

# Hands-on Open vSwitch and Software-defined Networking

Jorge Crichigno, Shahrin Sharif, Elie Kfoury

University of South Carolina

<http://ce.sc.edu/cyberinfra>

[jcrichigno@cec.sc.edu](mailto:jcrichigno@cec.sc.edu), [ssharif@email.sc.edu](mailto:ssharif@email.sc.edu), [ekfoury@email.sc.edu](mailto:ekfoury@email.sc.edu)

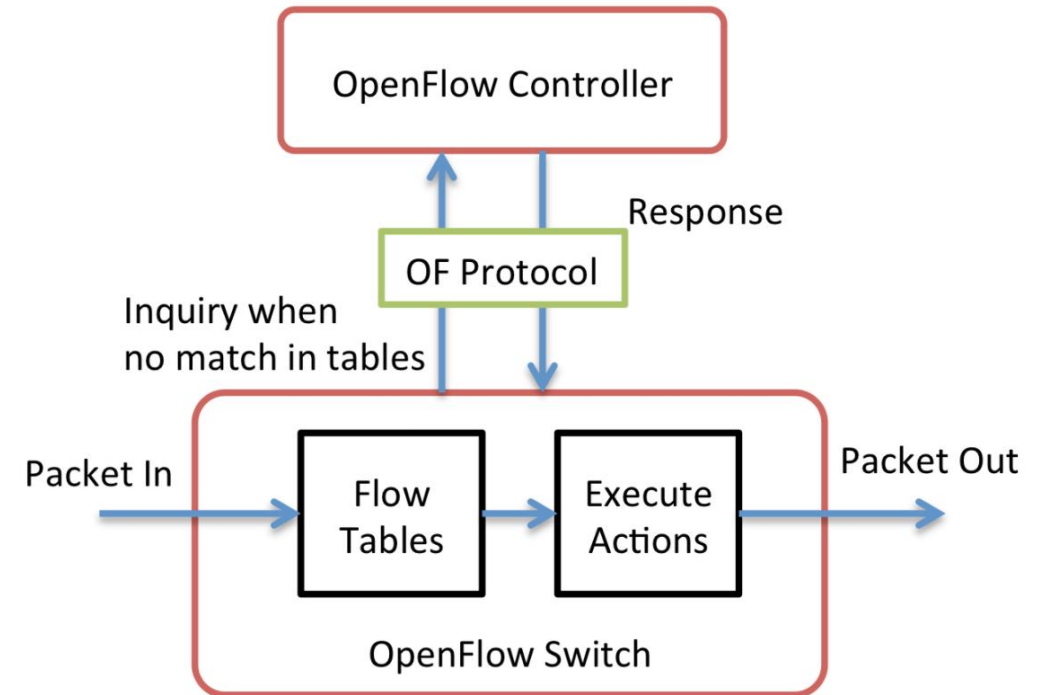
WASTC 2021 virtual Faculty Development Weeks (vFDW)

June 24, 2021

# Lab 4: Open vSwitch Flow Table

# OpenFlow overview

- OpenFlow is a protocol specification that describes the communication between OpenFlow switches and an OpenFlow controller
- OpenFlow consists of three components
  - OpenFlow controller
  - OpenFlow protocol
  - OpenFlow switch



