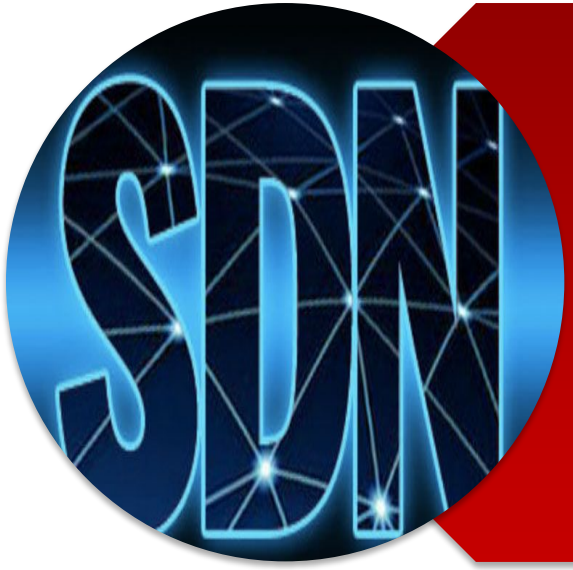
# Lecture 6




## Investigating OpenFlow negotiations

# Objectives

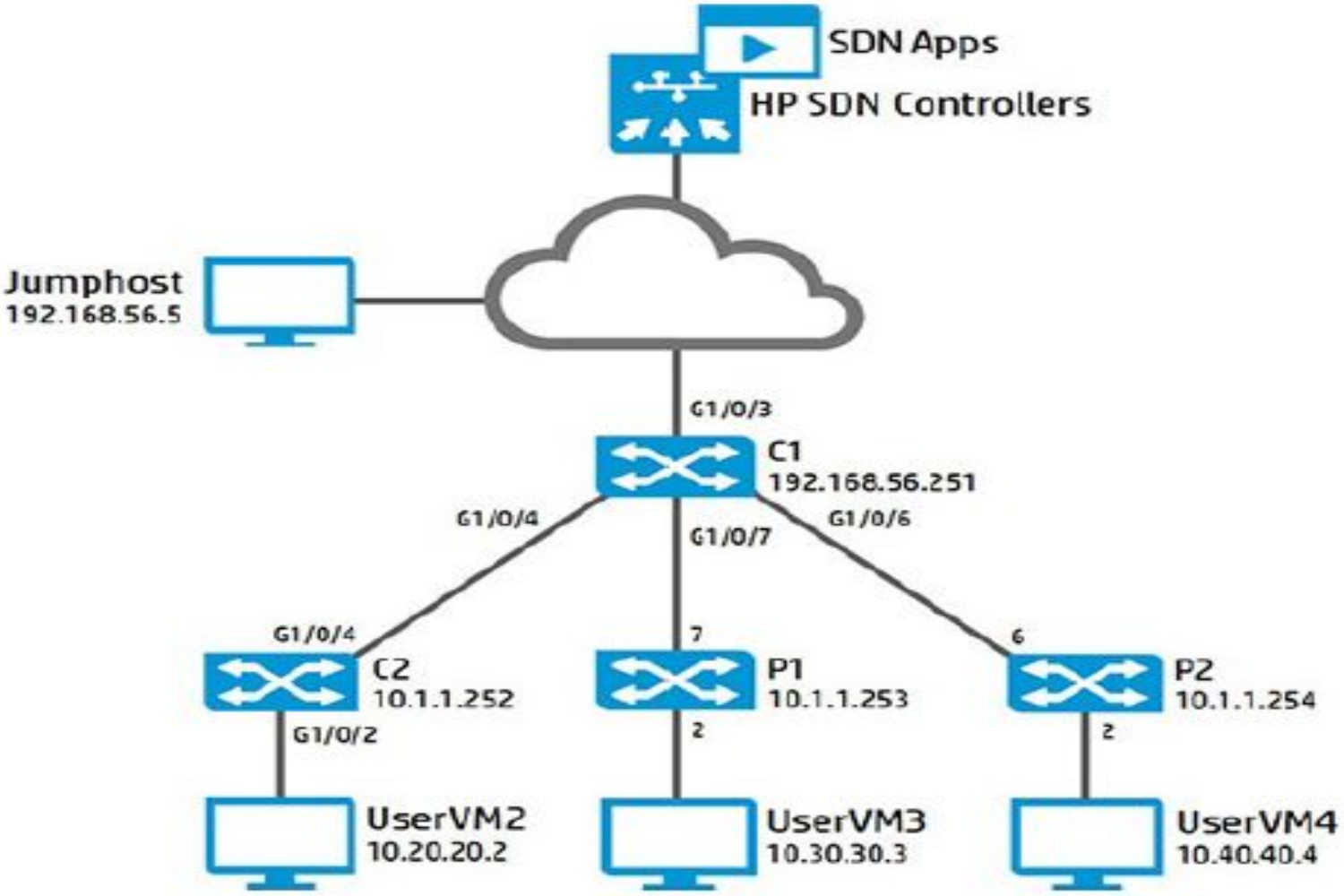
# Introduction to the lecture

- Capture OpenFlow messages
- Switch features
- Switch datapath IDs (DPIDs)
- Packet in messages
- Buffering
- Tables and pipelines
- Flow modifications (Flow mod)
- Error messages


# Investigating OpenFlow negotiations

# Capture OpenFlow messages

# Capture OpenFlow messages

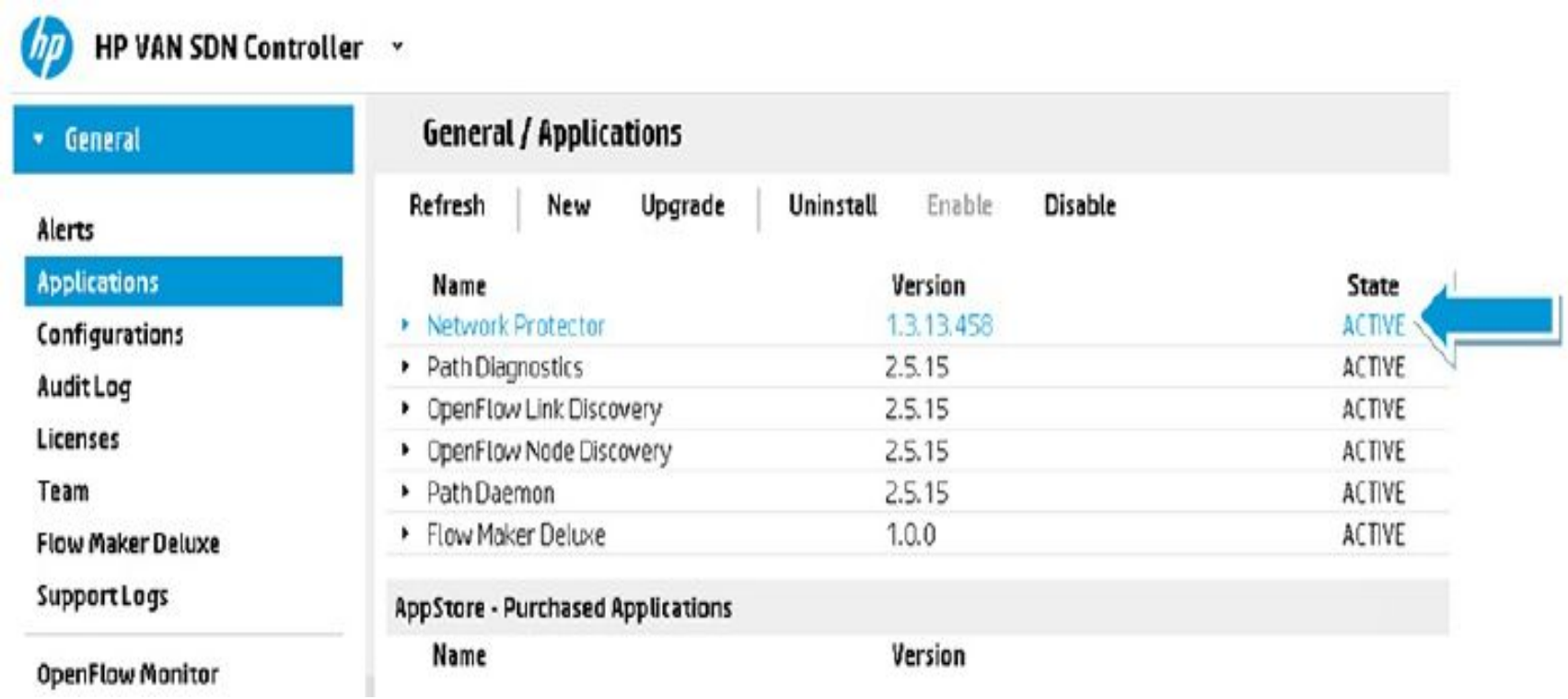


# Capture OpenFlow messages

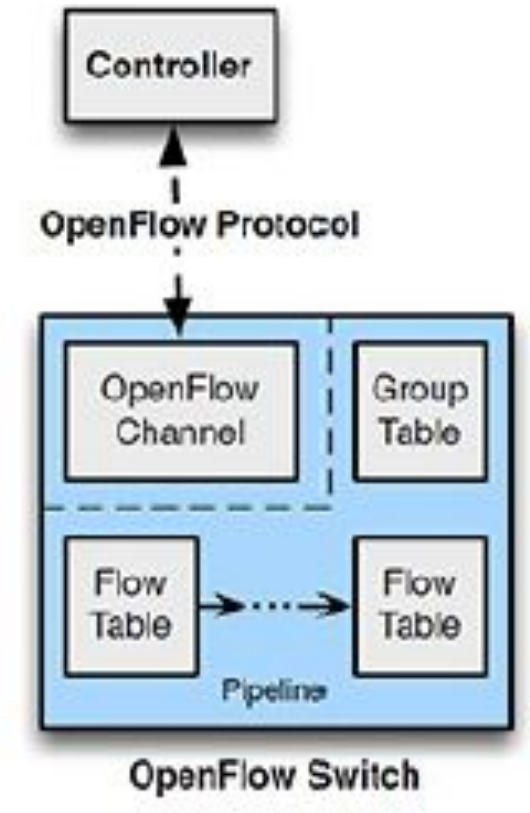
| Device | Description | IP Address |
| --- | --- | --- |
| C1 (Comware switch 1) | HP Comware switch | VLAN 1: 10.1.1.251<br>VLAN 20: 10.20.20.251<br>VLAN 30: 10.30.30.251<br>VLAN 40: 10.40.40.251<br>VLAN 192: 192.168.56.251 |

# Capture OpenFlow messages

**Applications**
**Disable**



# Capture OpenFlow messages




**controller-to-switch, asynchronous, and symmetric.**

# Capture OpenFlow messages

- Switch to Controller = Hello
- Controller to Switch = Hello
- Controller to Switch = Features request
- Switch to Controller = Features reply
- Controller to Switch = Set config
- Switch to Controller = Error

# Capture OpenFlow messages

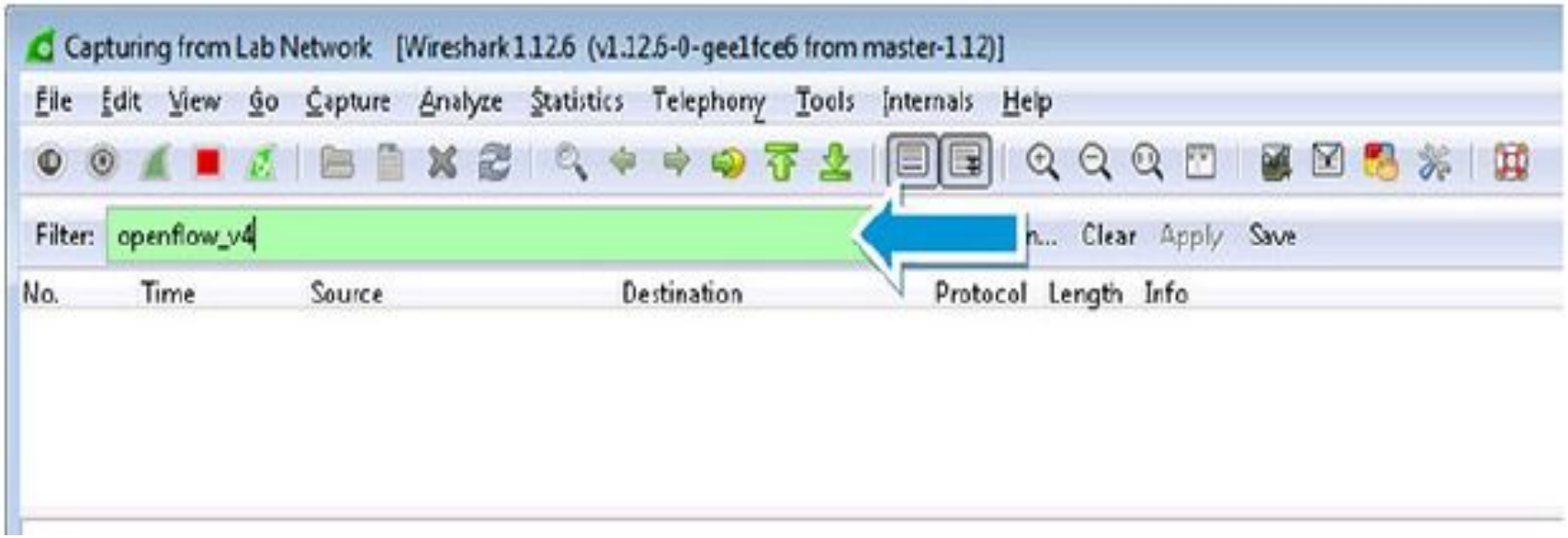
```
<C2> system-view
[C2] openflow instance 1
[C2-of-inst-1] undo classification
[C2-of-inst-1] active instance
```

```
P1# config
P1(config)# openflow
P1(openflow)# disable
```

```
P2# config
P2(config)# openflow
P2(openflow)# disable
```

# Capture OpenFlow messages

openflow_v4

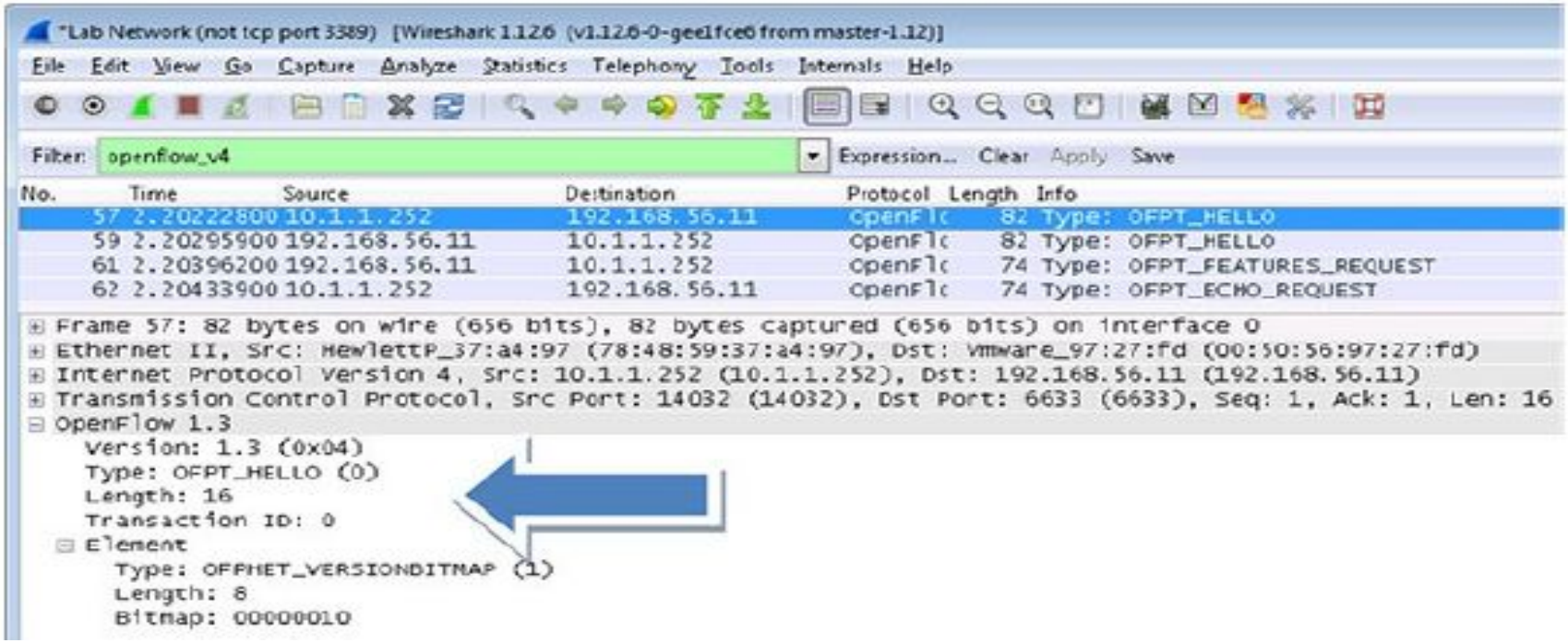



**Result:**

# Capture OpenFlow messages

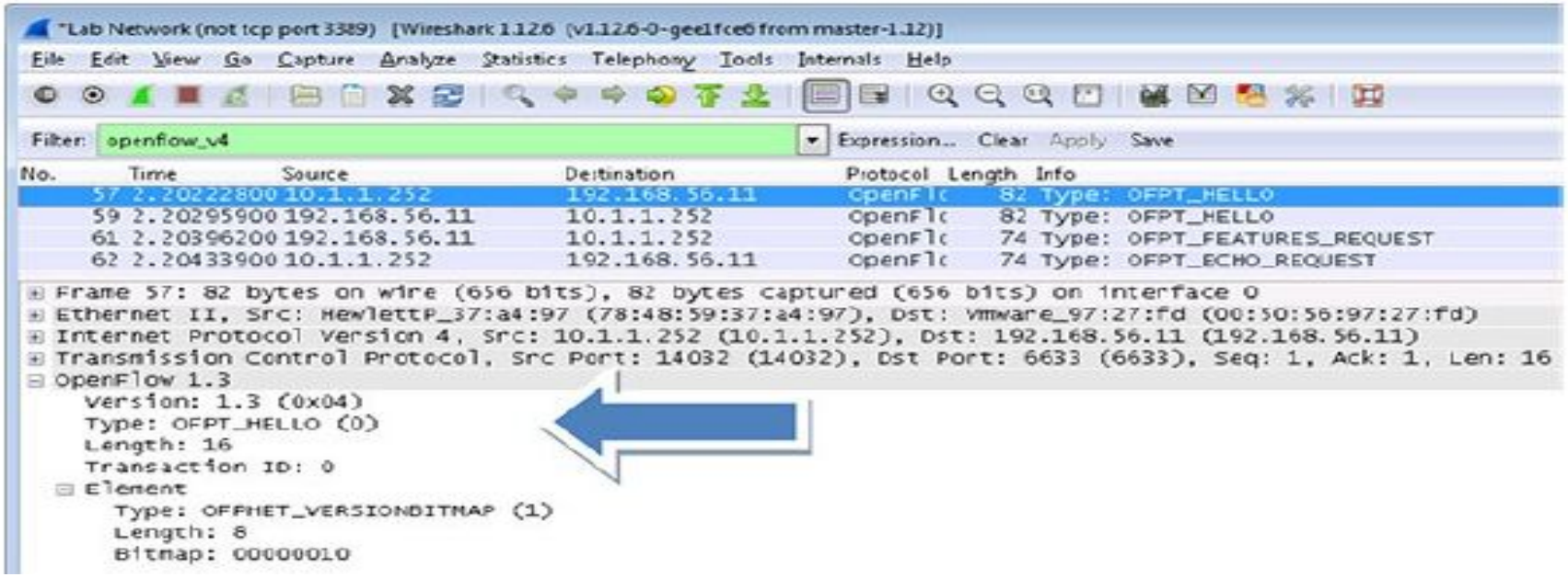
```
[C2-of-inst-1] classification vlan 20
[C2-of-inst-1] active instance
```

# Capture OpenFlow messages




**Answer:**

# Capture OpenFlow messages




**Answer:**

# Message types

```
enum ofp_type {
/* Immutable messages. */
OFPT_HELLO = 0, /* Symmetric message */
OFPT_ERROR = 1, /* Symmetric message */
OFPT_ECHO_REQUEST = 2, /* Symmetric message */
OFPT_ECHO_REPLY = 3, /* Symmetric message */
OFPT_EXPERIMENTER = 4, /* Symmetric message */
/* Switch configuration messages. */
OFPT_FEATURES_REQUEST = 5, /* Controller/switch message */
```

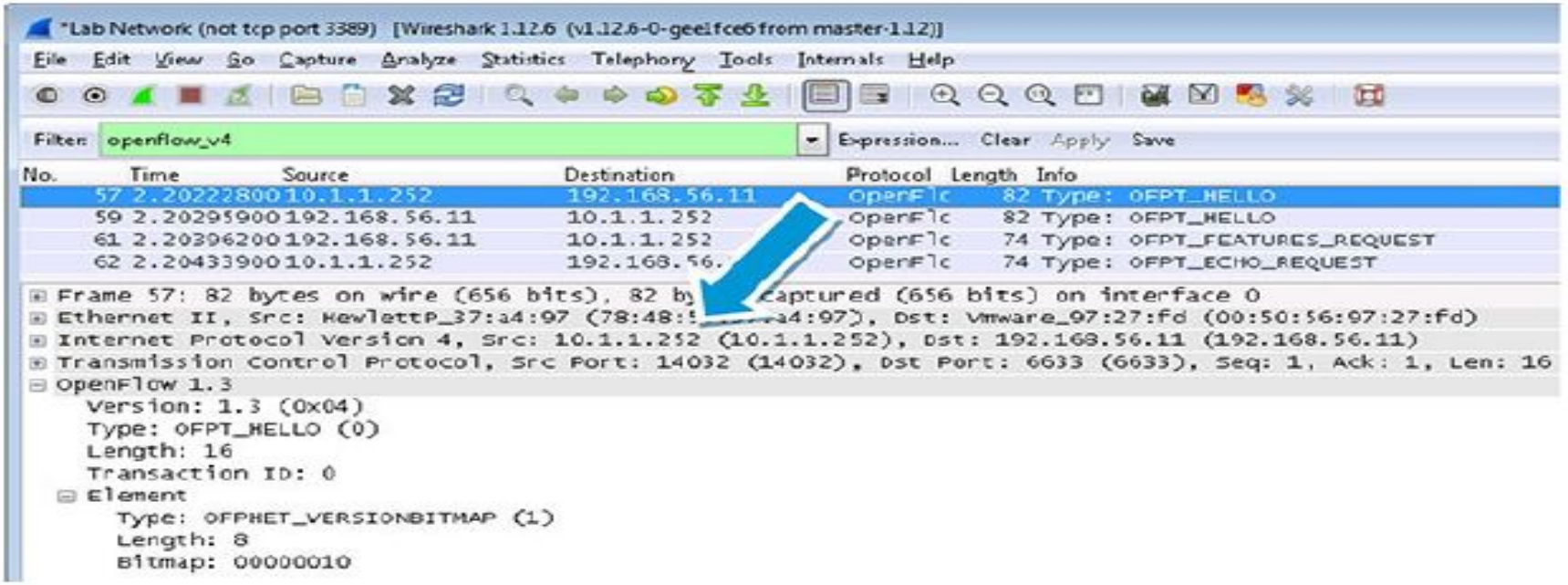
# Message types

```
OFPT_FEATURES_REPLY = 6, /* Controller/switch message */
OFPT_GET_CONFIG_REQUEST = 7, /* Controller/switch message */
OFPT_GET_CONFIG_REPLY = 8, /* Controller/switch message */
OFPT_SET_CONFIG = 9, /* Controller/switch message */
/* Asynchronous messages. */
OFPT_PACKET_IN = 10, /* Async message */
OFPT_FLOW_REMOVED = 11, /* Async message */
OFPT_PORT_STATUS = 12, /* Async message */
/* Controller command messages. */
OFPT_PACKET_OUT = 13, /* Controller/switch message */
OFPT_FLOW_MOD = 14, /* Controller/switch message */
OFPT_GROUP_MOD = 15, /* Controller/switch message */
OFPT_PORT_MOD = 16, /* Controller/switch message */
OFPT_TABLE_MOD = 17, /* Controller/switch message */
/* Multipart messages. */
OFPT_MULTIPART_REQUEST = 18, /* Controller/switch message */
```

# Message types

```
OFPT_MULTIPART_REPLY = 19, /* Controller/switch message */
/* Barrier messages. */
OFPT_BARRIER_REQUEST = 20, /* Controller/switch message */
OFPT_BARRIER_REPLY = 21, /* Controller/switch message */
/* Queue Configuration messages. */
OFPT_QUEUE_GET_CONFIG_REQUEST = 22, /* Controller/switch message */
OFPT_QUEUE_GET_CONFIG_REPLY = 23, /* Controller/switch message */
/* Controller role change request messages. */
OFPT_ROLE_REQUEST = 24, /* Controller/switch message */
OFPT_ROLE_REPLY = 25, /* Controller/switch message */
/* Asynchronous message configuration. */
OFPT_GET_ASYNC_REQUEST = 26, /* Controller/switch message */
OFPT_GET_ASYNC_REPLY = 27, /* Controller/switch message */
OFPT_SET_ASYNC = 28, /* Controller/switch message */
/* Meters and rate limiters configuration messages. */
OFPT METER MOD = 29, /* Controller/switch message */
```

