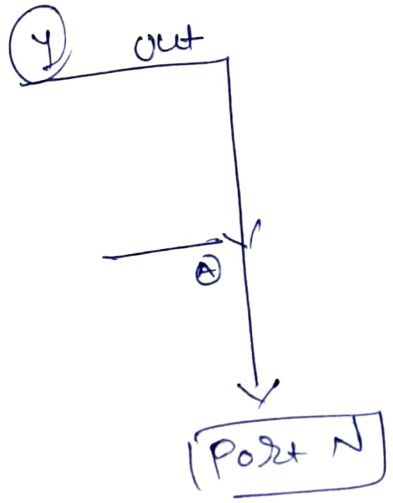
example.
like there are 2 cases. The first case is just to directly route the packet to an output port. or just to send the packet to the matching function.



- And in the left most ~~port~~ path (logical path) the controller indicate that it wishes to differ the forwarding decision to the packet matching logic.
- this given open flow switch implementation is easier. If you have this switch and you want to implement this logic, that is simple.
- There are 2 possibilities available, the first one is OF only and the second one is OF hybrid approach.

- An OF only switch is one - that will forward packets only according to the OF logic described here while the hybrid approach is a switch that can also switch packets in its legacy mode for eg:- the ethernet switches or ip router, they can also handle those packets.
- one can view the hybrid case is an OF switch that will actually reside next to a completely independent traditional switch. However such hybrid switch will require pre-processing classification mechanism - that directs the packet to either the OF processing or the traditional processing.
- So this was the basic functionality of the OF swith.

In this article, we will get to know the basics of OpenFlow protocol, which is the standard protocol in traditional software-defined networks. This article will go through the basics of Software Defined Networking (SDN) and OpenFlow (OF), the working of OF protocol and some terminologies, advantages over traditional networking architecture, and alternatives to OpenFlow be used in SDN.

What is the OpenFlow protocol?

The **OpenFlow (OF) protocol** is a standard in software-defined networking (SDN) architecture. This protocol defines the communication between an SDN controller and the network device/agent.

Basics of OpenFlow

As mentioned, the OpenFlow protocol lays out the foundation for communication between an SDN controller and a dumb network device. This protocol was developed first by researchers at Stanford University in 2008 and was first adopted by Google in their backbone network in 2011-2012. It is managed now by the Open Networking Foundation (ONF). The latest version used in the industry is V1.5.

OpenFlow is the standard southbound protocol used between the SDN controller and the switch. The SDN controller takes the information from the applications and converts them into flow entries, which are fed to the switch via OF. It can also be used for monitoring switch and port statistics in network management.

- The OpenFlow protocol is only established between a controller and the switch. It does not affect the rest of the network. If a packet capture were to be taken between two switches in a network, both connected to the controller via another port, the packet capture would not reveal any OF messages between the switches. It is strictly for the use between a switch and the controller. The rest of the network is not affected.

Initiation of OpenFlow channel

- OpenFlow protocol works on the TCP protocol. The standard protocol is TCP 6633 for OF V1.0 and 6653 for OF V1.3+. There needs to be IP connectivity between the controller and the switches to establish an OF connection. OF channel is formed only after a successful TCP 3-way handshake.

The switch sends a "HELLO" packet to introduce it to the controller to start the OF channel communication. The switch also sends information like the highest version of OF it supports. The controller replies to the hello message with its highest supported OF version. Then, the switch negotiates on the highest level of the OpenFlow version that they both supports.

Once the version is negotiated, the controller sends a "FEATURE_REQUEST" message. This message essentially asks the switch for its supported OF capabilities like the number of flow tables supported, supported actions, etc. The switch replies to it with a "FEATURE_REPLY" message stating all its capabilities along with its unique identifier or Datapath ID (DPID).

After this, it is said that the OpenFlow channel is successfully established between the switch and the controller. The connection between the controller and switch is essential as it is the only way for a switch to communicate with a controller.

To secure this connection, a protocol like TLS can also be used instead of a TCP connection. Here, the controller and switch need to have the proper certificates and keys for a

successful TLS connection. This prevents snooping on the OF channel.

OpenFlow tables and Flow entries

Flow tables are like a traditional switch's MAC/CAM table that stores the hosts' hardware addresses. Flow tables store flow entries or flows that tell the SDN switch what to do with a packet when it comes to an incoming port.

The switch will match specific parameters like IP address, port number, MAC address, VLAN ID, etc. and select the best matching flow entry from the table and execute the action associated with that entry. Actions could be to drop the packet, forward it out a different port, flood the packet, or send it to the controller to further inspect it.

