PICA8, INC.

# OVS Configuration Guide

## PicOS 2.0.4

**Yachal Chen, Zoneson Chen**

**2013-9-20**

This document provides all OVS configuration commands for PicOS 2.0.4.

**OVS Configuration Guide, PicOS 2.0.4**

# Table of Contents

# Preface

## *Intended Audience*

This guide is intended for data center administrators, system administrators, and customer service staffs responsible for configuring the PicOS Open vSwitch (OVS).

## *Documents*

PicOS documents are available at the following website:
http://www.pica8.com/portal/trial.php
Open vSwitch software documents are available at the following website:
http://openvswitch.org/

OpenFlow documents are available at the following website:
http://www.openflow.org/

## *Organization*

This configuration guide is organized as follows:

| Chapter | Descriptions |
|---|---|
| Chap 1. Overview | Provides an overview of the Pica8 switch |
| Chap 2. System Update and Boot | Describes the system update and boot |
| Chap 3. Configurating Open vSwitch | Describes the configuration of OVS in PicOS 2.0.4 |
| Chap 4. Example Configurations | Provides configuration examples |

# Chapter 1. Overview

This chapter provides an overview of PicOS OVS features. Open vSwitch is a production-quality, multilayer virtual switch licensed under the open source Apache 2.0 license. PicOS OVS is the implementation of Open vSwitch on Pica8 hardware switches.

## *PicOS OVS Features List*

PicOS OVS supports the following features:

**Table 1-1. PicOS OVS Features List.**

| |
|---|
| NetFlow, sFlow |
| Standard 802.1Q VLAN model with trunking |
| Link monitoring |
| MPLS, GRE |
| Multiple virtual bridges |

# Chapter 2. System Update and Boot

This chapter describes how to update and boot the system.

## *The Boot Process*

Before receiving the switch's boot information, make sure you have connected the console port with the correct baud rate, data bits value, and stop bits value.

- The baud rate is **115200**.
- The data bits value is **8**.
- The stop bits value is **1**.

The output message of the boot-up is shown below:

```
U-Boot 1.3.0 (Mar  8 2011 - 16:39:03)

CPU:   8541, Version: 1.1, (0x80720011)
Core:  E500, Version: 2.0, (0x80200020)
Clock Configuration:
       CPU: 825 MHz, CCB: 330 MHz,
       DDR: 165 MHz, LBC:  41 MHz
L1:    D-cache 32 kB enabled
       I-cache 32 kB enabled
I2C:   ready
DRAM:  Initializing
initdram robin1
initdram robin2
robin before CFG_READ_SPD
robin after CFG_READ_SPD
initdram robin3
    DDR: 512 MB
FLASH: 32 MB
L2 cache 256KB: enabled
In:    serial
Out:   serial
Err:   serial
Net:   TSEC0, TSEC1
IDE:   Bus 0: OK
  Device 0: Model: CF 512MB Firm: 20060911 Ser#: TSS25016070309051750
            Type: Hard Disk
            Capacity: 495.1 MB = 0.4 GB (1014048 x 512)
Hit any key to stop autoboot:  5
```

Select Option 2, Open vSwitch, to boot up. The default choice is Option 1, Pica8 XorPlus, a switching software that provides the L2/L3 protocol stack and enables the community to innovate. You can edit the default choice with Option 4, Boot menu editor, if you wish.

```
File system OK
net.netfilter.nf_conntrack_acct = 1
net.ipv6.conf.all.forwarding = 1
 7 Apr 07:08:58 ntpdate[900]: no servers can be used, exiting
System initiating...Please wait...
    Please choose which to start: Pica8 XorPlus, OpenFlow, or System shell:
        (Will choose default entry if no input in 10 seconds.)
        [1]  Pica8 XorPlus   * default
        [2]  Open vSwitch
        [3]  System shell
        [4]  Boot menu editor
```

```
Enter your choice (1,2,3,4):2
```

Next, enter the OVS CLI as follows:

```
Open vSwitch is selected

  Note: Defaultly, the OVS server is runned with static local management IP and port 6633.
        The default way of vswitch connecting to server is PTCP.
        If you do not want default configuration, choose manual start!

Do you want start the OVS by manual? (yes/no)
```

### (1) Manually starting OVS

You can choose to start the OVS software manually or by the system. To manually start OVS, enter **yes**, and the system will enter into the system shell as follows:

```
Do you want start the OVS by manual? (yes/no) yes
You need start the OVS by manual!
root@PicOS-OVS#
root@PicOS-OVS#
```

Specify the configuration database file, which contains the configurations needed by OVS initialization. You only have to create it once.

The created file is located in **/ovs/ovs-vswitchd.conf.db**.

```
root@PicOS-OVS# ovsdb-tool create /ovs/ovs-vswitchd.conf.db
/ovs/share/openvswitch/vswitch.ovsschema
Nov 13 06:55:55|00001|lockfile|INFO|/ovs/.ovs-vswitchd.conf.db.~lock~: lock file does not exist,
creating
root@PicOS-OVS#
```

Next, configure the IP addresses of the management interface eth0 and its gateway.

You can use DHCP:

```
root@PicOS-OVS# udhcpc
udhcpc (v1.13.3) started
Sending discover...
Sending select for 10.10.50.215...
Lease of 10.10.50.215 obtained, lease time 3600
root@PicOS-OVS#
PHY: 24520:01 - Link is Up - 1000/Full
root@PicOS-OVS#
```

Or, configure them manually:

```
root@PicOS-OVS# ifconfig eth0 10.10.50.215 netmask 255.255.255.0 up
root@PicOS-OVS# route add default gw 10.10.50.1
root@PicOS-OVS#
```

Start the OVS database server. The command's parameters are the configuration database file and the connection method.

```
root@PicOS-OVS# ovsdb-server /ovs/ovs-vswitchd.conf.db --remote=ptcp:6633:10.10.50.215 &
root@PicOS-OVS#
```

