انت على طول كنت بتقول راوترز و هي ممكن تكون حاجات تانية

Abstract—In a line with the noteworthy growth of the networks (e.g., networks properties and features,) the design of the network become more complex and maybe we cannot achieve these properties and features with the traditional architecture of the networks (e.g., architecture of dedicated monolithic routers and switches that implement both data and control plane.) In some cases, we need to take more complex decisions for routing process and security and managing the network flow and prevent some of the undesirable issues, with relying on the old architecture of the networks we can’t make these decisions. The way we can go through to make these properties of the network easy to achieve is making a change in the ordinary network architecture, making it \_the architecture\_ easier to be configured and be modified and by the way enhance its performance. Making use of the SDN (software defined network) is a great way to get benefit of the desired features of the expected and desired network architecture. SDN is a very rising architecture in today’s networks that is been adapted by a very influential technical institution (e.g., Google that uses SDN to manage the data center from inside and between data centers from outside).

**What is SDN (software defined network)**

SDN is the separation of the control plane and data plane. Control plane (i.e., the area where the routing protocols is performed) functions is performed by the SDN controller and network-control applications network, and the data plane (i.e., the area where the forwarding decision is performed) function is performed by the network routers and switches. This separation makes the process of modifying the network state and protocols done in an easier way than the old architecture (with monolithic data and control plane.) Also, we should mention that this separation allows more security options (i.e., firewall) that we can now achieve by SDN, simply by defining which packets from which sources will be dropped and another function this article will cover like traffic control.

SDN forwarding function is done in a different way. As we learned from the traditional routing algorithms, the packet forwarding decision depends only on the destination IP address (i.e., 3rd layer address). In SDN forwarding scheme, the forwarding decisions depends on many fields not only the IP address, and these fields not only from the IP layer (i.e., but it can also be from data link layer and/or transport layer.) fields from 2nd later like source and destination addresses and from 4th later like source and destination port. And this forwarding decision is done using protocols like OpenFlow protocol that depend on the “match-plus-action” scheme that will be discussed.

**SDN and traditional network**

The main difference between the SDN and monolithic data and control plane architecture is that the SDN dedicate the control plane function to some hardware devices (i.e., SDN controller that deliver configuration data to the underlying routers, and the network-control applications which perform the routing algorithms and network state updates and packet flow decisions.) leaving the only function performed in the network router and switches is the data plane functions which is not the case in the other architecture which the two planes are merged and done in the routers and switches.

Traditional architecture

Control

plan

data

plan

Control

plan

Data

plan

Control

plan

Data

plan

Device 1

Device 2

Device 3

Data plane

Control plane

SDN

Controller

Device

2

Device

1

Device

3

App2

App1

App3

Network-control applications

SDN architecture

Northbound APIs

Southbound APIs

**SDN architecture**

This discussion will be divided into two parts as the SDN consists of two main parts (i.e., the data and control plane.) First, we will talk about control plane and control plane components and the functionality of each component as a part of interactive cooperative system.

Control plane is divided into two parts, the SDN controller and the network-control application:

* Network-control application: the part of the control plane that is responsible for the routing function. These applications take the network information (e.g., the links states and the nodes states form the SDN controller and perform the defined routing algorithm (e.g., Dijkstra algorithm) to determine the shortest path also it is triggered by the SDN controller if there is any change in the network state, in this case the applications make the decision depending on the changes in the network state and pass the appropriate decision to the SDN-controlled routers. We will notify that the SDN controller does not do any routing processes, it is the job of the applications and the SDN controller transfer this routing information to the SDN-controlled routers. Other function the is performed in the applications is the control access (i.e., firewall) that provide the network with some security by defining which packets will be forwards and which will be dropped, and this decision can be performed depending on several information (e.g., the source IP address, and this mean not to receive any data from this host.) The IP address is not the only way to control the access, it can be controlled using the TCP/UDP source and/or destination ports to prevent any date transmissions between hosts and a specific port in transport layer, we will discuss this in more detail when we talk about OpenFlow protocol. Other function done in network-control applications is the load balancing of the network by defining the action taken by matching in the flow table (i.e., tables used by the SDN-controlled routers to determine the action taken by the matching of an incoming data header with an entry of the table) in the SDN-controlled routers. It is done by destitute the routes to a single destination over multiple paths, and this service make use of the counter entry (i.e., entry in the flow table that determine the number of matchings that is done on a specific entry in the table) of an entry in the flow table. By considering the above graph, we can say the application1 may be the control access application and application2 is the routing and application3 is the traffic controller. Of course this not the only applications can be provided from the SDN architecture, many services can be added to the network and this is one of the most advantage for the SDN over the traditional architecture that we can easily and add and remove modules to enhance the network without making changes in all network parts, for example we do not need to make any change in the routers configuration if we change the routing algorithm used or add a new service to the network, all we do is make the new changes in the control plane (i.e., the network-control application) add pass this change information and parameters the routers of the network through the SDN controller. Changing the routers configuration is not simple, any change in the network that need special router configuration to each router is very complicated problem and it becomes harder if the router is manufactured in different places. So, it is reasonable to make a radical change in the old network architecture, and of course SDN provide an accepted solution to this issue of adding new features and overcome this constrains.
* SDN controller: As comes to mind that the controller plays the role of a link between the controlled-application network and the SDN controlled devices (i.e., The SDN controller defines the data flows between the centralized control plane and the data planes on individual routers and switches.) The controller does the complex functions in the network (with the help of the controlled applications); the controller receives different messages from the SDN-controlled devices (using southbound APIs) and process these messages which may be a message to notify the controller that there is a failed router in the network, or in other scenario may be a message carrying an information of a change in a link in the network. The controller deals with the incoming messages be determining which type is this message and who sent it and by the way the SDN controller can gather the information that will help in determining to which application that the parameters taken from the messages should be passed (using northbound APIs). When the controller receives the message and determining its type, it extract an information from this message that help the application module to handle the issue concerned with this message. And in the other direction, the SDN controller receive the results of the processing done int the application and populate this information in the devices .When we talk about SDN abstraction, we must mention the role that the controller plays to take off the responsibility of constructing the routing table (i.e., flow table) from the SDN-controlled routers. The SDN-controlled routers only are concerned with the execution of the forwarding process depending on the flow table that the controller deliver to it after it has been computed by the routing application (i.e., The controller populates flow tables.) We must mention here that it can be seen that the SDN controller is only one device that control all this functions in the SDN network, but in fact the SDN controller is a combination of many devices that are seen as only one device (i.e., SDN controller is logically centralized but physically distributed.) And this distribution come with many advantages (e.g., distribute the incoming message from either application or the devices over many servers and fault tolerance.) These advantages reduce the delay time in the connection between the application and the devices and make that connection more reliable.