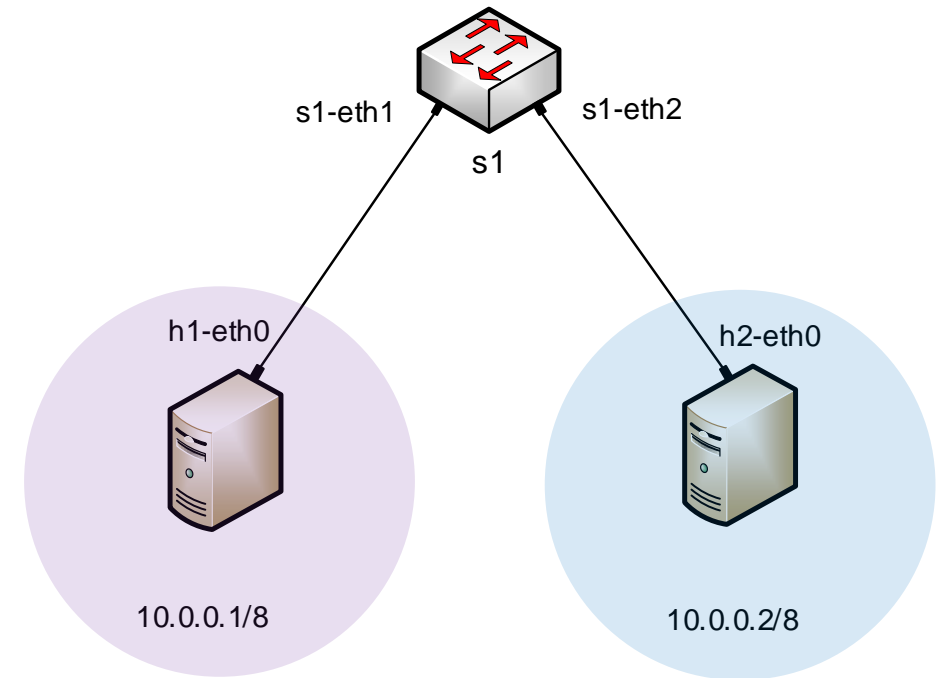
# Flow Table

- A flow table consists of flow entries
- A flow entry consists of header fields, counters, and actions associated with that entry

| Flow Entry 0  |  | Flow Entry 1  |                                |     | Flow Entry F  |                                      |     | Flow Entry M  |                                      |
|---------------|--|---------------|--------------------------------|-----|---------------|--------------------------------------|-----|---------------|--------------------------------------|
| Header Fields | Inport 12<br>192.32.10.1,<br>Port 1012 | Header Fields | Inport *<br>209.***,<br>Port * | ■■■ | Header Fields | Inport 2<br>192.32.20.1,<br>Port 995 | ■■■ | Header Fields | Inport 2<br>192.32.30.1,<br>Port 995 |
| Counters      | val                                    | Counters      | val                            |     | Counters      | val                                  |     | Counters      | val                                  |
| Actions       | val                                    | Actions       | val                            |     | Actions       | val                                  |     | Actions       | val                                  |

# Lab Topology

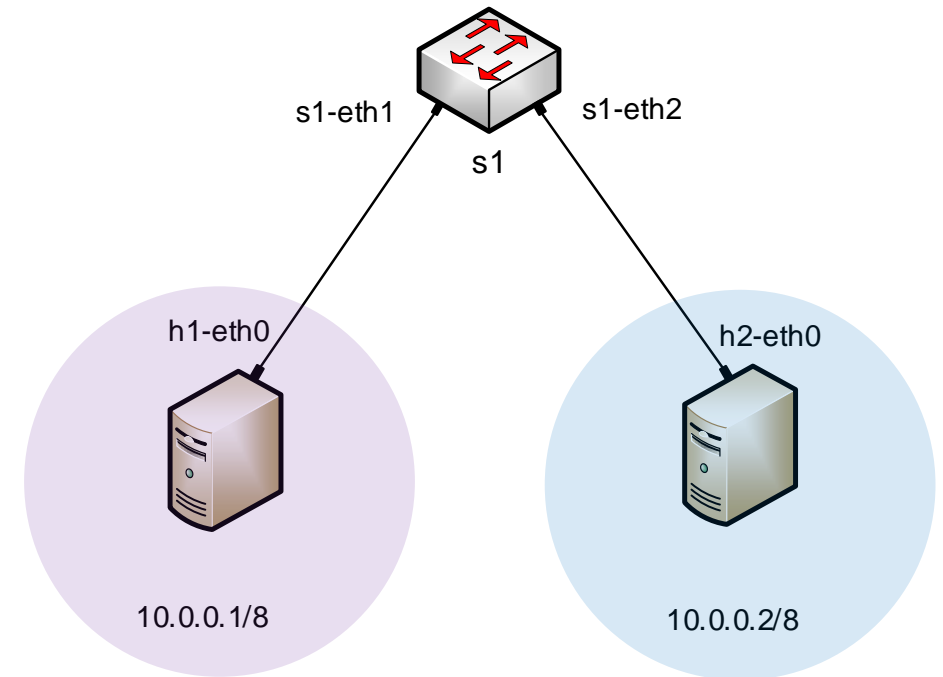
- Switch s1 connected to hosts h1 and h2
- Hosts h1 and h2 belong to network 10.0.0.0/8
- The lab aims to demonstrate how to manage flows manually in the switch s1



