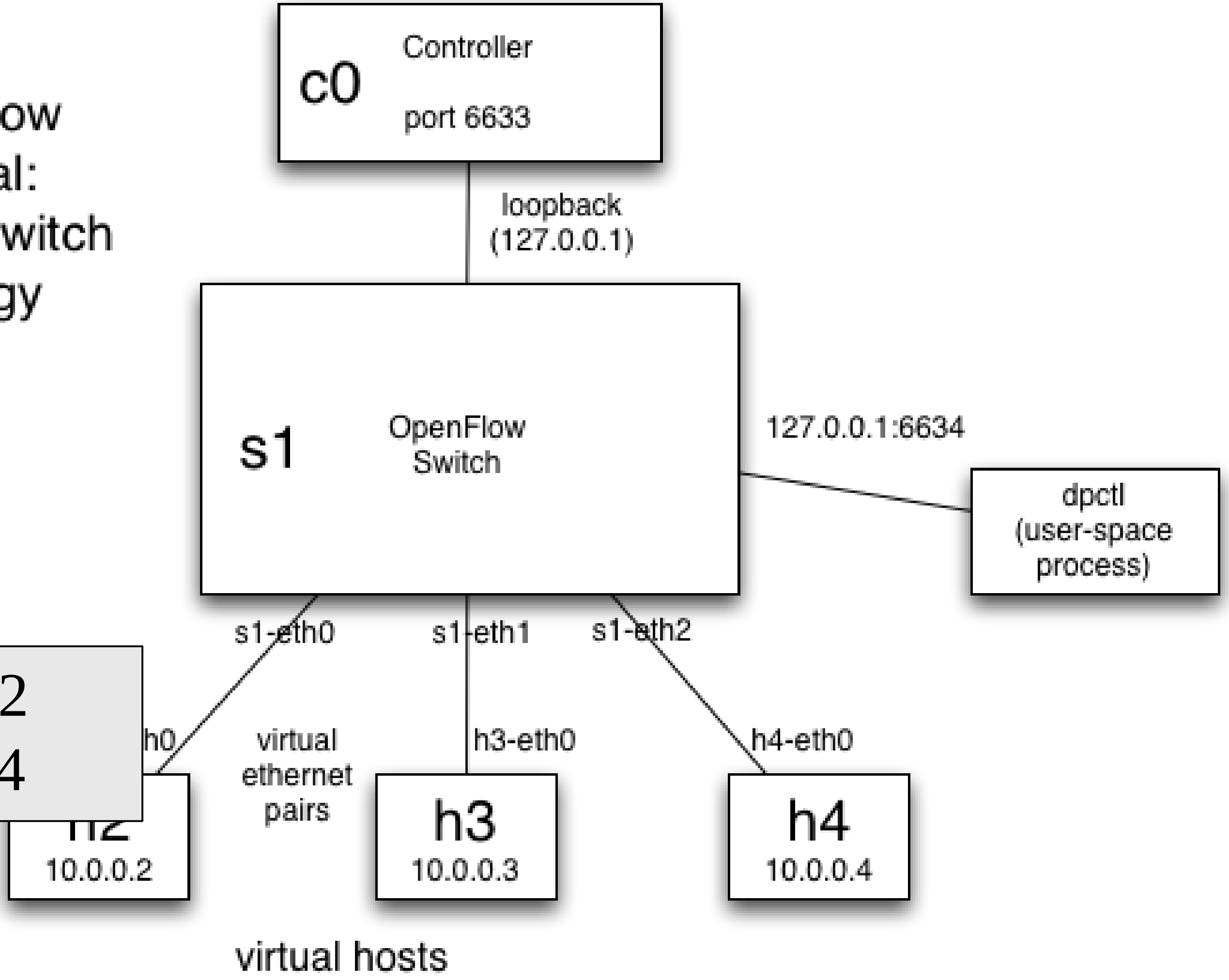
h4 10.0.0.4

virtual hosts

SRC: H2
DST: H4

OpenFlow - Example

OpenFlow Tutorial: 3hosts-1switch topology