Finally, start the OVS daemon:

```
root@PicOS-OVS# ovs-vswitchd tcp:10.10.50.215:6633 --pidfile=ovs-vswitchd.pid --overwrite-
pidfile > /var/log/ovs.log 2>/dev/null &
```

### (2) Automatically starting OVS

You can also choose to start the OVS software automatically: if you enter **no**, the system will enter into the system shell as follows:

```
System initiating...Please wait...
    Please choose which to start: Pica8 XorPlus, OpenFlow, or System shell:
         (Will choose default entry if no input in 10 seconds.)
         [1]  Pica8 XorPlus   * default
         [2]  Open vSwitch
         [3]  System shell
         [4]  Boot menu editor
Enter your choice (1,2,3,4,5):2

Open vSwitch is selected.

  Note: Defaultly, the OVS server is runned with static local management IP and port 6633.
        The default way of vswitch connecting to server is PTCP.
        If you do not want default configuration, choose manual start!

Do you want start the OVS by manual? (yes/no) no

Please set a static IP and netmask for the switch (e.g. 128.0.0.10/24):
```

You can then input the parameters, step by step:

```
Please set a static IP and netmask for the switch (e.g. 128.0.0.10/24) : 10.10.50.215/24

Please set the gateway IP (e.g 172.168.1.2):10.10.50.1

Specify the file name of database for server, if not exist, it will be created:
Choose the default database file /ovs/ovs-vswitchd.conf.db!
System have found the database file!

Waitting for eth0 up ......
Done!

Adding the gateway ......
route: SIOCADDRT: File exists
Run the ovsdb-server with 10.10.50.215 and port 6633 with ptcp ......
Waitting for ovsdb-server ......
Done!

Run the ovs-vswitchd with 10.10.50.215 and port 6633 with ptcp ......
Waitting for ovs-vswitchd ......
Done!

Startup finished!
root@PicOS-OVS#
```

# Chapter 3. Configurating Open vSwitch

This chapter describes the configuration steps of Open vSwitch, including NetFlow, sFlow, 802.1Q VLAN, and monitoring.

## *Connecting to an OpenFlow controller*

With the command **ovs-vsctl** in our example below, the switch connects to an OF controller whose IP address is 10.10.53.50, and whose port number is 6636.

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.215:6633 set-controller br0 tcp:10.10.53.50:6636
root@PicOS-OVS#
```

## *Creating a Bridge and Adding Ports*

You can create one or more bridges in a Pica8 switch. Each physical port can be added to only one bridge.

### (1) Creating a bridge and adding ports

In our following example, we create a bridge br0 and add ports ste-1/1/1, te-1/1/2, and te-1/1/3 to br0. The default VLAN ID for each port is 1.

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.215:6633 add-br br0 -- set bridge br0
datapath_type=pica8
device br0 entered promiscuous mode
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.215:6633 add-port br0 te-1/1/1 vlan_mode=access tag=1
-- set Interface te-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.215:6633 add-port br0 te-1/1/2 vlan_mode=access tag=1
-- set Interface te-1/1/2 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.215:6633 add-port br0 te-1/1/3 vlan_mode=access tag=1
-- set Interface te-1/1/3 type=pica8
root@PicOS-OVS#
```

### (2) Configuring the default VLAN ID for a port

Here we add the port te-1/1/3 to bridge br0. The default VLAN ID is 1000.

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.215:6633 add-port br0 te-1/1/3 vlan_mode=trunk tag=1
trunks=1000 -- set Interface te-1/1/3 type=pica8
root@PicOS-OVS#
```

### (3) Displaying bridge information

```
root@PicOS-OVS# ovs-ofctl show br0
OFPT_FEATURES_REPLY (xid=0x1): ver:0x1, dpid:0000e89a8f503d30
n_tables:1, n_buffers:256
features: capabilities:0x87, actions:0x3f
 1(ge-1/1/1): addr:e8:9a:8f:50:3d:30
     config:     0
     state:      LINK_DOWN
     current:    10MB-FD COPPER AUTO_NEG AUTO_PAUSE AUTO_PAUSE_ASYM
     advertised: 10MB-FD AUTO_PAUSE
     supported:  10MB-HD 10MB-FD 100MB-HD 100MB-FD 1GB-FD AUTO_NEG AUTO_PAUSE AUTO_PAUSE_ASYM
     peer:       10MB-FD AUTO_PAUSE
 2(ge-1/1/2): addr:e8:9a:8f:50:3d:30
     config:     0
     state:      LINK_DOWN
     current:    10MB-FD COPPER AUTO_NEG AUTO_PAUSE AUTO_PAUSE_ASYM
     advertised: 10MB-FD AUTO_PAUSE
     supported:  10MB-HD 10MB-FD 100MB-HD 100MB-FD 1GB-FD AUTO_NEG AUTO_PAUSE AUTO_PAUSE_ASYM
```

```
     peer:       10MB-FD AUTO_PAUSE
 3(ge-1/1/3): addr:e8:9a:8f:50:3d:30
     config:     0
     state:      LINK_DOWN
     current:    10MB-FD COPPER AUTO_NEG AUTO_PAUSE AUTO_PAUSE_ASYM
     advertised: 10MB-FD AUTO_PAUSE
     supported:  10MB-HD 10MB-FD 100MB-HD 100MB-FD 1GB-FD AUTO_NEG AUTO_PAUSE AUTO_PAUSE_ASYM
     peer:       10MB-FD AUTO_PAUSE
 LOCAL(br0): addr:e8:9a:8f:50:3d:30
     config:     PORT_DOWN
     state:      LINK_DOWN
     current:    10MB-FD COPPER
OFPT_GET_CONFIG_REPLY (xid=0x3): frags=normal miss_send_len=0
root@PicOS-OVS#
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.200:6633 list-ports br0
ge-1/1/1
ge-1/1/2
ge-1/1/3
root@PicOS-OVS#
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.200:6633 list-ifaces br0
ge-1/1/1
ge-1/1/2
ge-1/1/3
root@PicOS-OVS#
root@PicOS-OVS#
```

**(4) Deleting a port from a bridge, and deleting a bridge**

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.215:6633 del-port br0 te-1/1/3
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.215:6633 del-br br0
```