# OpenFlow specification




**Answer:**

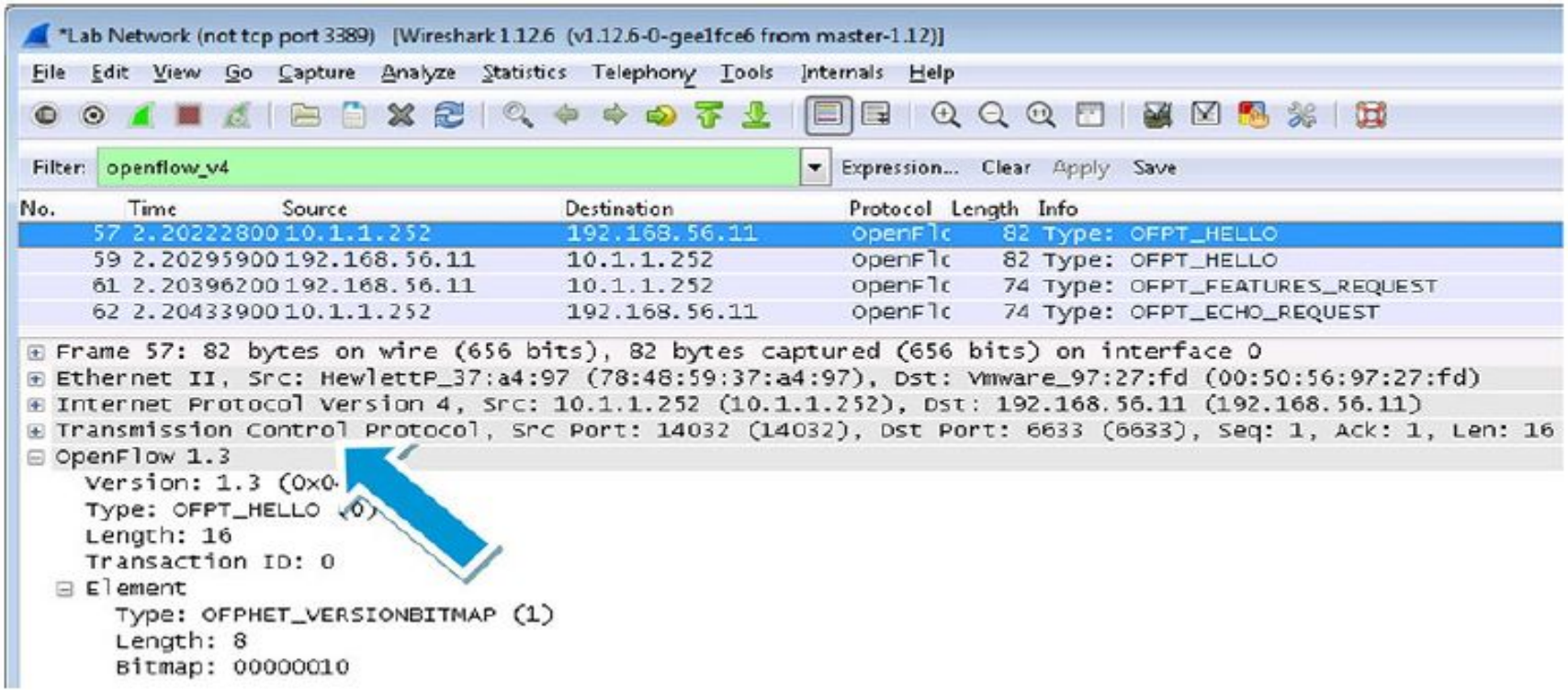
# OpenFlow specification

**Answer:**

**Optional:**

# OpenFlow specification

# OpenFlow specification

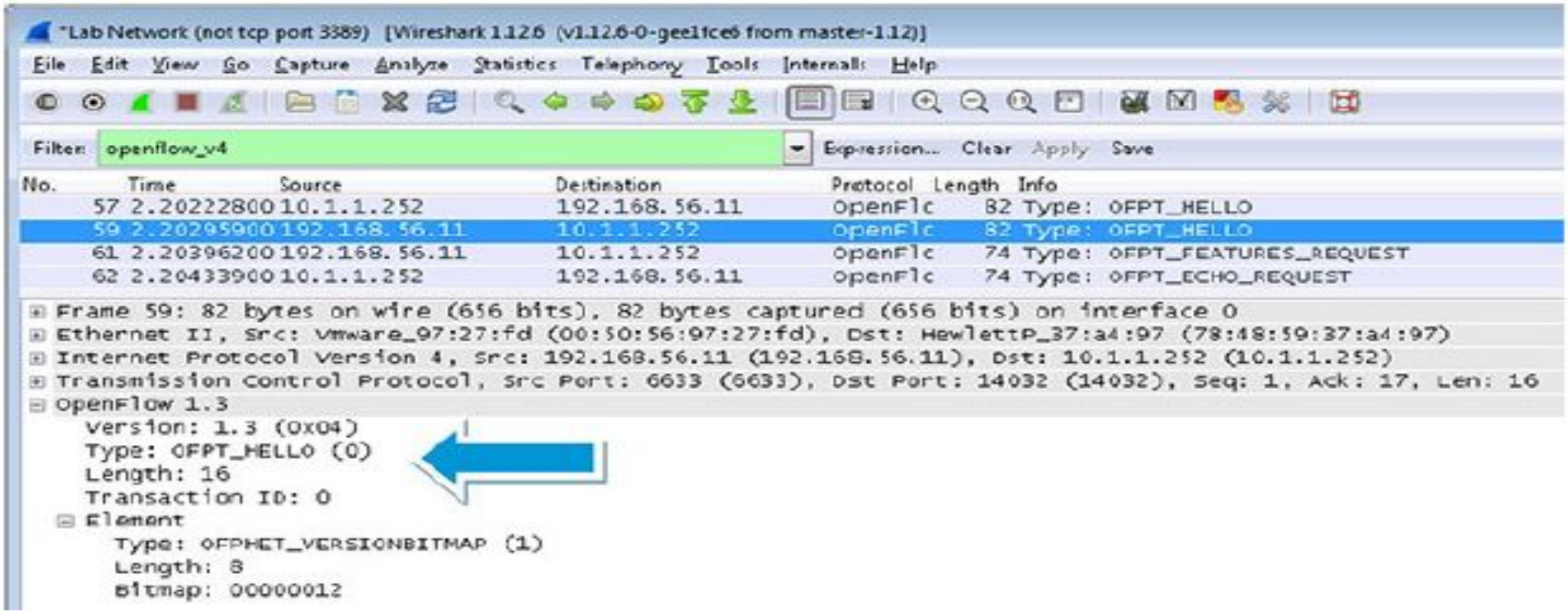



**Answer:**

# OpenFlow specification

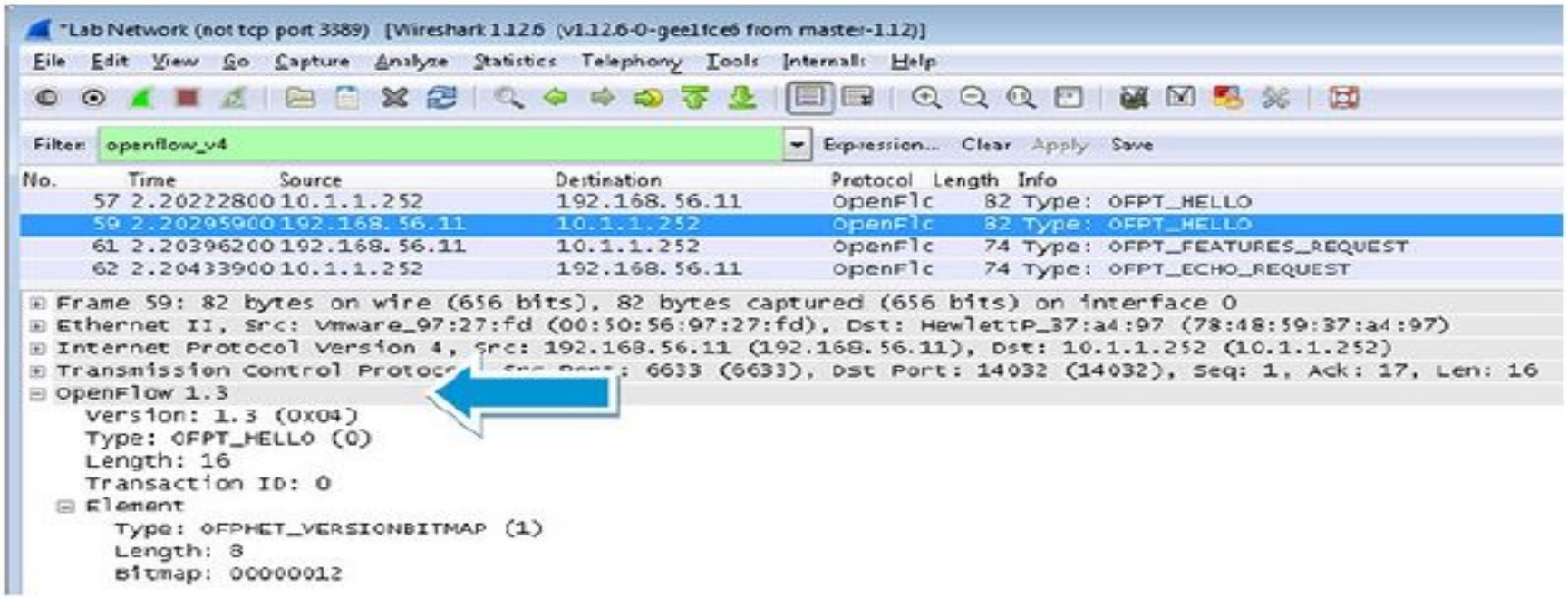
**Answer:**

# OpenFlow specification



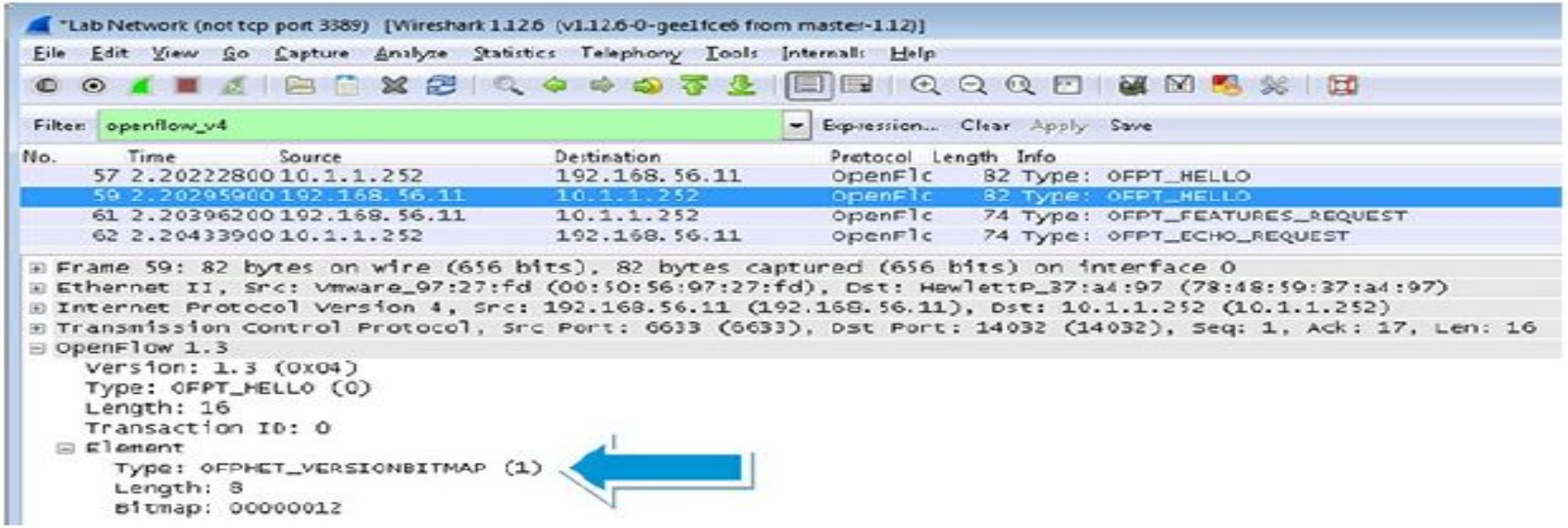

**Result:**

# OpenFlow specification



# OpenFlow specification

**Answers:**

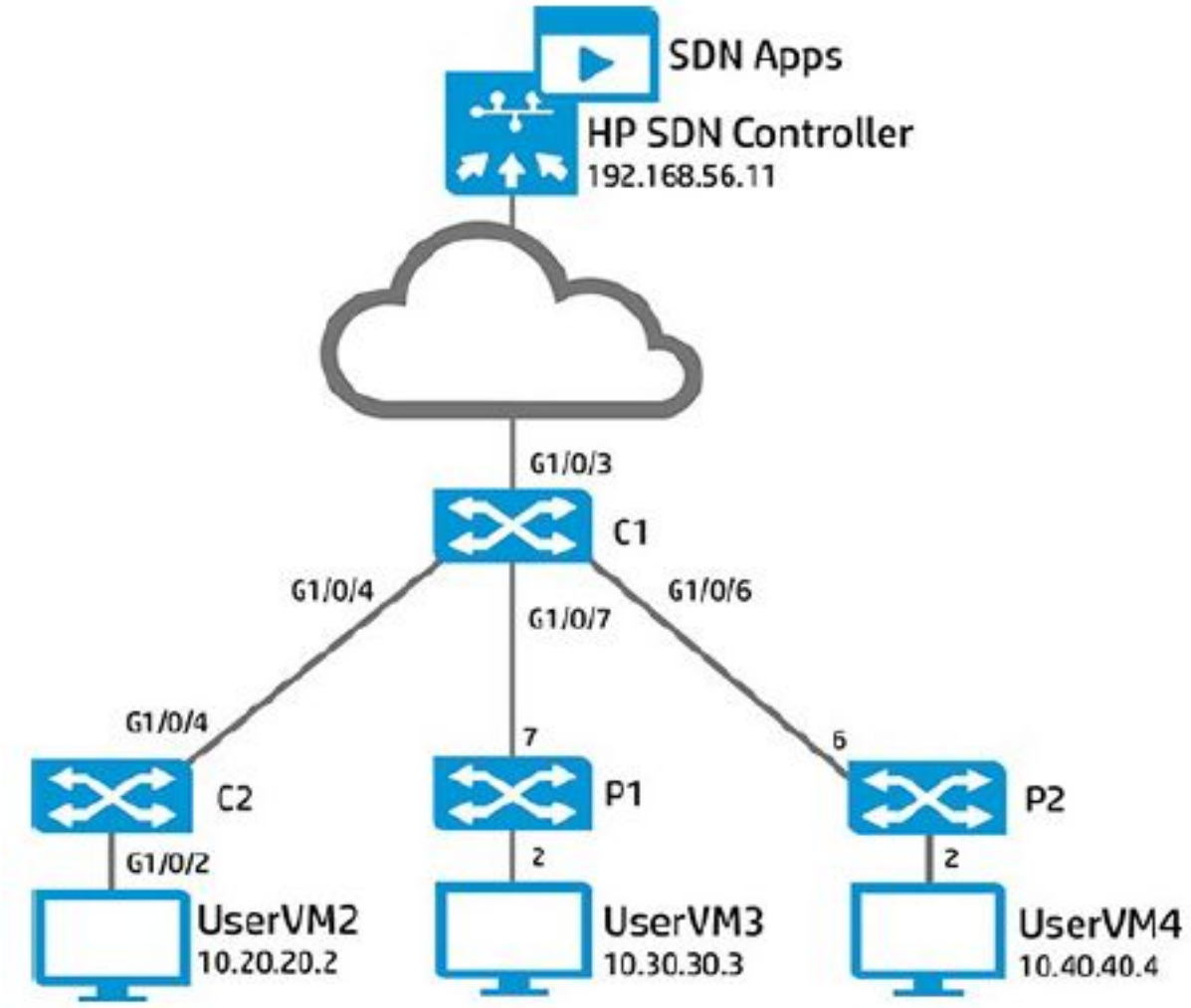



**Answer:**

# OFPT_HELLO message


# Investigating OpenFlow negotiations

# Network topology



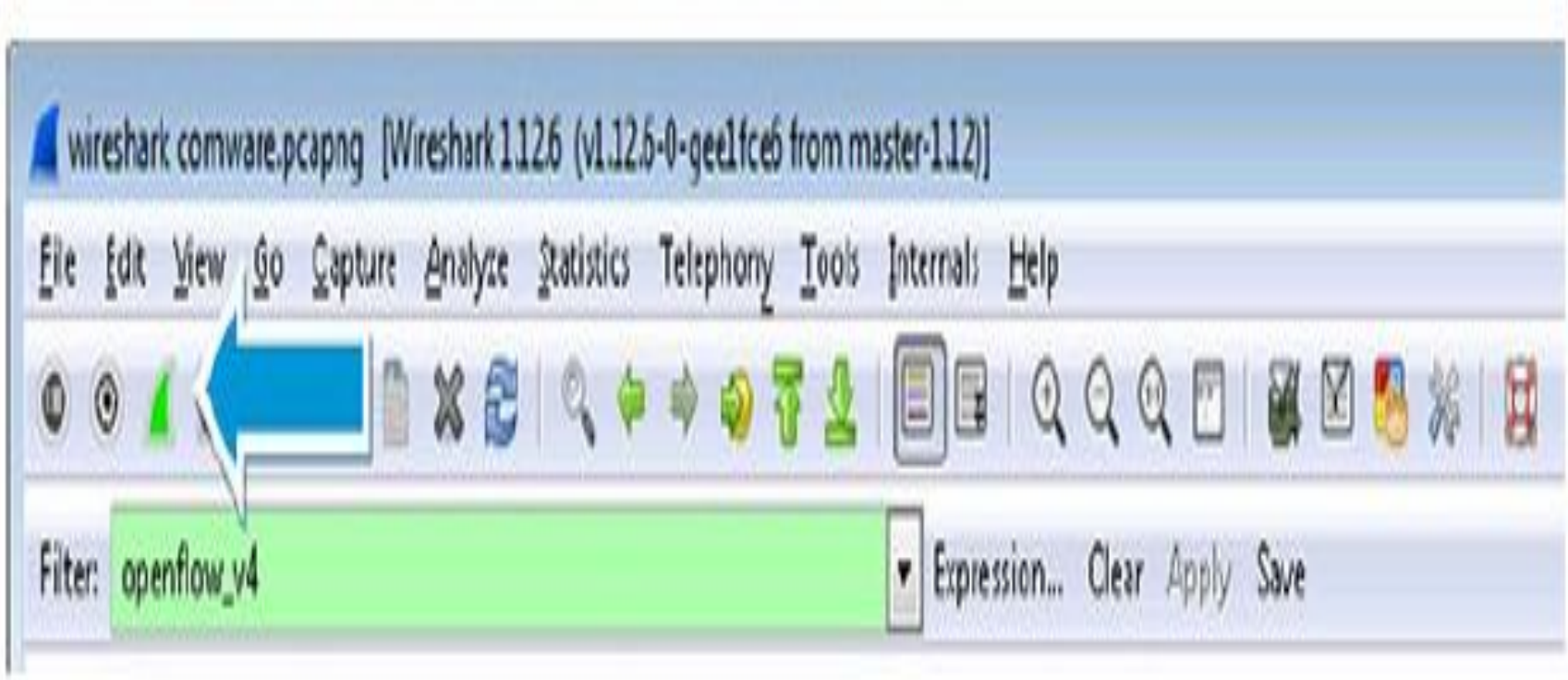
# Configure Comware switch

```
<C2> system-view
[C2] openflow instance 1
[C2-of-inst-1] undo classification
[C2-of-inst-1] active instance
[C2-of-inst-1] quit
```

# Configure Provision switch

```
P1# config

P1(config)# openflow

P1(openflow)# instance vlan30

P1(of-inst-vlan30)# disable

P1(of-inst-vlan30)# no controller-id 3

P1(of-inst-vlan30)# controller-id 1
```

# Start Wireshark



# Enable OpenFlow switchs

```
P1(of-inst-vlan30)# enable
P1(of-inst-vlan30)# exit
P1(openflow)# enable
```

```
[C2-of-inst-1] classification vlan 20
[C2-of-inst-1] active instance
```

# Clear the Address Resolution Protocol (ARP)


```
C:\Windows\system32> arp -a -d

C:\Windows\system32>
```

# 7. Ping UserVM3's default gateway and hp.com:

```
C:\Windows\system32> ping 10.30.30.251

Pinging 10.30.30.251 with 32 bytes of data:
Reply from 10.30.30.251: bytes=32 time=1ms TTL=255
Reply from 10.30.30.251: bytes=32 time<1ms TTL=255
Reply from 10.30.30.251: bytes=32 time<1ms TTL=255
Reply from 10.30.30.251: bytes=32 time<1ms TTL=255

Ping statistics for 10.30.30.251:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```

# 7. Ping UserVM3's default gateway and hp.com:

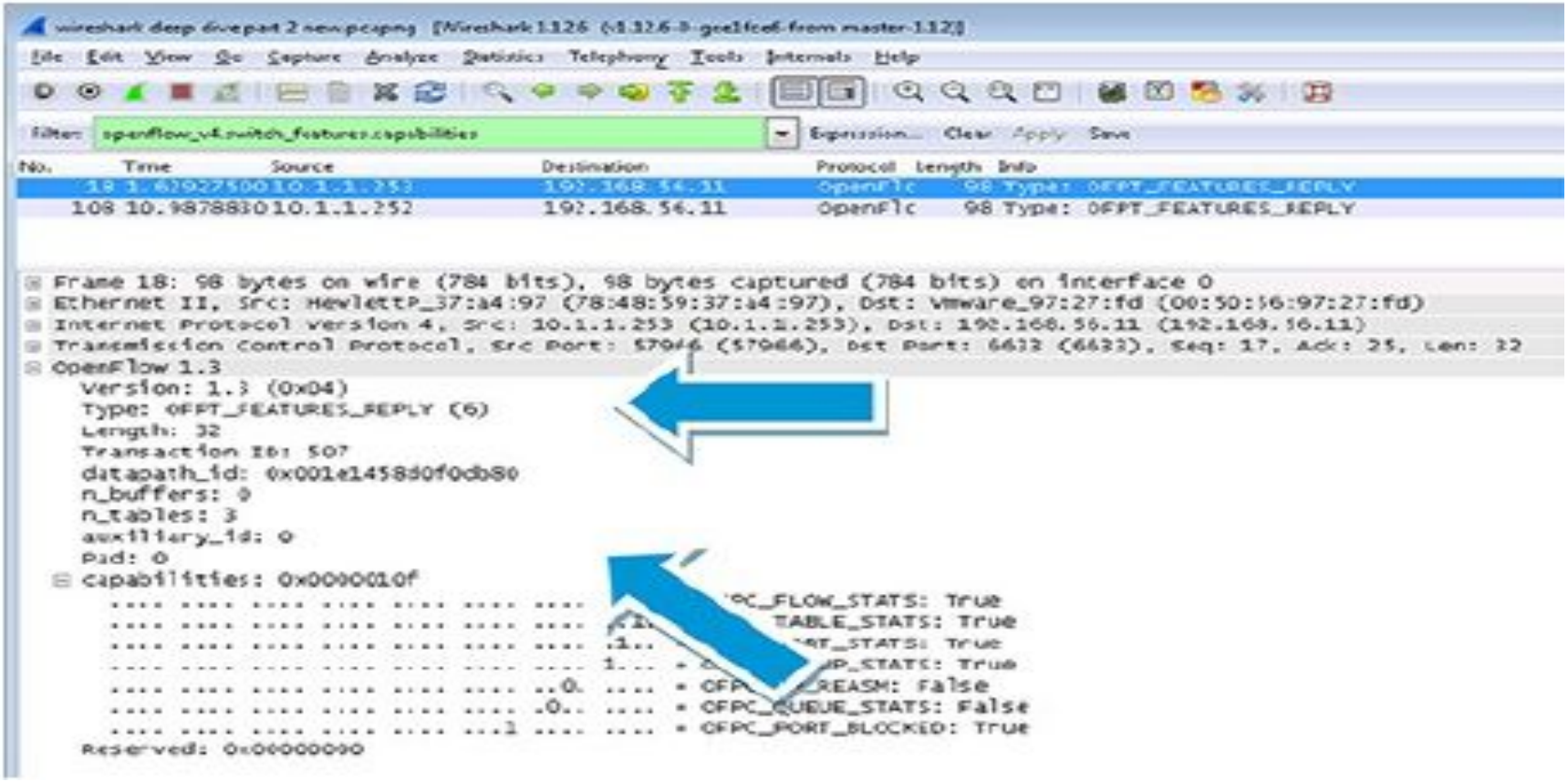
```
C:\Windows\system32> ping hp.com

Pinging hp.com [192.168.56.51] with 32 bytes of data:
Reply from 192.168.56.51: bytes=32 time<1ms TTL=63
Reply from 192.168.56.51: bytes=32 time<1ms TTL=63
Reply from 192.168.56.51: bytes=32 time<1ms TTL=63
Reply from 192.168.56.51: bytes=32 time<1ms TTL=63

Ping statistics for 192.168.56.51:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

# Stop Wireshark

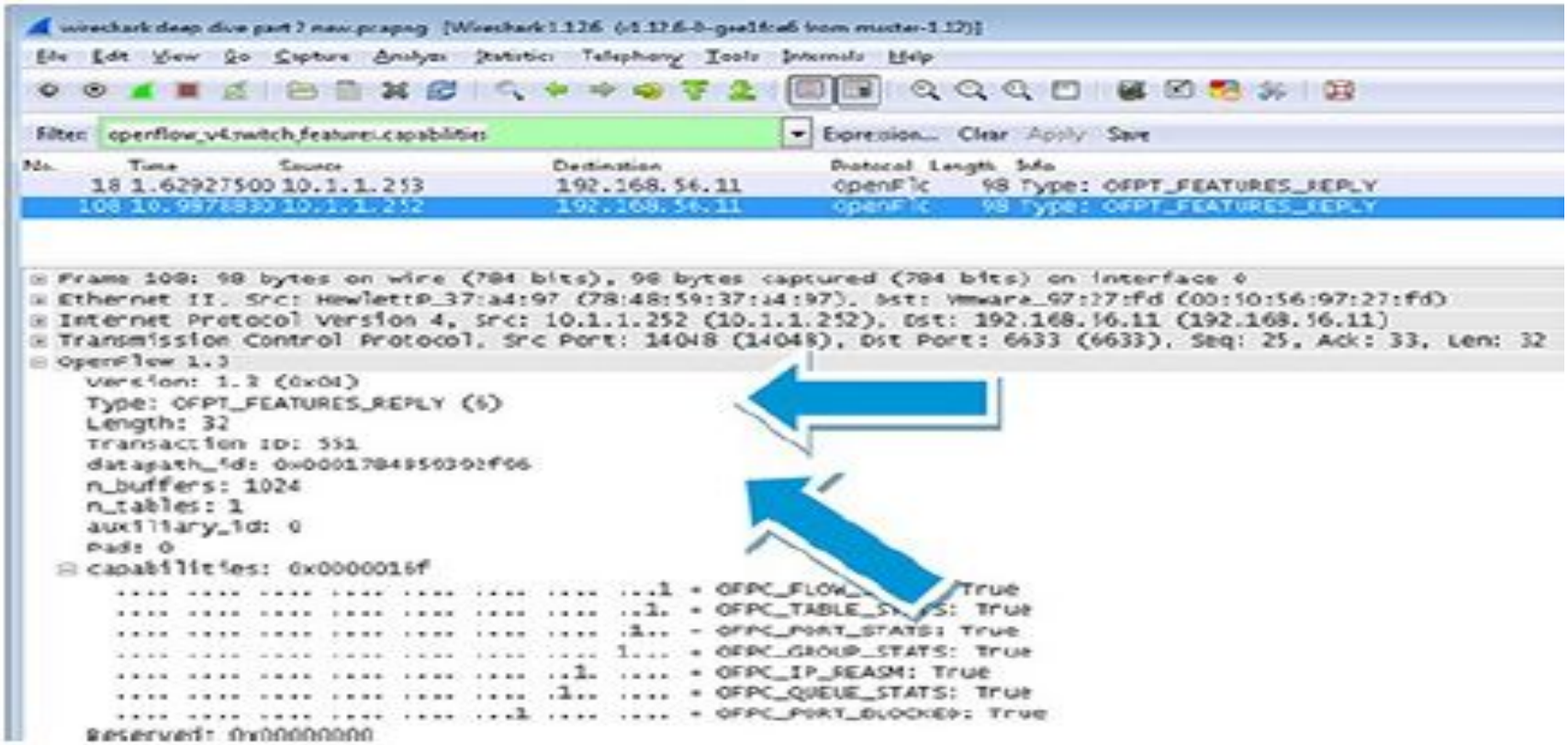

**Apply**

**Answer:**

# Features Reply message



# Features Reply message



# OpenFlow specification

**datapath_id**

# ProVision switch DPIDs calculation

**Answer:**

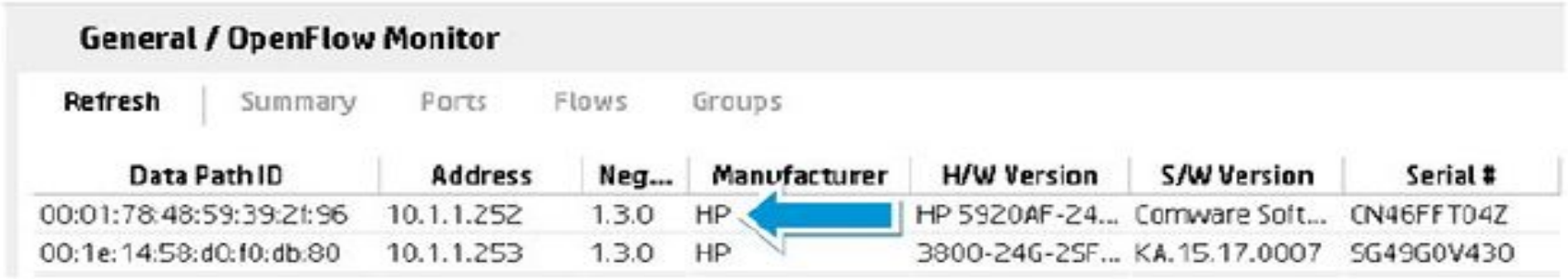
- **The most significant 16 bits are the VLAN number associated with the OpenFlow instance. So for the OpenFlow instance configured on VLAN 30, this number will be “1E”. (30 in decimal equates to 1E in hexadecimal). Hence the switches are identified by 00:1E.**
- **Least significant 48 bits are the switch MAC address.**

# OpenFlow Monitor


# Comware switch DPIDs calculation

**DPIDs on Comware switches:**

- **Comprised of the instance ID and the bridge MAC address. The most significant 16 bits are the instance ID.**
- **The least significant 48 bits are the bridge MAC address.**