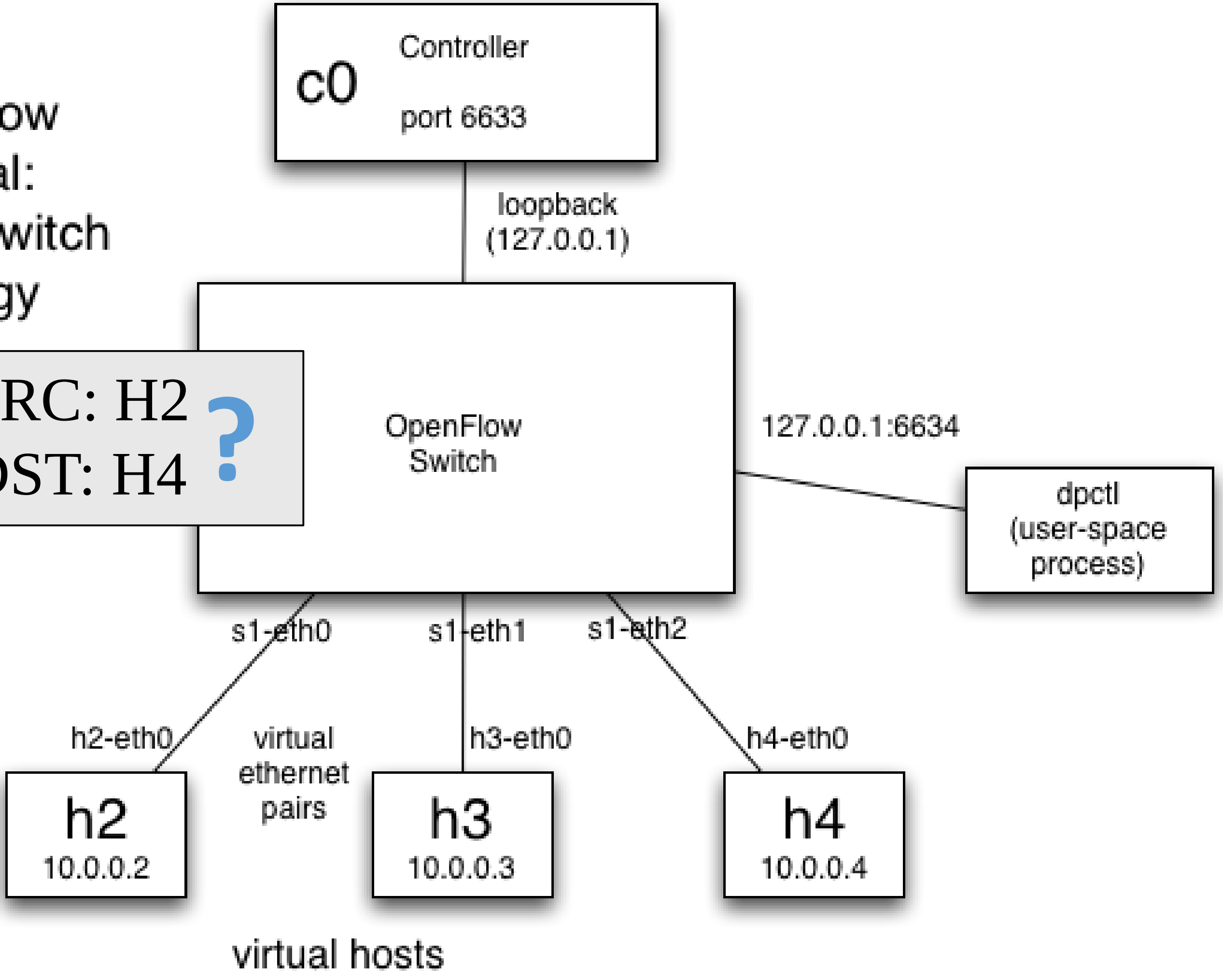


SRC: H2

DST: H4

OpenFlow - Example

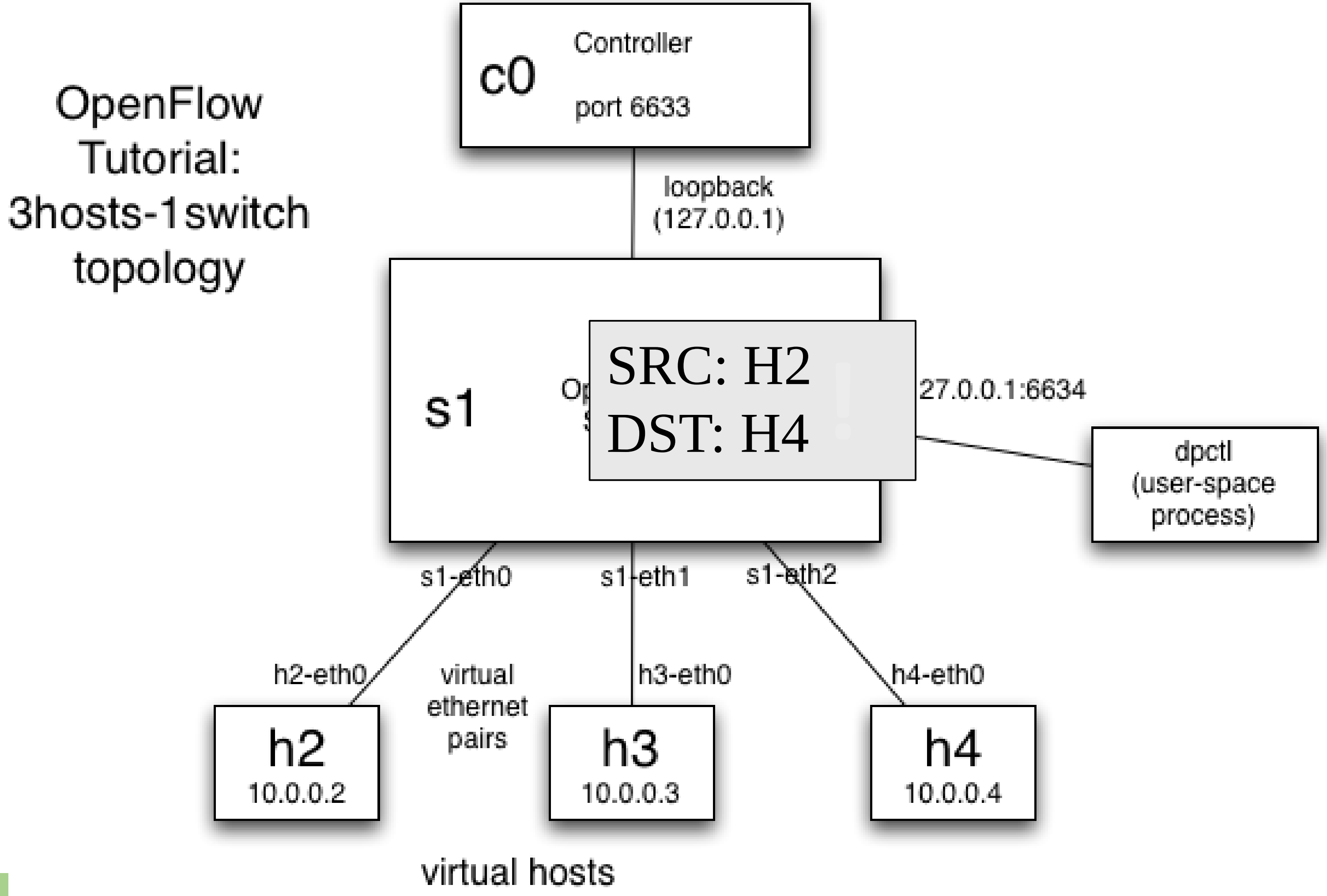
OpenFlow Tutorial: 3hosts-1switch topology