If a switch does not have an entry for a packet, the switch might have a default entry or "TABLE_MISS" entry. This entry has the lowest priority, and the actions can either be to drop the packet or send it to the controller.

When the controller receives this kind of packet from a switch, it sends it to the application running at the application layer, which processes the packet and let the controller know if a new flow entry needs to be inserted in the switch's flow table. If that's the case, the controller will insert a flow entry on the switch.

The next packet of the same kind will be dealt with by the switch at the Data layer as it already has an entry, and appropriate actions would be taken. This improves the efficiency of the network by a huge factor.

Advantages of using OpenFlow

There are several advantages of OpenFlow and SDN rather than traditional networks:

- SDN enables separation of control and data plane, which means switches can use all their hardware resources in just forwarding data instead of computing routes.
- OpenFlow provides an easy way of communication between controller and switch, easily implemented in an existing network.
- Most current devices support OpenFlow, it is not enabled by default, but we can easily enable and use it to transition to SDN.
- It provides security with a TLS connection to prevent snooping and DoS attacks on the controller and/or the network.

- OpenFlow does **NOT** change the configuration for a switch. It just updates the flow tables, which define the path for a packet.

Alternatives and other means to SDN

Although it seems that OpenFlow is ingrained with SDN, it is just an industry-standard practice to implement SDN in an organization efficiently. There are several other means to program the SDN switches and insert flow entries in its flow tables.

- **REST APIs:** REST APIs are the most popular choice after OpenFlow to communicate with the switches. All the controller applications also have a REST module supported with them. The user can query the switch database to POST or GET data. "Curl" is one of the most popular tools for this.

An example of a REST call on the RYU controller to get all the switches in your network would be:

```
curl -XGET http://localhost:8080/stats/switches
```

- **Management protocols:** Protocols like SSH, NETCONF, and SNMP could also manage an OF enabled switch from the controller. This method is

generally used for old devices that do not support OpenFlow.

35th slide ①

→ This component is related to protocol.

→ the different type of messages that are defined.

→ The messaging b/w the controller and switch is transmitted over a secure channel.

→ We know this secure channel is implemented via the TLS over TCP.

- If the switch knows the IP address of the controller, then the switch will initiate this connection.
- Each message between the controller and the switch starts with the OF header.
- This header specifies the OF version number, the type of the message, the length of the message, the transaction id of the message and other info.
- So the various message type in version 1.0, they are listed here in this table.
- The messages fall in 3 general categories.

symmetric, and we have the controller switch and asynchronous.

- But these are only the most important types of messages in the normal context.
- We do have other types of messages that are not mentioned over here.
- But just for overview all the messages are being defined.
- And how actually the communication take place is also shown here (36th slide)

→ Message type

→ Hello

↳ hello messages are exchanged after the secure channel has been established to determine the highest OF version number supported by the peers.

→ ECHO

- Echo_request and reply, they are used by either side during the life of the channel to assertion that the connection

still alive and to measure the current bw.

→ Similarly we have vendor message and they are available for vendor specific experimentation or maybe enhancement.

→ Packet_In
→ Packet_Out
} they are important and we have seen this.

*) Packet_In → how the switch, this is a message generated by the switch. how the switch passes data packet back to the controller for exception handling. And the control plane traffic will usually be sent back to the controller with the help of this message.

*) Packet_Out → which is used by the controller to send data to the switch.

→ Flow_removed

Port_ Status – which is used to communicate changes in port status whether by direct user intervention or may be by a physical change in the communication medium itself.

→ Every message has a category ~~that~~ the cat

→ So the comm^n is blw like the controller and switch so, it is going to be like in the category of controller switch.

But sometime a message may be like for eg:- a packet_in and packet_out → they are also for the communication blw the switch and the controller but it is like asymmetric.

* Actions and Packet Forwarding 34th slide ①

3. After packet matching, we need to perform certain Actions,

- may be we need to forward the packet, to drop the packet and things like that.
- So the required actions, that must be supported by a Flow entry are either forward the packet or drop the packet.
- The most common case is that the output actions specify a physical port on which the packet should be forwarded.
- There are diff^n virtual ports defined in this version 1.0.
- They have special significance for the output action data that is performed.
- what are those virtual ports,

They are Local
All
Controller
In-Port and
Table.

- Here you can see them in bold and capital letters.
- These are actually the actions related to forwarding.