# Open vSwitch DPIDs calculation

**Open vSwitch switch DPIDs:**

- **Comprised of 00:00 and the MAC address of the OVS bridge.**

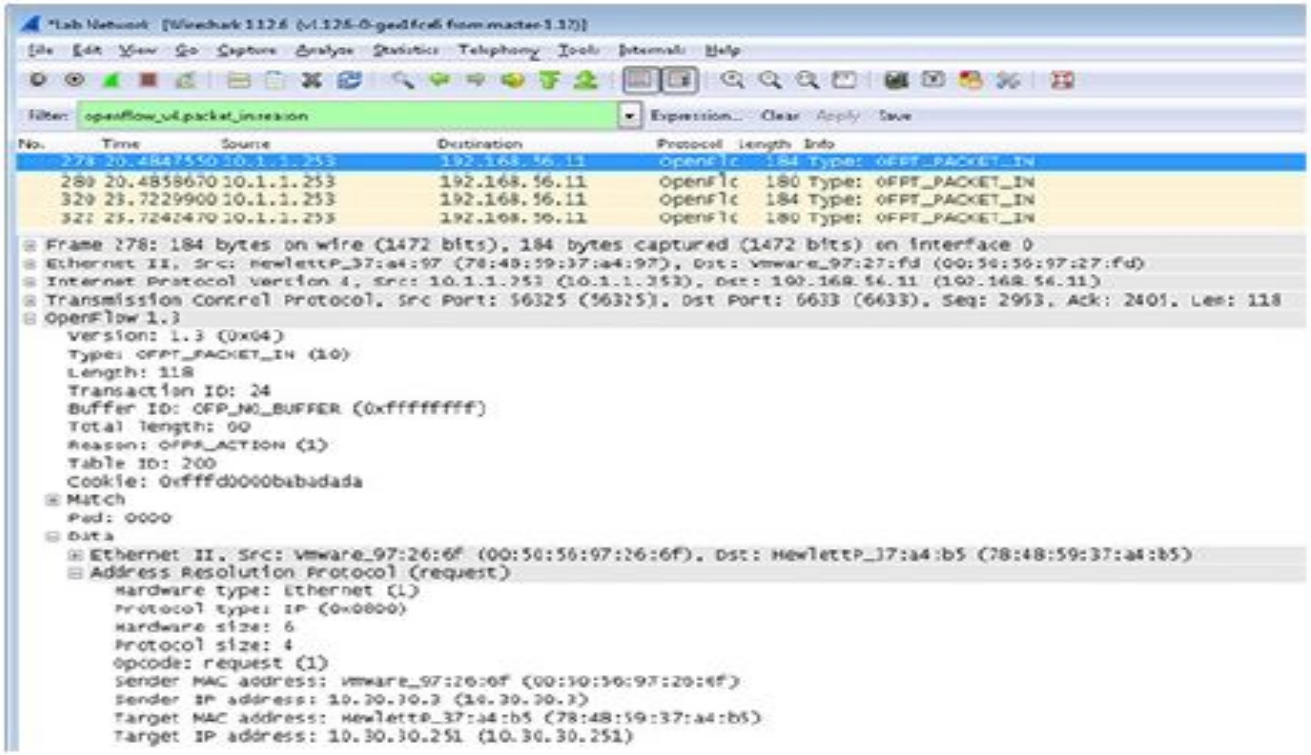

```
sdn@KVMServer1:~$ ifconfig
...
ovsbr0    Link encap:Ethernet HWaddr 00:0c:29:e8:22:de
  inet addr:172.16.1.13 Bcast:172.16.1.255 Mask:255.255.255.0
```


# Investigating OpenFlow negotiations

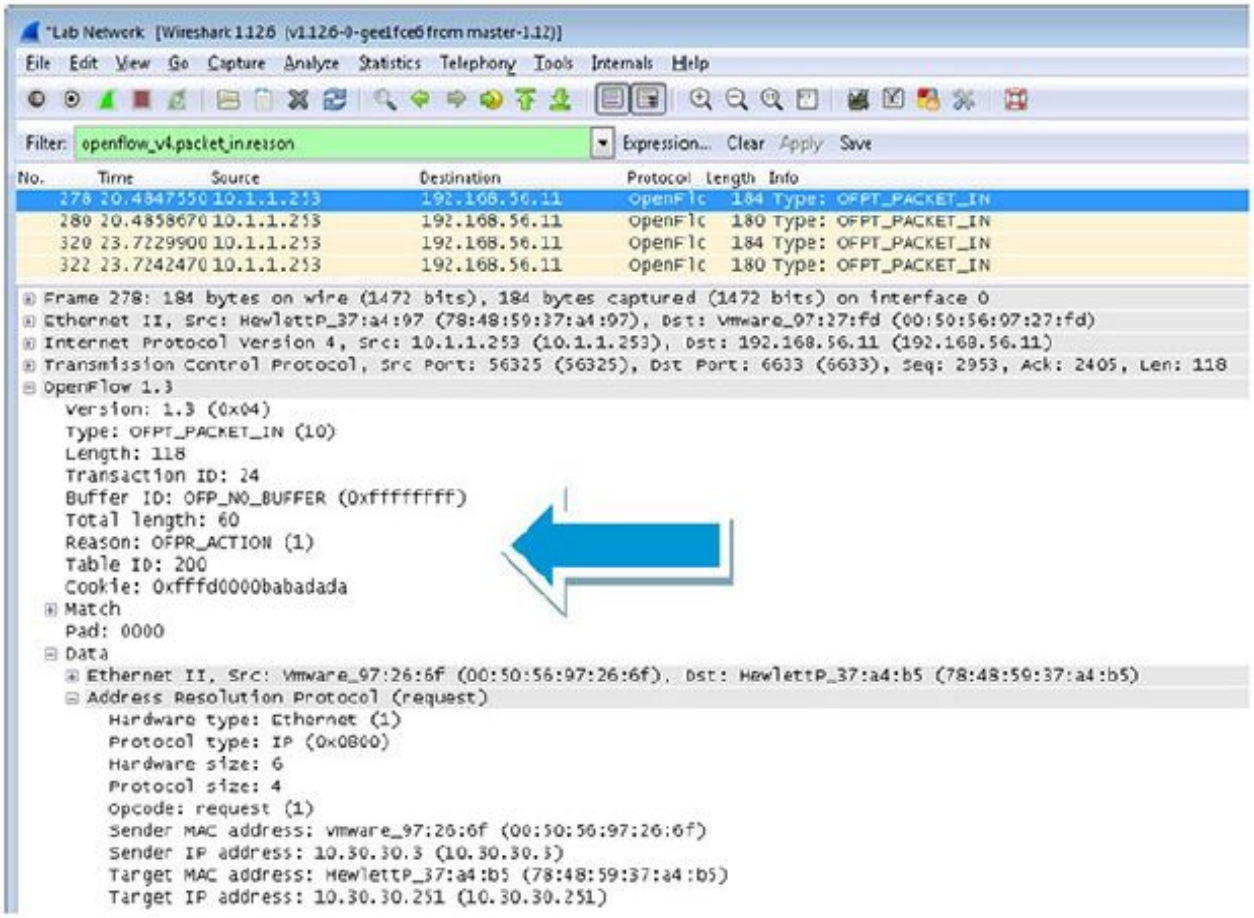
# *Packet_in* message

**Apply**

openflow_v4.packet_in.reason



# *Packet_in* message



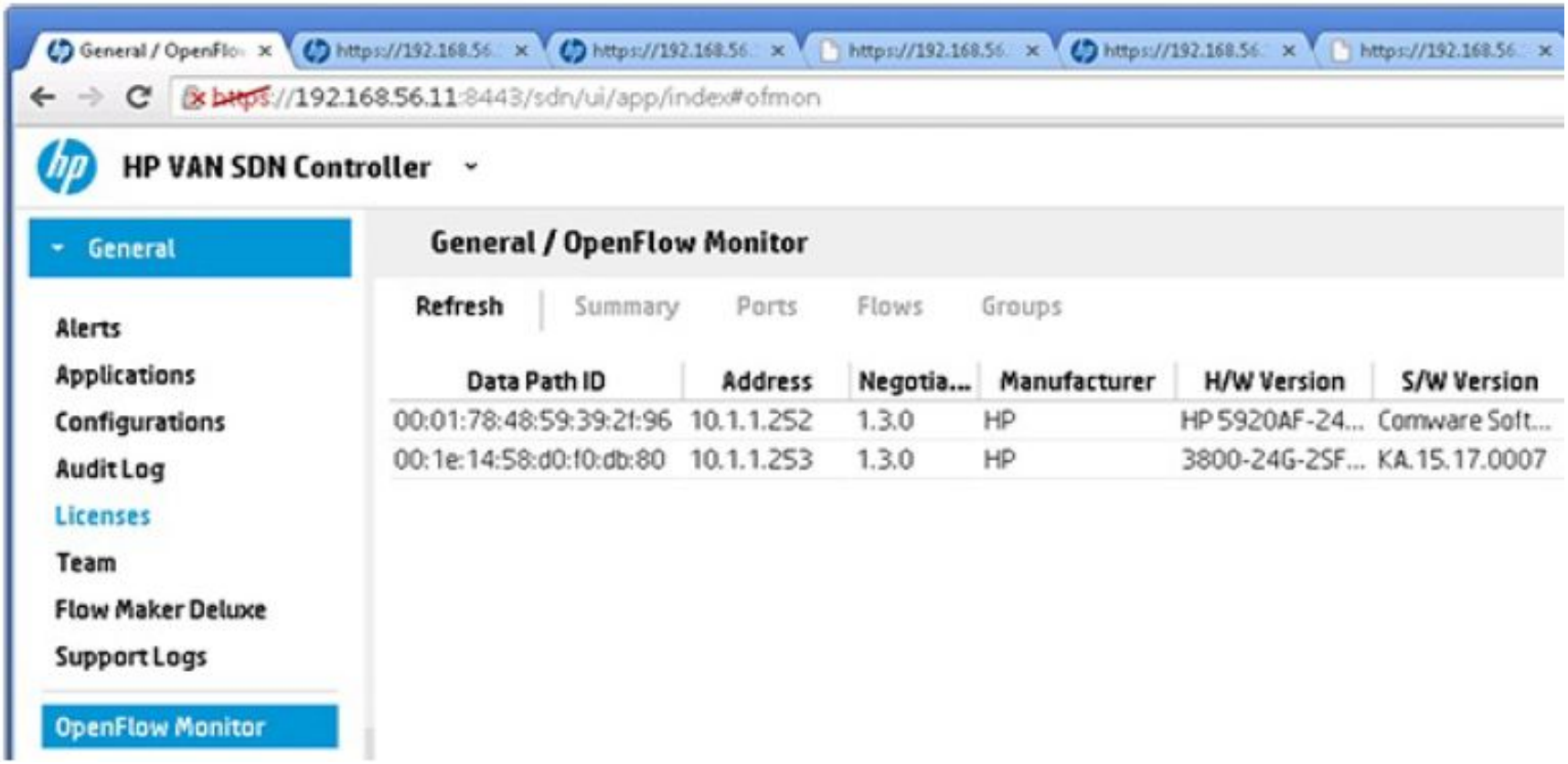

**Answer:**

**OFPR_ACTION (1)**

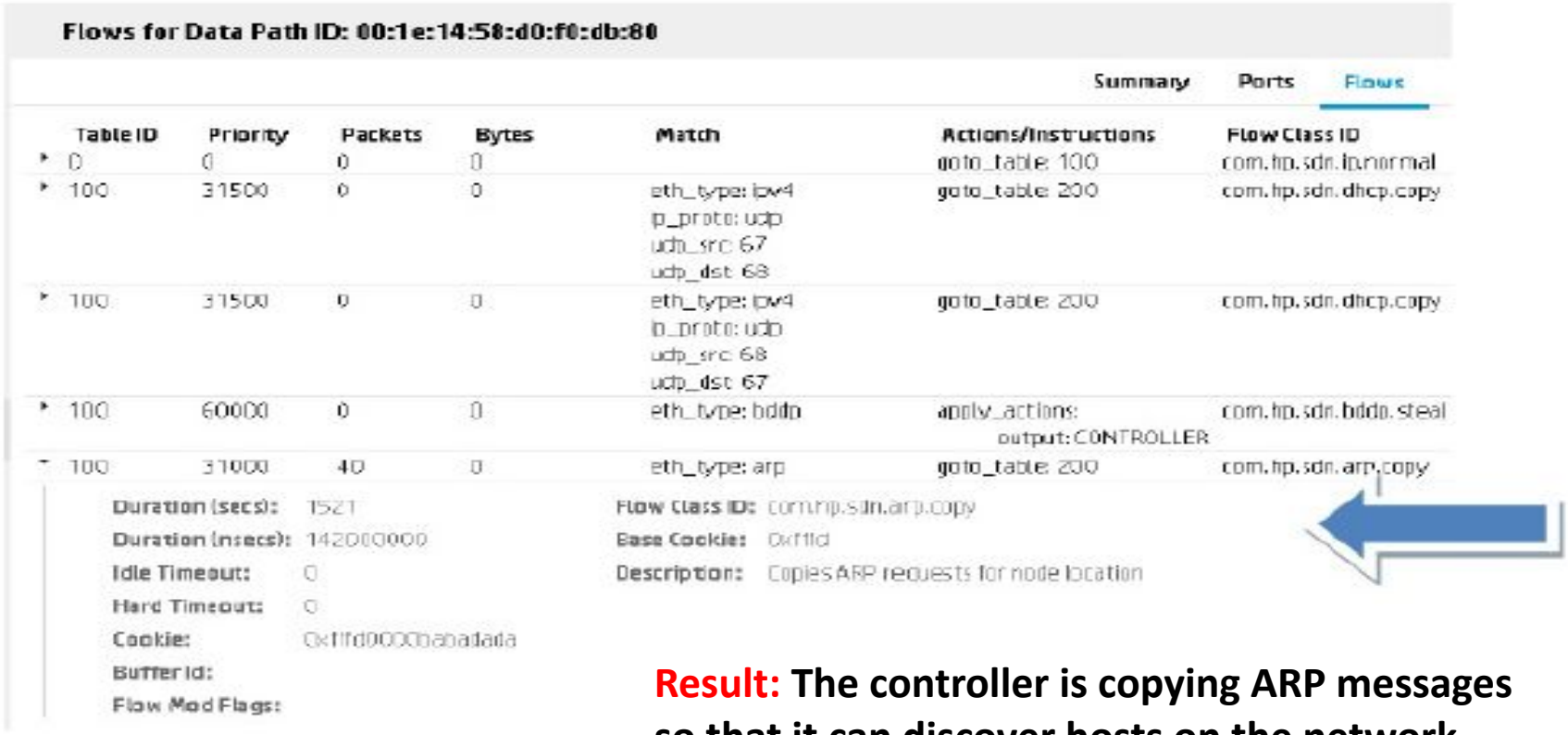
# *Packet_in* message

```
/* Why is this packet being sent to the controller? */

enum ofp_packet_in_reason {

  OFPR_NO_MATCH = 0, /* No matching flow (table-miss flow entry). */

  OFPR_ACTION = 1, /* Action explicitly output to controller. */

  OFPR_INVALID_TTL = 2, /* Packet has invalid TTL */
```

# *Packet_in* message



# *Packet_in* message

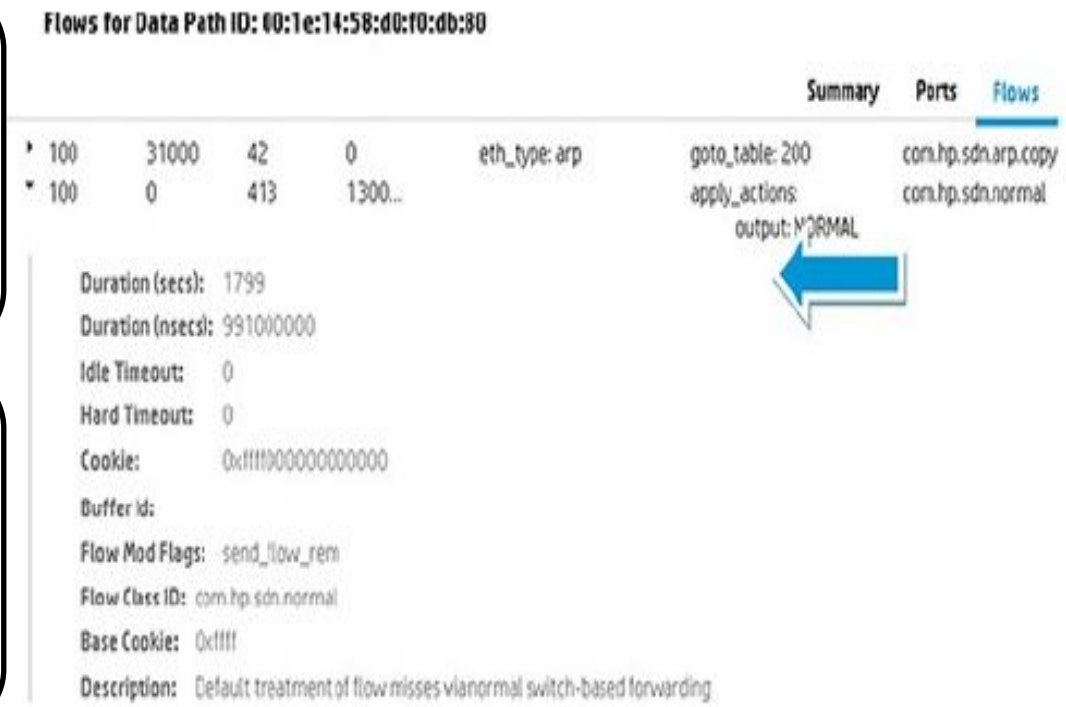



**Result:** **The controller is copying ARP messages so that it can discover hosts on the network.**

# *Table-miss flow entry*

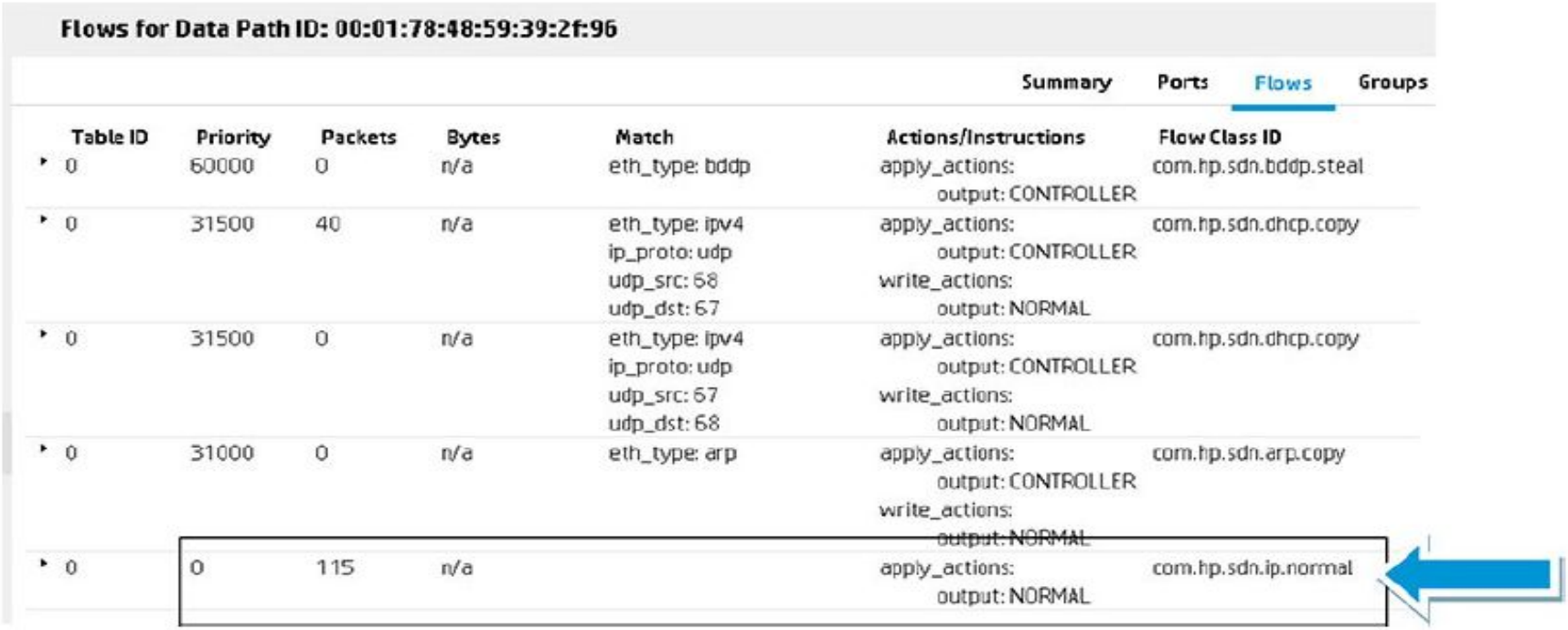
**Answer:**

**Answer:**




**Result:**

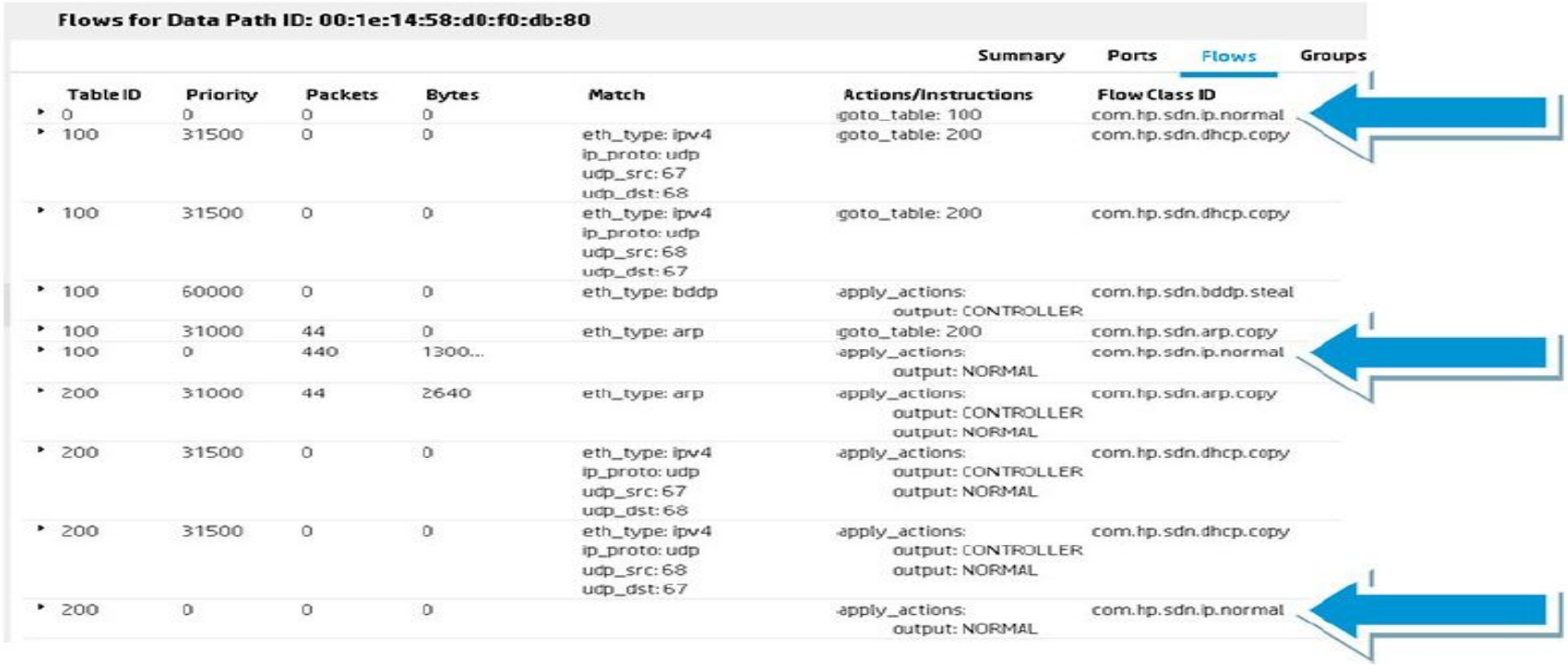
# *Table-miss flow entry*



# *Table-miss flow entry*

# *Table-miss flow entry*

**Answer:**



Flows for Data Path ID: 00:1e:14:58:d0:f0:db:80

Summary | Ports | Flows | Groups

| Table ID | Priority | Packets | Bytes | Match | Actions/Instructions | Flow Class ID |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | | goto_table: 100 | com.hp.sdn.ip.normal |
| 100 | 31500 | 0 | 0 | eth_type: ipv4<br>ip_proto: udp<br>udp_src: 67<br>udp_dst: 68 | goto_table: 200 | com.hp.sdn.dhcp.copy |
| 100 | 31500 | 0 | 0 | eth_type: ipv4<br>ip_proto: udp<br>udp_src: 68<br>udp_dst: 67 | goto_table: 200 | com.hp.sdn.dhcp.copy |
| 100 | 60000 | 0 | 0 | eth_type: bddp | apply_actions:<br>output: CONTROLLER | com.hp.sdn.bddp.steal |
| 100 | 31000 | 44 | 0 | eth_type: arp | goto_table: 200 | com.hp.sdn.arp.copy |
| 100 | 0 | 440 | 1300... | | apply_actions:<br>output: NORMAL | com.hp.sdn.ip.normal |
| 200 | 31000 | 44 | 2640 | eth_type: arp | apply_actions:<br>output: CONTROLLER<br>output: NORMAL | com.hp.sdn.arp.copy |
| 200 | 31500 | 0 | 0 | eth_type: ipv4<br>ip_proto: udp<br>udp_src: 67<br>udp_dst: 68 | apply_actions:<br>output: CONTROLLER<br>output: NORMAL | com.hp.sdn.dhcp.copy |
| 200 | 31500 | 0 | 0 | eth_type: ipv4<br>ip_proto: udp<br>udp_src: 68<br>udp_dst: 67 | apply_actions:<br>output: CONTROLLER<br>output: NORMAL | com.hp.sdn.dhcp.copy |
| 200 | 0 | 0 | 0 | | apply_actions:<br>output: NORMAL | com.hp.sdn.ip.normal |

# *Table-miss flow entry*

**Answer:**

**Answer:**


# Investigating OpenFlow negotiations

# OpenFlow tables

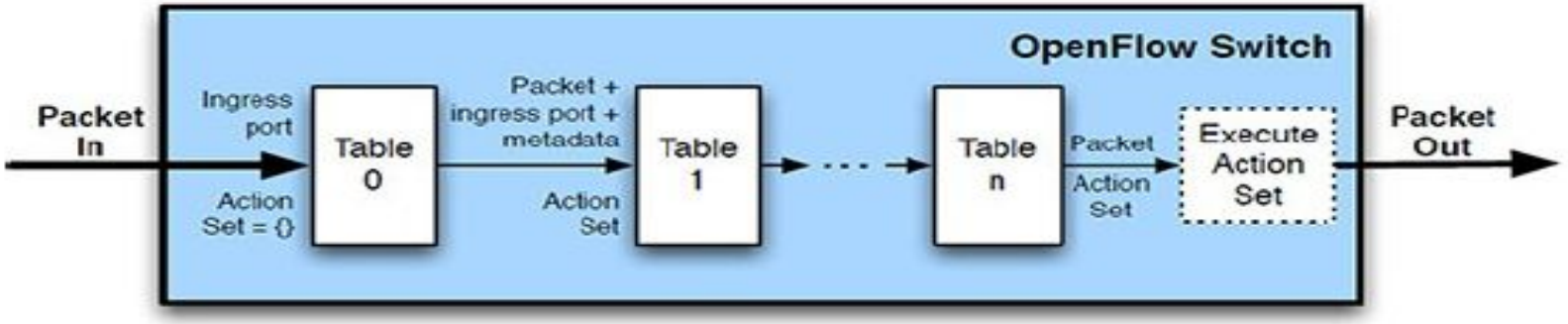
**Answer:**

**Answer:**

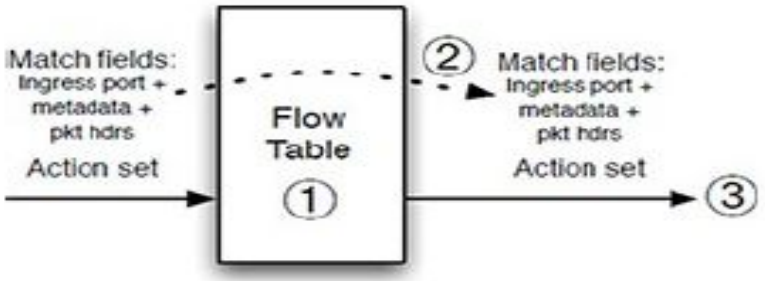
**Answer:**

**Answer:**

# OpenFlow tables




(a) Packets are matched against multiple tables in the pipeline




① Find highest-priority matching flow entry

② Apply instructions:
- i. Modify packet & update match fields (apply actions instruction)
- ii. Update action set (clear actions and/or write actions instructions)
- iii. Update metadata