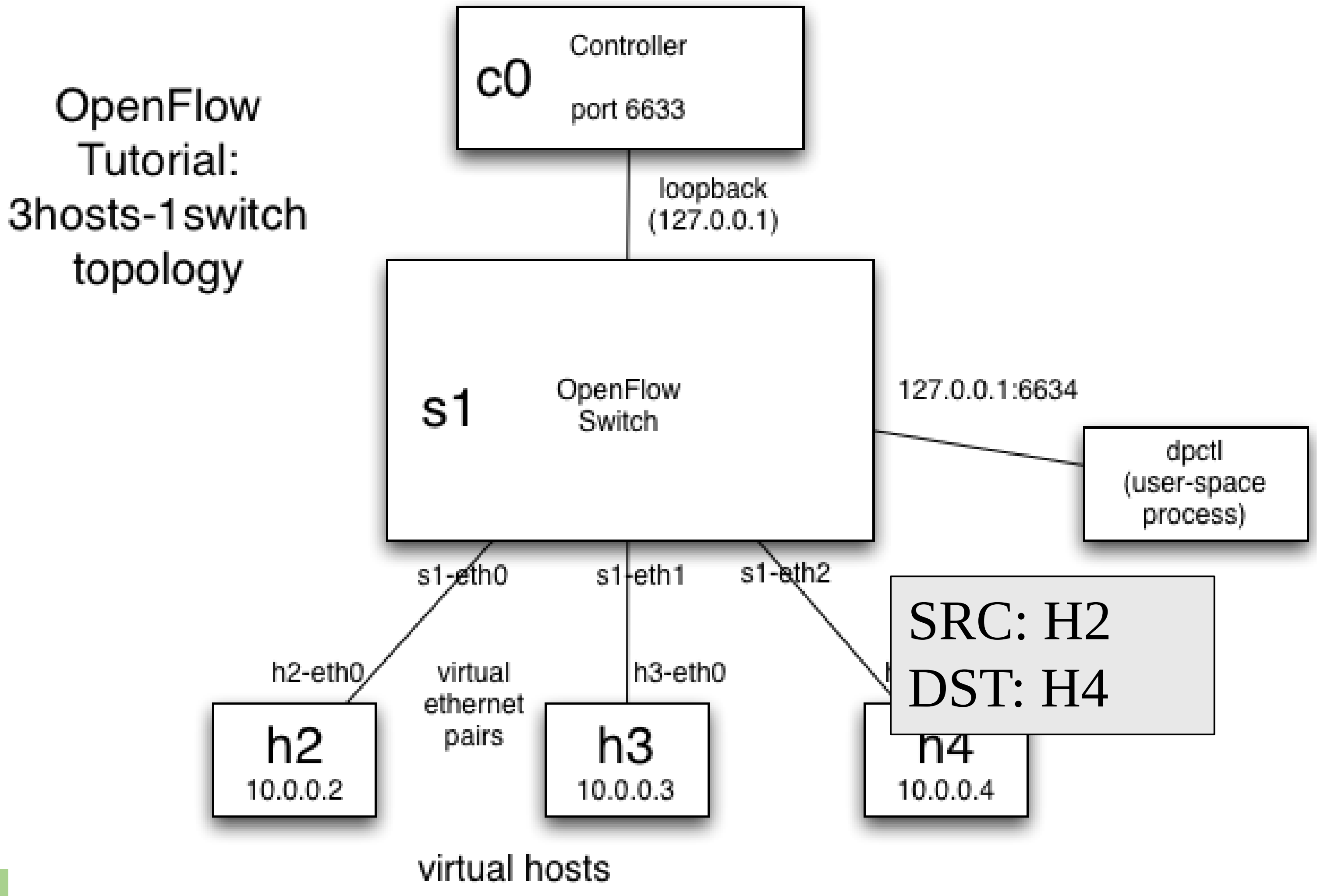
SRC: H2

DST: H4 ?