# Forwarding Using Layer 1

- Everything that comes from s1-eth1 is sent out to s1-eth2
- Everything that comes from s1-eth2 is sent out to s1-eth1

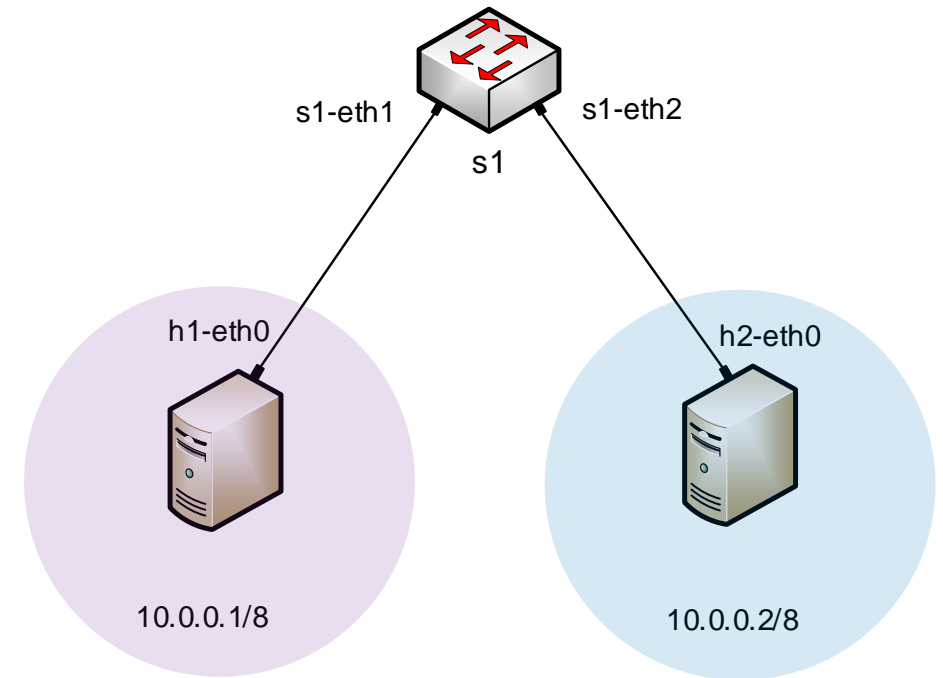
```
ovs@admin: ~  
File Actions Edit View Help  
ovs@admin: ~  
ovs@admin:~$ sudo ovs-ofctl add-flow s1 in_port=1,action=output:2  
ovs@admin:~$
```



# Forwarding Using Layer 2

- Flow entries based on MAC addresses of the hosts

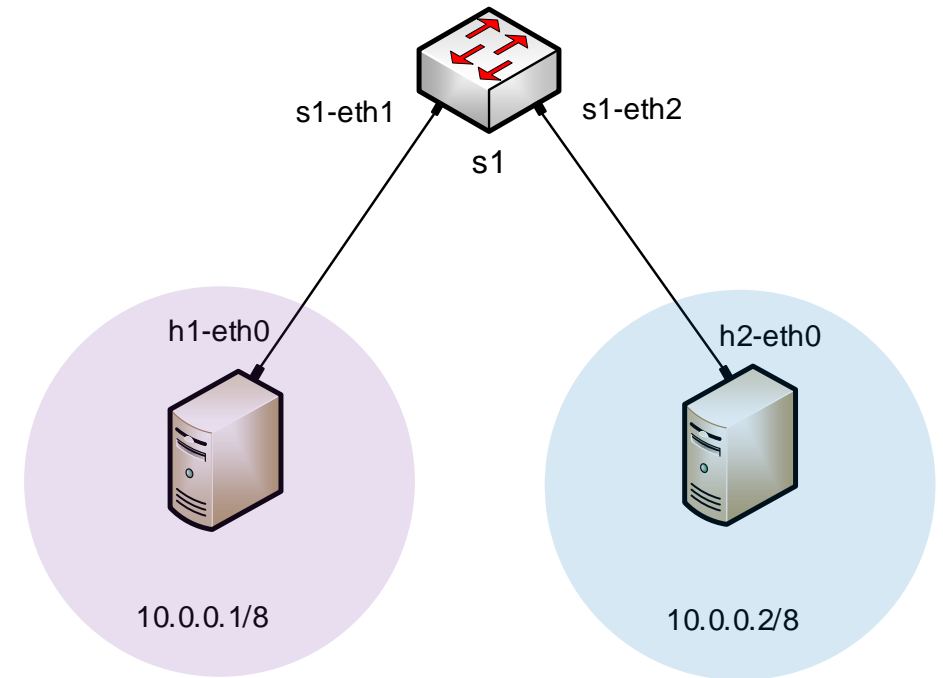
```
ovs@admin: ~  
File Actions Edit View Help  
ovs@admin: ~  
ovs@admin:~$ sudo ovs-ofctl add-flow s1 dl_dst=00:00:00:00:00:01,action=output:1  
ovs@admin:~$
```