## *Configuring the Link Speed of a Port*

In PicOS OVS, you can configure the link speed of each port as follows:

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.215:6633 add-port br0 te-1/1/1 vlan_mode=access tag=1
type=pica8 options:link_speed=1G
root@PicOS-OVS#
```

## *Configuring an 802.1Q Trunk Port*

In PicOS OVS, each port has its default VLAN ID. If you want a port to belong to more than one VLAN, you can configure the port to trunk mode.

**(1) Configuring a trunk port for multiple VLANs**

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.215:6633 add-port br0 te-1/1/1 vlan_mode=trunk
trunks=100,200,300 -- set Interface te-1/1/1 type=pica8
root@PicOS-OVS#
```

## *Configuring sFlow*

PicOS OVS supports sFlow v5. Configure sFlow as follows:

**(1) Configuring sFlow**

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.215:6633 -- --id=@s create sFlow agent=eth0
target=\"10.10.50.207:9901\" header=128 sampling=64 polling=10 -- set Bridge br0 sflow=@s
```

```
root@PicOS-OVS#
```

In the CLI above, the parameters are shown as follows:

COLLECTOR_IP=10.10.50.207
COLLECTOR_PORT=9901
AGENT_IP=eth0
HEADER_BYTES=128
SAMPLING_N=64
POLLING_SECS=10

**(2) Deleting sFlow**

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.215:6633 -- clear Bridge br0 sflow
root@PicOS-OVS#
```

## *Configuring NetFlow*

PICOS OVS supports NetFlow. Configure NetFlow as shown below:

**(1) Configuring NetFlow**

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.215:6633 -- set Bridge br0 netflow=@nf -- --id=@nf
create NetFlow targets=\"10.10.50.207:5566\" active-timeout=30
root@PicOS-OVS#
```

In the CLI above, the parameters are shown as follows:

COLLECTOR_IP=10.10.50.207
COLLECTOR_PORT=5566
ACTIVE_TIMEOUT=30

**(2) Deleting NetFlow**

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.215:6633 --  clear Bridge br0 netflow
```

## *Configuring Mirroring*

PicOS OVS supports Mirroring. Configure Mirroring as shown below:

**(1) Configuring Mirroring**

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.215:6633 -- set bridge br0 mirrors=@m -- --id=@te-
1/1/1 get Port te-1/1/1 -- --id=@te-1/1/2 get Port te-1/1/2 -- --id=@te-1/1/3 get Port te-1/1/3 -
- --id=@m create Mirror name=mymirror select-dst-port=@te-1/1/1,@te-1/1/2 select-src-port=@te-
1/1/1,@te-1/1/2 output-port=@te-1/1/3
root@PicOS-OVS#
```

- In the configuration above, we configured te-1/1/1, te-1/1/2, and te-1/1/3 as mirror ports. The source ports are te-1/1/1 and te-1/1/2 (including the ingress and egress), and the output port (monitor port) is te-1/1/3.
- **select-dst-port** denotes that the packet (in the switching chip) will go out from the specified port (egress).

- **select-src-port** denotes that the packet will enter the specified port (ingress).

**(2) Deleting Mirroring**

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.215:6633 -- remove bridge br0 mirrors mymirror
```

## *Configuring IPv4 Flows*

PicOS OVS supports IPv4 flows in OpenFlow.

**(1) Creating an IPv4 flow**

```
root@PicOS-OVS# ovs-ofctl add-flow br0
dl_src=22:11:11:11:11:11,dl_dst=22:00:00:00:00:00,in_port=1,dl_type=0x0800,nw_src=128.1.1.1,nw_ds
t=128.1.1.2,nw_proto=6,actions=output:2,3,4
root@PicOS-OVS#
root@PicOS-OVS# ovs-ofctl dump-flows br0
NXST_FLOW reply (xid=0x4):
cookie=0x0, duration=12.758s, table=0, n_packets=0, n_bytes=0,
tcp,in_port=1,dl_src=22:11:11:11:11:11,dl_dst=22:00:00:00:00:00,nw_src=128.1.1.1,nw_dst=128.1.1.2
actions=output:2,output:3,output:4
cookie=0x0, duration=2180.111s, table=0, n_packets=0, n_bytes=0, priority=0 actions=NORMAL
root@PicOS-OVS#
```

**(2) Deleting an IPv4 flow**

```
root@PicOS-OVS# ovs-ofctl del-flows br0 dl_src=22:11:11:11:11:11,dl_dst=22:00:00:00
:00:00,in_port=1,dl_type=0x0800,nw_src=128.1.1.1,nw_dst=128.1.1.2,nw_proto=6
root@PicOS-OVS#
```

**(3) Removing all flows**

```
root@PicOS-OVS# ovs-ofctl del-flows br0
root@XorPlus
```

## *Configuring GRE Tunnel*

PicOS OVS supports IP GRE Tunnel.

**(1) Creating a GRE tunnel**

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.243:6633 add-port br0 gre1 -- set Interface gre1
type=pica8_gre options:remote_ip=10.10.60.10 options:local_ip=10.10.61.10 options:vlan=1
options:src_mac=00:11:11:11:11:11 options:dst_mac=00:22:22:22:22:22 options:egress_port=ge-1/1/5
```

To create a GRE tunnel, configure a GRE tunnel and two flows, which are used for sending traffic to GRE, and sending output from GRE.