OpenFlow - Example



OpenFlow - Example



SRC: H2

DST: H4

OpenFlow Protocol Messages

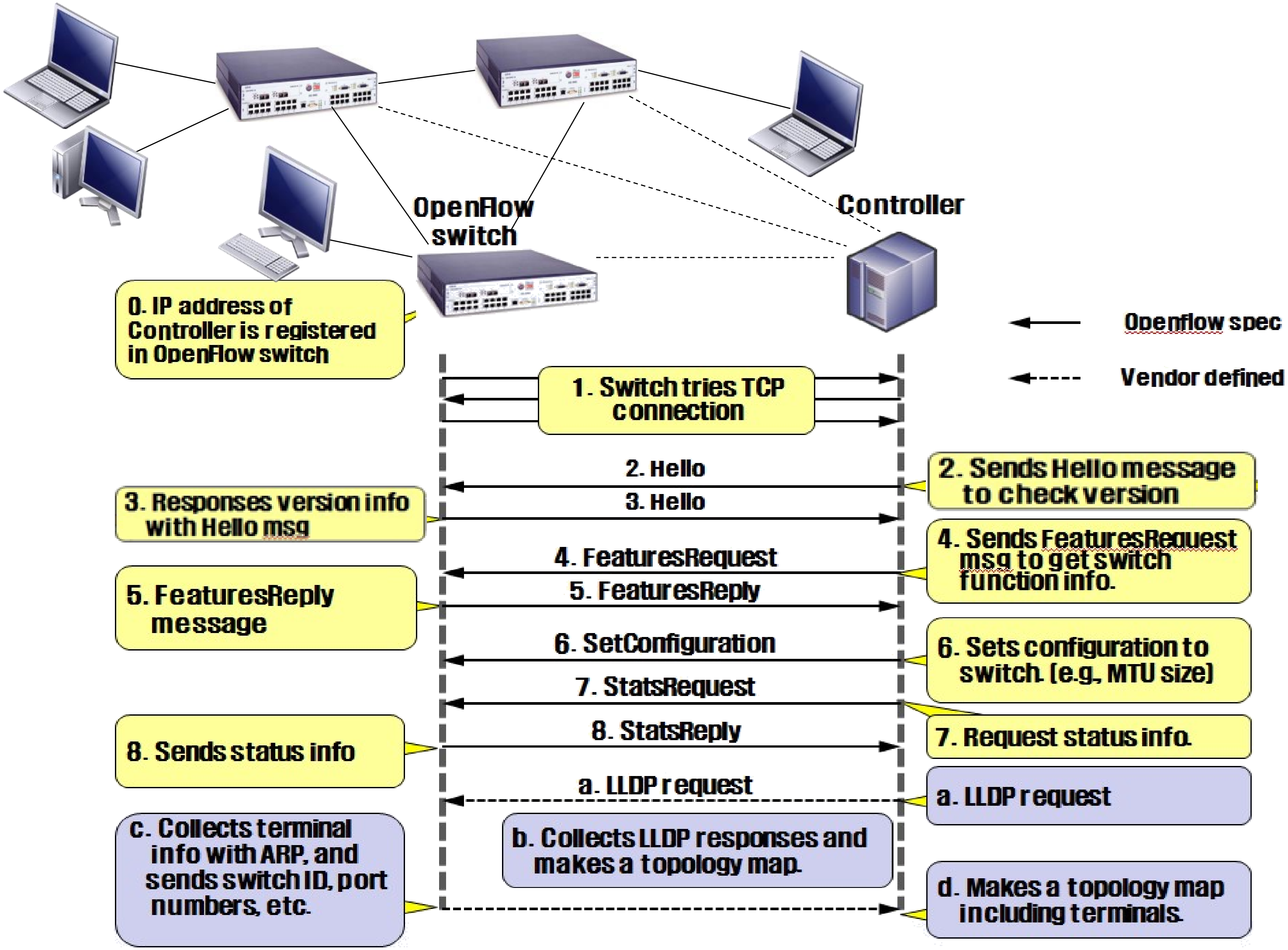
C: OpenFlow Controller
S: OpenFlow Switch

AM: Asynchronous message
SM: Symmetric Message

CSM: Control/Switch Message

Category	Message	Type	Description
Meta Info. Configuration	**Hello (SM)**	C → S	following a TCP handshake, the controller sends its version number to the switch.
	Hello (SM)	S → C	the switch replies with its supported version number.
	Features Request (CSM)	C → S	the controller asks to see which ports are available.
	Set Config (CSM)	C → S	in this case, the controller asks the switch to send flow expirations.
	Features Reply (CSM)	S → C	the switch replies with a list of ports, port speeds, and supported tables and actions.
	Port Status	S → C	enables the switch to inform that controller of changes to port speeds or connectivity..
Flow Processing	**Packet-In (AM)**	S → C	a packet was received and it didn't match any entry in the switch's flow table, causing the packet to be sent to the controller.
	Packet-Out (CSM)	C → S	Instructs a switch to send a packet out to one or more switch ports.
	Flow-Mod (CSM)	C → S	instructs a switch to add a particular flow to its flow table.
	Flow-Expired (CSM)	S → C	a flow timed out after a period of inactivity.

OpenFlow Communication



0. IP address of Controller is registered in OpenFlow switch

2. Sends Hello message to check version

3. Responses version info with Hello msg

4. Sends FeaturesRequest msg to get switch function info.

5. FeaturesReply message

6. Sets configuration to switch. (e.g., MTU size)

7. Request status info.

8. Sends status info

a. LLDP request

b. Collects LLDP responses and makes a topology map.

