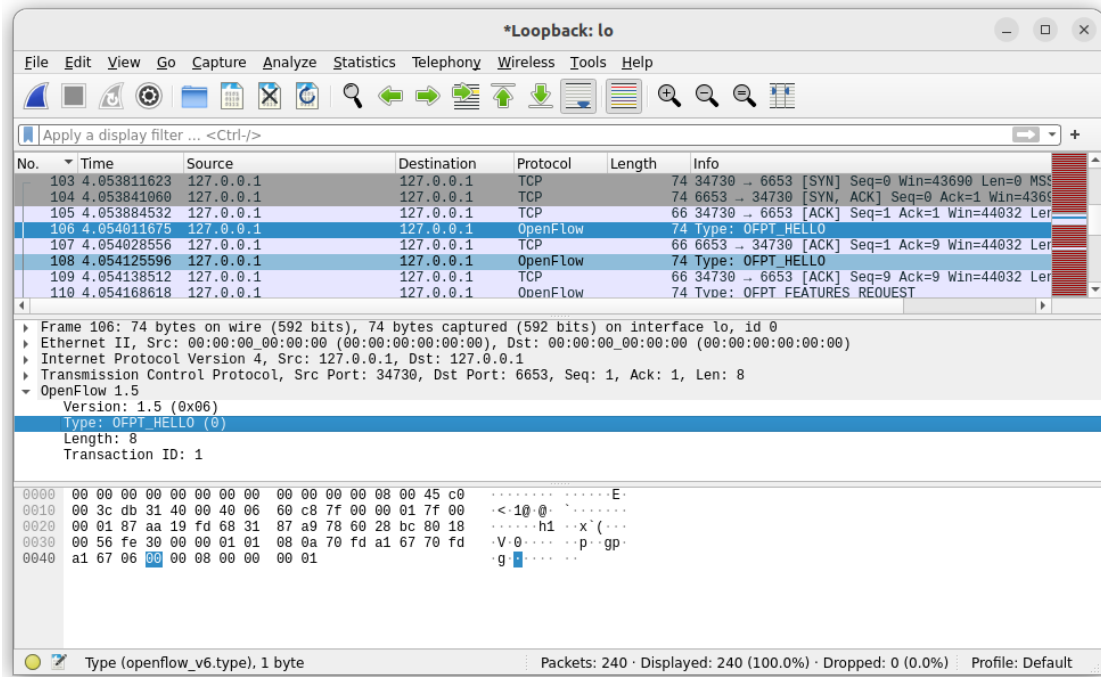


Mokhtari (9831143) – Computer Networks 2 – PRJ 02

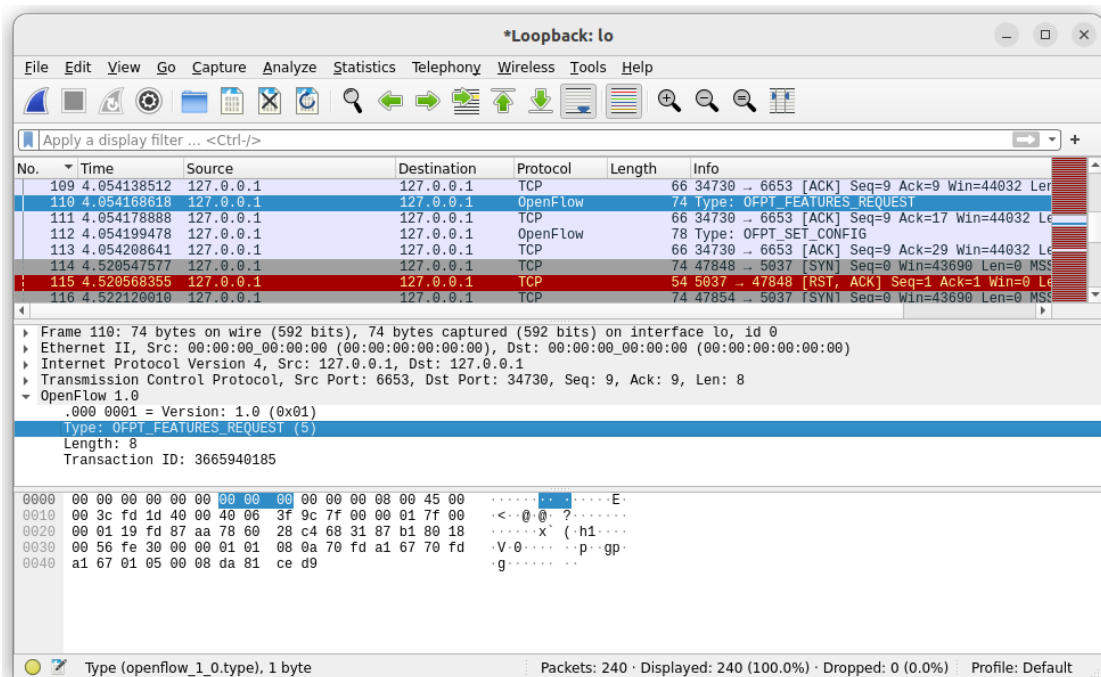
P1:

1.1. As shown in the screenshots below, messages have only used the TCP and OpenFlow protocols, and we know that OpenFlow also uses TCP. To be more specific, this is the OpenFlow protocol.

1.2.



1.3. Initially, the controller sends a feature request message to the switch to obtain the Data path ID and port number. As a response, the switch sends a feature reply message that includes information such as the Data path ID, etc.



Mokhtari (9831143) – Computer Networks 2 – PRJ 02

The screenshot shows a Wireshark capture on the 'Loopback: lo' interface. The packet list displays several TCP and OpenFlow packets. Packet 134 is selected, showing an OpenFlow 1.0 OFPT_FEATURES_REPLY packet. The packet details pane shows the following information:

- Frame 134: 338 bytes on wire (2704 bits), 338 bytes captured (2704 bits) on interface lo, id 0
- Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)
- Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
- Transmission Control Protocol, Src Port: 34730, Dst Port: 6653, Seq: 9, Ack: 29, Len: 272
- OpenFlow 1.0
 - .000 0001 = Version: 1.0 (0x01)
 - Type: OFPT_FEATURES_REPLY (6)
 - Length: 272
 - Transaction ID: 3665940185
 - Datapath unique ID: 0x0000000000000001

The packet bytes pane shows the raw data in hexadecimal and ASCII. The status bar at the bottom indicates: Type (openflow_1_0.type), 1 byte | Packets: 240 · Displayed: 240 (100.0%) · Dropped: 0 (0.0%) · Profile: Default

1.4.

The screenshot shows a Wireshark capture on the 'Loopback: lo' interface. The packet list displays several TCP and OpenFlow packets. Packet 235 is selected, showing an OpenFlow 1.0 OFPT_PACKET_IN packet. The packet details pane shows the following information:

- Type: OFPT_PACKET_IN (10)
- Length: 88
- Transaction ID: 0
- Buffer Id: 0xffffffff
- Total length: 70
- In port: 1
- Reason: No matching flow (table-miss flow entry) (0)
- Pad: 00
- Ethernet II, Src: 3e:11:01:51:08:9d (3e:11:01:51:08:9d), Dst: IPv6mcast_02 (33:33:00:00:00:02)
- Internet Protocol Version 6, Src: fe80::3c11:1fff:fe51:89d, Dst: ff02::2

The packet bytes pane shows the raw data in hexadecimal and ASCII. The status bar at the bottom indicates: Type (openflow_1_0.type), 1 byte | Packets: 240 · Displayed: 240 (100.0%) · Dropped: 0 (0.0%) · Profile: Default