# Forwarding Using Layer 3

- Flow entries based on IP addresses of the hosts

```
ovs@admin: ~  
File Actions Edit View Help  
ovs@admin: ~  
ovs@admin:~$ sudo ovs-ofctl add-flow s1 ip,nw_dst=10.0.0.1,action=output:1  
ovs@admin:~$
```

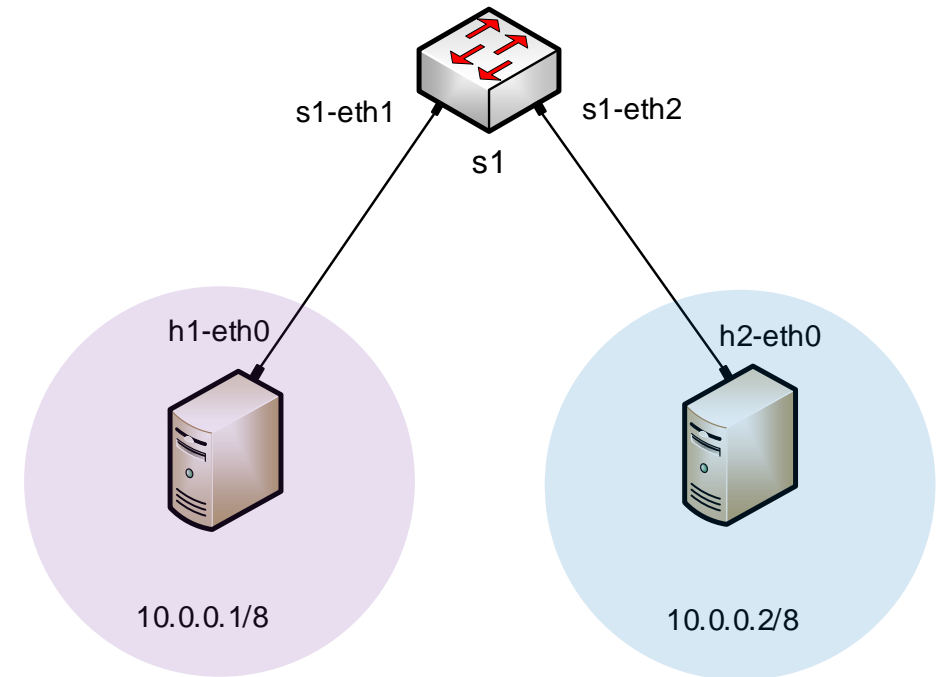




# Forwarding Using Layer 4

- Flow entries based on TCP
- A simple python web server is running in host h2
- Host h1 can connect to the server using port 80

```
ovs@admin: ~  
File Actions Edit View Help  
ovs@admin: ~  
ovs@admin:~$ sudo ovs-ofctl add-flow s1 tcp,tp_dst=80,action=output:2  
ovs@admin:~$
```



# Setting Match Priority

- Packets are matched against flow entries based on prioritization

```
ovs@admin: ~  
File Actions Edit View Help  
ovs@admin: ~  
ovs@admin:~$ sudo ovs-ofctl dump-flows s1  
cookie=0x0, duration=170.568s, table=0, n_packets=0, n_bytes=0, arp_actions=NORMAL  
cookie=0x0, duration=147.367s, table=0, n_packets=0, n_bytes=0, priority=500, ip,nw_dst=10.0.0.  
2 actions=drop  
cookie=0x0, duration=88.072s, table=0, n_packets=0, n_bytes=0, priority=400, dl_dst=00:00:00:00  
:00:02 actions=output:"s1-eth2"  
cookie=0x0, duration=80.535s, table=0, n_packets=0, n_bytes=0, priority=400, dl_dst=00:00:00:00  
:00:01 actions=output:"s1-eth1"  
ovs@admin:~$
```