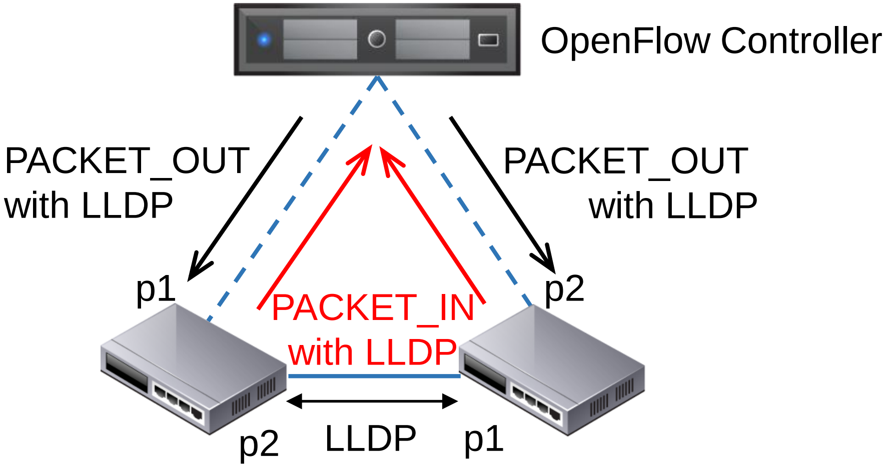
c. Collects terminal info with ARP, and sends switch ID, port numbers, etc.

d. Makes a topology map including terminals.

Topology Discovery in OpenFlow

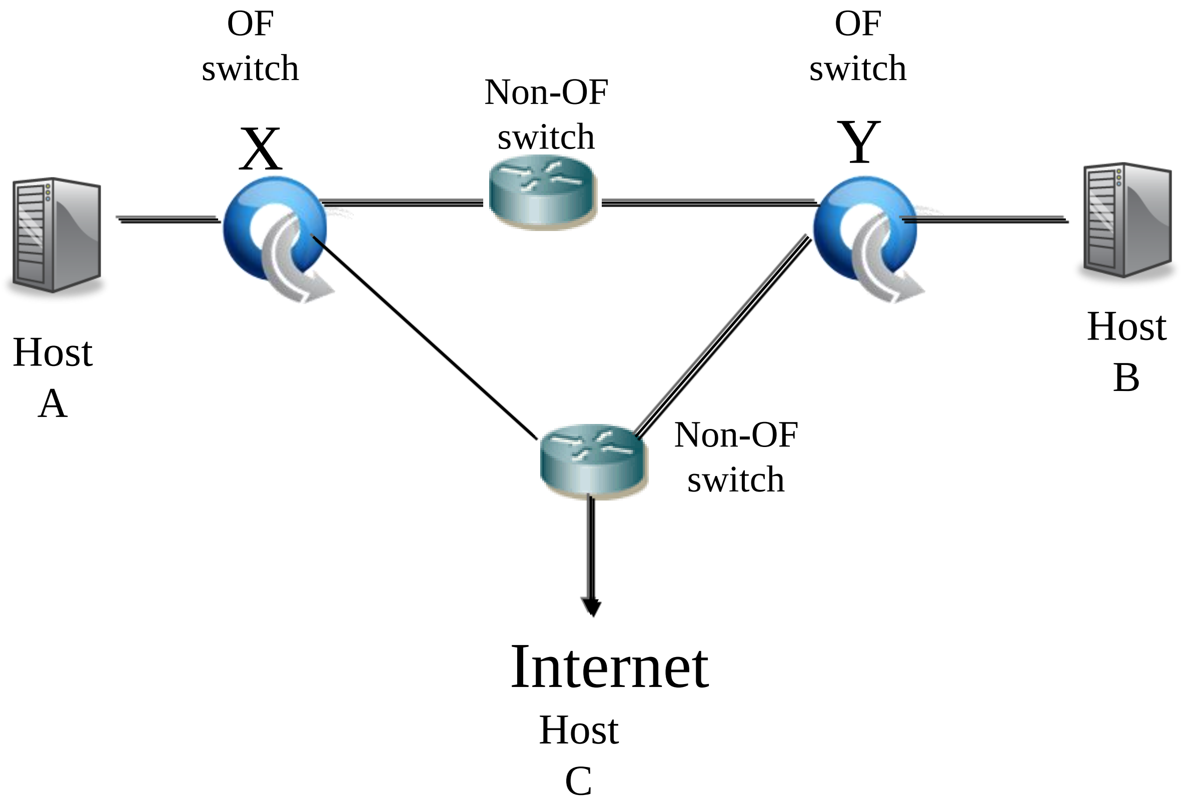
- Purpose
  - To construct an entire network view
- Method
  - Use the Link Layer Discovery Protocol (LLDP)

IDX	SRC	DST	SRC PORT	DST PORT
153	sw. A	sw. B	p2	p1
…	…	…	…	…
357	sw. B	sw. A	P1	p2



Topology discovery

- OpenFlow controller view is not always complete. For instance, what does the controller see here?