```
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,actions=output:109
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=5,actions=mod_dl_src:00:11:11:11:11:11,
mod_dl_dst:00:33:33:33:33:33,output:1
```

The GRE ports start at 109, which is the port number of GRE1. The first flow displayed above indicates that all traffic from port ge-1/1/1 will be sent to the GRE tunnel whose port number is 109. The second flow indicates that all traffic from the GRE tunnel will be forwarded to port ge-1/1/1. You must configure the MAC address of the traffic that is sent out.

## *Configuring MPLS*

- PicOS OVS supports MPLS, which is specified in OpenFlow v1.2. The basic actions of MPLS are push, swap, and pop.
- You can add flows to modify and copy the MPLS TTL and IP TTL.
- In the current version, you can push up to 2 MPLS labels for a flow.
- Note that every untagged packet will be tagged with the default VLAN ID before push, swap, and pop.

**(1) Pushing an MPSL header for flows**

Below we specify a flow, which should match `{in_port=1,dl_type=0x0800,dl_src=22:11:11:11:11:11,dl_dst=22:00:00:00:00:00,dl_vlan=1}`. The action is to push a MPLS header whose label is 10, and forward the packet to port te-1/1/2. Note that the MPLS TTL will copy from the IP header and decrease.

```
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,dl_type=0x0800,dl_src=22:11:11:11:11:11
,dl_dst=22:00:00:00:00:00,dl_vlan=1,actions=push_mpls:0x8847,set_field:10-\>mpls_label
,output:2
```

**(2) Pushing two MPSL headers for flows**

Below we specify a flow, which should match `{in_port=1,dl_type=0x0800,dl_src=22:11:11:11:11:11,dl_dst=22:00:00:00:00:00,dl_vlan=1}`. The action is to push two MPLS headers, whose labels are 10 and 20, and forward the packet to port te-1/1/2.

```
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,dl_type=0x0800,dl_src=22:11:11:11:11:11
,dl_dst=22:00:00:00:00:00,dl_vlan=1,actions= push_mpls:0x8847,set_field:10-\>mpls_label,
set_field:20-\>mpls_label,output:2
root@PicOS-OVS#
```

**(3) Swapping an MPLS packet**

Below we specify a flow, which should match `{in_port=1,dl_type=0x0800,dl_src=22:11:11:11:11:11,dl_dst=22:00:00:00:00:00,dl_vlan=1,mpls_label=10}`. The action is to swap and set the Label as 20, and forward the packet to port te-1/1/2.

```
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=1,dl_type=0x8847,dl_src=22:11:11:11:11:11,dl_dst=22:00:00:00:00:00,dl_vlan=1,dl_type=0x88
47,mpls_label=10,actions= set_field:20-\>mpls_label,output:2
root@PicOS-OVS#
```

**(4) Popping an MPLS header for flows**

Below we specify a flow, which should match `{in_port=1,dl_type=0x0800,dl_src=22:11:11:11:11:11,dl_dst=22:00:00:00:00:00,dl_vlan=1,mpls_label=10}`. The action is to pop the MPLS header and forward the packet to port te-1/1/2. Note that the MPLS TTL will be copied to the IP header TTL and decrease.

```
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,dl_type=0x8847,dl_src=22:11:11:11:11:11
,dl_dst=22:00:00:00:00:00,dl_vlan=1,mpls_label=10,actions=pop_mpls:0x8847,output:2
```

**(5) Popping an MPLS header for flows that have two MPLS headers**

Below we specify a flow that has two MPLS headers (10 and 20). The pop action always pops the outer MPLS header (note that only this label is popped). The output packet is also an MPLS packet, and "pop_mpls:0x8847" must be configured.

```
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,dl_type=0x8847,dl_src=22:11:11:11:11:11
,dl_dst=22:00:00:00:00:00,dl_vlan=1,mpls_label=10,actions=pop_mpls:0x8847,output:2
```

**(6) Popping two MPLS headers for flows that have two MPLS headers**

Below we specify a flow that has two labels to pop. The output flow is the IP packet. Configure two pop entries to pop the flow.

```
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,dl_type=0x8847,dl_src=22:11:11:11:11:11
,dl_dst=22:00:00:00:00:00,dl_vlan=1,actions=pop_mpls:0x0800,output:2
```

## *Configuring LAG and LACP ports*

- PicOS OVS supports LAG and LACP.
- PicOS can support 24 LAGs or LACPs at most. Each LAG can have up to 8 member ports.

**(1) Creating a static LAG**

In the following configuration, we create LAG ae1, and add port 2 and port 3 to this LAG.

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.156:6633 add- port br0 ae1 vlan_mode=trunk tag=1
trunks=2000,4094 -- set Interface ae1 type=pica8_lag
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.156:6633 -- set Interface ae1 options:lag_type=static
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.156:6633 set Interface ae1 options:members=ge-
1/1/2,ge-1/1/3
```

**(2) Creating an LACP port**

You can create an LACP port and configure its parameters.

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.156:6633 add-port br0 ae1 vlan_mode=trunk tag=1
trunks=2000,4094 -- set Interface ae1 type=pica8_lag
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.156:6633 -- set Interface ae1 options:lag_type=lacp
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.156:6633 set Interface ae1 options:members=ge-
1/1/2,ge-1/1/3

root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.156:6633 -- set Interface ae1 options:lacp-system-
id=00:11:11:11:11:11
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.156:6633 -- set Interface ae1 options:lacp-system-
priority=32768
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.156:6633 -- set Interface ae1 options:lacp-time=fast
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.156:6633 -- set Interface ae1 options:lacp-time=slow
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.156:6633 -- set Interface ae1 options:lacp-
mode=active
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.156:6633 -- set Interface ae1 options:lacp-
mode=passive

root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.156:6633 -- set Interface ge-1/1/2 options:lacp-port-
id=2
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.156:6633 -- set Interface ge-1/1/2 options:lacp-port-
priority=32768
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.156:6633 -- set Interface ge-1/1/2 options:lacp-
aggregation-key=0
```

**(3) Creating static flows for LAG or LACP**

You can create a static flow whose output port is a LAG or LACP port.