③ Send match data and action set to next table

(b) Per-table packet processing

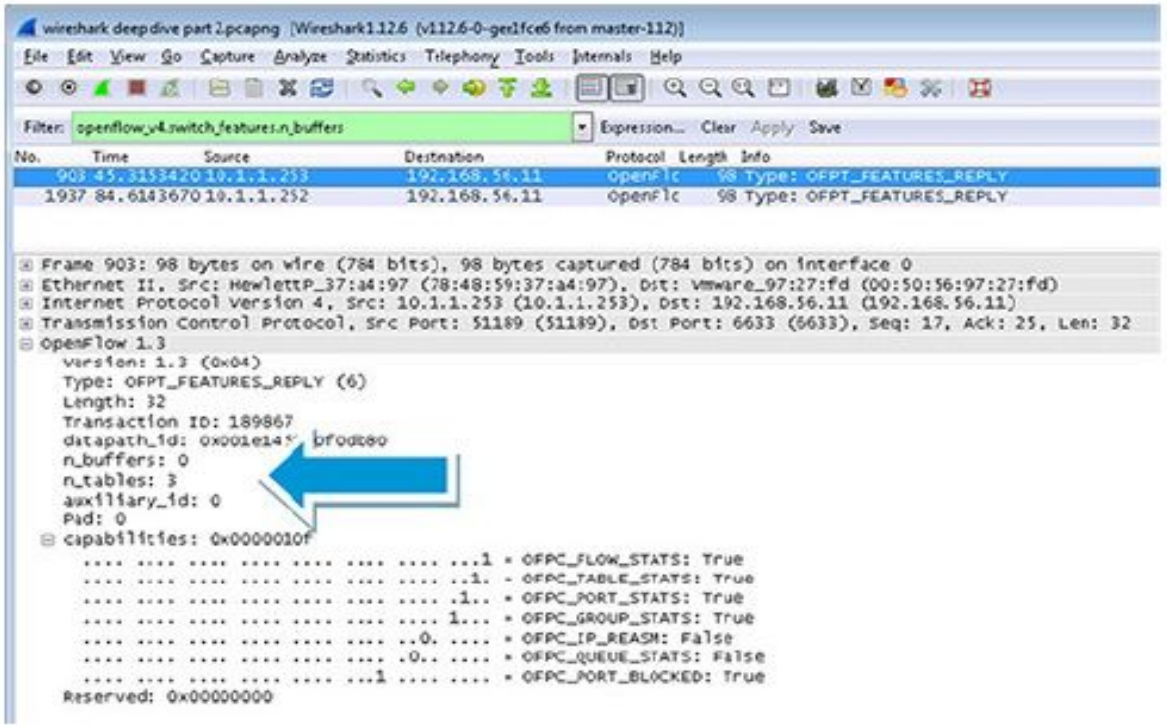
# OpenFlow tables

# OpenFlow tables

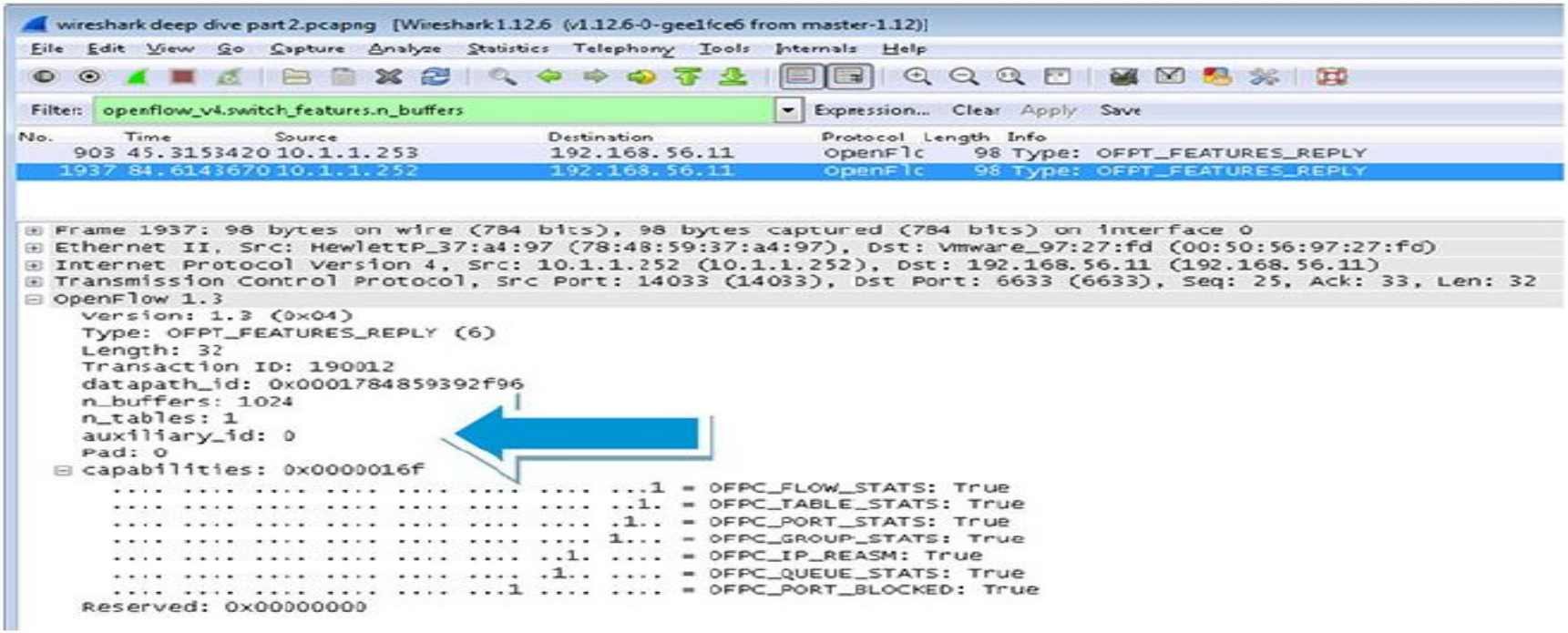
# OpenFlow tables

**Apply**
**openflow_v4.switch_features.n_buffers**

**Result:**



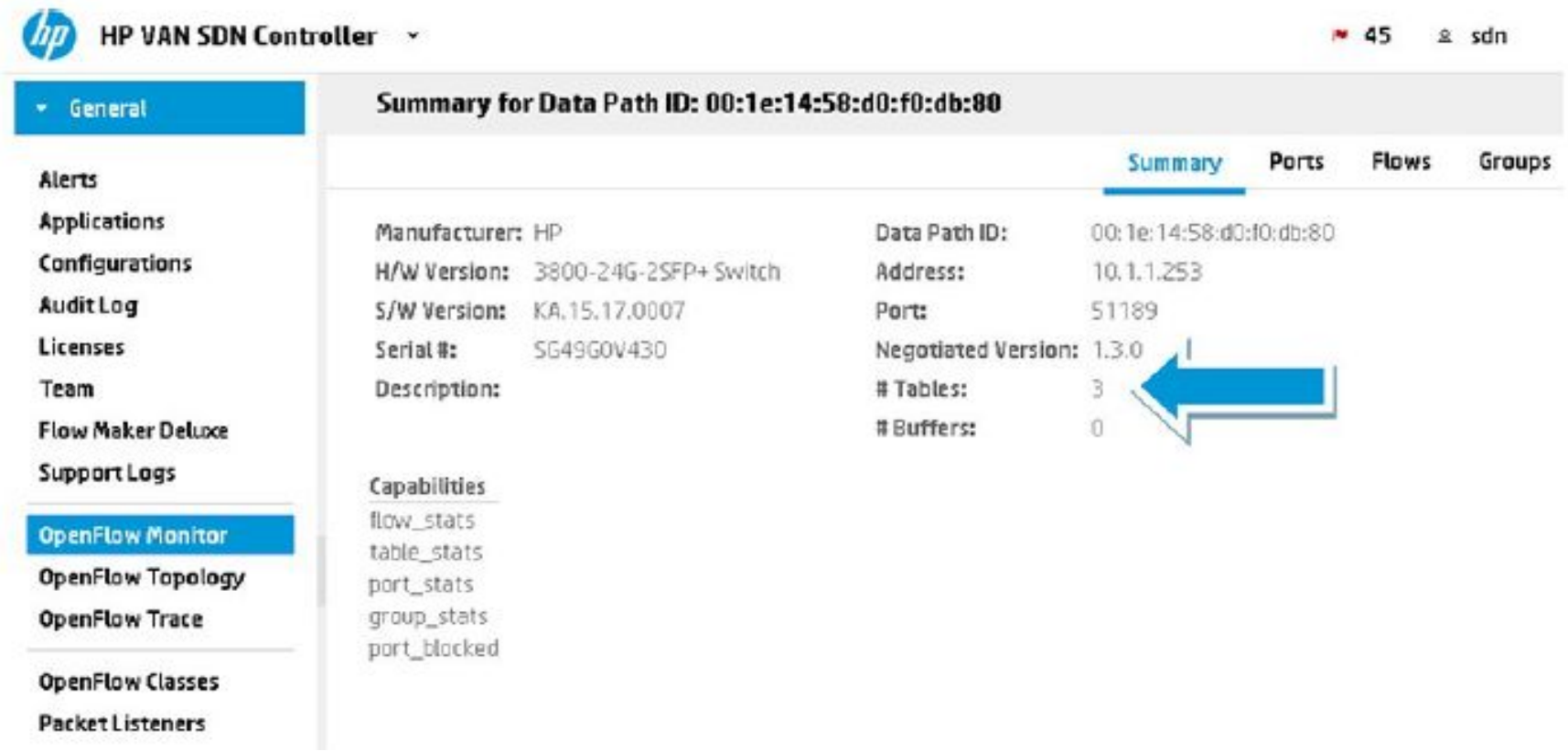
# OpenFlow tables



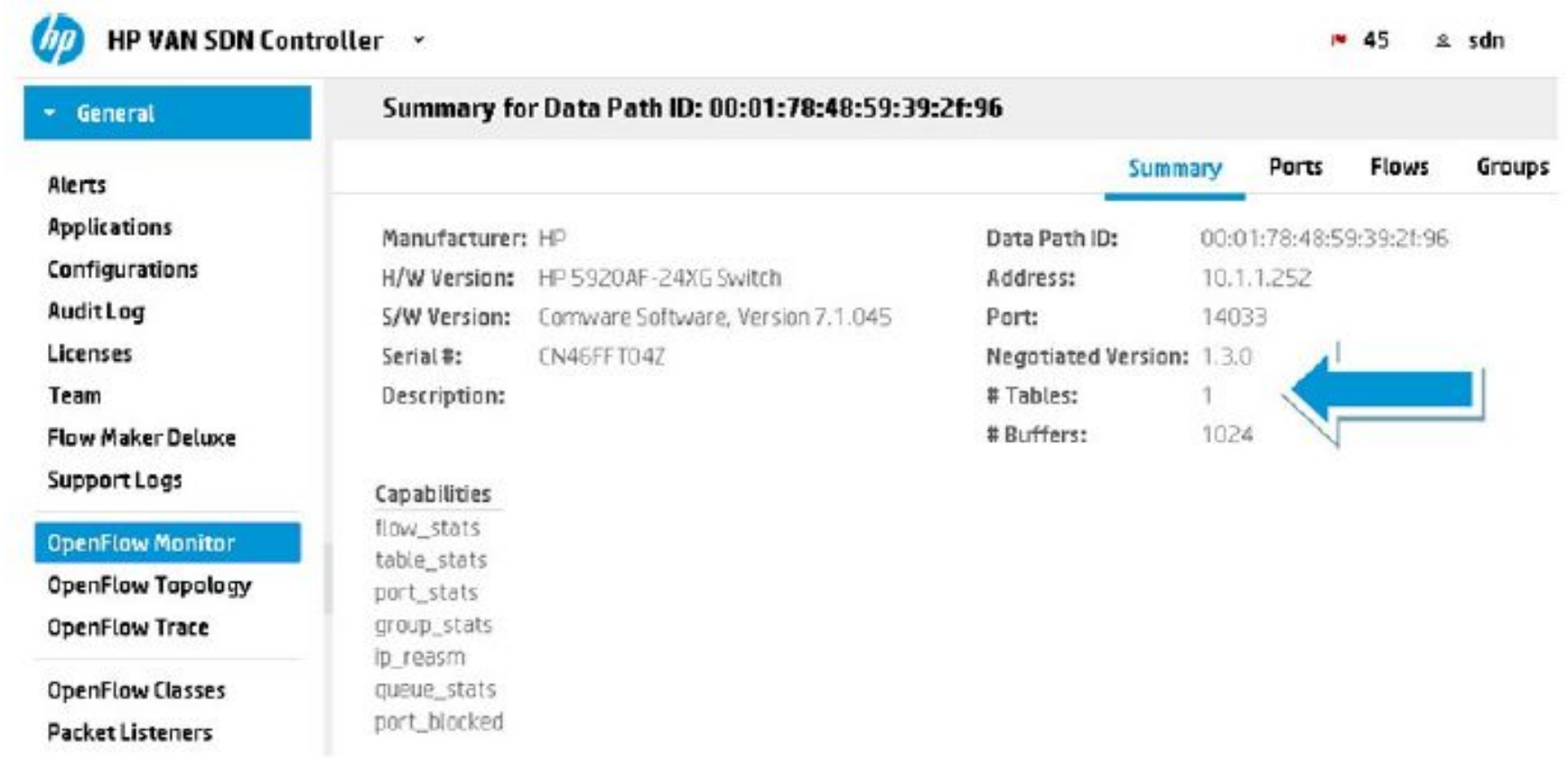

**Result:**

# OpenFlow tables

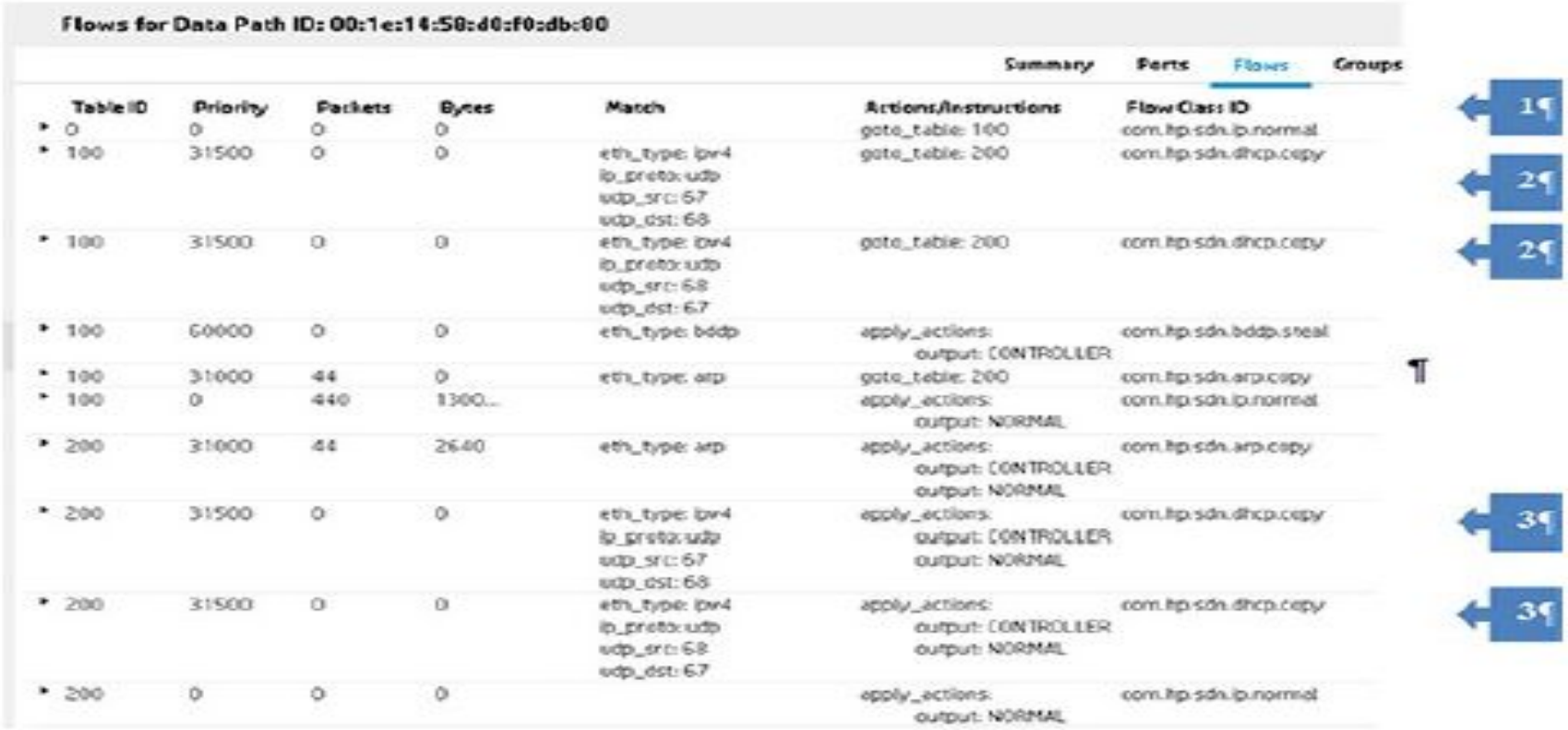
# OpenFlow Monitor



# OpenFlow Monitor



# Flow table of the ProVision switch



Flows for Data Path ID: 00:1e:14:50:d0:f0:db:00

Summary | Ports | Flows | Groups

| Table ID | Priority | Packets | Bytes | Match | Actions/Instructions | Flow Class ID |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | | goto_table: 100 | com.hp.sdn.ip.normal |
| 100 | 31500 | 0 | 0 | eth_type: ipv4<br>ip_proto: udp<br>udp_src: 67<br>udp_dst: 68 | goto_table: 200 | com.hp.sdn.dhcp.copy |
| 100 | 31500 | 0 | 0 | eth_type: ipv4<br>ip_proto: udp<br>udp_src: 68<br>udp_dst: 67 | goto_table: 200 | com.hp.sdn.dhcp.copy |
| 100 | 60000 | 0 | 0 | eth_type: bddp | apply_actions:<br>output: CONTROLLER | com.hp.sdn.bddp.steal |
| 100 | 31000 | 44 | 0 | eth_type: arp | goto_table: 200 | com.hp.sdn.arp.copy |
| 100 | 0 | 440 | 1300... | | apply_actions:<br>output: NORMAL | com.hp.sdn.ip.normal |
| 200 | 31000 | 44 | 2640 | eth_type: arp | apply_actions:<br>output: CONTROLLER<br>output: NORMAL | com.hp.sdn.arp.copy |
| 200 | 31500 | 0 | 0 | eth_type: ipv4<br>ip_proto: udp<br>udp_src: 67<br>udp_dst: 68 | apply_actions:<br>output: CONTROLLER<br>output: NORMAL | com.hp.sdn.dhcp.copy |
| 200 | 31500 | 0 | 0 | eth_type: ipv4<br>ip_proto: udp<br>udp_src: 68<br>udp_dst: 67 | apply_actions:<br>output: CONTROLLER<br>output: NORMAL | com.hp.sdn.dhcp.copy |
| 200 | 0 | 0 | 0 | | apply_actions:<br>output: NORMAL | com.hp.sdn.ip.normal |

# Flow table of the ProVision switch

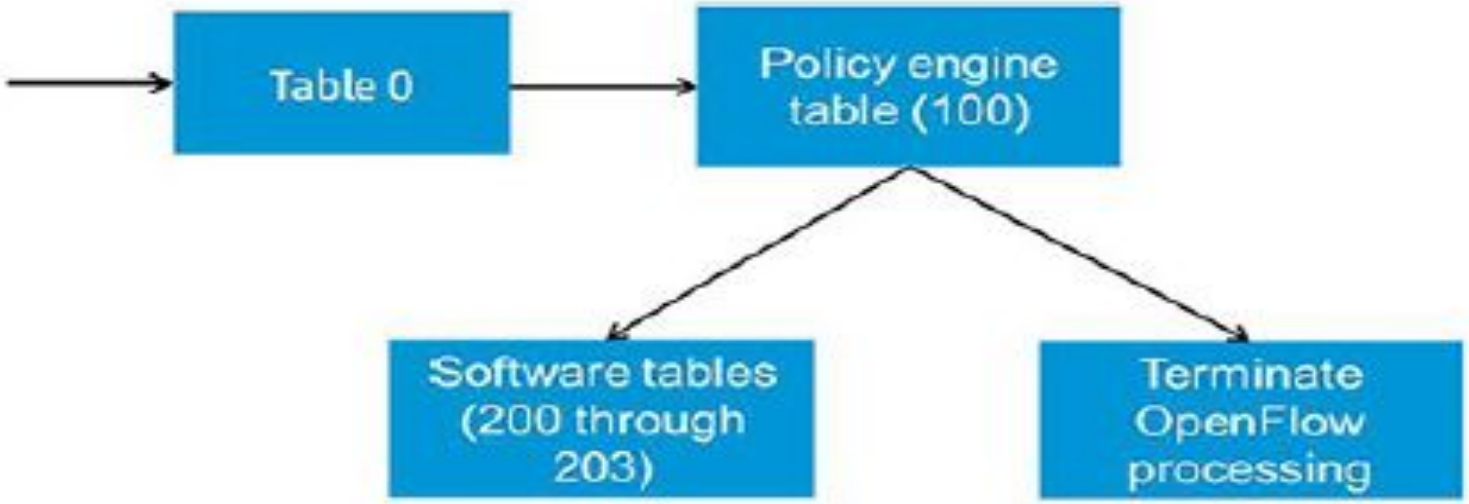
**Answer**

- **(1) Arriving traffic is first matched on the entries in the first OpenFlow table in the pipeline (Table 0). This table has a single entry, which is a table-miss entry (priority zero, match anything). The entry has a goto statement to send traffic to Table 100.**
- **(2) DHCP traffic will match either of the entries in Table 100 with priority of 31500. The entries have a goto statement to send traffic to Table 200. There are two DHCP entries — UDP port 67 is the client DHCP listen port and UDP port 68 is the server DHCP listen port. Two entries have been added to match flows in both directions.**
- **(3) DHCP traffic will match either of the entries in Table 200 with priority of 31500. The entries have an apply action of both output to CONTROLLER and output NORMAL. Thus, DHCP traffic is sent to the normal pipeline (traditional routing and switching) and is also copied to the controller.**

# Flow table of the ProVision switch

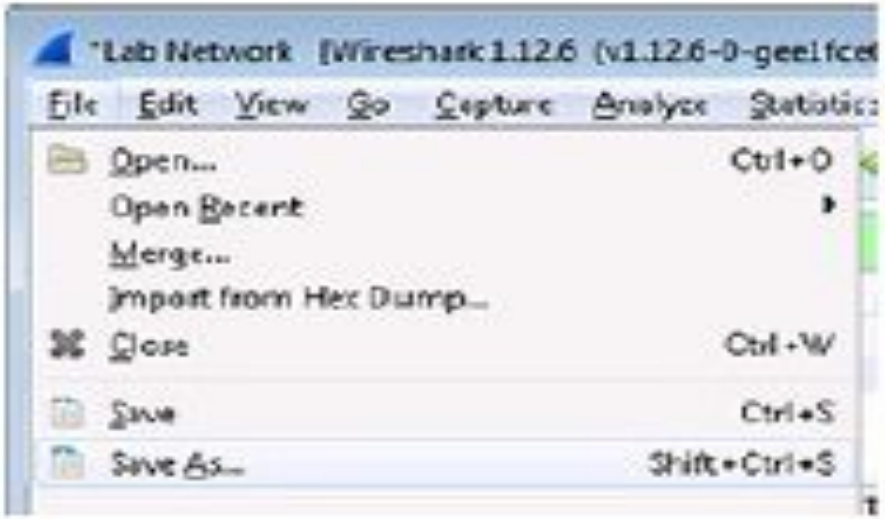
```
P1# show openflow instance vlan30 flow-table

OpenFlow Instance Flow Table Information

Table                    Flow      Available Free Miss
ID    Table Name         Count     Flow Count     Count     Goto Table
----- ------------------ --------- -------------- --------- -------------
0     Start              1         NA             0         100
100   Policy Table       5         NA             469       200
200   SW Table 1         4         NA             0         *

* Denotes that the pipeline could end here.
P1#
```

# Flow table of the ProVision switch

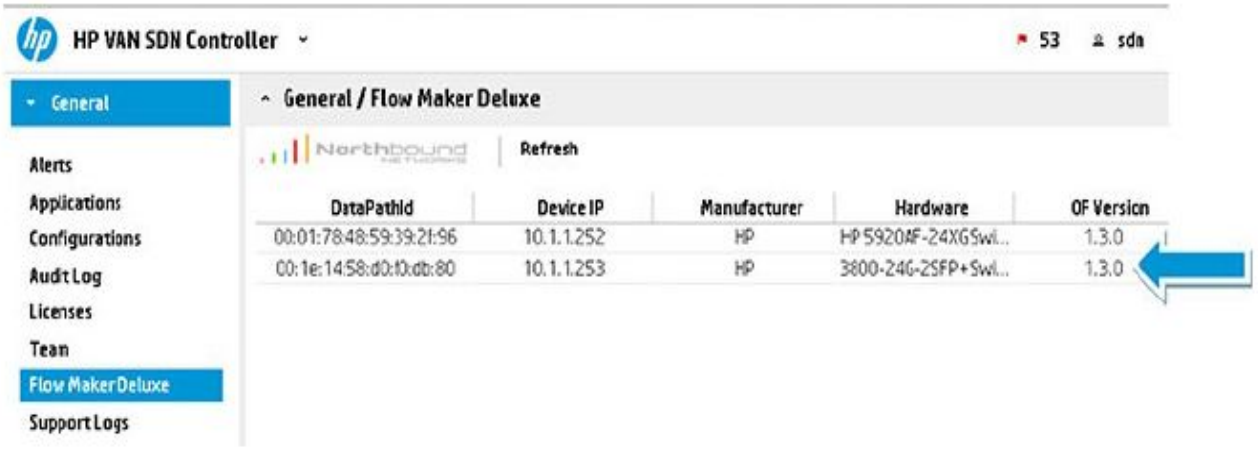



**Result:**

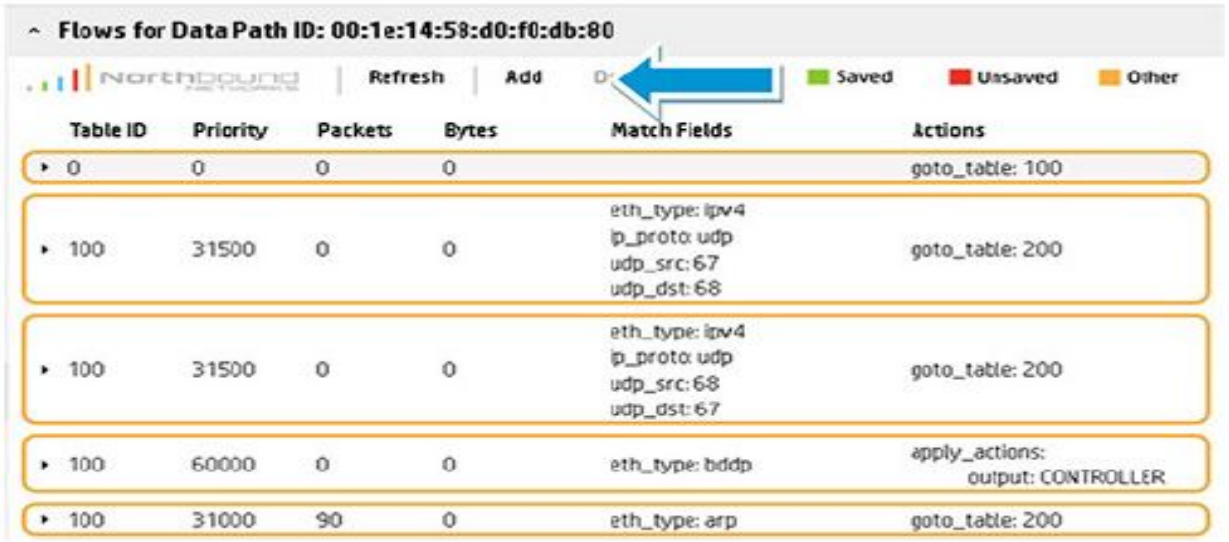
# Flow table of the ProVision switch




# Flow table of the ProVision switch




Add



# Flow table of the ProVision switch

**31. Program a new flow entry in Table 0 of the ProVision switch 1 (P1) that drops all ingress traffic on port 2.**
**Will this rule work?**
**Add a flow entry with the following attributes and then click Add (see Figure).**

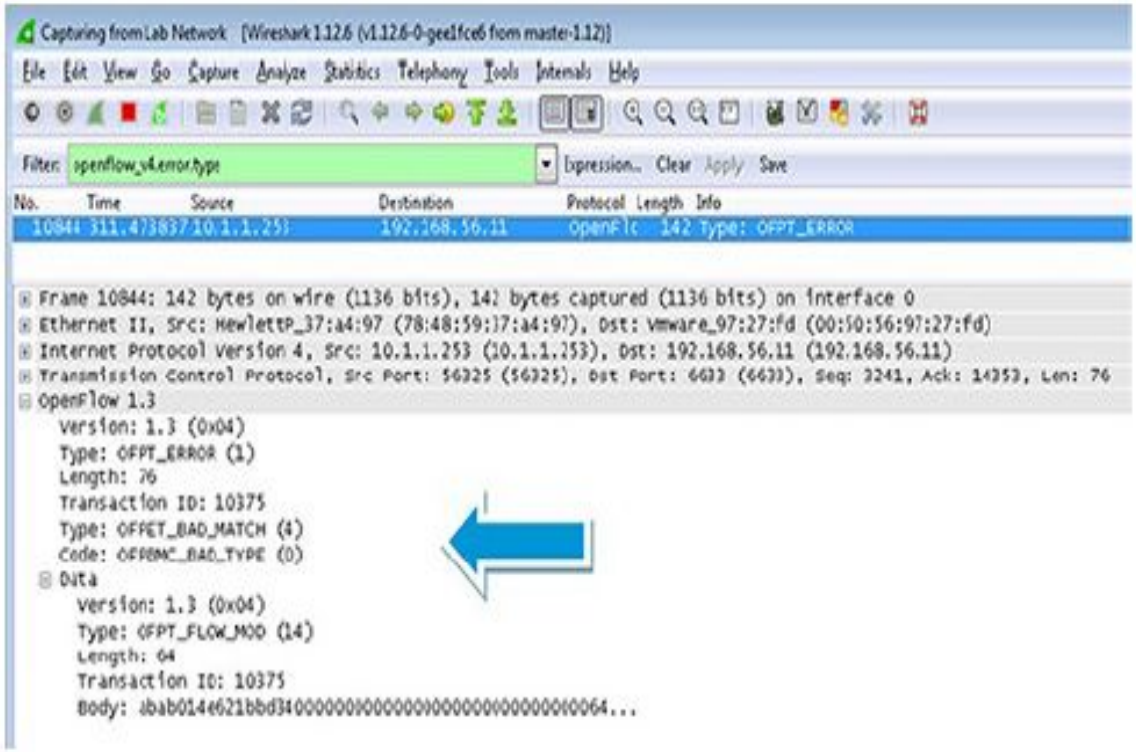
- **Table ID:0**
- **Priority: 100**
- **In Port: 2**
- **Instructions: Apply Action**
- **Action 1: No Action**
- **Save Flow = True**


