SDN MID TERM

1.


```
{
   "dpid": 1,
   "table_id": 5,
   "idle_timeout": 0,
   "hard_timeout": 0,
   "priority": 100,
   "match":{
      "eth_type": 2048,

"ipv4_src": "0.0.0.0/0",
         "ipv4_dst": "192.0.2.1",
         "ip_proto": 6,
         "tcp_dst": "0xffff"
      "ip_proto": 1
   },
   "actions":[
   ]
}
```


2.

The id given is a class c network 192.168.15.0/25 with a subnet mask of 255.255.255.128, which means that it has 126 usable IP addresses.

126/6 = 21

To create 6 equal-sized VPCs, each VPC would need to have 21 usable IP addresses, excluding broadcast ip and network id.

we can create a new subnet mask of /27, which give us 32 usable IP addresses in each vp. The new subnet mask would be 255.255.255.224

The following shows the network ID, subnet mask, broadcast IP and usable IP addresses range for each vpc 1 to 6:

VPC -1:

Network ID : 192.168.15.0

Broadcast IP: 192.168.15.31

Subnet Mask: 255.255.255.224

Usable IP Addresses: 192.168.15.1 - 192.168.15.30

VPC -2:

Network ID : 192.168.15.32

Broadcast IP: 192.168.15. 63

Subnet Mask: 255.255.255.224

Usable IP Addresses: 192.168.15.33- 192.168.15.62

VPC -3:

Network ID : 192.168.15.64

Broadcast IP: 192.168.15. 95

Subnet Mask: 255.255.255.224

Usable IP Addresses: 192.168.15.65 - 192.168.15.94

VPC -4:

Network ID : 192.168.15.96

Broadcast IP: 192.168.15.127

Subnet Mask: 255.255.255.224

Usable IP Addresses: 192.168.15.97- 192.168.15.126

VPC -5:

Network ID : 192.168.15.128

Broadcast IP: 192.168.15.159

Subnet Mask: 255.255.255.224

Usable IP Addresses: 192.168.15.129 - 192.168.15.158

VPC -6:

Network ID : 192.168.15.160

Broadcast IP: 192.168.15.191

Subnet Mask: 255.255.255.224

Usable IP Addresses: 192.168.15.161 - 192.168.15.190

3.

To reduce the number of flows in SDN switches for the client, we can:

Split flows into categories and direct each to a separate flow processor.

Change the flow processing algorithms to prioritize certain types of flows.

Use flow-based load balancing to control the p

4. 1-We can use multiple flow tables, an option would be to create separate flow tables for each category of the flow, and then configure the SDN switch to process the flow tables in the desired order. This would allow us to specify the order in which the flows are processed without modifying the priority field in the flows.

2-We can use flow-based load balancing to control the order in which flows are processed. This will involve dividing the flows into different categories, directing each category of flows to a separate flow processor or controller. The flow processors then can be configured to process flows in the desired order.

3-Modifying the flow processing algorithms to ensure that flows are processed in the desired order. For eg:, we can prioritize certain types of flows over others, or we can implement an queuing mechanism to hold flows until the necessary resources are available.

5. What sort of network topology will the following code build? What line should you add to make it a ring topology?

A: Code built for linear topology, to make it we have to add below code:

"self.addlink(s3,s1)" after the line self.addLink(s2, s3) in python code

6. No, SDN switch is designed to work in conjunction with a centralized controller that provides it with forwarding rules. The switch relies on these rules to determine the appropriate port where to forward packets. If the controller fails, the switch will not have access to the necessary information to make forwarding decisions based on the MAC address of the packets.

In the absence of a functional controller, the switch will not be able to handle packets and forward them to the appropriate port. So, in this scenario, traditional switching comes in place

7. Some commonly used symmetric messages are

Hello: This message is used to initiate a connection between the OpenFlow switch and the OpenFlow controller.

The Hello message is sent from the switch to the controller to initiate the connection and to provide information about the switch.

Echo Request: This message is used to test the connectivity between the OpenFlow switch and the OpenFlow controller. The Echo Request message is sent from the controller to the switch and the switch is reply back with an Echo Reply message.

Echo Reply: This message is sent by the OpenFlow switch in response to an Echo Request message, this message confirms the connectivity between the switch and the controller.

8.

1.The network should be governed by policies declared over high-level name – they should know the rules that are declared by policies and abide by rules

2.Network routing should be policy aware - Network policies dictate the nature of connectivity between communicating entities and, therefore, naturally affect the paths that packets take. This is in contrast to today's networks in which forwarding and filtering use different mechanisms rather than a single integrated approach

3. The network should enforce a strong binding between a packet and its origin - Addresses are dynamic and change frequently, and they are easily spoofed. The loose binding between users and their traffic is a constant target for attacks in enterprise networks. If the network is to be governed by a policy declared over high-level names (e.g., users and hosts),then packets should be identifiable, without doubt.

9.

Stress testing, performance testing, resource monitoring, load testing can be done as show the results to vendor and convince them to get beneficial for them.