Mokhtari (9831143) – Computer Networks 2 – PRJ 02

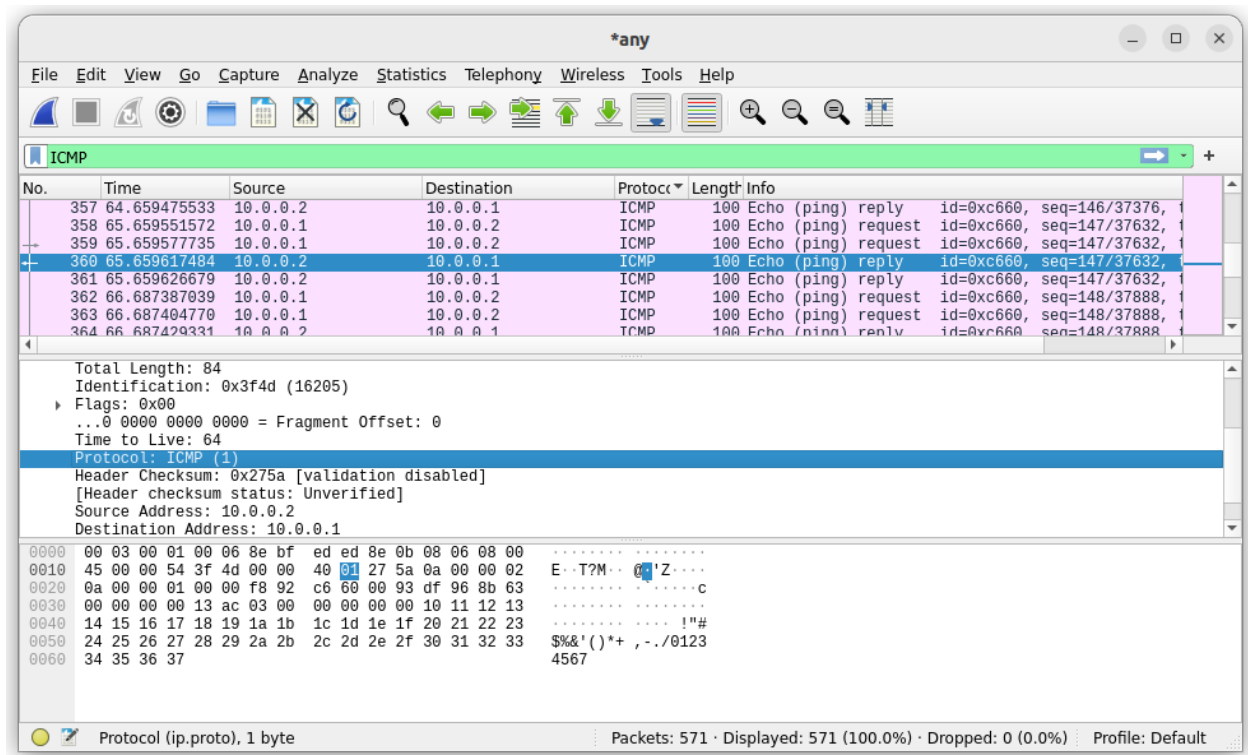
1.5. In OpenFlow, when a switch receives a packet on a port, it will try to match the packet to a flow entry in the switch's default flow table. If the switch cannot locate a flow that matches the packet, it will, by default, send it to the controller as a packet-in for closer inspection and processing.

In fact, PACKET_IN events occur when a flow isn't matched on the switch or when the TTL is not valid and the packet sent to the controller. As a result, no event is created; the switch forwards the packet according to its flow rules, and the controller is unaware.

In another case, this is because of the reverse connection, when the action belongs to the controller!

1.6. The ICMP protocol was used to accomplish this.

Host 1 sent ICMP packets under the OpenFlow protocol to host 2, and host 2 responded. Moreover, these packages indicate the time it took to send and receive the packages, as well as the validity of the connection.



P2:

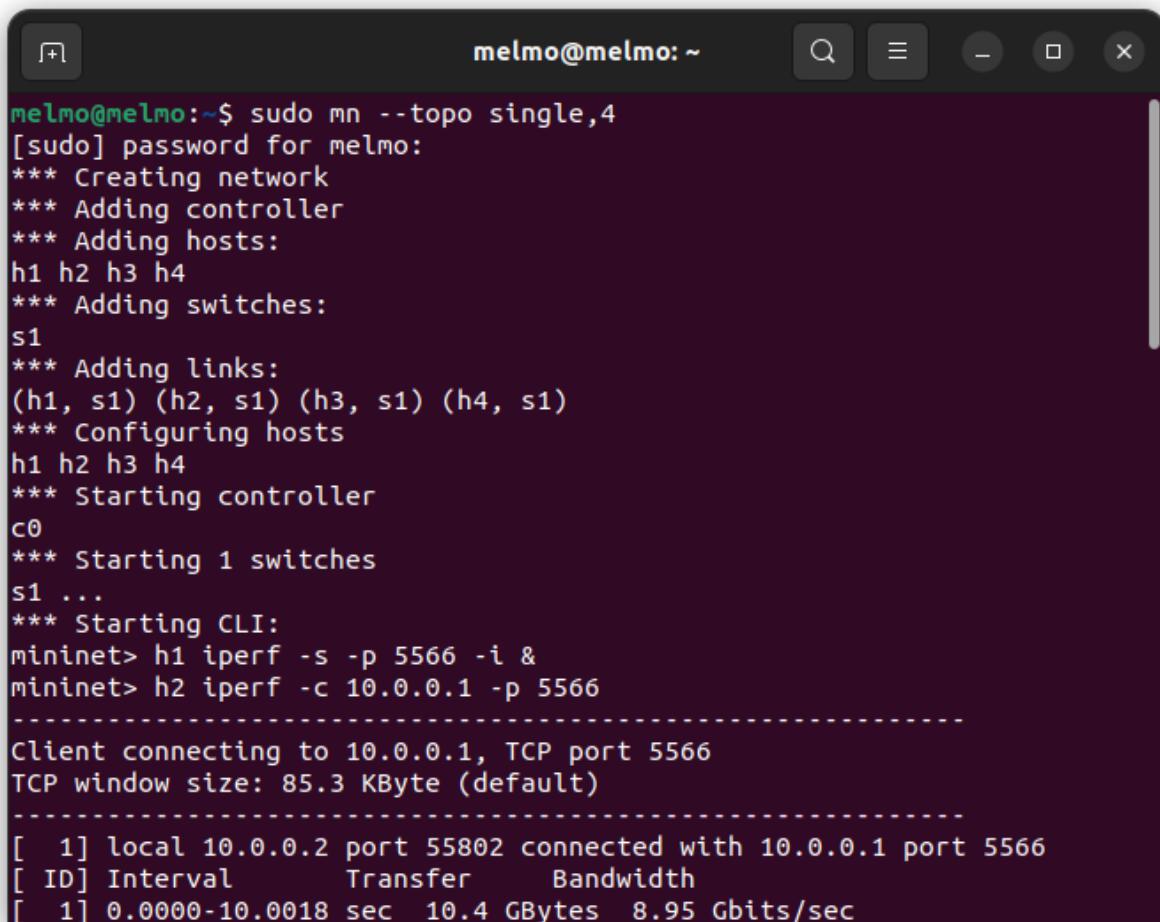
This command tests the bandwidth between hosts via a TCP or UDP connection (in my case, TCP).

My case involved checking the bandwidth between two hosts, h1 and h2, over a four-node single topology.

The "iperf node1 node2" command can be used to examine connections between two hosts, but I used the command in the screenshot below for better results.

Mokhtari (9831143) – Computer Networks 2 – PRJ 02

More specifically, the iperf command is a network performance test tool that can generate TCP and UDP data streams and measure the throughput of a network that is carrying them. IPerf actively measures the maximum achievable bandwidth on IP networks. It supports tuning various parameters related to timing, buffers, and protocols (TCP, UDP, and SCTP with IPv4 and IPv6). For each test, it reports the bandwidth, loss, and other parameters.



```
melmo@melmo: ~  
melmo@melmo:~$ sudo mn --topo single,4  
[sudo] password for melmo:  
*** Creating network  
*** Adding controller  
*** Adding hosts:  
h1 h2 h3 h4  
*** Adding switches:  
s1  
*** Adding links:  
(h1, s1) (h2, s1) (h3, s1) (h4, s1)  
*** Configuring hosts  
h1 h2 h3 h4  
*** Starting controller  
c0  
*** Starting 1 switches  
s1 ...  
*** Starting CLI:  
mininet> h1 iperf -s -p 5566 -i &  
mininet> h2 iperf -c 10.0.0.1 -p 5566  
-----  
Client connecting to 10.0.0.1, TCP port 5566  
TCP window size: 85.3 KByte (default)  
-----  
[ 1] local 10.0.0.2 port 55802 connected with 10.0.0.1 port 5566  
[ ID] Interval      Transfer    Bandwidth  
[ 1] 0.0000-10.0018 sec 10.4 GBytes 8.95 Gbits/sec
```