**Result:**

# Flow table of the ProVision switch

Apply

openflow_v4.error.type




- Type: OFPET_BAD_MATCH = 4
- Code: OFPBMC_BAD_TYPE = 0

# Flow table of the ProVision switch

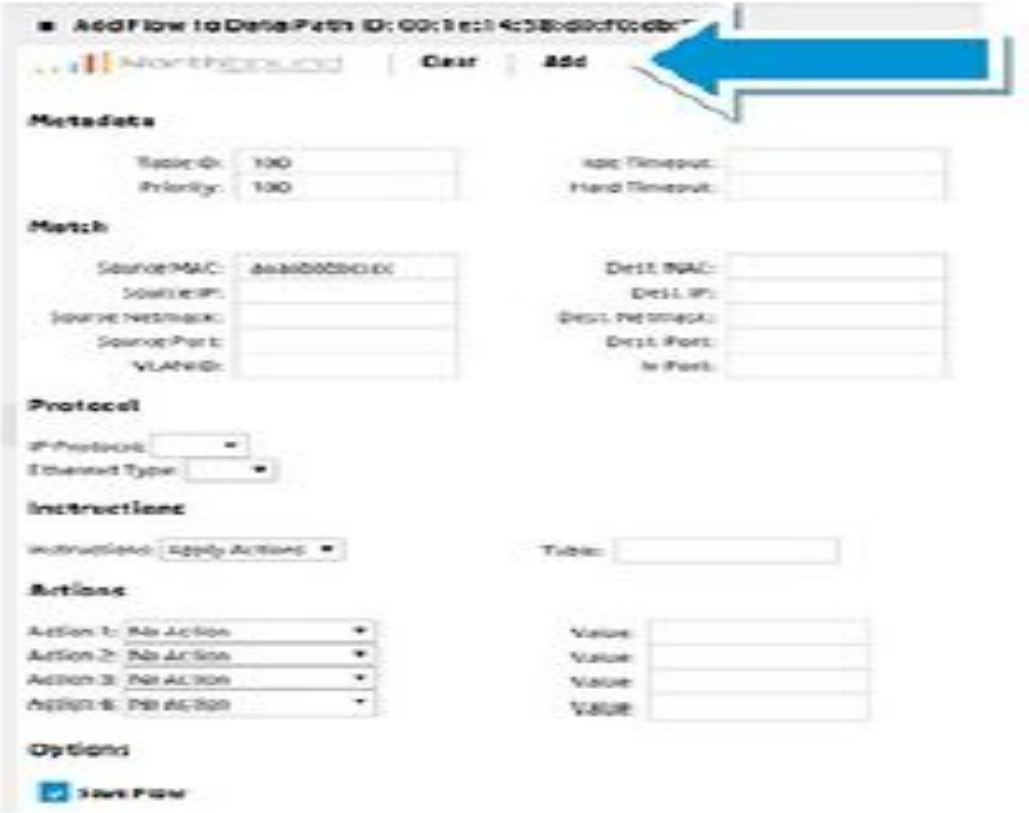
```
P1# show openflow instance vlan30 flow-table 0 table-capability

OpenFlow Flow Table Properties


Table Match Capabilities:

Table Instructions:

Table-Miss Instructions:

  GoTo 100

P1#
```

**Result:**

# Flow table of the ProVision switch

**Will this rule work?**
**Add a flow entry with the following attributes and then click Add (see Figure).**

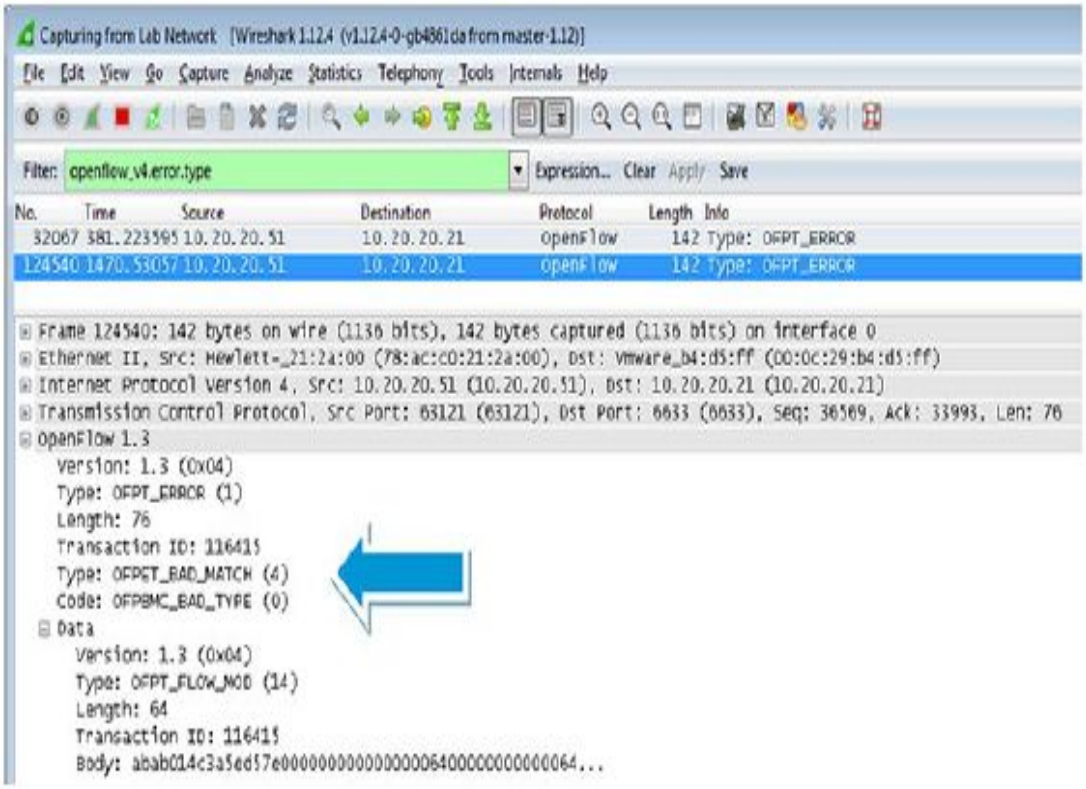
- **Table ID: 100**
- **Priority: 100**
- **Src Mac: aaaabbbbcccc**
- **Instructions: Apply Action**
- **Action 1: No Action**
- **Save Flow = True**



# Flow table of the ProVision switch

**The same error code displays:**

- **Type: OFPET_BAD_MATCH = 4**
- **Code: OFPBMC_BAD_TYPE = 0**

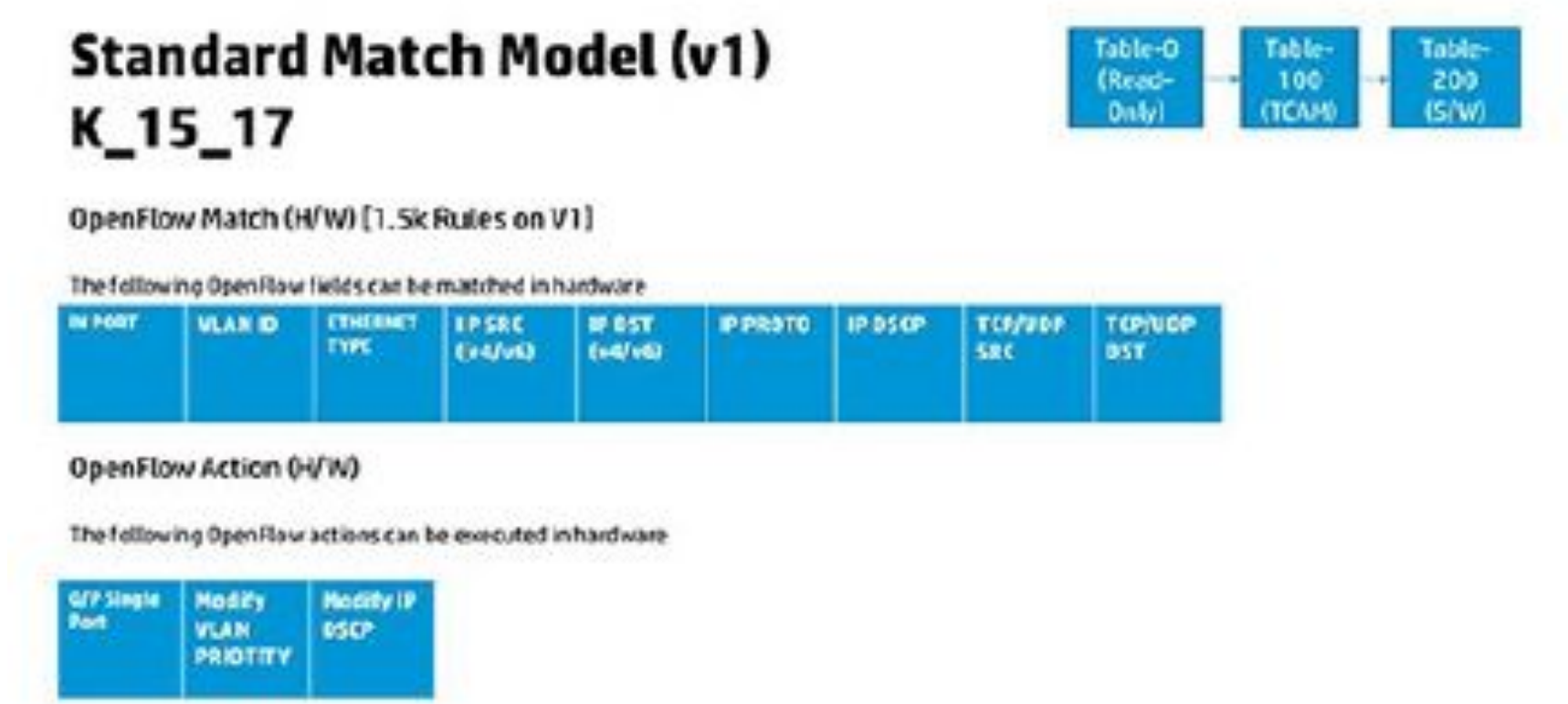


# Flow table of the ProVision switch

3800

V2 ASICs 3500 V1 ASICs

# Flow table of the ProVision switch



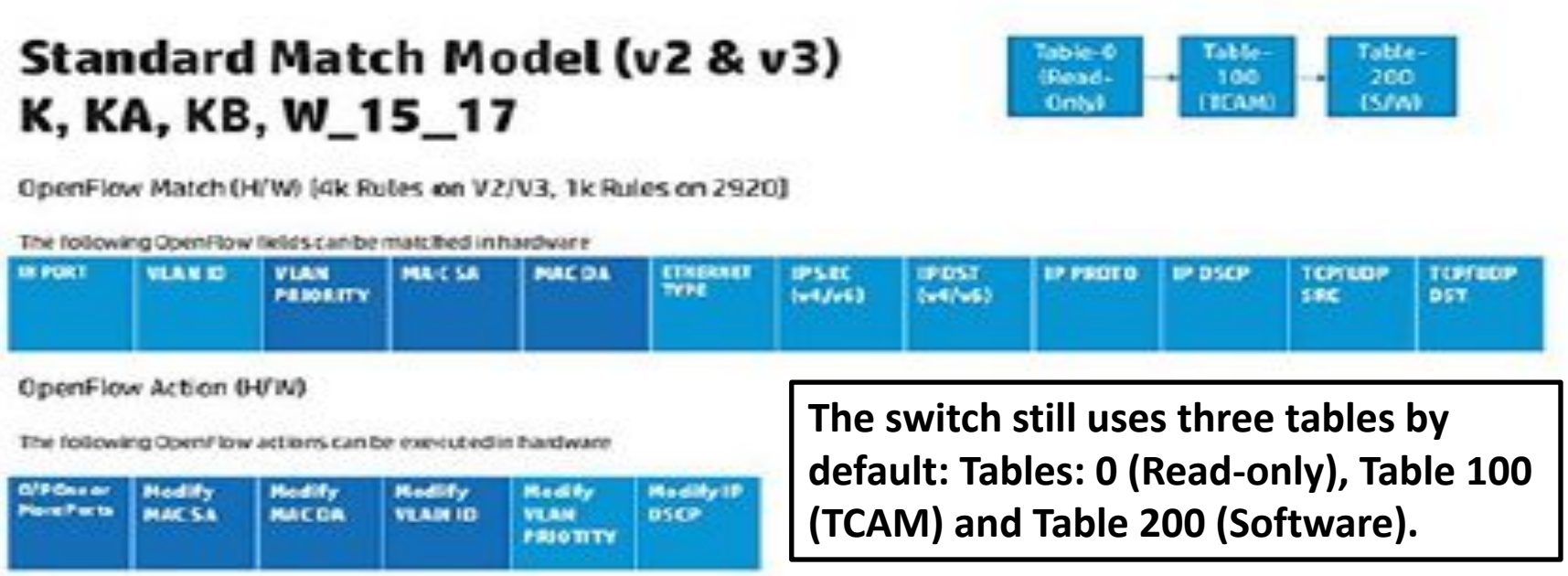
# Flow table of the ProVision switch

**Details V1 ASICs:**

- **Tables: 0 (read-only), Table 100 (TCAM), Table 200 (Software).**
- **Values that can be matched in hardware: in port, VLAN ID , Ethernet type, IP source address (IPv4 and IPv6), IP destination address (IPv4 and IPv6), IP protocol, TCP/UDP source port, and TCP/UDP destination port.**
- **Actions that can be executed in hardware: destination port (single port), modify VLAN priority, and modify IP differentiated services code point (DSCP).**

**Result:**

# Flow table of the ProVision switch




**The switch still uses three tables by default: Tables: 0 (Read-only), Table 100 (TCAM) and Table 200 (Software).**

**Result:**

# Flow table of the ProVision switch

**Details V2 ASICs:**

- **Ingress Port**
- **VLAN ID**
- **VLAN Priority**
- **Source MAC**
- **Destination MAC**
- **Ethertype**
- **IP source address (IPv4/IPv6)**
- **IP destination address (IPv4/IPv6)**
- **IP protocol**
- **IP DSCP**
- **TCP/UDP source port**
- **TCP/UDP destination port**

# Flow table of the ProVision switch

## Details V2 ASICs:

- Destination port (one or more ports)
- Modify source MAC address
- Modify destination MAC address
- Modify VLAN ID
- Modify VLAN priority
- Modify IP DSCP

# Flow table of the ProVision switch

## Custom Pipeline Model (v3)
## KB_15_17

OpenFlow Match (H/W) [Scale discussed in subsequent slides]

The following OpenFlow fields can be matched in hardware

| IN PORT | VLAN ID | VLAN PRIORITY | MAC SA | MAC DA | ETHERNET TYPE | IP SRC (v4/v6) | IP DST (v4/v6) | IP PROTO | IP DSCP | TCP/UDP SRC | TCP/UDP DST |
|---|---|---|---|---|---|---|---|---|---|---|---|

OpenFlow Action (H/W)

The following OpenFlow actions can be executed in hardware

| O/P One or More Ports | Modify MAC SA | Modify MAC DA | Modify VLAN ID | Modify VLAN PRIOTITY | Modify IP DSCP | Modify IPv4 SRC | Modify IPv4 DST | Modify TCP/UDP SRC (IPv4) | Modify TCP/UDP DST (IPv4) |
|---|---|---|---|---|---|---|---|---|---|

**The OpenFlow tables can be configured dynamically by SDN Applications.**

**Result:**

# Flow table of the ProVision switch

**Details V3 ASICs:**

- **Ingress Port**
- **VLAN ID**
- **VLAN Priority**
- **Source MAC**
- **Destination MAC**
- **Ethertype**
- **IP source address (IPv4/IPv6)**
- **IP destination address (IPv4/IPv6)**
- **IP protocol**
- **IP DSCP**
- **TCP/UDP source port**
- **TCP/UDP destination port**

# Flow table of the ProVision switch

**Details V3 ASICs:**

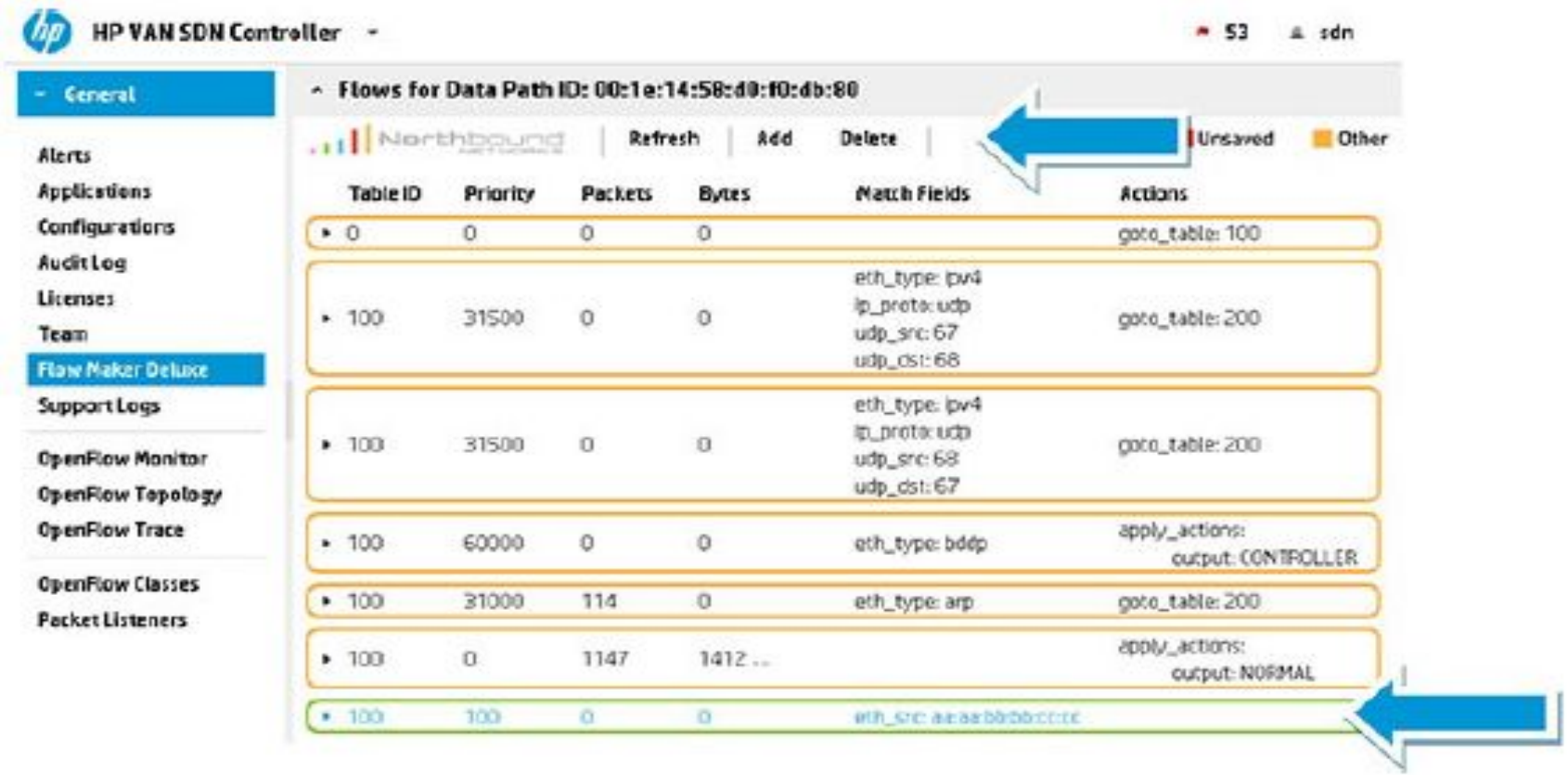
- **Destination port (one or more ports)**
- **Modify source MAC address**
- **Modify destination MAC address**
- **Modify VLAN ID**
- **Modify VLAN priority**
- **Modify IP DSCP**
- **Modify IPv4 source address**
- **Modify IPv4 destination address**
- **Modify TCP/UDP source address (IPv4)**
- **Modify TCP/UDP destination address (IPv4)**

# Flow table of the ProVision switch

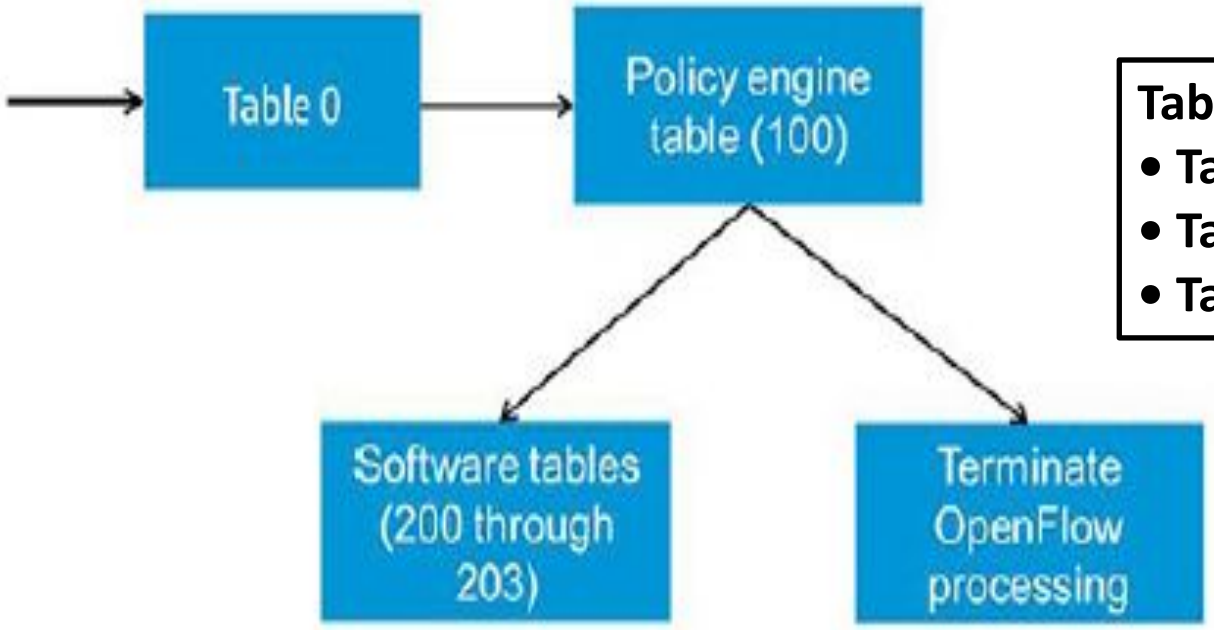
**3800 example**

```
P1# show openflow instance vlan30 flow-table 100 table-capability
OpenFlow Flow Table Properties
Table Match Capabilities:
Incoming Port                          Ethernet Destination
Ethernet Source                        Ethernet Type
VLAN ID                                VLAN PCP
IP DSCP                                IP Protocol
IPv4 Source Address                    IPv4 Destination Address
TCP Source Port                        TCP Destination Port
UDP Source Port                        UDP Destination Port
ICMP Type                              ICMP Code
IPv6 Source Address                    IPv6 Destination Address
Table Instructions:
 Metering
 Apply Actions
  Set-Field
   Ethernet Destination   Ethernet Source
   VLAN ID            VLAN PCP
   IP DSCP
  Output
 GoTo 200
```

# Flow table of the ProVision switch

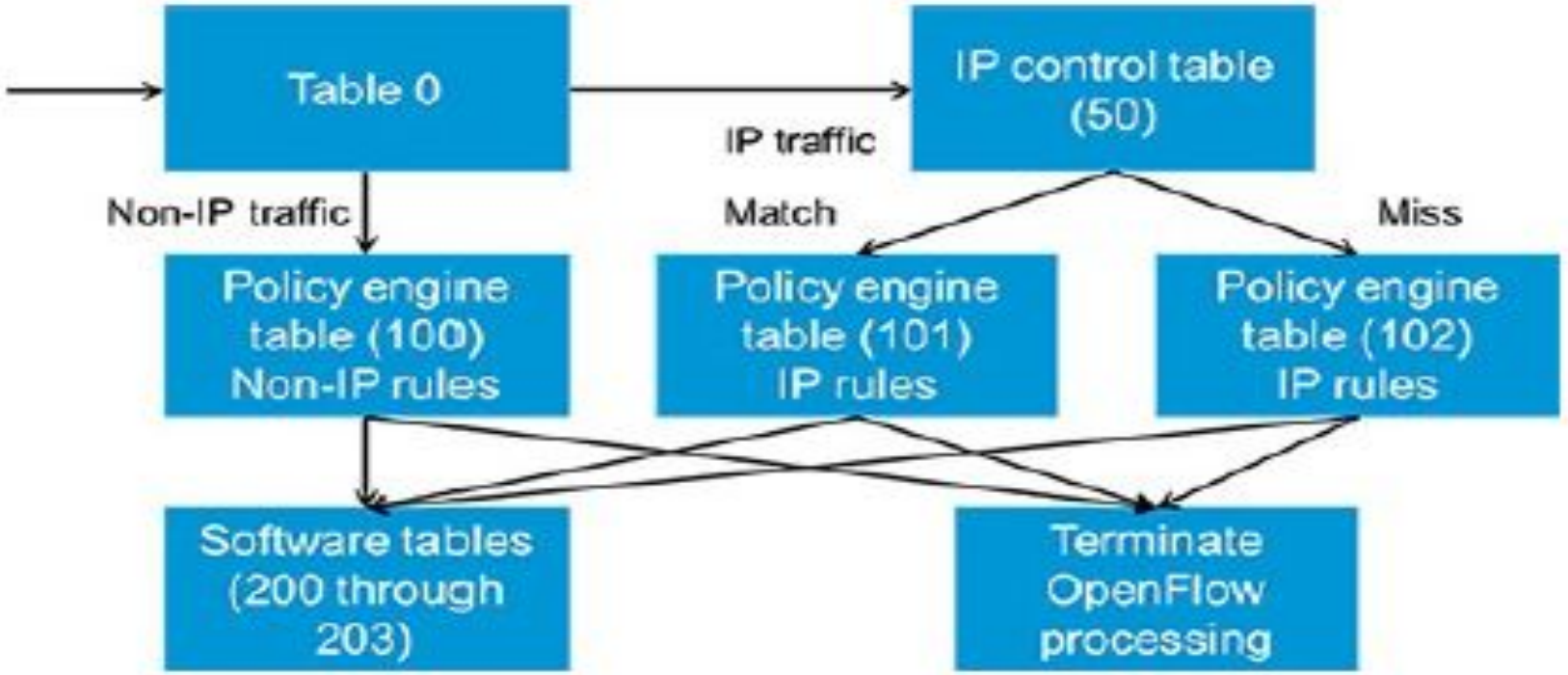


# Flow table of the ProVision switch




**Tables in pipeline:**
- **Table 0: Read-only**
- **Table 100 (TCAM)**
- **Table 200 (Software)**

# Flow table of the ProVision switch



# Flow table of the ProVision switch

## Custom Pipeline Model – Default Pipeline

- All the OpenFlow Tables in this mode are in **hardware**.

| Table-Number/ Property | Table-0 (HASH) | Table-1 (HASH) | Table-2 (HASH) | Table-3 (TCAM) |
|---|---|---|---|---|
| Match Key | VLAN_VID & ETH_SRC | VLAN_VID & ETH_DST | ETH_TYPE, IP_SRC, IP_DST, IP_PROTO, SRC_PORT & DST_PORT | IN_PORT, ETH_SRC, ETH_DST, VLAN_VID, VLAN_PCP, ETH_TYPE, IP_SRC, IP_DST, IP_DSCP, IP_PROTO, SRC_PORT & DST_PORT |
| Instructions | APPLY, WRITE, CLEAR, METER, GOTO | APPLY, WRITE, CLEAR, METER, GOTO | APPLY, WRITE, CLEAR, METER, GOTO | APPLY, WRITE, CLEAR, METER, |
| Scale | 8k | 8k | 8k | 2k |


And what's more - The pipeline can be changed on the fly by the controller using standard OpenFlow constructs (OFPMP_TABLE_FEATURES message). **Create a network pipeline of your choice!**