Flow Routing vs. Aggregation

Flow-Based

- Every flow is individually set up by controller
- Exact-match flow entries
- Flow table contains one entry per flow
- Good for fine grain control, e.g. campus networks

Aggregated

- One flow entry covers large groups of flows
- Wildcard flow entries
- Flow table contains one entry per category of flows
- Good for large number of flows, e.g. backbone

Reactive vs. Proactive (pre-populated)

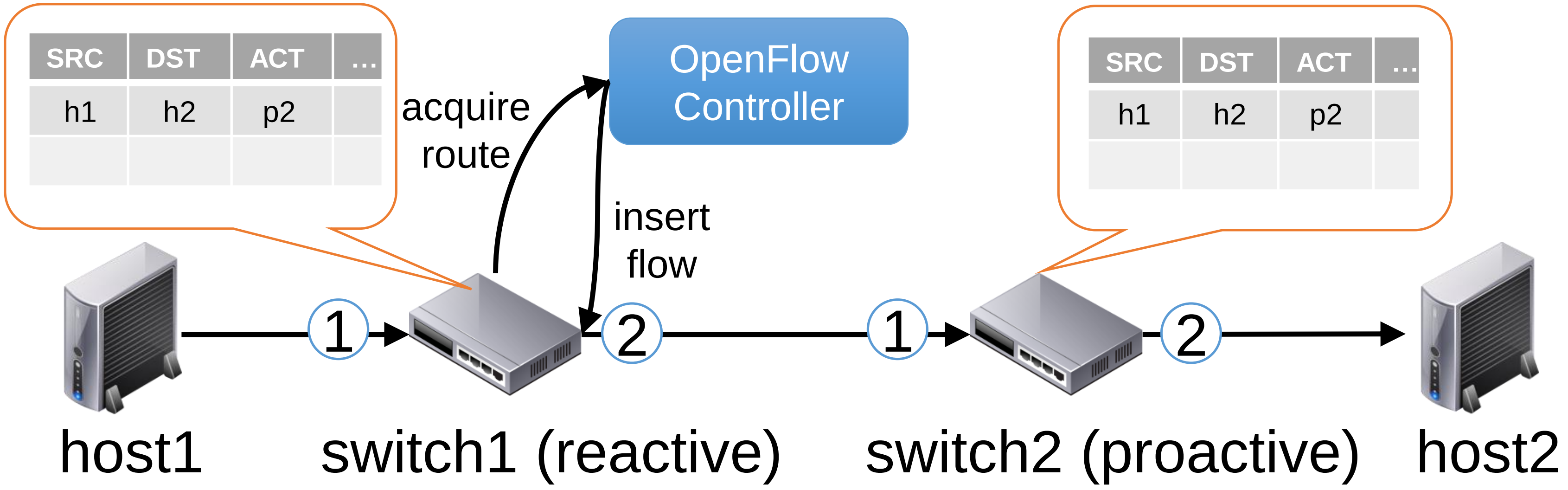
Reactive

- First packet of flow triggers controller to insert flow entries
- Efficient use of flow table
- Every flow incurs small additional flow setup time
- If control connection lost, switch has limited utility

Proactive

- Controller pre-populates flow table in switch
- Zero additional flow setup time
- Loss of control connection does not disrupt traffic
- Essentially requires aggregated (wildcard) rules

