(Introduction to OpenFlow Switch)

* Hardware-based OpenFlow Switches

- Commercial hardware switches with OpenFlow capability
- Show high processing speed.
- Have space limitation on saving the flow table entries.
  - Approximately store 1500 flow entries
- Not easy to upgrade.
  - Most switches only support OpenFlow ~~to~~ up to version 1.0

---

* Software-based OpenFlow Switches

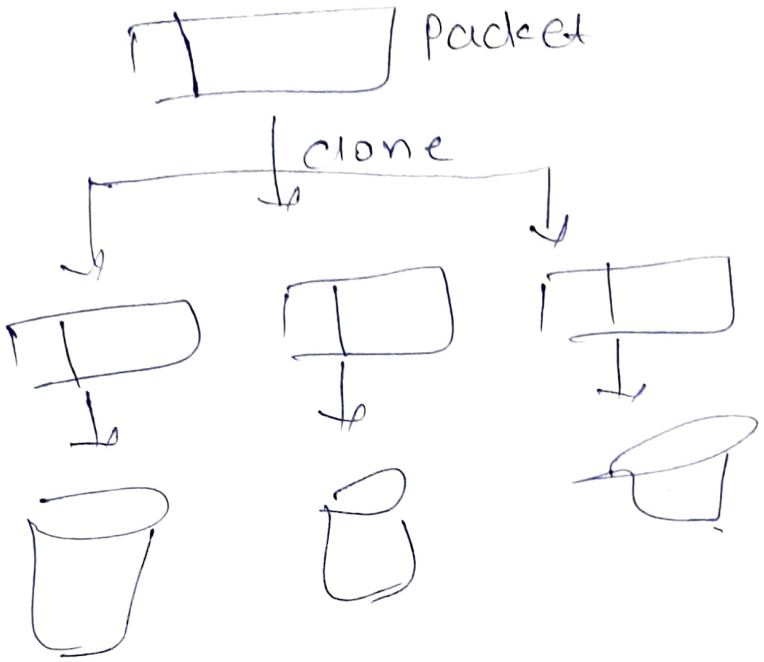
- OpenFlow enabled slow switch (runs on X86 commodity computer)
- Performance is relatively low
- Store large amount of flow entries with bound (theoretically)
- Under Active development, support most recent OpenFlow spec.

* Hybrid OpenFlow Switch.
  - A virtual switch with specialized h/w device.
  - Much faster than s/w-based switches.

---

* s/w-based OpenFlow Switches
  - OpenFlow Switch (OVS)
    → Overview
      - A virtual switch or virtual Ethernet Bridge (VEB)
      - used with hypervisors to interconnect to virtual machines within a host
      - and virtual machines b/w diff hosts across networks.

* Group Table. (3)



Group Type = all

→ Flow table forwards the packets towards the Group table.

→ Here we have 4 groups.

→ All the actions are performed on the group.

→ Here OpenFlow will enable a set of Ports.

→ Here 4 groups are multicasting, broadcasting, Unicasting and things like that.

→ Now, each group table will consist of no. of rows.

→ rows means entries. In this case group entries.

→ Inside the group table we have 4 components

① Identifier = Group Identifier = which will uniquely identify the groups.

- A group is defined as an entry in the group table.
- If we have 4 group then we identify them by group number like, group 1, group 2 - - - - .

② Group type = to determine group Semantics like every group no. one is working on Database, group no. 2 is working on networking, group no. 3 is working on web and things like that.

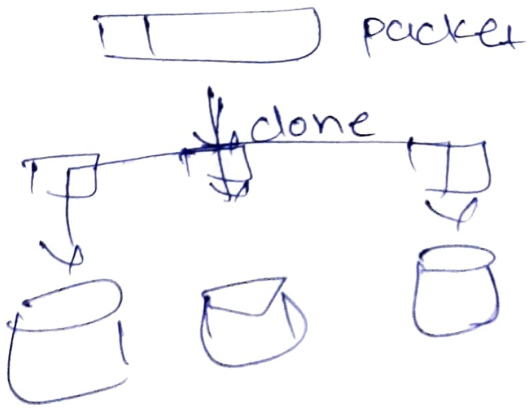
③ Counters = That needs to be updated when packets are processed by a group.

- Total no. of packets received
- Total no. of packets transmitted
- Total no. of packets dropped ... etc.

④ Action Buckets = It is an actually an order list of Action buckets where each Action bucket contains a set of Actions to ~~execute~~ execute.

- So each group includes a set of one or more action Buckets and each bucket will containing a list of Actions.
- An action list in bucket will execute when packet reaches a bucket.
- Action list is executed in a sequence.

(1)



→ first group type is "all".

→ It executes all the buckets in the group.