# Flow table of the Comware switch

**Answer:**

**Answer:**

**Answer:**

**Answer:**

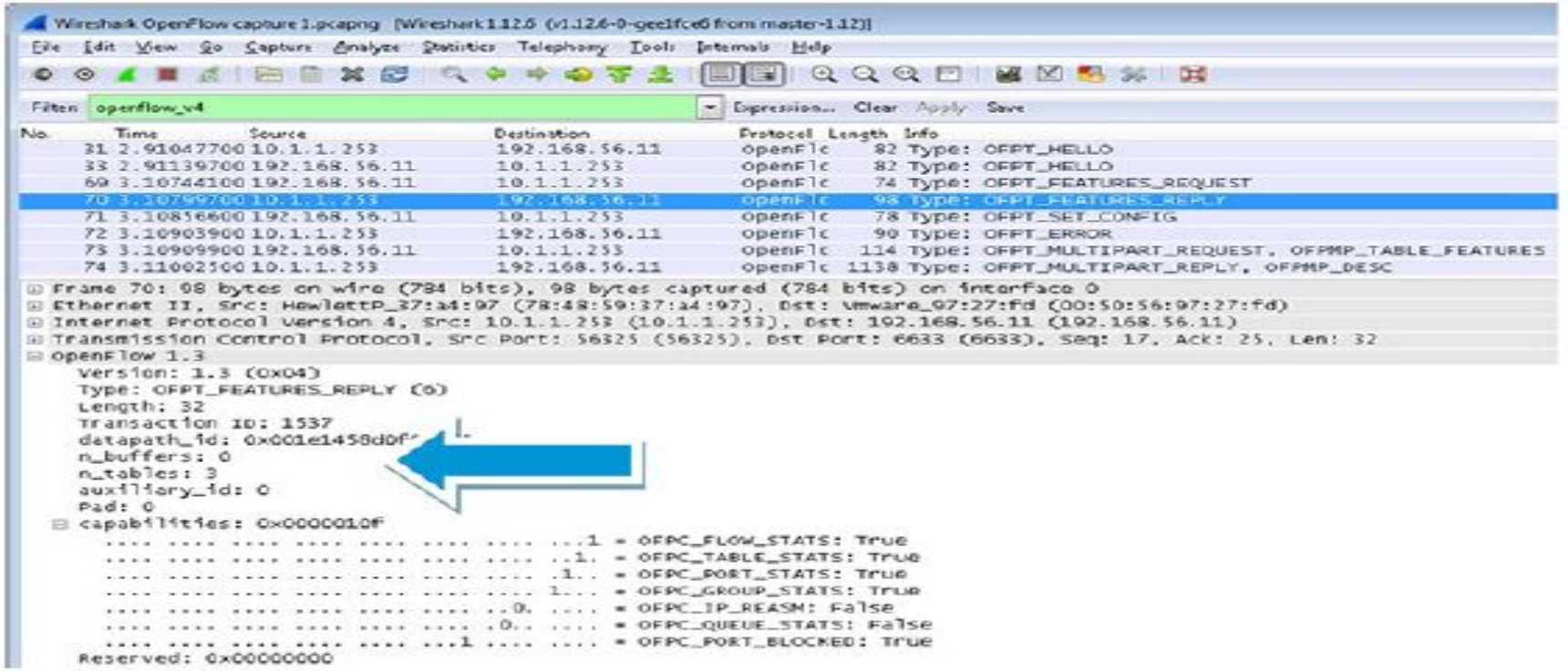
# Flow table of the Comware switch

**Flows for Data Path ID: 00:01:78:48:59:39:2f:96**

Summary | Ports | Flows | Groups

| Table ID | Priority | Packets | Bytes | Match | Actions/Instructions | Flow Class ID |
|---|---|---|---|---|---|---|
| 0 | 60000 | 0 | n/a | eth_type: bddp | apply_actions:<br>output: CONTROLLER | com.hp.sdn.bddp.steal |
| 0 | 31500 | 248 | n/a | eth_type: ipv4<br>ip_proto: udp<br>udp_src: 68<br>udp_dst: 67 | apply_actions:<br>output: CONTROLLER<br>write_actions:<br>output: NORMAL | com.hp.sdn.dhcp.copy |
| 0 | 31500 | 0 | n/a | eth_type: ipv4<br>ip_proto: udp<br>udp_src: 67<br>udp_dst: 68 | apply_actions:<br>output: CONTROLLER<br>write_actions:<br>output: NORMAL | com.hp.sdn.dhcp.copy |
| 0 | 31000 | 0 | n/a | eth_type: arp | apply_actions:<br>output: CONTROLLER<br>write_actions:<br>output: NORMAL | com.hp.sdn.arp.copy |
| 0 | 0 | 634 | n/a | | apply_actions:<br>output: NORMAL | com.hp.sdn.normal |

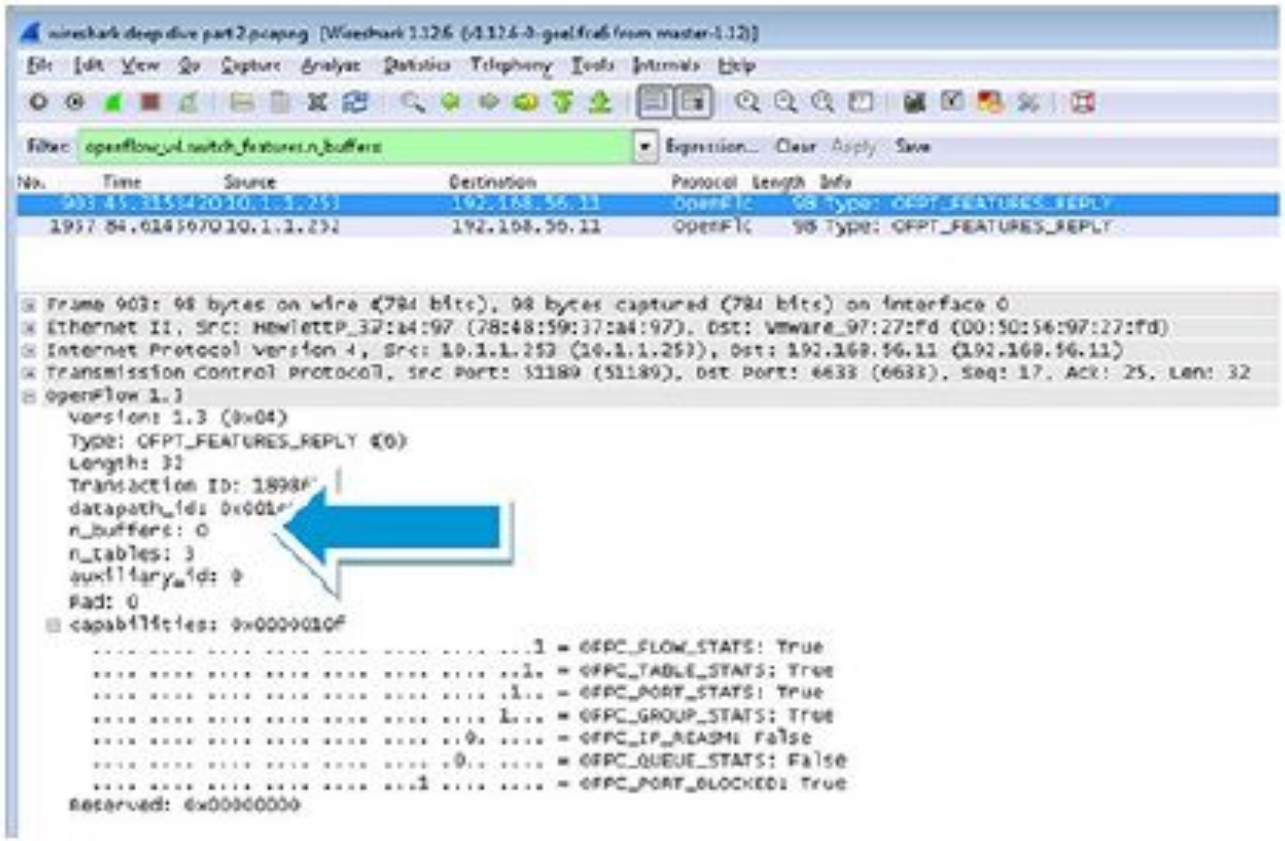
# Flow table of the Comware switch



# Flow table of the Comware switch

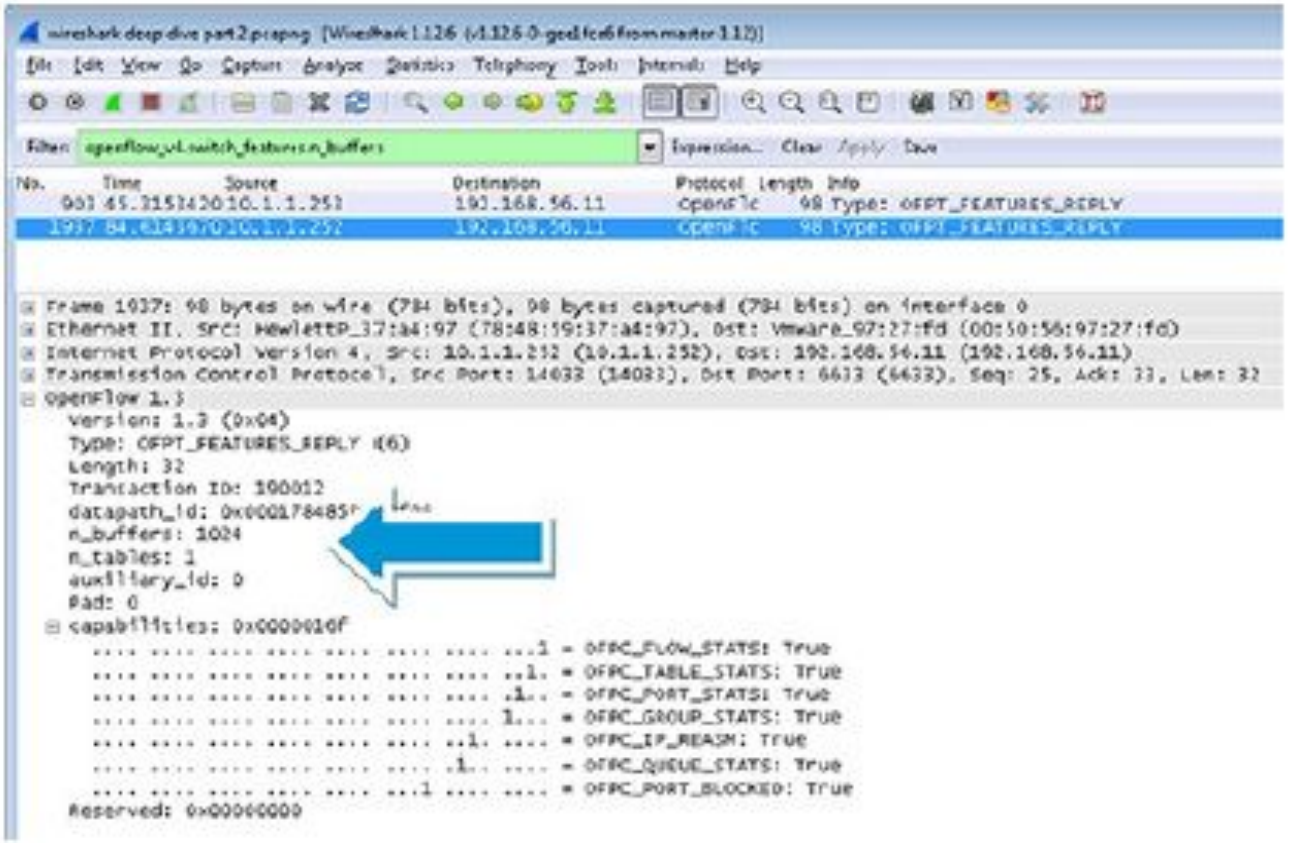
openflow_v4.switch_features.n_buffers.

**ProVision number of buffers**



# Flow table of the Comware switch

**Comware number of buffers**




**n_buffers:**

# Flow table of the Comware switch

**Answer:**

**Answer:**

**Answer:**

**Answers:**

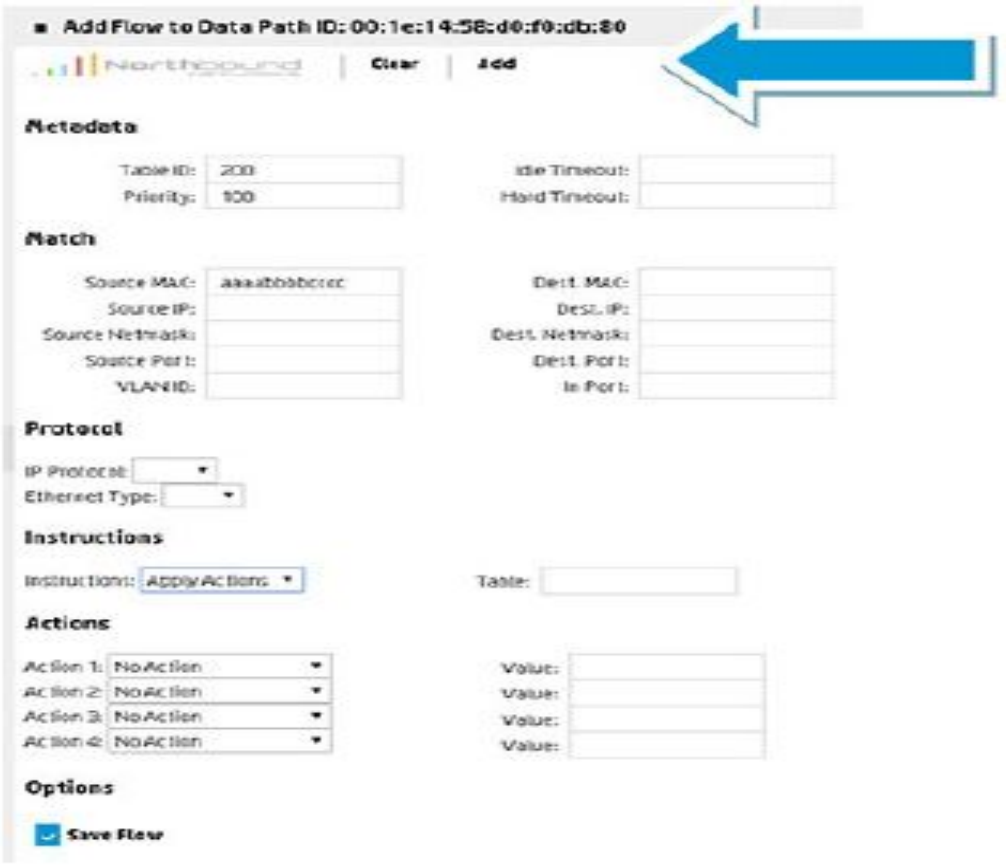
# Flow table of the Comware switch

# Flow table of the Comware switch

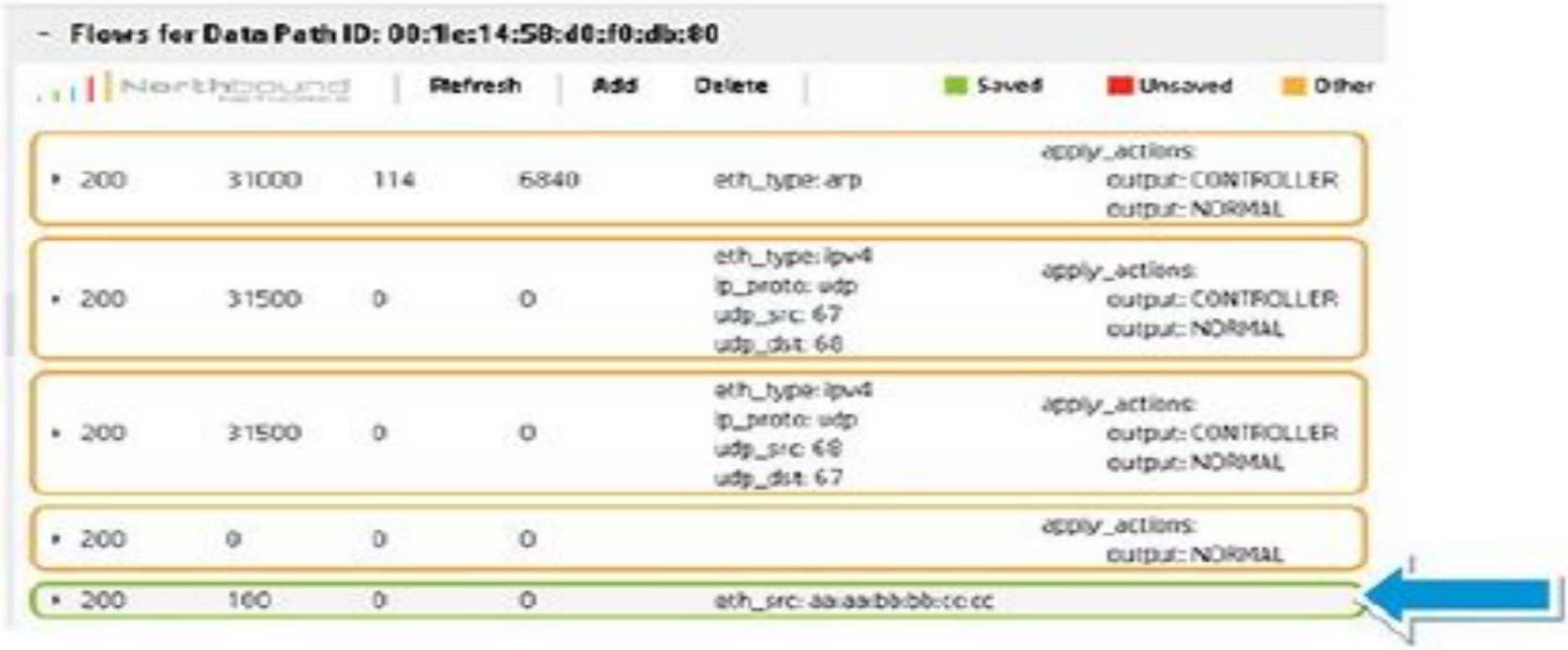
**ProVision switch**

**Will this rule work? Add a flow entry with the following attributes and then click Add, as illustrated in Figure.**

- **Table ID: 200**
- **Priority: 100**
- **Src Mac: aaaabbbbcccc**
- **Instructions: Apply Actions**
- **Action 1: No Action**
- **Save Flow = True**



# Flow table of the Comware switch

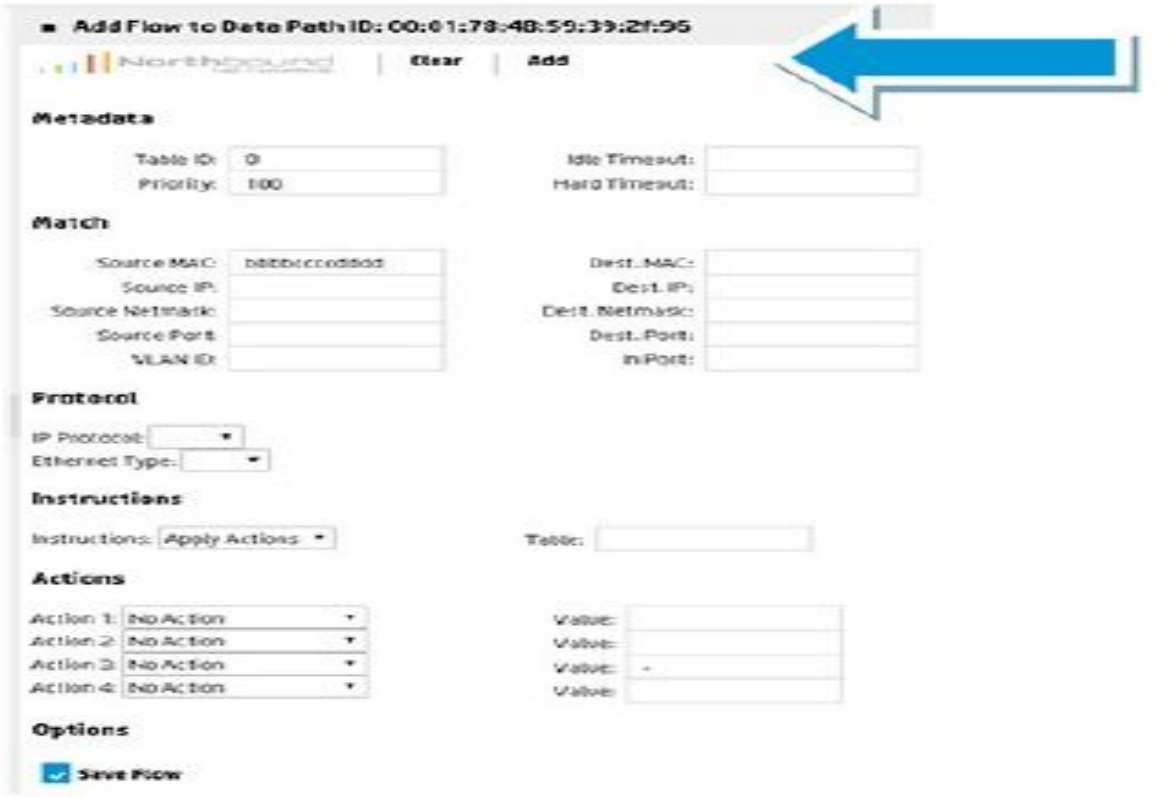



**Result:**

# Flow table of the Comware switch

## Comware switch

- Table ID: 0
- Priority: 100
- Src Mac: bbbbccccdddd
- Instructions: Apply Actions
- Action 1: No Action
- Save Flow = True



# Flow table of the Comware switch


**Result:**

# Flow table of the Comware switch


# Investigating OpenFlow negotiations

# Flow modifications (Flow mod)

- **Set the Wireshark filter to: openflow_v4.flowmod.command.**
- **Click Apply.**

# Flow modifications (Flow mod)

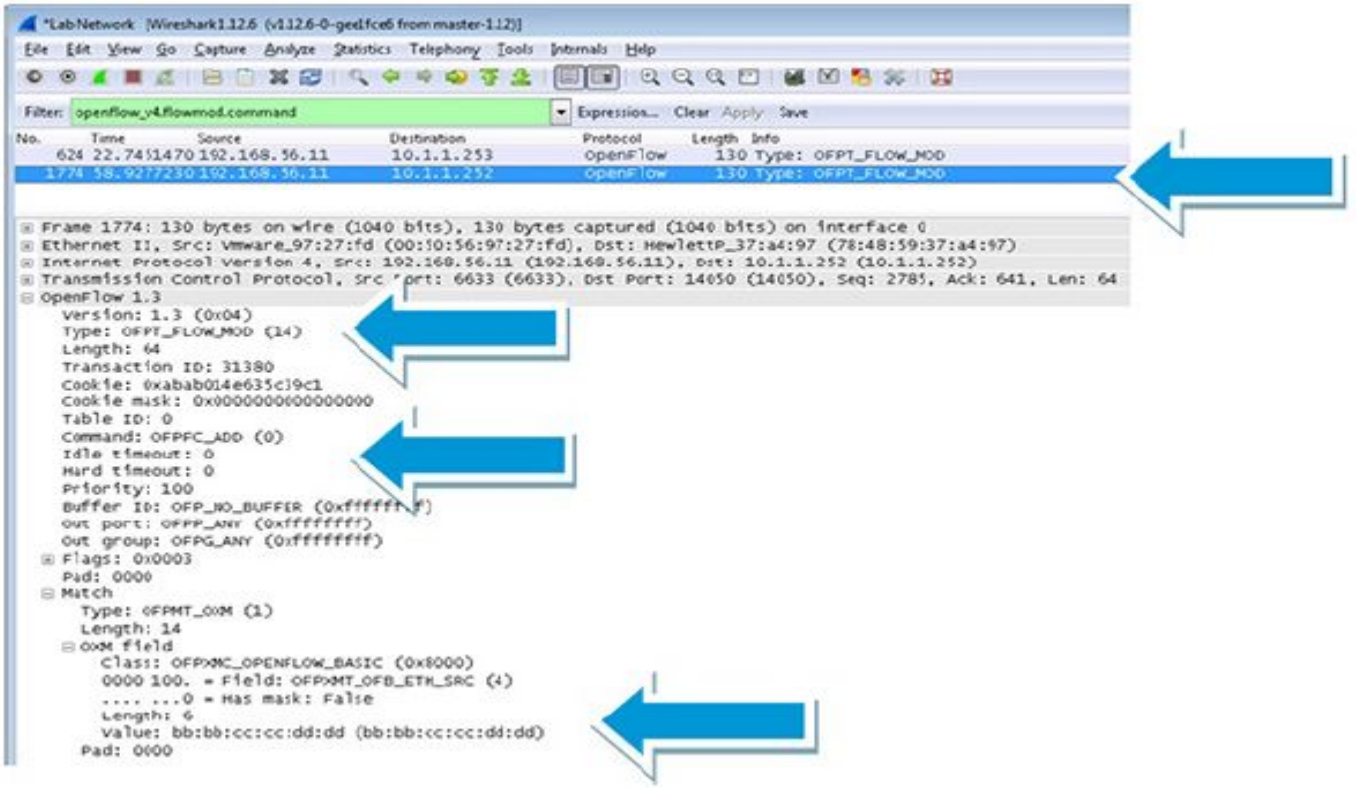
**Answer:**

```
enum ofp_flow_mod_command {
  OFPFC_ADD = 0, /* New flow. */
  OFPFC_MODIFY = 1, /* Modify all matching flows. */
  OFPFC_MODIFY_STRICT = 2, /* Modify entry strictly matching wildcards and priority.*/
  OFPFC_DELETE = 3, /* Delete all matching flows. */
  OFPFC_DELETE_STRICT = 4, /* Delete entry strictly matching wildcards and priority. */
```

# Flow modifications (Flow mod)

**Comware switch**

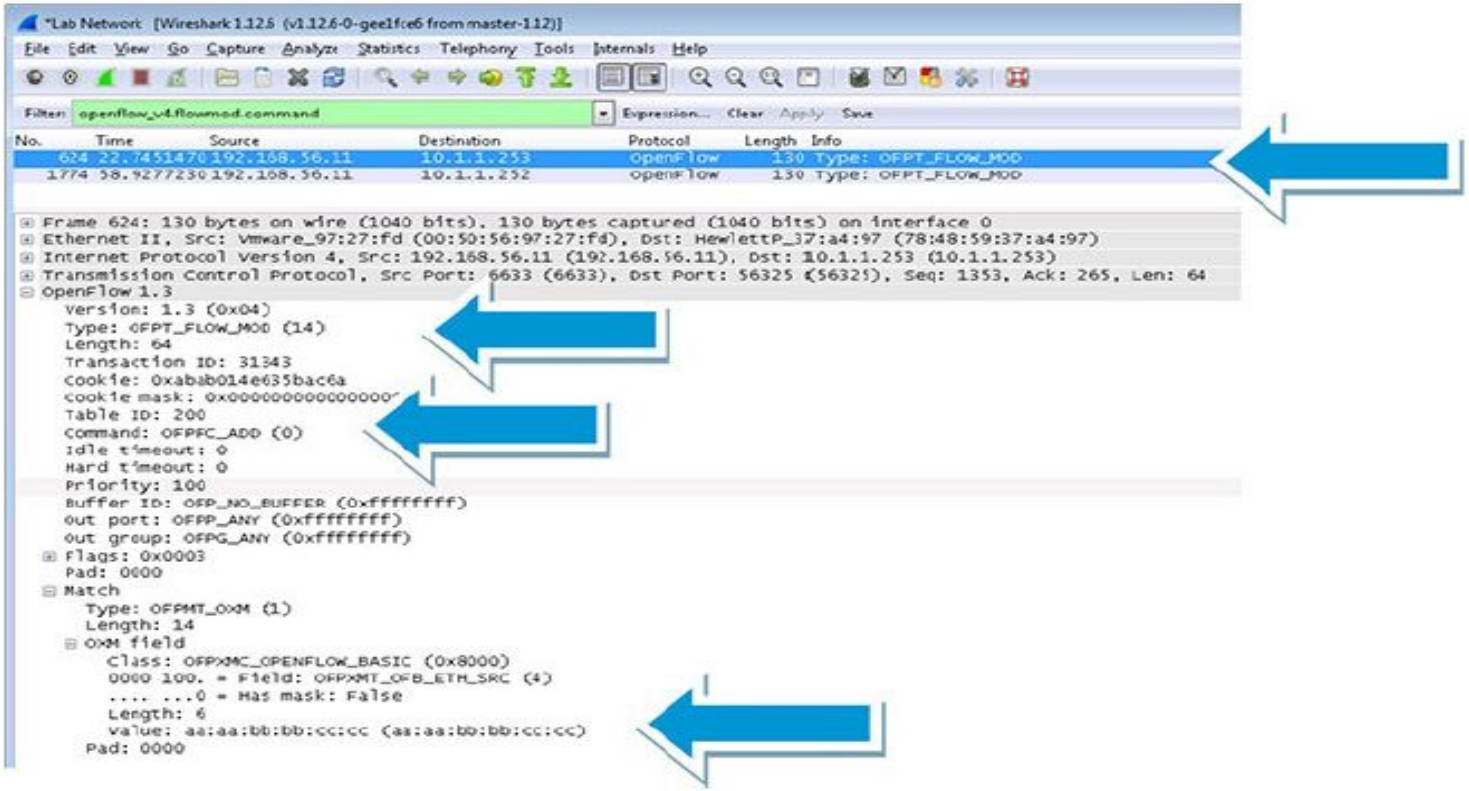
**Result:** This is an OFPFC_ADD message. The controller is adding a new flow to the switch.