Packet Forwarding in OpenFlow



SRC	DST	ACT	…
h1	h2	p2	

OpenFlow Controller

acquire route

insert flow

1

2

1

2

SRC	DST	ACT	…
h1	h2	p2	

host1

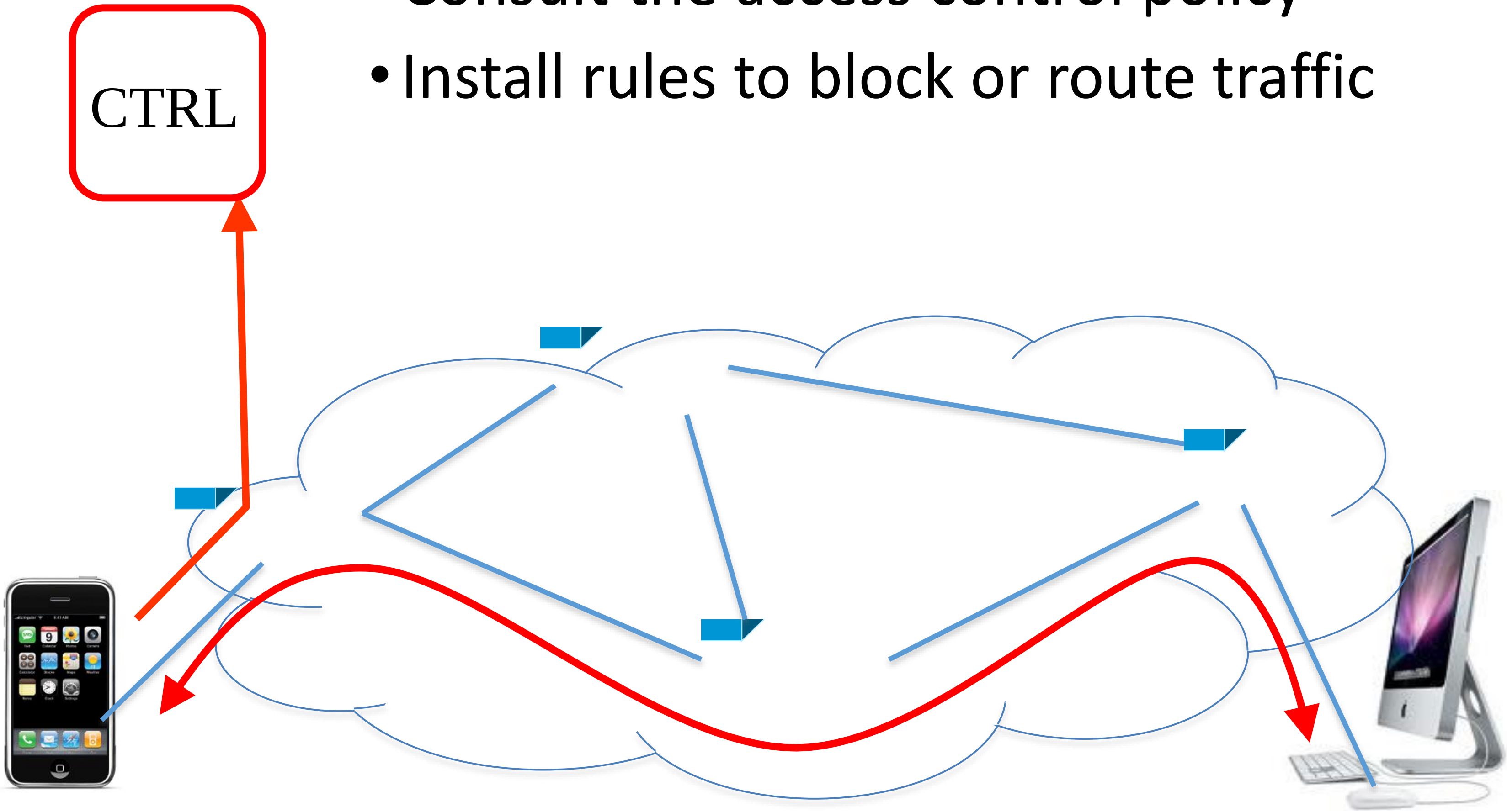
switch1 (reactive)

switch2 (proactive)