→ Each arriving packets is effectively cloned means each bucket will designate a different output.

→ If you have an incoming packet to the bucket, then it will be transmitted on multiple output ports.

→ In the terminology of networking this group is used for multicasting and broadcasting.

→ like you have a packet, you may have copies of that packet and you send them to all the buckets. Or in another words, you forward packets on all the output ports.

② Type = "select"

→ Random - select.

→ Executes one bucket in the group.

→ Here you have 3 buckets, you are executing only a single one in group and that decision is based on a switch computed selection algorithm, like which bucket you need to select. So, we need a algorithm for that.

→ And that could be based on several parameters. like, we need to share equal load, so who is actually with less load - you assign to that bucket.

→ load sharing is based on bucket sharing. And this is the responsibility of the SDN controller to decide which

packet will be put in which bucket.

(3) Type: failover.

- Executes the first live bucket. The buckets are evaluated in order and the first live bucket is selected.

→ like for eg. the port liveness is managed by SDN in this case.

→ If first bucket is not responding for eg, then what is next in the active list is selected.

(4) Type: Indirect.

- It allows multiple packet flows to point to a common group identifier. This type provides for more efficient mgmt by the controller.

- for eg, there are 100 flow entries that have same SA and DA. And the feild is match. But the source & dest. addl. are same, But may be

there are difference in some other feild.

- So, in this case all the packets based on the common feilds, they will be forwarded to a common port. And the action perform in this case is for same destination address.
- This is like Unicast communication.
- All the packets for same destination.

Controller - switch secure channel) 29th slide - (1)

→ This channel is b/w the controller and the OF switch.

- The secure channel is actually the path and this path is used for comm^n b/w the OF controller and the OF device.
- Generally this comm^n is secure as I just mentioned - It is secured by using TLS based asymmetric encryption.
- This connection may be in-band or out-of-band - These are the 2 variants of the secure channel.
- In the out-of-band eg:- It is like the secure channel connection, you can see here it will enters in the switch by using a dedicated port and that port is represented by 2, here.
- This is not switched by the OF Data plane, some legacy network stake will deliver the OF messages via the secure channel to the Data plane.

where all the OF messages are passed and handled. so the **out-of-band** secure channel is only relavent in case if we are using OF hybrid switch.

- while normally we will be using or SDN will use the **In-band**. in this eg: you can see the OF messages from the controller coming by using the port and that **port is k** and this port is actually a part of the OF data plane.

- so now we know the main difference b/w port z and port k. This port k is actually part of the switch. while port z is not part of this. **Port z** is used when we are using the hybrid switch.

- while **Port k** is used when we are using the OF switch.

- All the packets will be handle by the OF packet matching function as shown here.

here we are using TLS for security, And to put TLS in practice one must obtain and configure security certificates for each devices which can be time Consuming and also error prone misconfiguration can lead to other vulnerabilities.

- So if you are not familiar with such stuff, it is also recommended not to use the TLS option.

* OF Protocol.

8th slide.

→ You all are familiar with the word protocol, the syntax and format and semantics, ~~of~~ the language of communication → so in this regard, the OF protocol actually describe the messages that exchanges, that take place b/w the OF controller and OF switch.

→ And this protocol is implemented on top of TLS ~~and in the previous~~

→ It is providing a secure OF channel.

→ So, this is the basic concept of the OF protocol. It defines the communication, the messages that exchange b/w the controller and the switch over a secure media.

→ ~~OF protocol defines the communication~~

→ this protocol is uniquely identifies the OF technology.

→ The protocol will consist of a set of messages and those messages are sent from the controller to the switch and corresponding set of messages

that are sent in the opposite direction. ③

42)

- The message collectively allow the controller to program this switch.
- So as to allow fine grain control over the switching of the traffic, the user traffic - How to route traffic from an input port to an output port.
- The most basic programming ~~are~~ is defines modifies and delete these flows.

→ What is Flow?

→ We know what is Flow, is a set of packet. They are transferred from one network end point to another end point.

→ Now these end points as we know it may be like ip they may be defined as the ip addresses, port number, TCP/UDP port or may be virtual lane endpoints. So it could be anything.

→ Now there are certain rules they describe the forwarding action, that the devices sould take ~~on~~ on the packets - That actually belong to the same flow.

hen the controller defines a flow it is providing the switch with the information it need to know.

- how to treat an incoming packet, it will match that flow. for eg!- we have a Flow that is dedicated for a specific and device_that is connected to a Vlan, So if a new packet is arrived for that, the device must know what to do with that specific packet because it belongs to a Flow.

- The most basic prescription for treatment of an incoming packet as we just saw they are denoted by (A), (B) and (C).