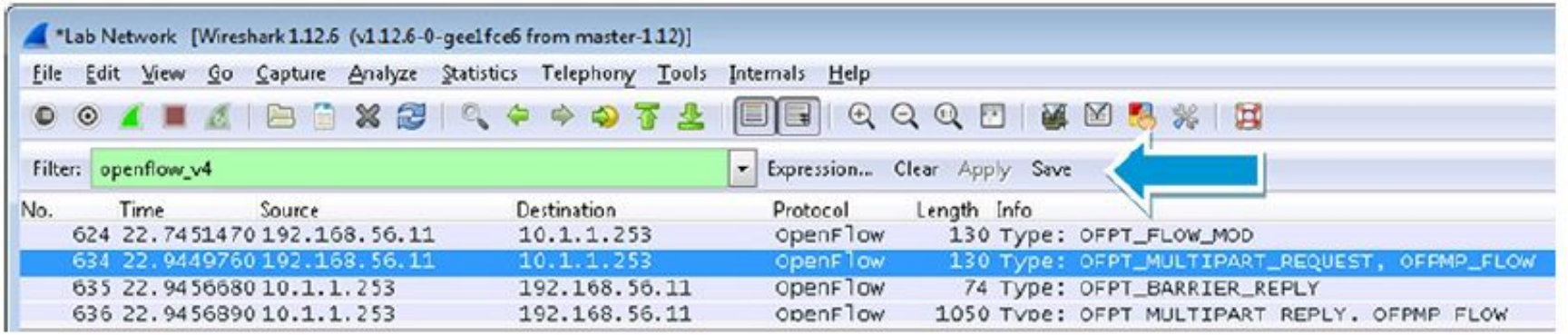
# Flow modifications (Flow mod)

**ProVision switch**

**Result:** This is also an OFPFC_ ADD message

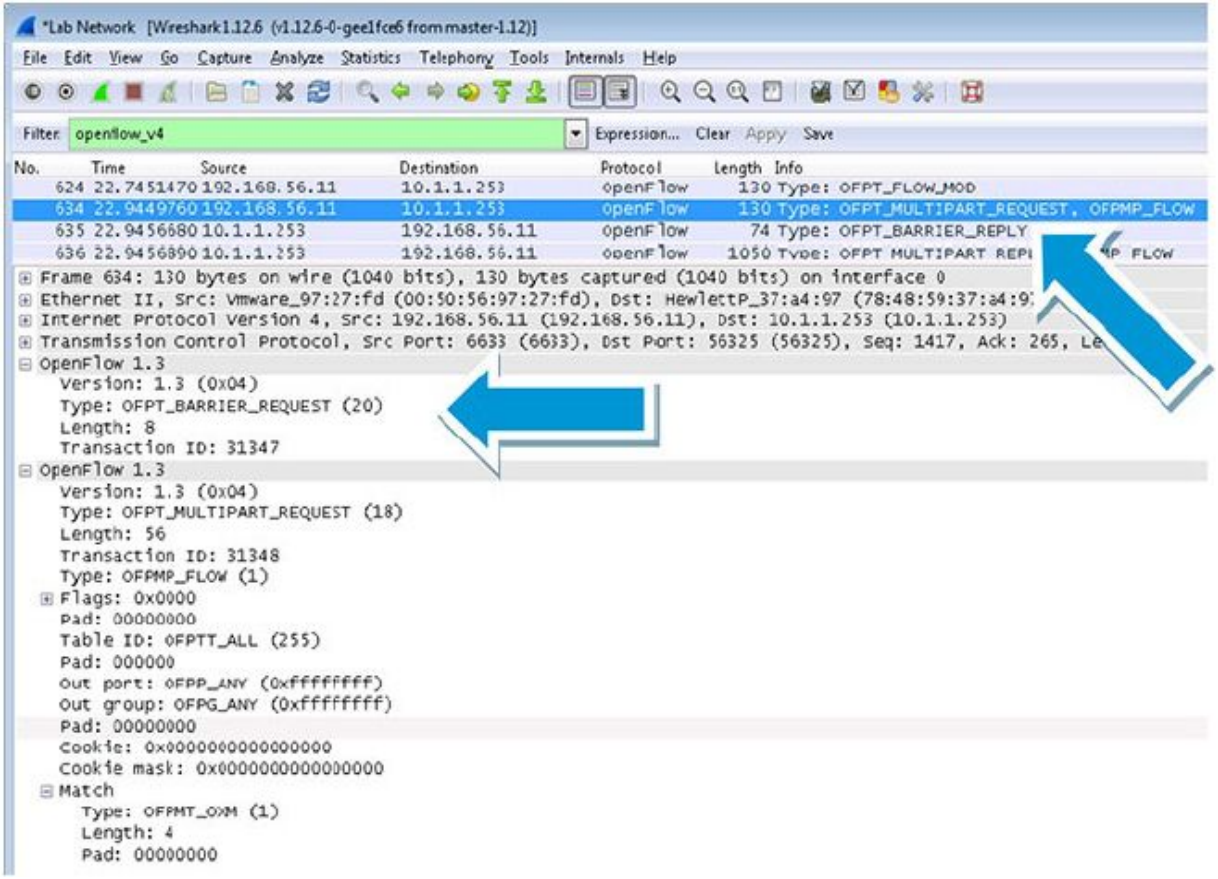


# Flow modifications (Flow mod)

# Flow modifications (Flow mod)

**Barrier:**

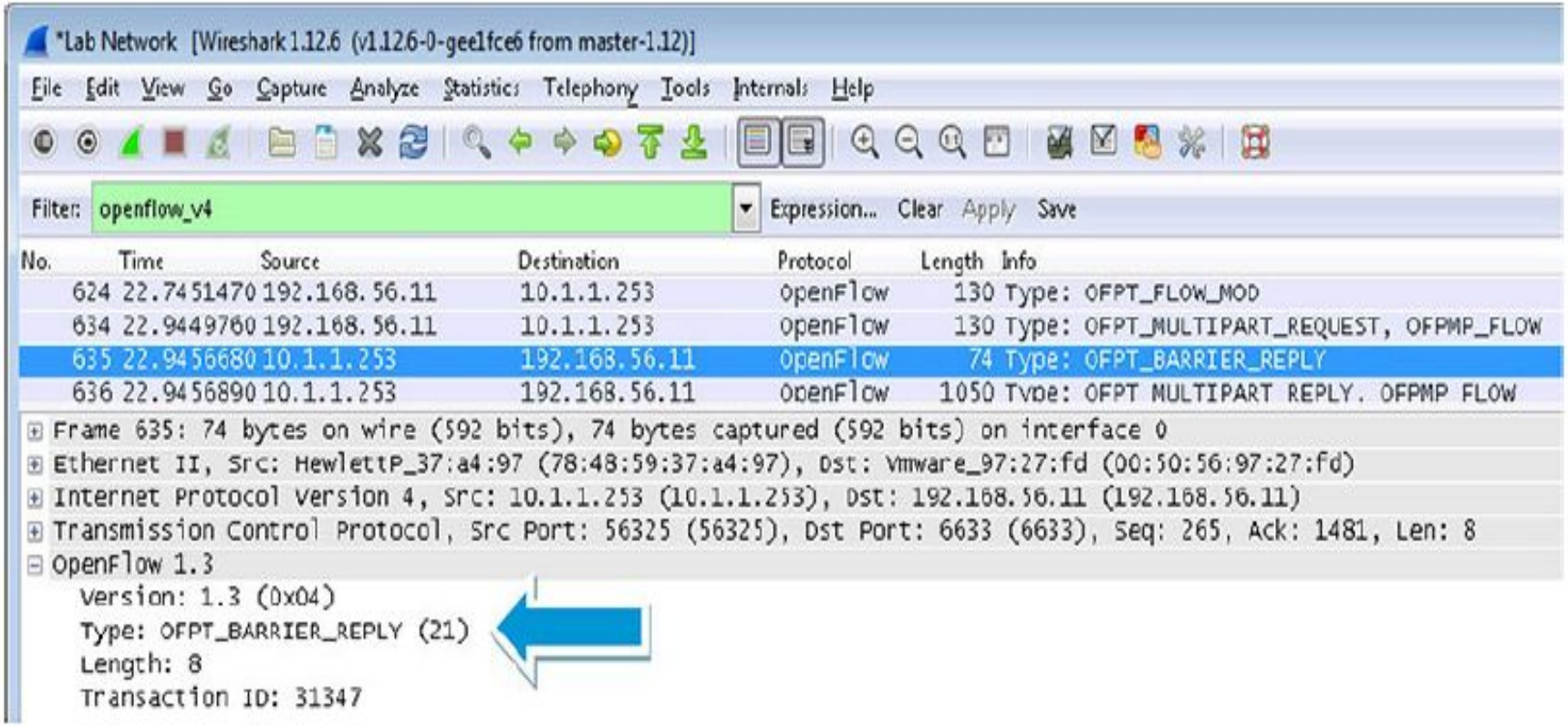


# Flow modifications (Flow mod)

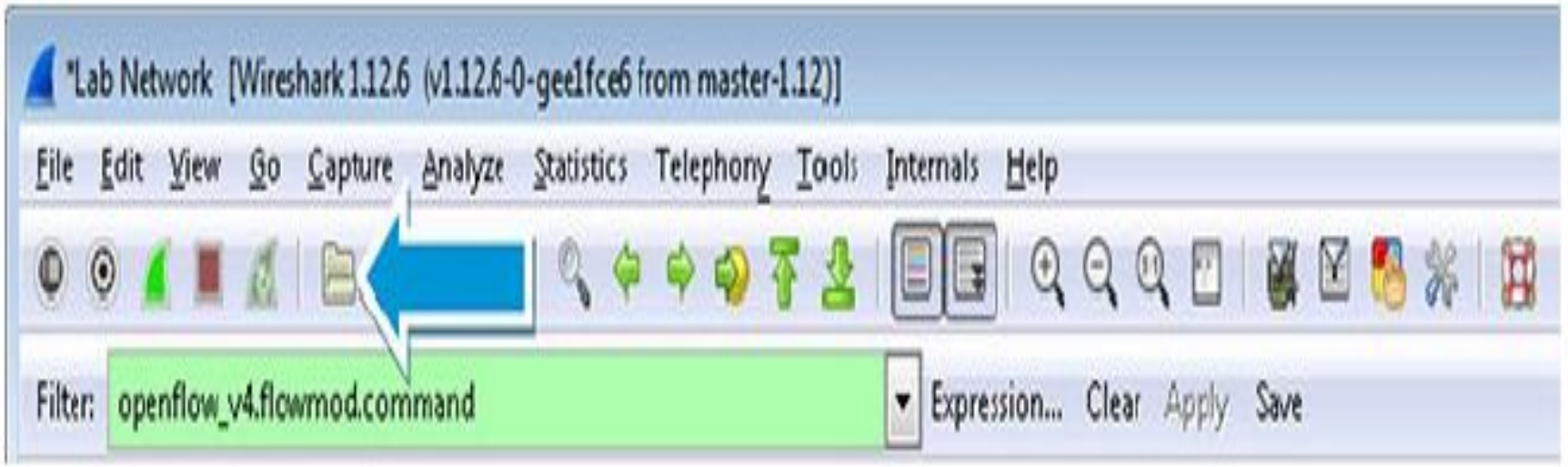
**Message ordering:**

- **1. Messages before a barrier must be fully processed before the barrier, including sending any resulting replies or errors.**
- **2. The barrier must then be processed and a barrier reply sent.**
- **3. Messages after the barrier may then begin processing.**

# Flow modifications (Flow mod)

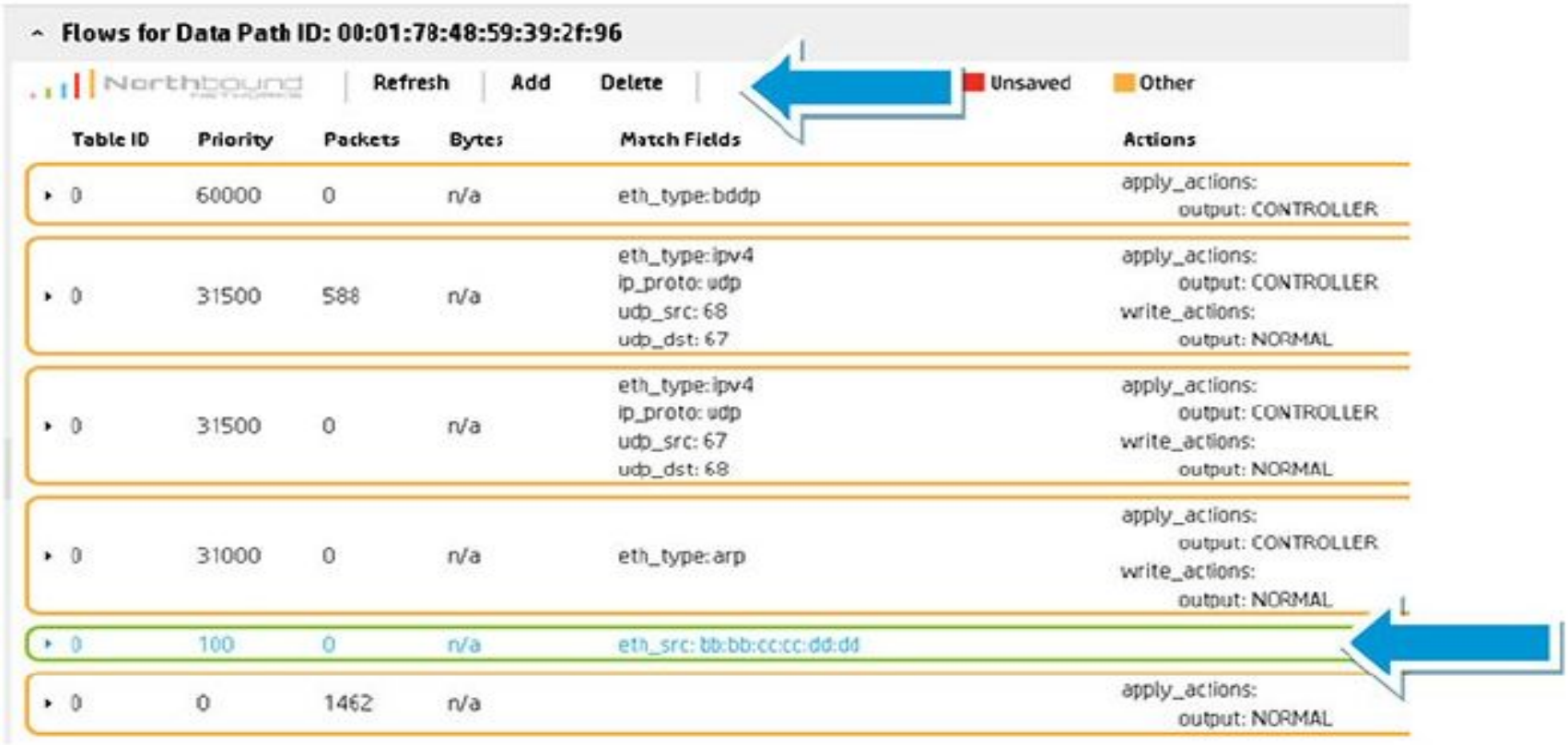


# Flow modifications (Flow mod)

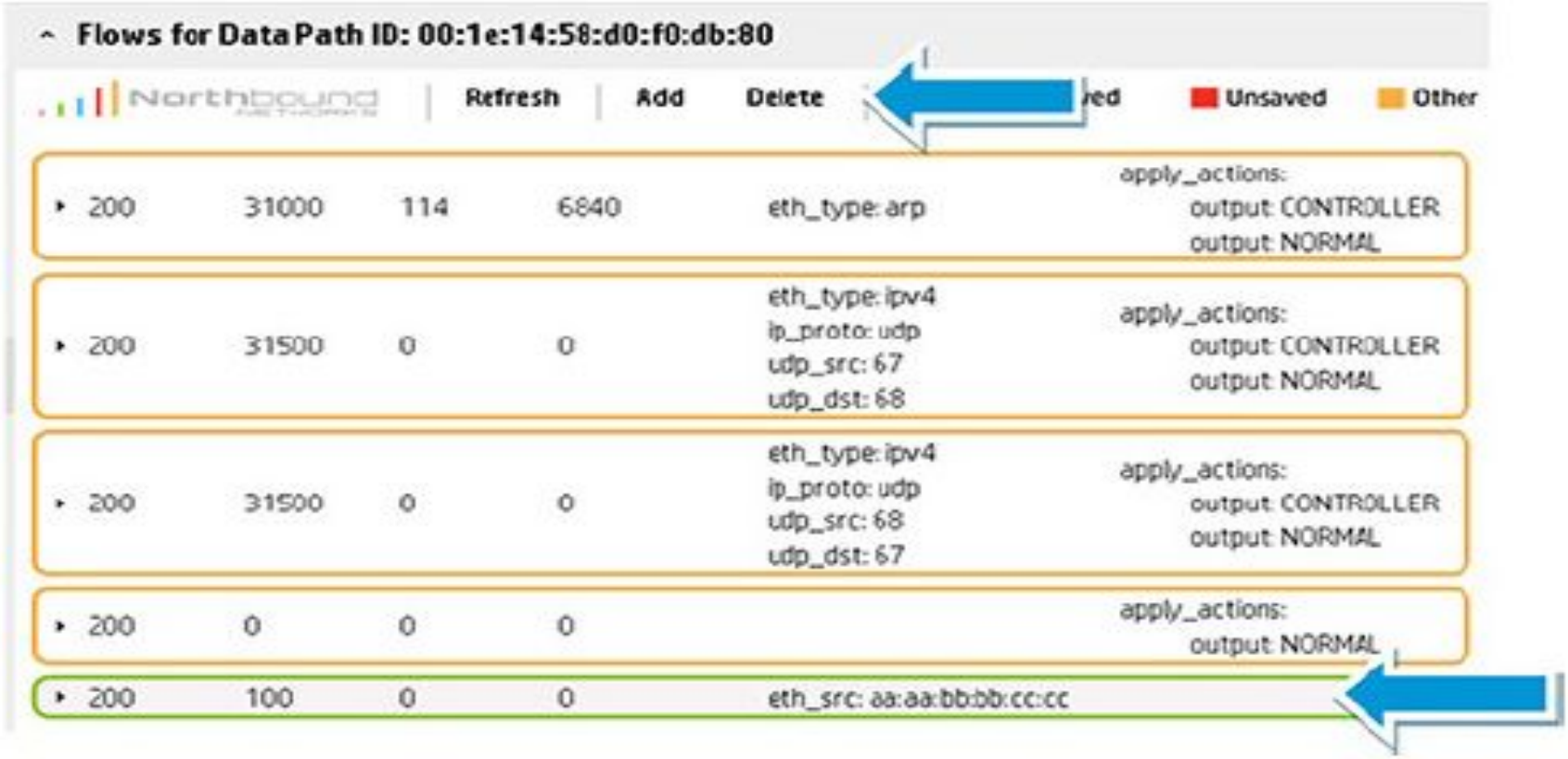


# Flow modifications (Flow mod)

## Comware switch

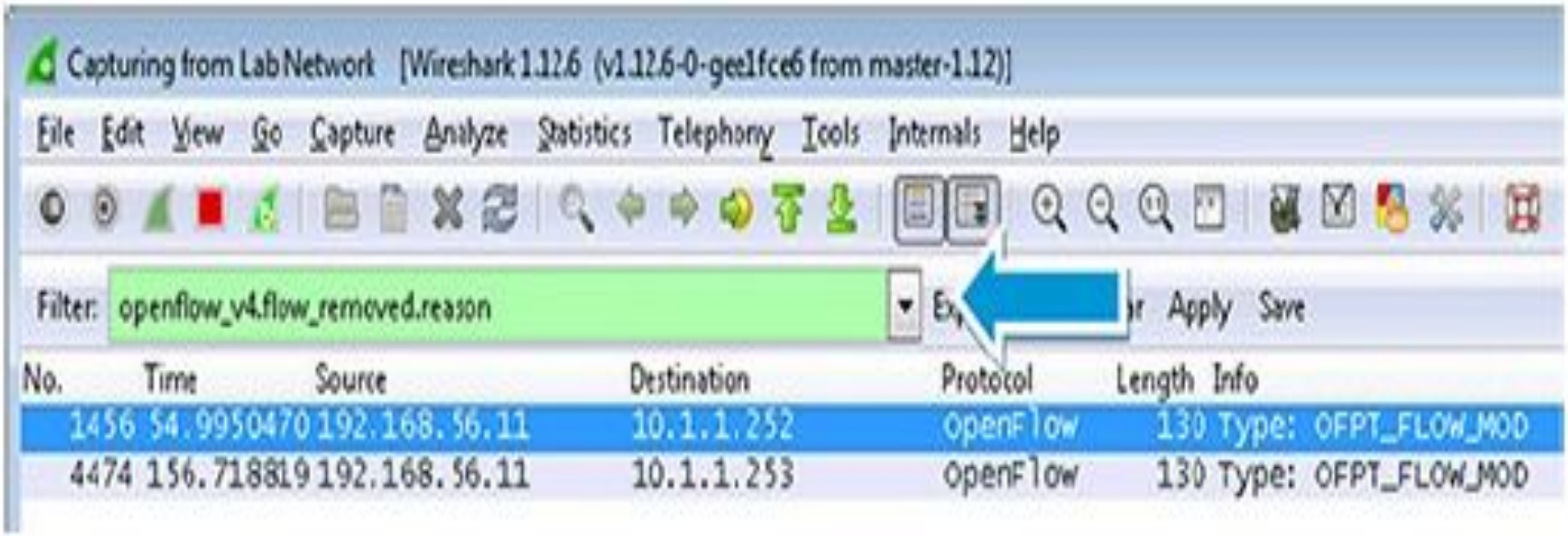


# Flow modifications (Flow mod)



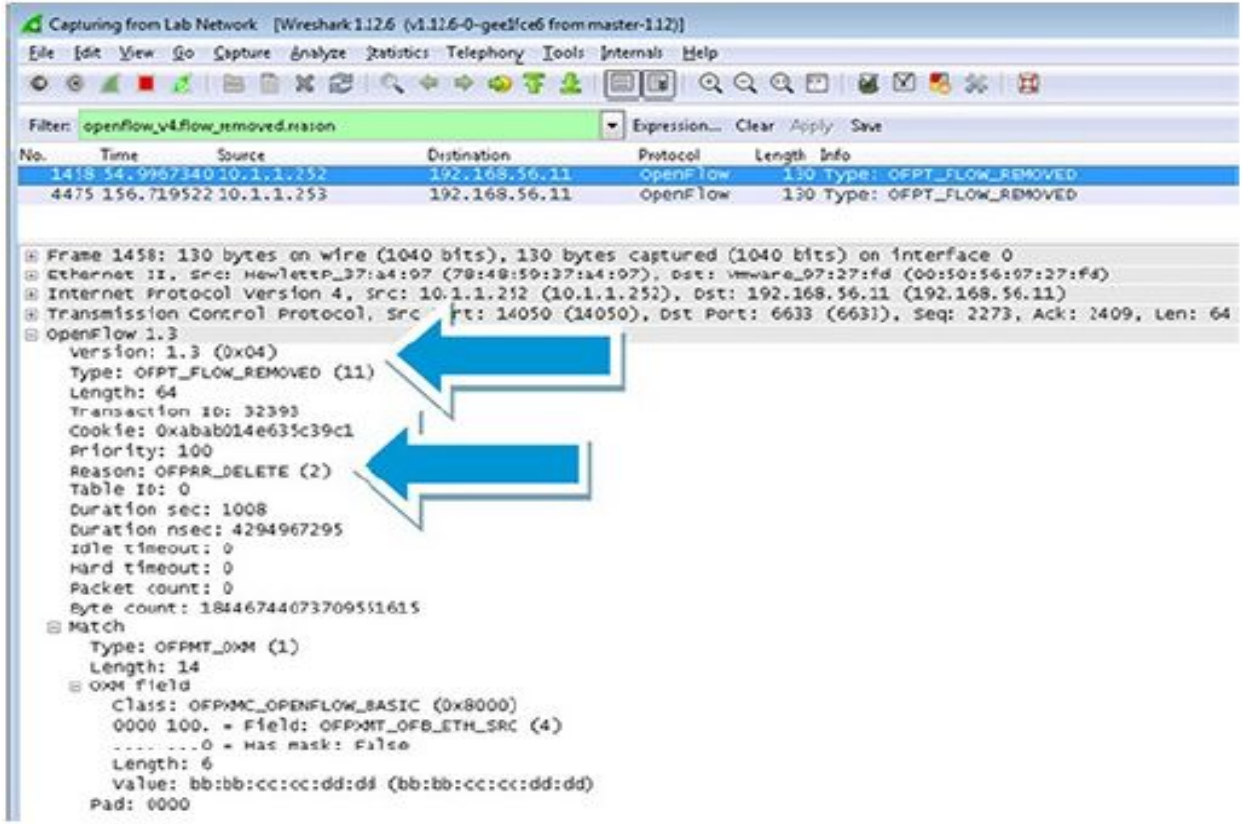
# Flow modifications (Flow mod)

**openflow_v4.flow_removed.reason**



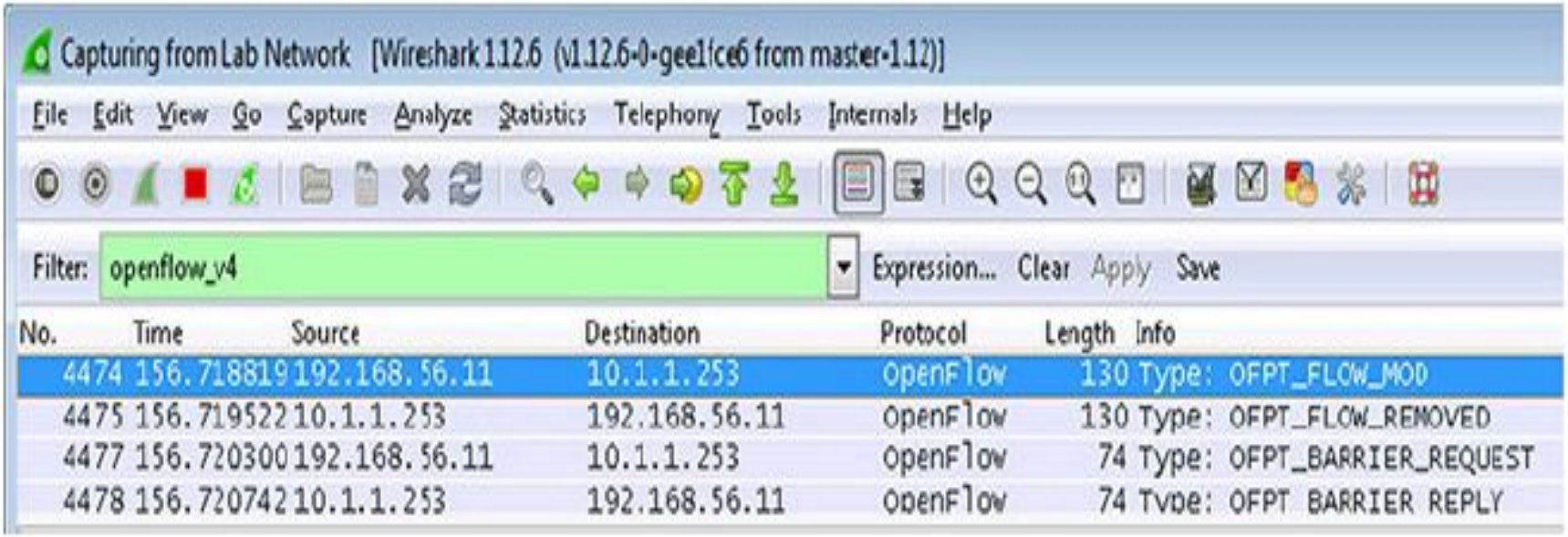
# Flow modifications (Flow mod)

In the Wireshark capture, the switch is informing the controller of the flow removal (OFPT_FLOW_REMOVED) and the reason for the removal (OFPRR_DELETE (2)).

