Option 1
Final Work – Detection of Elephant Flows

This final work consists of the implementation of a solution capable of detecting elephant flows in the network. Elephant flows are long flows that exceed a certain amount of packets. The solution must have the following characteristics:

- count the packets of each flow that passes through the switches. For the purposes of this work, consider a flow that has the same five TCP/IP header fields.
- define a threshold (length) that characterizes an elephant flow. When this threshold is reached, the switch must be able to create a header (elephant header) in the current packet to carry the switch ID where the elephant flow was detected. This packet with the new header must travel the rest of the route towards the destination. On the last switch of the route, the packet with the extra header must be cloned and sent to a sink node. At this point, the extra header should be removed and the original packet sent to the destination.

Note 1: Note that when the threshold is reached on the switch and the elephant header is created, the counter for that flow must be reset (or set to some value) to prevent the next packet of the same flow from detecting the elephant flow again.

Note 2: An elephant flow must be detected on only one switch. So, when the elephant header is present, that flow does not need to be counted in subsequent switches along the way.

Note 3: The sink node must be able to receive the cloned packets, extract the elephant header, show the TCP/IP header and the switch ID where the elephant flow was detected. You can also use some simple storage solution (time series database) of these data for further analysis.

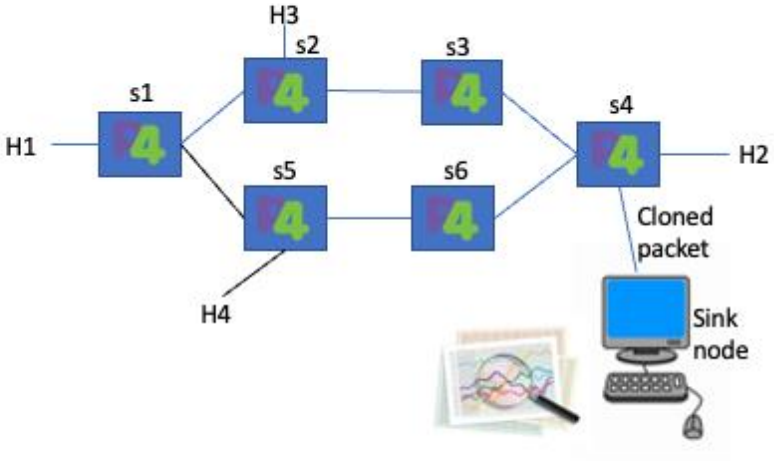
Note 4: Use registers and hash function to count elephant flows.

Note 5: Generate flows on hosts H1, H3 and H4 with H2 as destination.

Note 6: Probably the clone will have to be performed in the ingress and the elephant header removed from the original package in the egress.

Note 7: Any other design decision must be decided by the students.

The topology to be used is shown in the figure below.


H3
s2
s3
s1
s4
H1
H2
s5
s6
Cloned
packet
H4
Sink
node

Option 2
Final Work – Queue Monitoring

This final work consists of the implementation of a solution capable of monitoring the occupancy of the switch queues. The solution must have the following characteristics:
- count the number of packets that left each switch interface.
- collect information about queue occupancy on each switch.

The solution must generate monitoring packets at the source with a specific header to collect monitoring information. The header, called monitoring header, must contain the following fields:
• Switch ID;
• Queue depth at the time the packet was removed from the queue. To get this information, use `deq_qdept;`
• Number of packets sent by the outgoing interface.

The receiving node of the collection must extract the monitoring information and present it in an organized way so that the administrator can know the occupancy of the queue on each switch as well as the number of packets that passed through each output interface.

Note 1: Generate background traffic to occupy the queues so that the collection can obtain information. Generate traffic that uses the two routes available in the topology;

Note 2: Generate a collection packet every second;

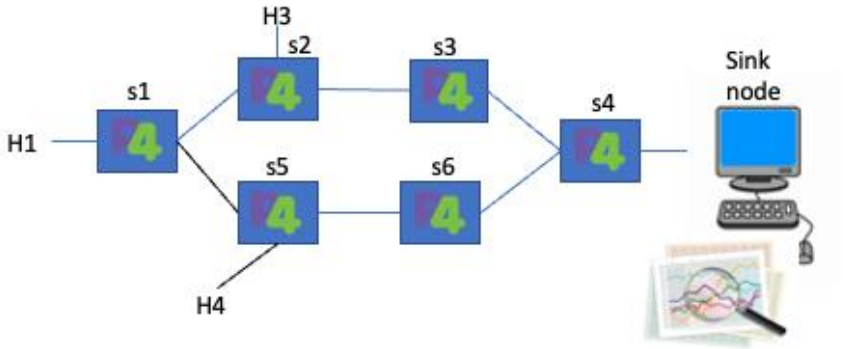
Note 3: Use counters to count the packets that pass on each outgoing interface;

Note 4: Queue occupancy is only available in egress.

Note 5: There is a code that can be used as inspiration here: https://github.com/leandrocalmeida/PoCSBRC2021/blob/main/code/int.p4. In this code, the student Leandro collected several monitoring information from switches.

Note 6: Any other design decision must be decided by the students.

The topology to be used is shown in the figure below.


H3
s2
s3
s1
s4
Sink
node
H1
s5
s6
H4