host2

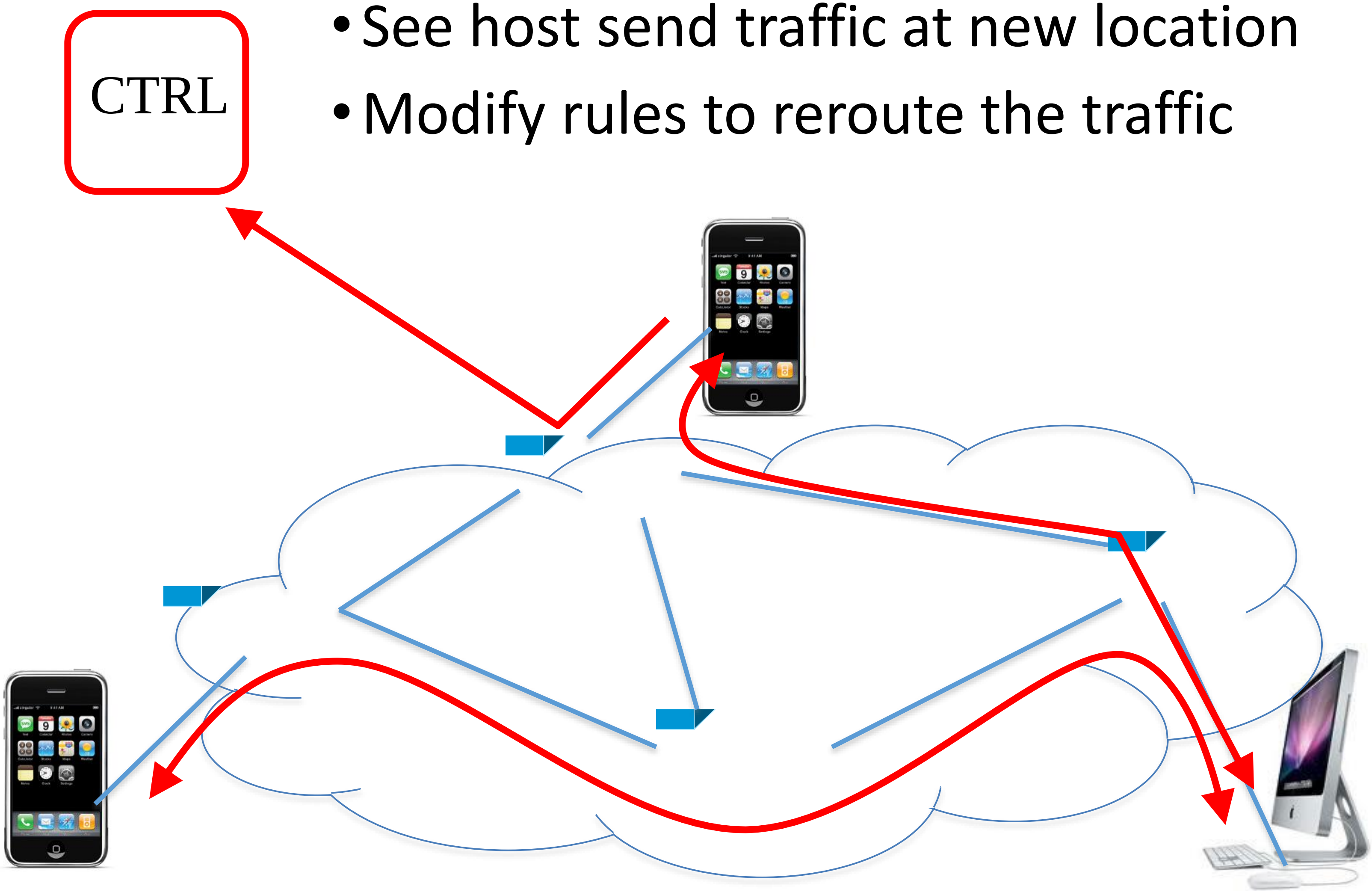
OF Applications: Dynamic Access Control

- Inspect first packet of a connection
- Consult the access control policy
- Install rules to block or route traffic



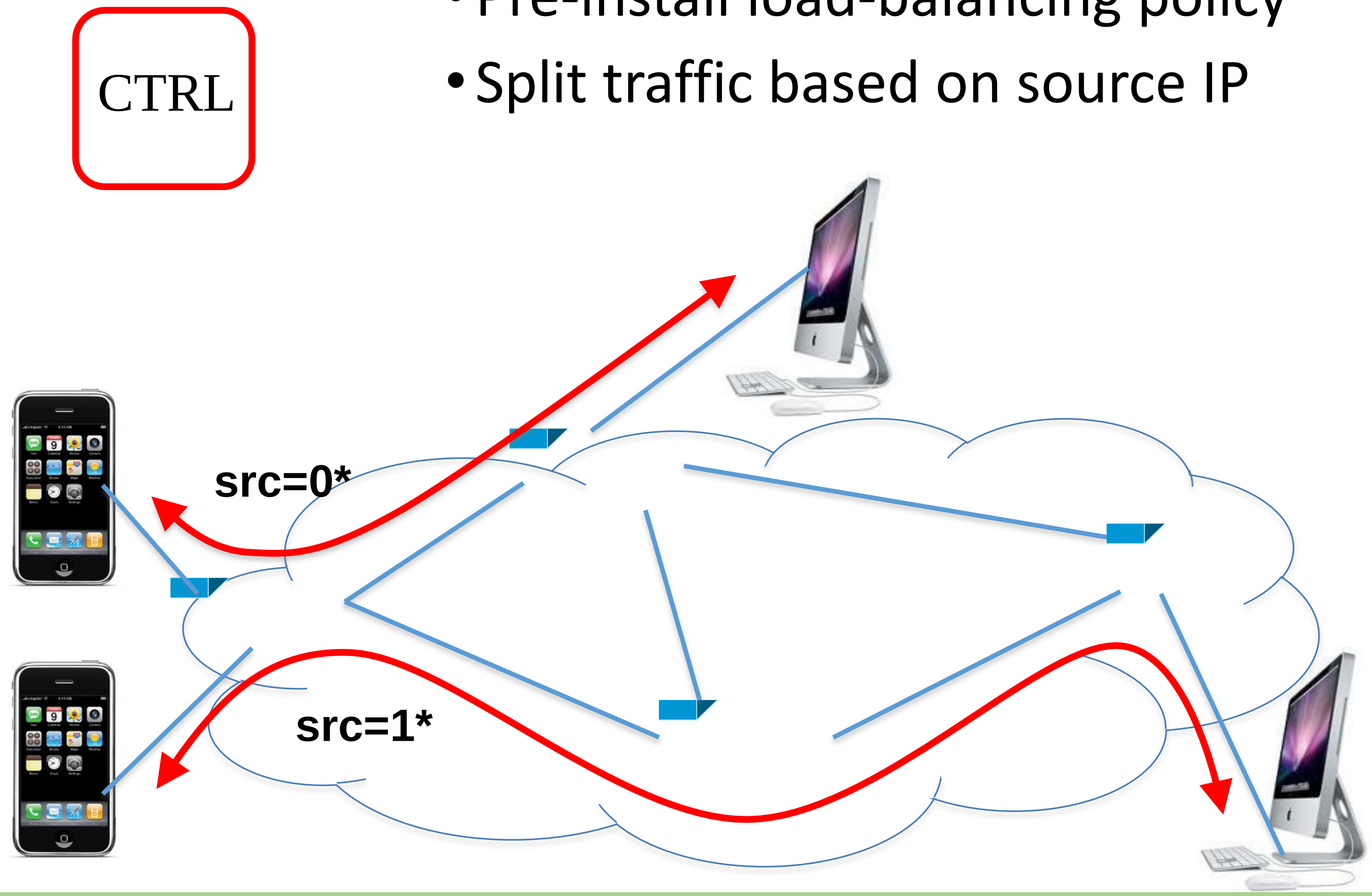
OF Applications: Seamless Mobility/Migration

- See host send traffic at new location
- Modify rules to reroute the traffic



OF Applications: Load Balancing

- Pre-install load-balancing policy
- Split traffic based on source IP



Examples of Current SDN Hardware

Juniper MX-series

NEC IP8800

WiMax (NEC)

HP Procurve 5400

Netgear 7324

PC Engines

Pronto 3240/3290

Ciena Coredirector

Summary

We have discussed:

- OpenFlow *as an example* of a Southbound interface
- OpenFlow as communication interface between control and data plane
- Various components of the OF standard