```
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=53,actions=output:1    (P3290/P3295/P3920)
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,actions=output:53
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=49,actions=output:1    (P3780)
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,actions=output:49

LAG number index is shown as following:
For the P-3290, P-3295, P-3920, lag number index is as follow.
-------------------------------------------------
|  lag  name          | ae1 |  ae2 |  ...  | ae24 |
-------------------------------------------------
|  lag number index   | 53  |  54  |  ...  |  76  |
-------------------------------------------------

For the P-3780, lag number index is as follow.
-------------------------------------------------
|  lag  name          | ae1 |  ae2 |  ...  | ae24 |
-------------------------------------------------
|  lag number index   |  49 |  50  |  ...  |  72  |
-------------------------------------------------
```

**(4) Displaying LACP information**

You can display LACP information with the command below:

```
root@PicOS-OVS# ovs-appctl -t ovs-vswitchd lacp/show
```

## *Configuring Group Tables*

- PicOS OVS supports group tables in OpenFlow v1.2.
- Due to ASIC limitations, not all buckets in a group table will be installed in the ASIC chip for a flow. The system will install as many as buckets as possible in the ASIC chip.

**(1) Creating a group table**

You can create a group table, and a flow whose action is to execute the group table.

1) type=all

```
root@PicOS-OVS# ovs-ofctl add-group br0 group_id=2238,type=all,bucket=output:2
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=1,dl_src=22:11:11:11:11:11,dl_dst=22:00:00:00:00:00,
dl_type=0x0800,nw_proto=6,nw_src=1.1.2.100,nw_dst=2.2.2.100,actions=group:2238
```

2) type=indirect

```
root@PicOS-OVS# ovs-ofctl add-group br0 group_id=2239,type=indirect,bucket=output:2
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=1,dl_src=22:11:11:11:11:11,dl_dst=22:00:00:00:00:00,
dl_type=0x0800,nw_proto=6,nw_src=1.1.2.100,nw_dst=2.2.2.100,actions=group:2239
```

3) type=fast_failover

```
root@PicOS-OVS# ovs-ofctl add-group br0 group_id=2,type= all,bucket=output:2
root@PicOS-OVS# ovs-ofctl add-group br0 group_id=3,type= all,bucket=output:3
root@PicOS-OVS# ovs-ofctl add-group br0 group_id=4,type= fast_failover,
bucket=watch_port:2,watch_group:2,output:4,bucket=watch_port:3,watch_group:3,output:5
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=1,dl_src=22:11:11:11:11:11,dl_dst=22:00:00:00:00:00,
dl_type=0x0800,nw_proto=6,nw_src=1.1.2.100,nw_dst=2.2.2.100,actions=group:4
```

**(2) Modifying buckets in a group table**

```
root@PicOS-OVS# ovs-ofctl mod-group br0 group_id=2238,type=all,bucket=output:3
root@PicOS-OVS# ovs-ofctl mod-group br0 group_id=2238,type=all,bucket=output:2,bucket=output:3
root@PicOS-OVS# ovs-ofctl mod-group br0
group_id=2238,type=all,bucket=mod_dl_src:22:11:11:22:22:22,
mod_dl_dst:22:00:00:11:11:11,output:2,bucket=mod_dl_src:22:11:11:22:22:22,mod_dl_dst:22:00:00:11:
11:11,output:3
```

**(3) Deleting a group table**

Delete a group table with the following command:

```
root@PicOS-OVS# ovs-ofctl del-group br0 group_id=2238
```

**(4) Displaying group table information**

Display information for the group table:

```
root@PicOS-OVS# ovs-ofctl dump-groups-desc br0
root@PicOS-OVS# ovs-ofctl dump-groups-stats br0 group_id=2238
root@PicOS-OVS# ovs-ofctl dump-groups-stats br0 group_id=all
root@PicOS-OVS# ovs-ofctl dump-groups-features br0
```

## *Configuring a Meter*

PicOS OVS supports meters in OpenFlow v1.3.

**(1) Creating a meter**

1) type=drop

a. `without burst_size`

30M packets will be forwarded to port 2; the support sending rate is 100M.

```
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,actions=output:2,meter:100
root@PicOS-OVS# ovs-ofctl -O OpenFlow13 add-meter br0
meter_id=100,flags=KBPS,band=type:drop,rate:30000
```

b. `with burst_size`

30M packets will be forwarded to port 2; the support sending rate is 100M.

```
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,actions=output:2,meter:100
root@PicOS-OVS# ovs-ofctl -O OpenFlow13 add-meter br0
meter_id=2,flags=KBPS/BURST,band=type:drop,rate:30000,burst_size=30000
```

2) type=dscp_remark

a. `without burst_size`

The DSCP value of 70M packets is changed to 14; the support sending rate is 100M.

```
root@PicOS-OVS# ovs-ofctl -O OpenFlow13 add-meter br0
meter_id=2,flags=KBPS/BURST,band=type:dscp_remark,rate:30000, prec_level:14
```

b. `with burst_size`

The DSCP value of 70M packets is changed to 14; the support sending rate is 100M.

```
root@PicOS-OVS# ovs-ofctl -O OpenFlow13 add-meter br0
meter_id=2,flags=KBPS/BURST,band=type:dscp_remark,rate:30000, prec_level:14,burst_size=30000
```

### (2) Modifying a meter

```
root@PicOS-OVS# ovs-ofctl -O OpenFlow13 mod-meter br0
meter_id=2,flags=KBPS/BURST,band=type:dscp_remark,rate:30000,  prec_level:12

root@PicOS-OVS# ovs-ofctl -O OpenFlow13 mod-meter br0
meter_id=2,flags=KBPS/BURST,band=type:drop,rate:10000,burst_size=30000
```

### (3) Deleting a meter

```
root@PicOS-OVS# ovs-ofctl del-meter br0
root@PicOS-OVS# ovs-ofctl del-meter br0 meter_id=1
```

### (4) Displaying meter information

```
root@PicOS-OVS# ovs-ofctl -O OpenFlow13 dump-meters-features br0
root@PicOS-OVS# ovs-ofctl -O OpenFlow13 dump-meters-cfg br0
root@PicOS-OVS# ovs-ofctl -O OpenFlow13 dump-meters-stats br0
```

## *Configuring PBB*

P-3920, P-3922, and P-3780 all support PBB, in OpenFlow v1.3.

### (1) Push

#### 1) Push pbb_isid,eth_src,eth_dst

Outer src mac is set as 00:00:00:11:11:11; dst mac is set as 00:00:00:22:22:22;
Vlan is set as 4094; pbb isid is set as 23.