- These 3 options are to forward the packet out to one or more output ports, to drop the packet or pass the packet as in case of (C) to the controller for exception handling. Or may be for further guidance from the controller.

- Now, the OF controller is evolved significantly with each version of the OF.

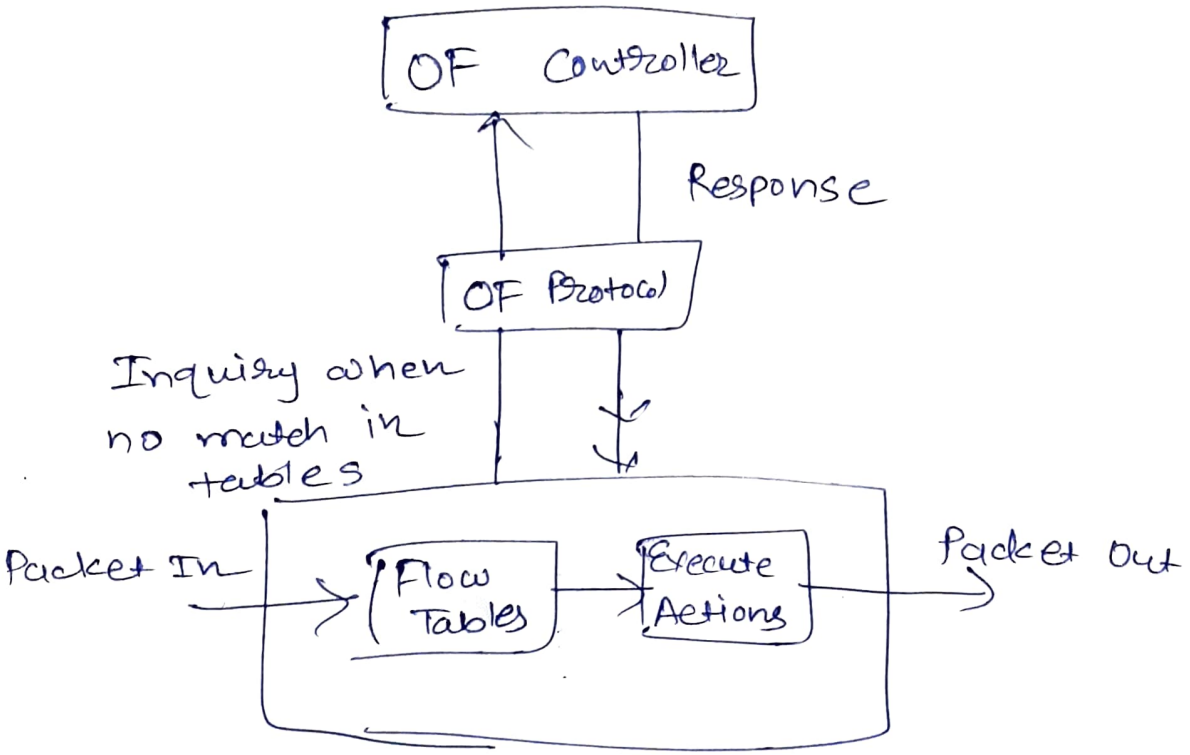
We know, there are different versions so in every version, the role of the OF, the duites, the responsibilities they are modified.

- There are certain limitations in the early version of OF and that limitations are fullfiled in the new versions of OF.

7th slide

* OpenFlow Controller.

28th slide



- Modern Internet switches, they actually makes thousands and even millions of decision per second and what are those decisions?, they are very simple decisions, whether to forward an incoming packet or to drop an incoming packet.
- If we need to forward what set of output port it should be forwarded and what header fields in the packet may need to be modified, added or removed.
- Now things are becoming complex for the switches because it's not just

forwarding, it's like taking some intelligent (2) decisions - ~~spreading the packet~~ headers, the ~~packet fields~~

- So the OF ~~Controller~~ plane actually its responsibilities are different from the traditional switches or the traditional control plane.

- It is differ in three ways,

→ We know that it can program different data plane element with a common standard language - that is what we have studied in SDN and that is what is known as the OF, which is like we want to program the data plane elements, the switches, the h/w with the common standard language.

→ Second difference b/w an OF controller and a traditional switch controller is that the OF controller it exists on a separate h/w devices then ~~as~~ that of the forwarding plane like we separate the two worlds the data plane and the ~~forwarding~~ controlling

plane. So actually it exists on a separate h/w device than that of the forwarding plane.

→ Unlike traditional switches where the control plane and the data plane are on the same physical device, this separation is made possible because the controller can program the data plane elements remotely over the internet.

→ And the third difference is that the controller can program multiple data plane elements from a single control plane instance

→ The OF controller is now responsible for programming all the packets.

