# Lab 3 SDN

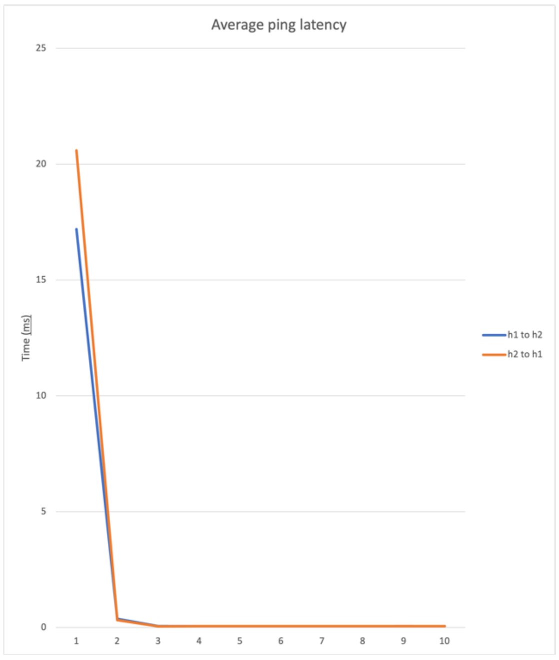
**Q1 - In a conventional network, how can a switch know where to forward an incoming packet? Please explain briefly the mechanism.**

In a conventional network, switches know where to forward an incoming packet by using a MAC address table, which is completed little by little when computers communicate. For example, a computer PC1 sends a frame received by the switch on port n.4, its MAC address appears in this frame, so the switch will know that PC1 is on port n.4.

**Q2 - Ping from one host to another. You should see that at the very beginning, the RTT is significantly longer. Why? Stop pinging and wait for about 30 seconds, and try again. Is the RTT now stable? Why?**

The RTT is much longer at the beginning, because the switch does not know where it should send the frame. So there is a "PacketIN" and "PacketOUT" event in our SDN. The controller sets up the uplink flow rule at the switch and forwards the packet over the corresponding outport of the switch.

**Q3 - Plot the Round Trip Time (RTT) evolution over time. You need to produce a plot with the runtime as the x-axis and the RTT time as the y-axis. You need two curves, the RTT from h1 to h2, and the RTT from h2 to h1O. ne measurement is not enough! You will need to restart both the network and the controller many times and average over multiple measures.**



**Q4 - Explain the roles of the different packets you see. You can use http: //flowgrammable.org/sdn/openflow/message-layer/ to understand them. We are using Openflow 1.3.Do not describe all the packets, but only the most important ones! Typically at the very beginning, once the switch contacts the controller, and when a new packet is received.**

Many different packets appear in OpenFlow. Among them we can mention the most important ones:

- OFPT_PACKET_IN: this is a way for a switch to send a packet to the controller for analysis
- OFPT_PACKET_OUT: this is a message containing a packet that the controller injects into the switch
- OFPT_ECHO_REQUEST: an echo request from the controller to the switch (can contain information like latency, bandwidth, or liveness)
- OFPT_ECHO_REPLY: an echo response from the switch to the controller

**Q5 - Explain what each flow represents. Briefly explain what each field is used for.**

These are all the flow entries in the switch table (which correspond to the OpenFlow 1.3 protocol). There are several fields:

- duration: the  time, in seconds, that the entry has been in the table
- table: flow table number
- n_packets: the number of packets that have matched the entry
- n_bytes: the total number of bytes from packets that have matched the entry
- idle_timeout: the time, in seconds, that the flow will expire after the given number of seconds of inactivity

**Q6 - Imagine now that a web server is running on h1. h2 starts an HTTP request to h1. What will happen? Describe briefly the messages exchanged between the switch and the controller, and the flows that would appear in the dump-flows result. Note: you can try to emulate the behavior by opening a terminal on h2 and by using the *wget* command and *python -m SimpleHTTPServer*.**

h2 sends a request to h1: this request is intercepted by the switch which makes a PacketIn to the controller, the controller makes a PacketOut to reinject the packet which will then be routed to h1. The same procedure will be done for the response from h1 to h2. We will then have two flow entries in the switch table.

**Q7 – How many switches are present? What kind of topology is it? How many switches must be programmed to ping from h1 to h8?**

It is a tree topology, there are 7 switches present. You must program 5 switches to ping from h1 to h8.

**Q8 - How does the first RTT evolve with the number of switches to cross?**

The more switches you have to go through, the longer the first RTT will take.