The second part in the SDN architecture is the data plane and it consists of the SDN-control devices that deal with the observed network operation. After talking about the controller and application, we can conclude thar these devices do not make any type of processing in the network, all its function is to deliver the network state information to the controller and receive the changes and the commands from it and perform actions depending on this information. And this action is not simple as regular routers in normal networks, it is special actions. Routers in networks that perform longest prefix matching have one action to perform in case of the incoming packet matched an entry in the routing table, and this action is simply FORWARD the packet to the defined (by routing algorithm) interface in the table. In SDN, routers perform different forwarding that implies many actions not only forward action, and this type of forwarding is known as match plus action forwarding. In match plus action forwarding, the action taken for the packet does not depend only on the IP destination address, it is more flexible and offer many types of matching that can be done on the received packet. These matching processes can be done on the fields in different layers as mentioned. One scenario that can be taken into consideration to make the match plus action clear is: suppose a switch that implies the match plus action forwarding. One action that can be taken is to drop all packet the came from the source MAC address AA.AA.AA.AA.AA.AA and forward all packet that came from the source MAC address source BB.BB.BB.BB.BB.BB. The different action that is not implemented in the match and forward forwarding is the drop action, this action is one of the features that can be achieved when the SDN architecture is applied int a certain network. Suppose another example when a router in the SDN receives a packet with a certain IP source and destination addresses, the match plus action offers a service of passing the received packet to the SDN controller to perform another proccing on this packet and pass it again with the actions that will be performed with this packet, using this feature the router after matching the packet with the source and destination addresses from the flow table and see that the appropriate action to be taken with this packet will pass the packet to be processed, and this feature can be useful when the packet is complex to be processed in the SDN deceives (i.e., routers). Match plus action forwarding provides modification action can be taken with the incoming packet. Suppose that a router received a packet and matched with an entry in the table, the action that is to be taken with this packet is to modify the IP destination field, IP type if service field, the transport layer destination port or even the destination MAC address, but the only field that can not be modified is the IP protocol field. A firewalling action can be taken with the match plus action forwarding, the router can only receive packet to an only defined router and not receive any data from a certain router (like BGP security-based policy actions), and provided action make a better security in the SDN that cannot be achieved in the network with the simple match and forward with longest prefix forwarding. How is this information transferred from the processing unit (SDN application) to the SDN devices? The answer is using the OpenFlow protocol. OpenFlow is not the only protocol used in this purpose, there is many other protocols, but OpenFlow is the most common protocol used in the relation between the controller and the controlled devices.

OpenFlow protocol

OpenFlow protocol in implemented in the area between the SDN controller and the SDN devices and is implemented based on the match plus action forwarding. The protocol does not make any configuration in the SDN devices, just exchange message between the controller and the devices (devices support the SDN architecture and the OpenFlow protocol.) When the OpenFlow-enabled device (e.g., router and switched that enable SDN) is powered on, the device will start looking for the SDN controller to connect to it, not as will rise in mind that the controller thar will first look for the device.