→ It is actually responsible for matching logic. It is also responsible for work for forwarding rules definition of the forwarding rules in the switch. The rules are followed by the switch but they are defined by the controller.

→In case of a traditional router, the router would run routing algorithms to determine how to program its forwarding table. Then that function or an equivalent replacement to it is now performed by the controller.

→ The logic is the same, we need forwarding rules, we need forwarding tables – but now we are programming them and we are controlling them from the controller.

→ And here we can see, at bottom we have the switches, which have the logic that we just saw. The Flow Tables and Actions performed by the switch and we have a secure communication channel.

→ And we have the bi-directional communications b/w the switch and the controller. And that is over a secure medium.

→ So we have like an enquiry and then a response from the controller – what to do with the packet ↔ Straigh forward → forward it

on an output port as we just saw previously or send it back to the logical part and make another decision.

→ So ~~this~~ that is actually a controller.

*OpenFlow Protocol

→ Overview of OpenFlow

- OpenFlow defines both the communications protocol b/w the SDN data plane and the SDN control plane, as well as part of the behavior of the data plane.
- General purpose protocol for programming the forwarding plane of SDN switches.

6th slide.

OF.org in 2008

↳ Their purpose was to promote and support OF.

5th slide (27th slide)

OF switch.

→ this shows the relationship b/w OF switch and OF controller.

→ We have switch and OF controller and they both communicate with one an another by using OF protocol.

→ Here we are going to focus on switch. And this is actually a core function of this switch is to take packets. and the packets actually arrived at one of the port, that are

mentioned here -- like port 1, port 2, port 3, port 4 and so on.

- Switch has different ports, packets arrived on those ports.
- In this particular case you can see, path X, that is on port 2, once this switch receives the packet on any of the port, it will forward it to another port, in this case port N.
- So, the basic responsibility is, forwarding a packet. It receives a packet on one port X and it forward the packet on another port for eg. port number n.
- However before forwarding the packet it is possible, in some cases it is necessary to do some modification to the packets along the journey, along the way.
- A unique aspect of OF switch is embodied in the packet matching function, As we can see here, we have packet matching function. and we do have a table that is known as the Flow Table and you all are familiar

ith the Flow Table. So we have Flow
table, we have a matching function.

- Now, you can see, there are different lines over here - some are single root and some are double a root. (↕)

- So the white grey double side arrow line here it starts in decision logic.

- It shows a match with a particular entry in the table and directs the new matched packet to an action box here on the right side.

- This action box has three fundamented options to deal with the arrive packet.

like → you arrive a packet at port before forwarding that packet - we need to take certain modifications to the packet if required. So the packet is handed over to the matching function and then the flow table and based on this entries of the flow table, certain actions are taken on the packet.

- Basically three actions needs to be taken.

Action - A :→ forward the packet out a local port, Possibly modifying certain headers but in most of the cases Just forward the packet → this is the simple case.

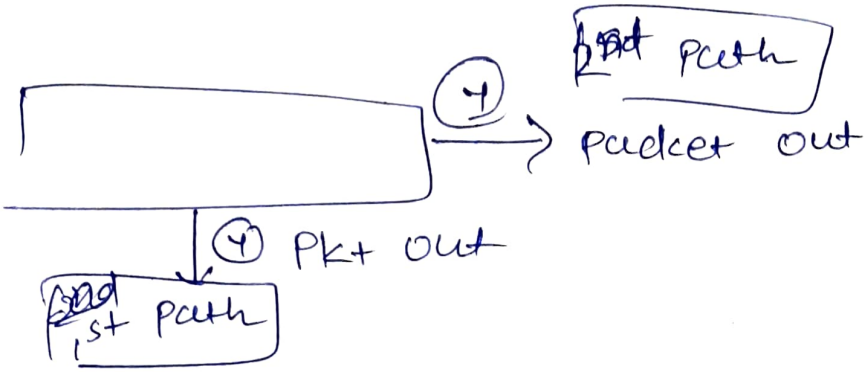
Action - B :→ Drop the packet, if the packet doesn't meet any criteria in the table, then we can drop the packet although we can send it to the controller but we will see that in Action - C

Action - C :→ Pass the packet to the controller. Packet is pass to the controller over a secure channel.

- A controller has either a control message or a data packet to give to the switch, the controller also uses the same secure channel

in the reverse direction.

- When controller has a packet to forward out through the switch, it uses the OF message and that particular message used by the controller is known as **packet out**.
- On the other side, if switch wants to communicate, it will use a packet and that packet is known is **packet in**.
- So, we can see in this fig, that such a data packet which is coming from the controller may take two different paths.



- The first path is OF logic, both are denoted by (4) but the 1st one is the logic path.
- In the right most case the controller directly specify the output port and the packet is pass to that **Port N** in this