A packet may be forwarded to one of these and these are like the five virtual ports. Not the actual ports.

- The actual ports are Port 1, 2, 3, 4, --k--N.
- But these are like the virtual ports, just for our understanding.
- What does LOCAL mean?

→ local dictates that the packet should be forwarded to the switch.

→ local control slw. We have OF switch side controller, The actual controller is OF controller but inside the switch we have local OF switch controller.

→ So the local mean that, the packet has to be forwarded to this a local OF Controller slw.

→ And this local is used when OF messages from the controller are received.

→ You received a message from the controller not from the ports. then the local is used.

And these are the messages from the Controller are received on a port, there is receiving packets switched by the OF datea plane.

- this local actually indicate the packet need to be processed by the local OF control slw.

→ All

- All as the name indicates, is used to flood a packet out on all ports on the switch except from the input port on which it is received.

- this provid like the broadcast capability of the OF switch.
- If you want to flood the packet, if you want to send a packet, broadcast a packet, all the available output ports except from the one on which it is arrived.

→ Controller

- this indicates that the switch should forward this packet to the controller, the actual controller.

not the local one.

→ IN-PORT

- Instruct the switch to forward the packet back out on the port on which it arrived.
- You received a packet from Port 2 and then you are forwarding the packet back on the same port on which it was arrived.
- IN-PORT normally creats a loop back situation, which could be used for certain scenarios but could be avoided in certain scenarios as well.

→ TABLE

- Which only applies to packets that the controller sends to the switch.
- Such packet arrived is part of the packet ~~out message.~~ out message.
- When the controller send a message to the switch, then the message is the packet out message and it arrived from the controller and that we use the table.

And it include actually an action list. and this action list will generally contain an output action which will specify for eg:- a port number which on the packet should be sent.

- The controller may wish to directly specify the o/p port for this dated packet or if it wishes the output port to be determined by the normal of packet processing pipeline.

- ~~it may~~

⇒ And these 2 options are depicted by the (4) arrow. as we saw in previous slides.

⇒ Now there are 2 additional virtual ports and they are normal and Flood.

- these supports version 1.0
- 1st one is normal virtual port, when the output action forwards the packet to the Normal virtual port, it send the packet to the legacy component of hybrid switch.

The use of this Normal, only makes sense again in the case of a hybrid switch.

→ Flood.

- In this case the switch will send a copy of packet out all ports except against on the ingress port from which it was received.

* Packet Matching

- In this we will show how packets are matched with the flow.
- when a packet arrives at the OF switch from an input port or maybe in some cases the packet is arrived from a controller.
- Now the packet that it received or the packet received by the OF switch is matched against the FT to determine if there is any matching flow entry there in the table.
- The following match fields, that are actually associated with the incoming packet and these fields may be used for matching against the Flow entries.
- These are not all of the fields but some of the fields that are used for the matching purposes and these are like switching, input port, VLAN ID, and the priority for the VLAN and ip addresses, frame type, destination, source addresses, Destination add, ethernet SA, DA, Port no, ip protocol used and etc.

So, these are different criterias that we can use for matching packet to a flow.

→ ~~before 12 diffn~~

→ Actually, 12 diffn rules are defined for packet matching in the OF.

→ And these are referred to basic tuple of match field

→ The flow entries are processed in order. If you receive a packet, you have to compare it with the entries and then the comparison is made in order.

→ once a match is found, no further match cuttempts are made against the flow table.

→ There may be additional Flow tables against which packet matching may continue. But we stopped once we found the first match because of this it is possible ~~for their~~ that we have multiple matching flows entries for a packet to be present in a flow table but only the first flow entry to match is meaningful. The other will not be

found as packet matching. So we stop upon the first match. ②

- ONF has defines 3 types of conformance in version 1.0
- What are those three levels, The three levels are full conformance and layer 2 conformance and layer 3 conformance.
- full conformance meaning all 12 matching fields are supported.
- layer 2 conformance meaning only layer 2 header fields matching is supported.
- layer 3 conformance - only the layer 3 header fields matching is supported.
- so if the end of the flow table is reached, it is also possible we reach to the end, without finding a match, such situation is known as table miss.
- In this event of a table miss, the packet is forwarded to the controller, noting that this is no longer strictly true in the later versions.

If a matching flow entry is not found in the later versions, the action associated with the flow entry determine how the packet is handled.

- And mostly the OF switch entries is forwarding → is forword this packet may be drop this packet.
- But in this 1.0, the packet is forwarded to the controller.