```
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=11,dl_type=0x0800,dl_src=22:11:11:11:11:11,dl_dst=22:22:22:22:22:22,actions=push_pbb:0x88
e7,set_field:23-\>pbb_isid,set_field:00:00:00:11:11:11-\>eth_src,set_field:00:00:00:22:22:22-
\>eth_dst,push_vlan:0x8100,set_field:4094-\>vlan_vid,output:12
```

#### 2) Push PBB without pbb_isid,eth_src,eth_dst

Outer src mac is set as 22:11:11:11:11:11; dst mac is set as 22:22:22:22:22:22;
Vlan is set as 4094; pbb isid is set as 0.

```
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=11,dl_type=0x0800,dl_src=22:11:11:11:11:11,dl_dst=22:22:22:22:22:22,actions=push_pbb:0x88
e7,push_vlan:0x8100,set_field:4094-\>vlan_vid,output:12
```

#### 3) Push pbb_isid,eth_src,eth_dst for PBB packets

Outer src mac is set as 00:00:00:11:11:11; dst mac is set as 00:00:00:22:22:22;
Vlan is set as 4094; pbb isid is set as 21 (isid of primary pbb packet should not be 21).

```
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=11,dl_type=0x88e7,actions=push_pbb:0x88e7,set_field:21-
\>pbb_isid,set_field:00:00:00:11:11:11-\>eth_src,set_field:00:00:00:22:22:22-
\>eth_dst,push_vlan:0x8100,set_field:4094-\>vlan_vid,output:12
```

### (2) Pop

Pop PBB packets tagged with VLAN 1 (these should be primary PBB packets).

```
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=11,dl_type=0x88e7,dl_src=00:00:00:11:11:11,dl_dst=00:00:00:22:22:22,actions=pop_pbb,pop_v
lan,output:12
```

Pop PBB packets tagged with VLAN 2000 (these should be primary PBB packets).

```
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=11,dl_type=0x88e7,dl_src=00:00:00:11:11:11,dl_dst=00:00:00:22:22:22,actions=pop_pbb,pop_v
lan,output:12
```

**Notes:**

- Push PBB should be performed with **push_vlan**.
- When you execute push PBB, **primary src mac** and **dst mac** will be used if no configurations of `eth_src` and/or `eth_dst` exist.
- Use push PBB for PBB packets. The primary PBB ISID should be not same as the push PBB ISID.
- When you use pop PBB, primary packets should include the VLAN ID, and actions should include pop_vlan.

## *Configuring QoS/Queues*

PicOS OVS supports QoS queuing.

```
ovs-ofctl del-flows br0
ovs-ofctl add-flow br0
in_port=1,dl_src=22:11:11:11:11:11,actions=set_queue:0,output=3
ovs-ofctl add-flow br0
in_port=2,dl_src=22:11:11:11:11:12,actions=set_queue:7,output=3
ovs-vsctl --db=tcp:10.10.50.145:6633 -- set port ge-1/1/3 qos=@newqos -- --
id=@newqos create qos type=PRONTO_STRICT queues:0=@newqueue
queues:7=@newqueue1 -- --id=@newqueue create queue other-config:min-
rate=100000000 other-config:max-rate=100000000 other-config:priority=1 -- --
id=@newqueue1 create queue other-config:min-rate=100000000 other-config:max-
rate=100000000 other-config:priority=1

Result:
   Port 3 receive all packets from port 2, and a little from port 1.
   Receive rate of port 3 is about 10Mbps+10Mbps.
```

# Chapter 4. Example Configurations

This chapter provides several configuration examples for 802.1Q.

## *Configuring 802.1Q VLANs*

In the following topology, we need to configure 2 VLANs for switches A and B.

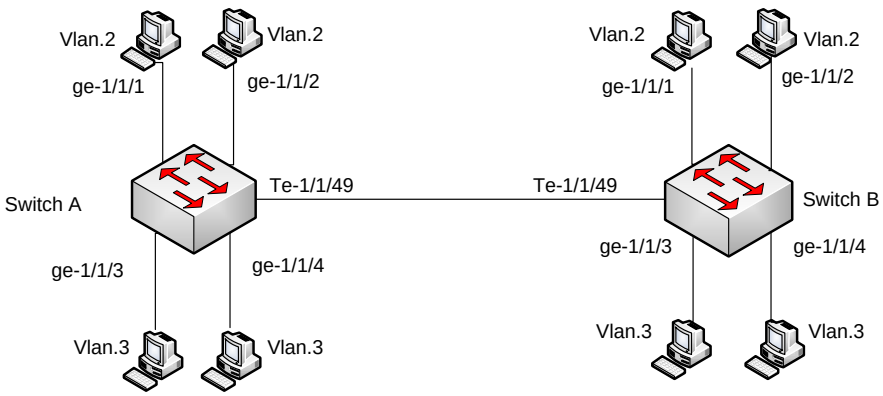


**Figure 4-1. 802.1Q network configuration.**

### (1) Configuring Switch A

For switch A, configure ge-1/1/1~ge-1/1/4 as access ports and te-1/1/49 as a trunk port, because the 10Gbit link will trunk the traffic of VLAN 2 and VLAN 3 .

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.100:6633 add-port br0 te-1/1/1 vlan_mode=access tag=2
-- set Interface te-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.100:6633 add-port br0 te-1/1/2 vlan_mode=access tag=2
-- set Interface te-1/1/2 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.100:6633 add-port br0 te-1/1/3 vlan_mode=access tag=3
-- set Interface te-1/1/3 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.100:6633 add-port br0 te-1/1/4 vlan_mode=access tag=3
-- set Interface te-1/1/4 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.100:6633 add-port br0 te-1/1/49 vlan_mode=trunk
trunks=2,3 -- set Interface te-1/1/49 type=pica8
root@PicOS-OVS#
```

### (2) Configuring Switch B

Configure ge-1/1/1~ge-1/1/4 as access ports and te-1/1/49 as a trunk port, because the 10Gbit link will trunk the traffic of VLAN 2 and VLAN 3.