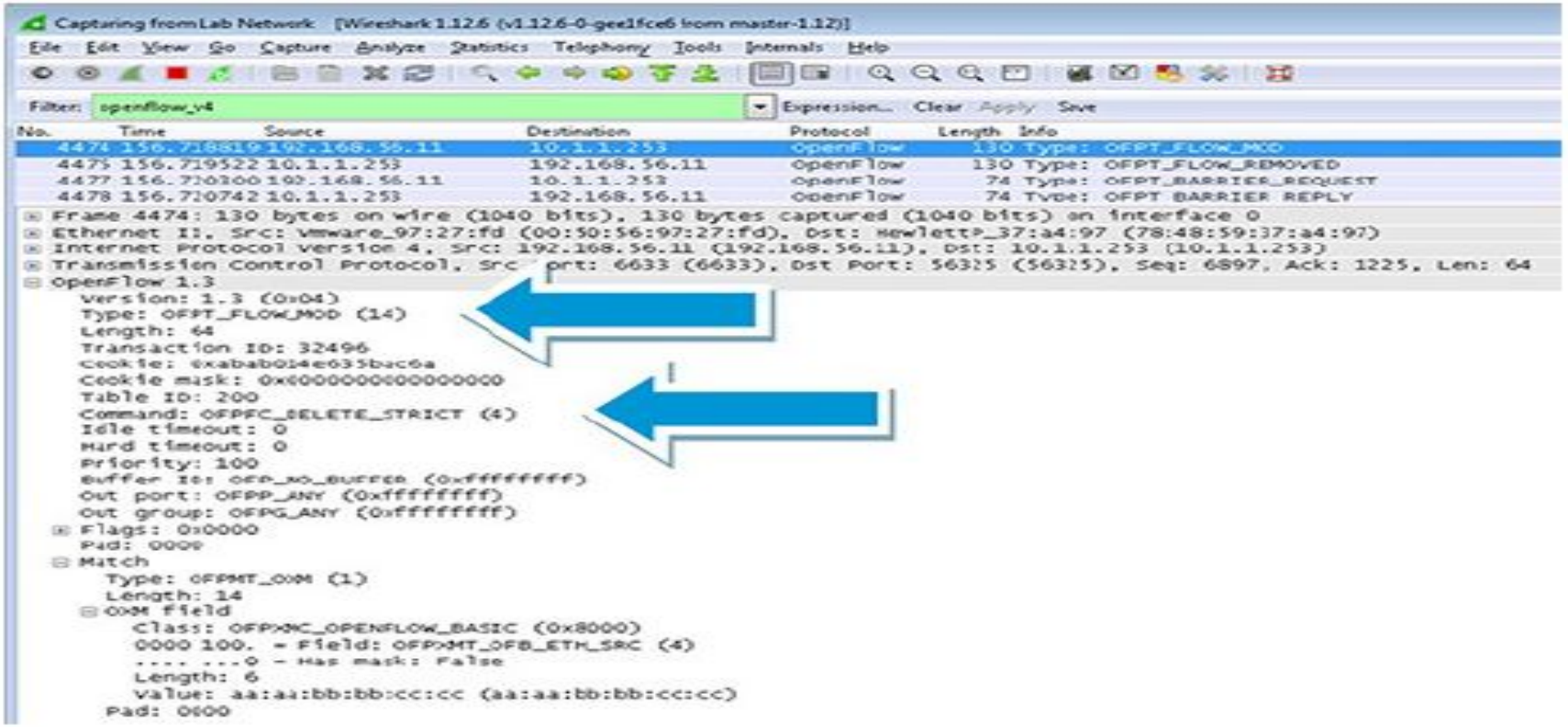


# Flow modifications (Flow mod)

# Flow modifications (Flow mod)



# Flow modifications (Flow mod)



# Flow modifications (Flow mod)

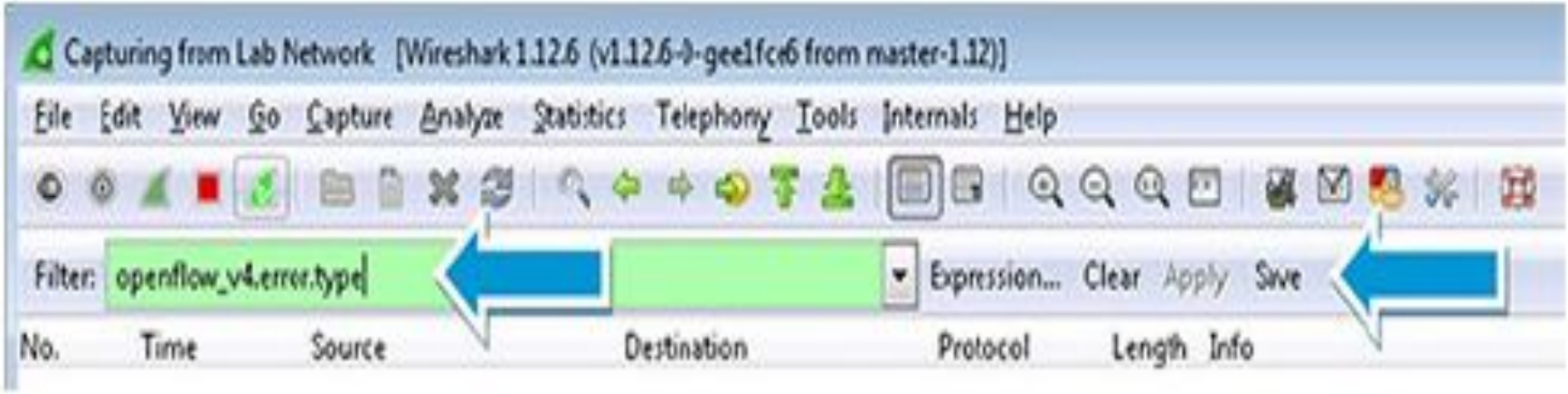
```
enum ofp_flow_mod_command {
  OFPFC_ADD =0,/* New flow. */
  OFPFC_MODIFY =1, /* Modify all matching flows. */
  OFPFC_MODIFY_STRICT = 2, /* Modify entry strictly matching wildcards and priority. */
  OFPFC_DELETE =3,/* Delete all matching flows. */
  OFPFC_DELETE_STRICT = 4, /* Delete entry strictly matching wildcards and priority. */
```

# Flow modifications (Flow mod)

- Type: OFPT_FLOW_MOD = 14
- Command: OFPFC_DELETE_STRICT = 4

# Flow modifications (Flow mod)

- Set the Wireshark filter to: openflow_v4.error.type
- Click Apply
- Click Restart a new live capture

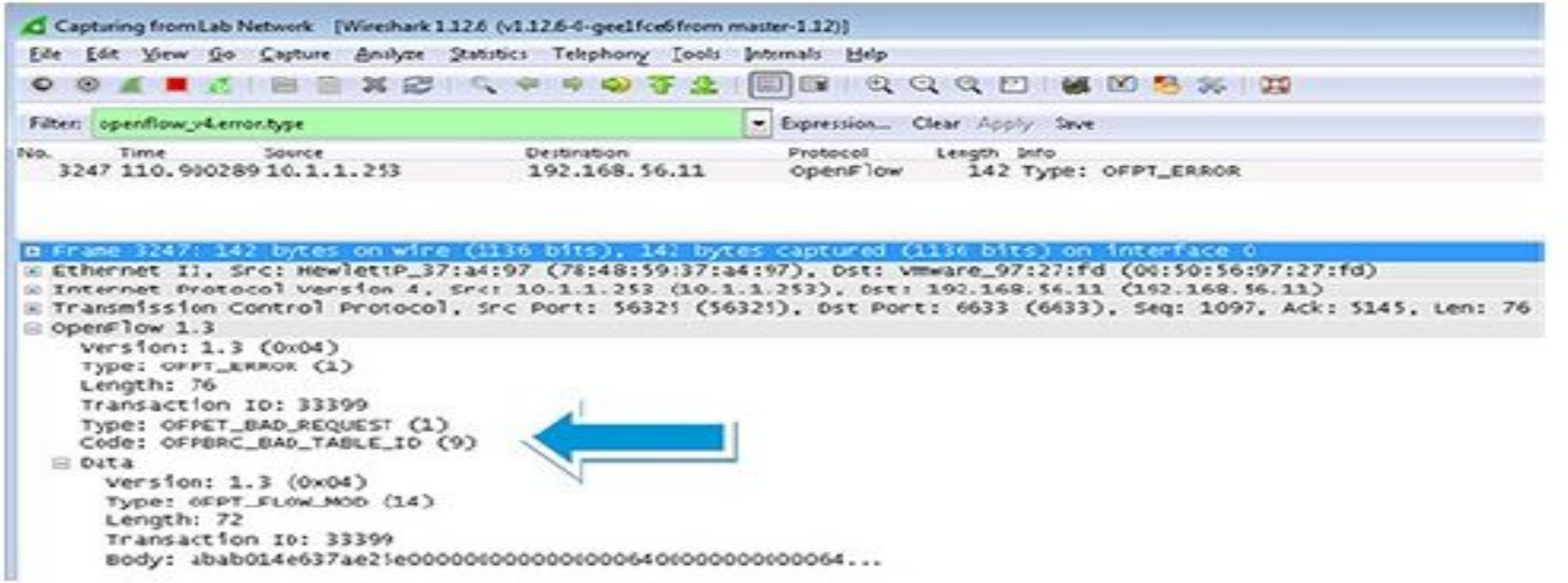


# Flow modifications (Flow mod)

In Flow Maker, add a flow entry with the following attributes to the Provision Switch (10.1.1.253) and then click Add (see Figure):

- Table ID: 100
- Priority: 100
- In Port: 2
- Instructions: Goto Table
- Table: 150
- Save Flow: True

# Flow modifications (Flow mod)

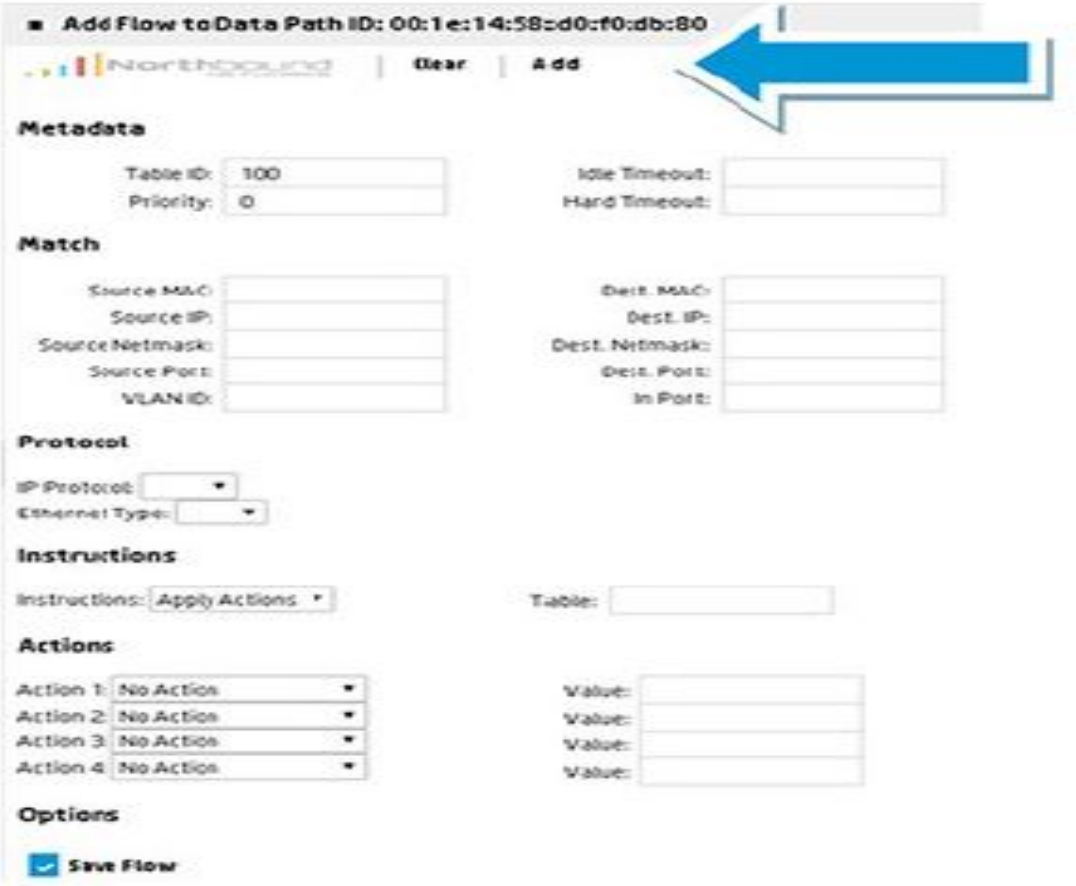



**Result:**

# Flow modifications (Flow mod)

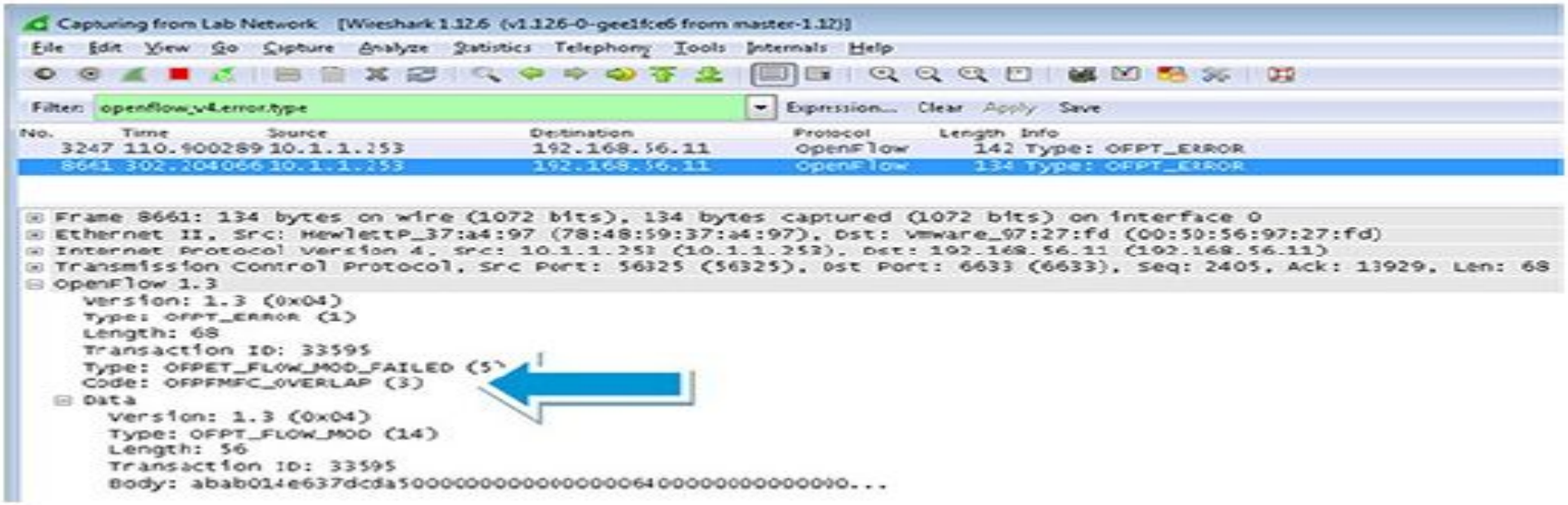
Add a flow entry to the ProVision switch (10.1.1.253) with the following attributes and then click Add (see Figure):

- Table ID: 100
- Priority: 0
- Instructions: Apply Actions
- Save Flow: True

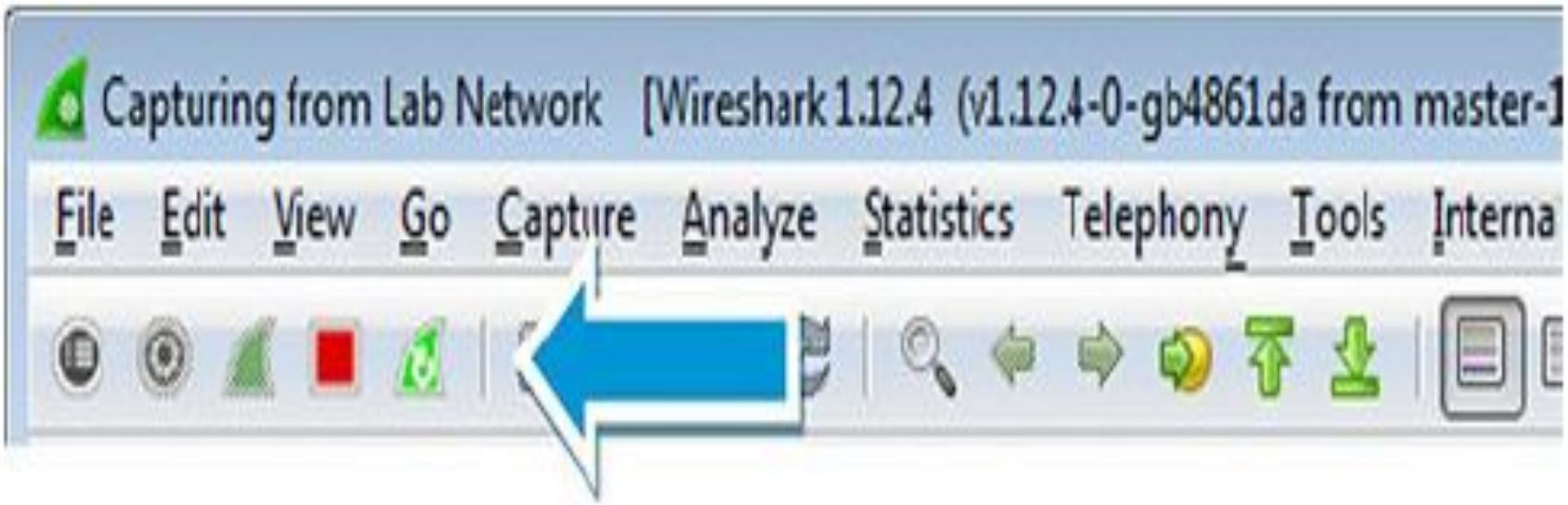


# Flow modifications (Flow mod)

**Result:**

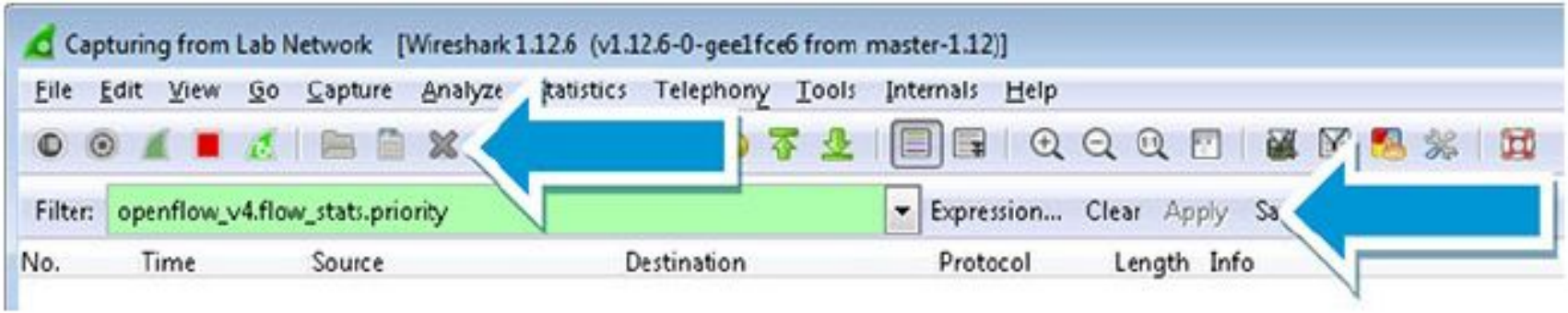


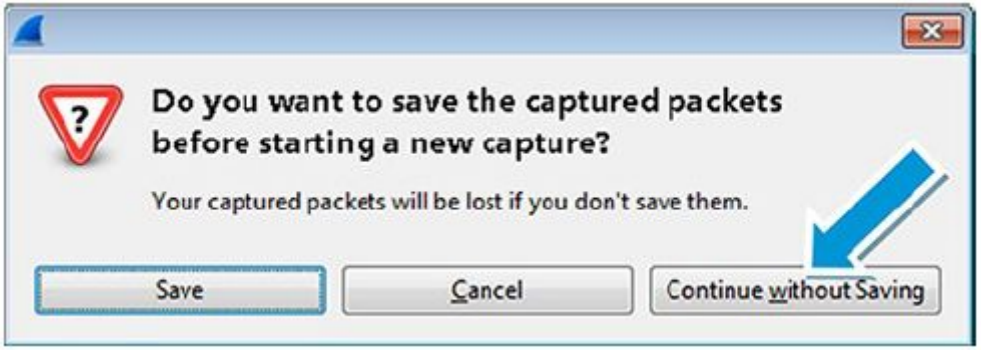
# Flow modifications (Flow mod)



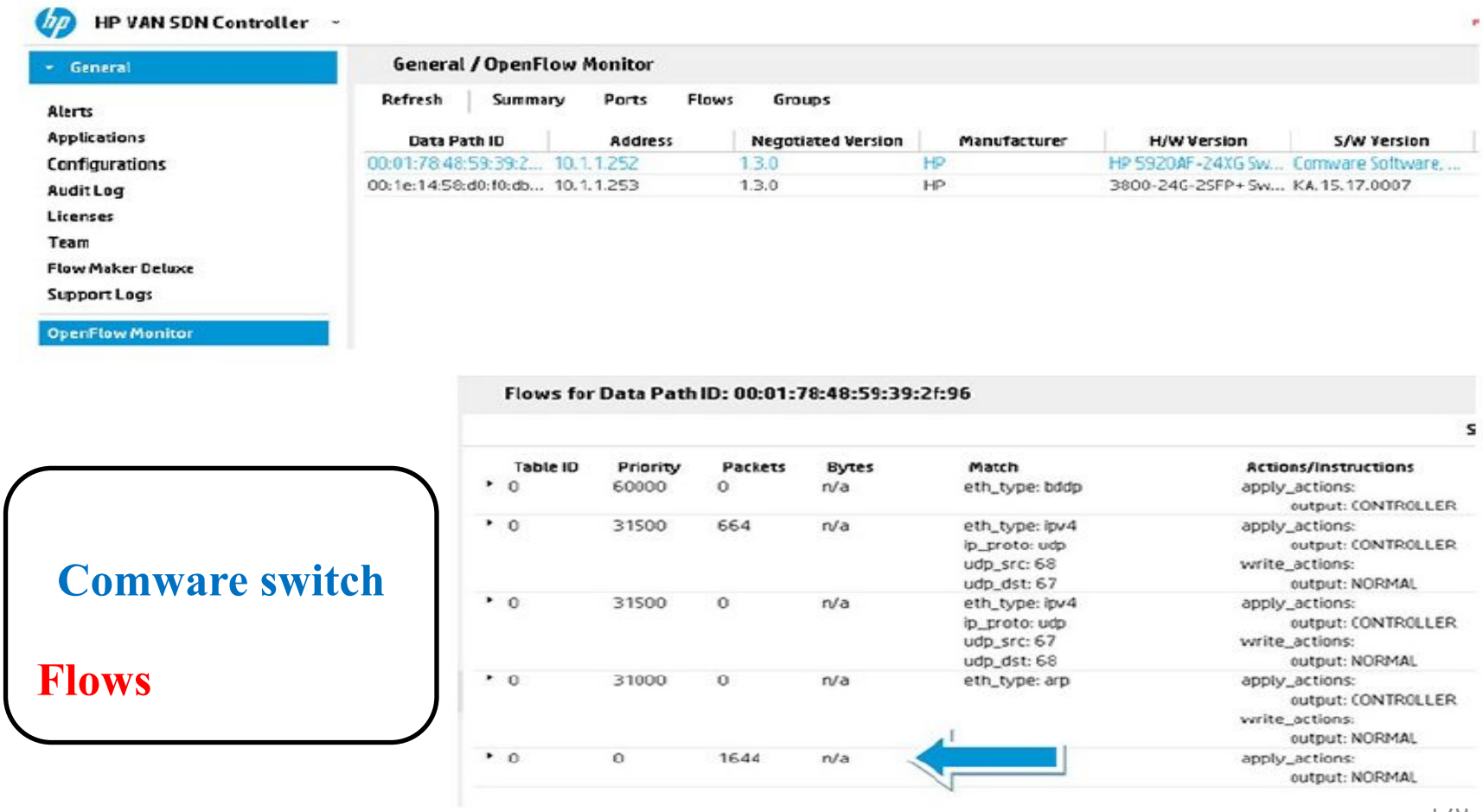
# Flow modifications (Flow mod)

- Set the Wireshark filter to: openflow_v4.flow_stats.priority
- Click Apply
- Click Start to start a new live capture
- Click Continue without Saving



# Flow modifications (Flow mod)

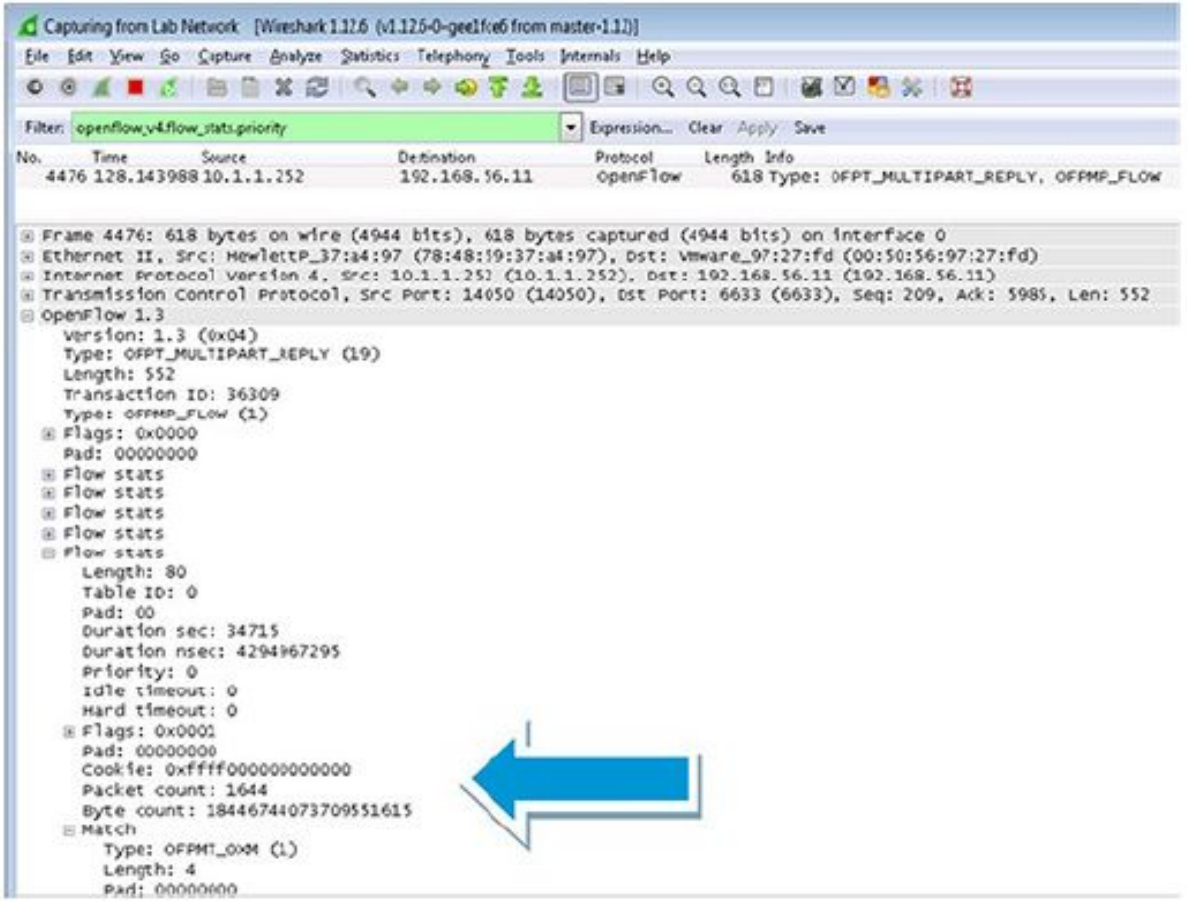



Comware switch

Flows

# Flow modifications (Flow mod)

**Packet count: 1644.**



# Flow modifications (Flow mod)

**Flows for Data Path ID: 00:01:78:48:59:39:2f:96**

| Table ID | Priority | Packets | Bytes | Match | Actions/Instructions |
|---|---|---|---|---|---|
| 0 | 60000 | 0 | n/a | eth_type: bddp | apply_actions:<br>output: CONTROLLER |
| 0 | 31500 | 671 | n/a | eth_type: ipv4<br>ip_proto: udp<br>udp_src: 68<br>udp_dst: 67 | apply_actions:<br>output: CONTROLLER<br>write_actions:<br>output: NORMAL |
| 0 | 31500 | 0 | n/a | eth_type: ipv4<br>ip_proto: udp<br>udp_src: 67<br>udp_dst: 68 | apply_actions:<br>output: CONTROLLER<br>write_actions:<br>output: NORMAL |
| 0 | 31000 | 0 | n/a | eth_type: arp | apply_actions:<br>output: CONTROLLER<br>write_actions:<br>output: NORMAL |
| 0 | 0 | 1651 | n/a | | apply_actions:<br>output: NORMAL |

# Flow modifications (Flow mod)