After the device find the controller, it will construct with it a secure channel (used for message exchange between the controller and the device) with it, and the SDN device may receive the flow table from the controller, they must define the version of the OpenFlow that will be used in the communication between them. Message exchange between controller and device is done over a TCP session because of the reliable data transfer that is essentially in this type of communication (e.g., the flow table must be transmitted correctly.) And this TCP sessions is encrypted using the transport layer service TLS for security. The connection between controller and SDN device take two shapes: physical connection in case of direct connection between then without any intermediate devices, and logical connection in case of there is intermediate devices between them (i.e., logically directed connected but they are separated), and this connection type means that the controller can be in an area away from the SDN device.

The flow information can be transmitted from the controller to device in two ways:

* Reactive method : When a devise wants to know a certain information about the action should be taken with a certain packet (e.g., to which interface should a certain packet be forwarded) it sends to the controller asks it about the action should be taken, the controller make its processing (depending on the information of this packet like IP address and destination TCP port) and replays to it by defining the suitable action should be taken, the device receives this reply and store its content to the next time it wants to make the same decision on similar packet. This scheme is useful when there is large number of devices attached to the controller and only a small number of them is active.
* Proactive method: the controller in this method will transmit all the flow table to the SDN device after the connection is established between them (i.e., the SDN device know what to do with the incoming packet unless it is a special packet need further processing, in this case it will be sent to controller for further processing and received with the suitable action associated with it.)

The communication between the SDN controller and devices is done by special messages, and these messages have been classified to messages from the controller to the devices (i.e., packet out messages) and messages from the devices to the controller (i.e., packet in messages.)

* Packet out messages: is supposed to play an important role in SDN, we can modify or delete entries in the flow table using packet out messages, we can configure the devices (i.e., setting of the device) and set ports priorities to control the flow of the packets, controller may collect some information about the nature of the flow of the packets (e.g., how many packet is forwarded from a certain port,) the controller may sent this information to the SDN application to be processed and make use of it in controlling the congestion of the network.
* Packet put messages: SDN-controlled switch uses this type of messages to inform the controller that a certain change has been happened in the flow table (e.g., an entry has been removed due to timeout,) or to inform it that a certain change in the physical connection (e.g., certain device connect to this device on a certain port is shut down or failed.) And messages can be sent to the controller carrying a packet that needs a further processing from the controller (e.g., it matched an entry in the table and the action with this entry is to send this packet to the controller for further processing because of the complexity of the processing of this message, or it did not match ant entry from the flow table.)

There is a third type of messages between the controller and the SDN-controlled devices is used as a *keep alive* messages to keep the connection between them if there is no information to be exchanged between them.

There is a very important feature in OpenFlow protocol which reflects the importance of configuring the network with a software (i.e., use SDN). OpenFlow protocol doesn’t need any special operating system to be installed in the device, it just needs a small OS to be installed to run over it reducing the resource needed in the device and so cheaper devices, and this is not the case in other protocols like Opflex protocol developed by cisco which need a special OS installed by cisco over the device to run over it. In case of a special OS run installed on the device to run the protocol (e.g., Opflex,) we cannot replace any device with a device from other vendor because we are associated with this vendor and its equipment but it is not the case of SDN and OpenFlow protocol which so not need any special OS, thus we can replace some devices in the network with another devices from another vendor easily and this one of the greatest feature that the SDN architecture offer, because the main idea of SDN is the flexibility.

Each entry in the flow table contains three fields; the first field is the *match* criteria that a packet will be compared, and the second field is the *action* will be taken with the packet when it matched a certain match rule. There a third field in the flow table called *counter*, and this field is used to know the number of times that a certain match rule is satisfied. This important field can be used in congestion control of the network.

Suppose a SDN with five routers and two of these routers send packet to other router by forwarding these packets to only one router leaving the other router idle and overwhelm the other router with all the flow of packets.



A



B

C

D

E

SDN

Controller

SDN controlled device

TCP/TLS

Attached devices

Secure channel

Flow table

Suppose in this graph that routers A and B always send packets to router E by forwarding to router C. The match field may be matching the incoming packet with an IP destination address SSS.SSS.\*.\* and TCP source port 47451 with the packets from A and B carrying an IP destination address and TCP source port that match this rule, in this case the counter field that belongs to the number of times a packet is forwarding to its destination using this interface will be very large and this is an indication to a hustle in this interface and this hustle should be handled. With the programming cappellettis that the SDN architecture offers, the SDN-controlled devices may be configured to deal with such states. When the devices are configured and provided with the flow table, a special command of monitoring the counter field and compare it with a pre-defined value for congestion control may be also configured in these devices. In this case, device C will see that the counter field associated with this match field is very large value, it will send this information to the SDN controller (typically to the statistics module in the controller) to handle this problem, the controller receive this message and inform the application associated with controlling the flow of the messages in between the devices and reprocess the routing flow to handle this congestion in device C and send the new configuration updates to the controller which will propagate it again to the devices. These updates may force device A to send to E by forwarding to D, and it will take some work off from device C which will handle only B packets.