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.200:6633 add-port br0 te-1/1/1 vlan_mode=access tag=2
-- set Interface te-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.200:6633 add-port br0 te-1/1/2 vlan_mode=access tag=2
-- set Interface te-1/1/2 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.200:6633 add-port br0 te-1/1/3 vlan_mode=access tag=3
-- set Interface te-1/1/3 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.200:6633 add-port br0 te-1/1/4 vlan_mode=access tag=3
-- set Interface te-1/1/4 type=pica8
root@PicOS-OVS#
```

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.200:6633 add-port br0 te-1/1/49 vlan_mode=trunk
trunks=2,3 -- set Interface te-1/1/49 type=pica8
root@PicOS-OVS#
```

## *Configuring GRE Tunnel*

In the following topology, we configure a GRE tunnel between switches A and B. The IP addresses of the GRE tunnel are 10.10.61.10/24 and 10.10.60.10/24.

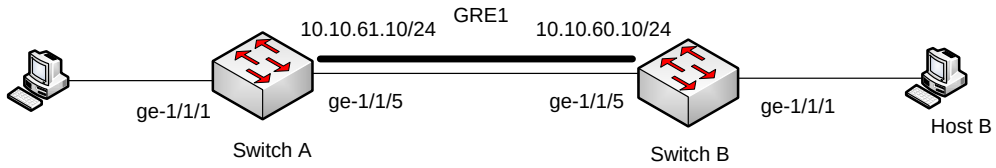


**Figure 4-2. GRE tunnel configuration.**

### (1) Configuring Switch A

For switch A, configure a GRE tunnel and two flows as follows:

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.243:6633 add-br br0 -- set bridge br0
datapath_type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.243:6633 add-port br0 ge-1/1/1 vlan_mode=trunk tag=1
-- set Interface ge-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.243:6633 add-port br0 ge-1/1/5 vlan_mode=trunk tag=1
-- set Interface ge-1/1/5 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.243:6633 add-port br0 gre1 -- set Interface gre1
type=pica8_gre options:remote_ip=10.10.60.10 options:local_ip=10.10.61.10 options:vlan=1
options:src_mac=00:11:11:11:11:11 options:dst_mac=00:22:22:22:22:22 options:egress_port=ge-1/1/5
root@PicOS-OVS#
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,actions=output:109
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=5,actions=mod_dl_src:00:11:11:11:11:11,mod_dl_dst:00:33:33:33:33:33,output:1
```

### (2) Configuring Switch B

Configure a GRE tunnel and two flows as follows:

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.200:6633 add-br br0 -- set bridge br0
datapath_type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.200:6633 add-port br0 ge-1/1/1 vlan_mode=trunk tag=1
-- set Interface ge-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.200:6633 add-port br0 ge-1/1/5 vlan_mode=trunk tag=1
-- set Interface ge-1/1/5 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.200:6633 add-port br0 gre1 -- set Interface gre1
type=pica8_gre options:remote_ip=10.10.61.10 options:local_ip=10.10.60.10 options:vlan=1
options:src_mac=00:22:22:22:22:22 options:dst_mac=00:11:11:11:11:11 options:egress_port=ge-1/1/5
root@PicOS-OVS#
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,actions=output:109
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=5,actions=mod_dl_src:00:22:22:22:22:22,mod_dl_dst:00:66:66:66:66:66,output:1
```

## *Configuring One (1) Label MPLS Network*

- In the following topology, we configure a simple MPLS network. Traffic from host A to host B (denoted by RED arrows) will be forwarded by the MPLS network with Label 10. Traffic from host C to host D (BLUE arrows) will be forwarded by the MPLS network with Label 20.

- Each flow pushes only **one** MPLS header.

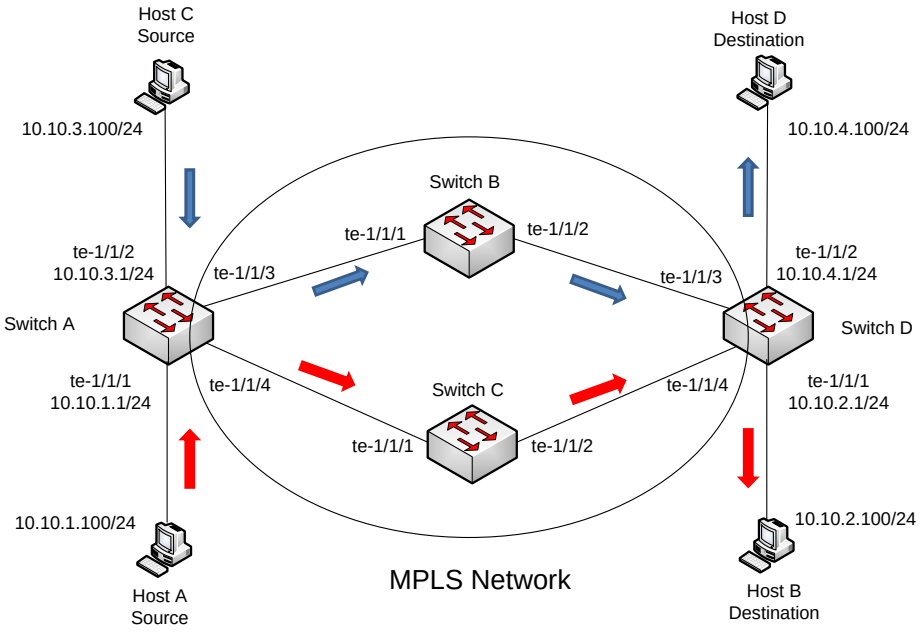


**Figure 4-2. MPLS network configuration.**

### (1) Configuring Switch A

For switch A, configure two flows, which will **push** MPLS Labels 10 and 20 for traffic RED and traffic BLUE, respectively.

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.10:6633 add-br br0 -- set bridge br0
datapath_type=pica8
device br0 entered promiscuous mode
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.10:6633 add-port br0 te-1/1/1 vlan_mode=access tag=1
 -- set Interface te-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.10:6633 add-port br0 te-1/1/2 vlan_mode=access tag=1
 -- set Interface te-1/1/2 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.10:6633 add-port br0 te-1/1/3 vlan_mode=access tag=1
 -- set Interface te-1/1/3 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.10:6633 add-port br0 te-1/1/4 vlan_mode=access tag=1
 -- set Interface te-1/1/4 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,dl_type=0x0800,nw_src=10.10.1.100,
nw_dst=10.10.2.100,dl_vlan=1,actions= push_mpls:0x8847,set_field:10-\>mpls_label
,output:4
root@PicOS-OVS#
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=2,dl_type=0x0800,nw_src=10.10.3.100,nw
_dst=10.10.4.100,dl_vlan=1,actions=push_mpls:0x8847, set_field:20-\>mpls_label,output:3
root@PicOS-OVS#
```

The packet format received by ports te-1/1/1 and te-1/1/2 is shown as follows (ingress):

| Ethernet | IP Header |
|---|---|

The transmitted packet format to ports te-1/1/3 and te-1/1/4 is shown as follows (egress):

| Ethernet | MPLS label 10 | IP Header |
|---|---|---|

| Ethernet | MPLS label 20 | IP Header |
|---|---|---|

**(3) Configuring Switch B**

Configure one flow, which will **swap** the MPLS Label 20 to 200, for traffic BLUE.

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.20:6633 add-br br0 -- set bridge br0
datapath_type=pica8
device br0 entered promiscuous mode
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.20:6633 add-port br0 te-1/1/1 vlan_mode=access tag=1
 -- set Interface te-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.20:6633 add-port br0 te-1/1/2 vlan_mode=access tag=1
 -- set Interface te-1/1/2 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,dl_type=0x08847,nw_src=10.10.3.100,nw
_dst=10.10.4.100,dl_vlan=1,mpls_label=20,actions= set_field:200-\>mpls_label,output:2
root@PicOS-OVS#
```

The transmitted packet format to port te-1/1/2 is shown as follows (egress):

| Ethernet | MPLS label 200 | IP Header |
|---|---|---|

**(4) Configuring Switch C**

Configure one flow, which will **swap** the MPLS Label 10 to 100, for traffic RED.

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.30:6633 add-br br0 -- set bridge br0
datapath_type=pica8
device br0 entered promiscuous mode
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.30:6633 add-port br0 te-1/1/1 vlan_mode=access tag=1
 -- set Interface te-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.30:6633 add-port br0 te-1/1/2 vlan_mode=access tag=1
 -- set Interface te-1/1/2 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,dl_type=0x08847,nw_src=10.10.1.100,nw
_dst=10.10.2.100,dl_vlan=1,mpls_label=10,actions= set_field:100-\>mpls_label,output:2
root@PicOS-OVS#
```

The transmitted packet format to port te-1/1/2 is shown as follows (egress):

| Ethernet | MPLS label 100 | IP Header |
|---|---|---|

**(5) Configuring Switch D**

Configure two flows, which will **pop** MPLS Labels 100 and 200 for traffic RED and traffic BLUE, respectively.

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.40:6633 add-br br0 -- set bridge br0
datapath_type=pica8
device br0 entered promiscuous mode
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.40:6633 add-port br0 te-1/1/1 vlan_mode=access tag=1
 -- set Interface te-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.40:6633 add-port br0 te-1/1/2 vlan_mode=access tag=1
 -- set Interface te-1/1/2 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.40:6633 add-port br0 te-1/1/3 vlan_mode=access tag=1
 -- set Interface te-1/1/3 type=pica8
```

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.40:6633 add-port br0 te-1/1/4 vlan_mode=access tag=1
 -- set Interface te-1/1/4 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=4,dl_type=0x08847,nw_src=10.10.1.100,nw
_dst=10.10.2.100,dl_vlan=1,actions=pop_mpls:0x8847,output:1
root@PicOS-OVS#
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=3,dl_type=0x08847,nw_src=10.10.3.100,nw
_dst=10.10.4.100,dl_vlan=1,actions=pop_mpls:0x8847,output:2
root@PicOS-OVS#
```

The transmitted packet format to ports te-1/1/1 and te-1/1/2 is shown as follows (egress):

| Ethernet | IP Header |
|---|---|

## *Configuring Multiple Virtual Bridges in Your System*

In PicOS OVS, you can create multiple virtual bridges, all independent of each other. One physical port can be added to only one virtual bridge. The flows of each virtual bridge can be configured by its respective controller.

```
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.30:6633 add-br br0 -- set bridge br0
datapath_type=pica8 other-config=datapath-id=0000d80aa99aaaaa
device br0 entered promiscuous mode
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.30:6633 add-port br0 te-1/1/1 vlan_mode=access tag=1
-- set Interface te-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.30:6633 add-port br0 te-1/1/2 vlan_mode=access tag=1
-- set Interface te-1/1/2 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.30:6633 set-controller br0 tcp:10.10.50.1:6633
root@PicOS-OVS#
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.30:6633 add-br br1 -- set bridge br1
datapath_type=pica8 other-config=datapath-id=0000d80bb99bbbbb
device br0 entered promiscuous mode
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.30:6633 add-port br1 te-1/1/3 vlan_mode=access tag=1
-- set Interface te-1/1/3 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.30:6633 add-port br1 te-1/1/4 vlan_mode=access tag=1
-- set Interface te-1/1/4 type=pica8
root@PicOS-OVS# ovs-vsctl --db=tcp:10.10.50.30:6633 set-controller br1 tcp:10.10.50.2:6633
root@PicOS-OVS#
```