Advancing IP/MPLS with Software Defined Network in Wide Area Network

Irena Šeremet, Samir Čaušević
Communication Technology
Faculty for traffic and communication
Sarajevo, Bosnia and Herzegovina
Irena.seremet.1@gmail.com , Samir.causevic@gmail.com

***Abstract*— New technologies introduce more services which will call for significant advances and changes in Multiprotocol Label Switching (MPLS) networks. Networks will have to go in the direction of network programmability, virtualization and cloud-based services. In this paper, we examined Software Defined Wide Area Network solution, its architecture and benefits of use. In order to analyze benefits of using SD-WAN two scenarios are compared: (i) Policy-based routing on WAN links in classical IP/MPLS network, and (ii) Using SDN controller to manage traffic on WAN links. Aim in both scenarios is to use the most optimal path for specific network.**

Key words: IP, MPLS, SD-WAN

I. INTRODUCTION

Multiprotocol Label Switching is a protocol-agnostic technique designed to direct data from source to destination based on labels rather than IP addresses. All packet-forwarding decisions are made on the contents of assigned label, without the need to open and examine the IP packet. Routers in an MPLS network exchange label information with each other using Label Distribution Protocol (LDP) protocol. When sending packets to specific network over MPLS, devices check Label Forwarding Information Base (LFIB) tables and examine which label to use for specific network. One of the key features that MPLS support are traffic engineering (TE), Virtual Routing and Forwarding (VRF) and L2/L3 Virtual Private Networks (VPNs). Traffic engineering enables service providers to route network traffic offering the best service to their users in terms of throughput and delay. MPLS traffic engineering automatically establishes and maintains Label Switched Paths - LSPs across the network, using Resource Reservation Protocol-RSVP. Depending on customer requirements, MPLS VPNs can be (i) point-to-point (ii) Layer 2 (iii) Layer 3 [1]. Layer 2 and Layer 3 MPLS VPNs enable customers to have point-to-multipoint VPN connections. A VRF consists of one or more routing tables, a derived forwarding table, the interfaces that use the forwarding table, and the policies and routing protocols that determine what goes into the forwarding table. Because each instance is configured for a particular VPN, each VPN has separate tables, rules, and policies that control its operation [2]. MPLS as a technique is very flexible, adaptive, reliable and scales very quickly. With IP/MPLS, the paths between end-points are dynamic and extremely resilient to failures; IP/MPLS will find a path as long as one exists, regardless of the number and locations of failures in the network [3]. LSPs from source to destination are pre-determined so devices in LSP do not have to make decision on every hop. This allows faster data transfer and less load for routers. In spite of the many advantages of MPLS, the development of new technologies results in different needs and requirements of end users. When MPLS was created, applications were not in the cloud and users were not accessing corporate applications from mobile devices [4]. MPLS was an adequate technology for that-time demands. Today, new technologies introduce more services which will call for significant advances and changes in MPLS networks. Networks will have to go in the direction of network programmability, virtualization and cloud-based services [5]. In second section of this paper, we examine programmable networks, especially Software Defined Networks - SDN. In the third section, we describe using SDN in WAN and SD-WAN architecture. In the fourth section, we compare routing and traffic management in two scenarios. In first scenario policy-based routing is on WAN links in IP/MPLS network used, and in second scenario control of routing and traffic management over WAN links is delegated to a SDN controller. We conclude our paper focusing on benefits of using Software Defined Wide Area Networks in section five.

II. PROGRAMMABLE NETWORKS

With higher demands in network predictability, reliability, and performance, better management in networks became crucial. Service providers started to implement more intelligent, flexible and programmability-enabled networks. Programmability-enabled networks [6] are driven by intelligent software and use Application Programmable Interfaces (APIs) which serve as the interface to the device or controller in order to gather

978-1-7281-1563-4/19/$31.00 ©2019 IEEE

data or intelligently build configurations. Software Defined Networking (SDN) is an architecture that decouples control plane and data plane achieving flexible and intelligent networks. Control plane is responsible for building and maintaining routing table while data plane is responsible for actual forwarding packets. In traditional IP networks, every network device has its own control and data plane. In full SDN solution, network devices will only have data plane and be responsible for forwarding data. Control plane for whole network segment will be centralized and placed in one or more SDN controllers. SDN controllers will make routing decision and maintain routing table. Through different protocols, controllers will instruct network devices how to handle the packet. SDN architecture contains three layers: (i) infrastructure layer, which represents physical routers and switches; (ii) control layer, which is centralized controller responsible for managing devices in infrastructure layer and (iii) application layer with applications interacting with lower layers. Applications communicate with controller through northbound interfaces or APIs and controller communicates with infrastructure devices through southbound interfaces such as OpenFlow, Border Gateway Protocol-Link State (BGP-LS), Path Computation Element Protocol (PCEP), Netflow, Netconf, etc. The SDN provides programmability of a control plane and automation of configurations through a centralized controller and open APIs. Network operators can implement their own protocols, rules and policies with common programming languages achieving flexible control over network services such as routing, traffic engineering, QOS and security [5].

III. SOFTWARE DEFINED WIDE AREA NETWORKS

With SD-WAN, the advantages of SDN are no longer limited to the data center. SD-WAN is a concept of implementing SDN to WAN connections such as broadband internet, MPLS, 4 or 5G mobile networks etc. SD-WAN is managed by a centralized controller and uses SDN to automatically determine the best route between two sites. Also it has the ability to monitor links and if needed, dynamically route traffic to links with enough bandwidth for each application's demand. Unlike other network connectivity services, SD-WANs use application-driven networking where application traffic is forwarded over different WANs based on QoS, Security and Business priority policies [7]. SD-WANs use policies to make application routing decisions for SD-WANs tunnels over each WAN link [8]. Policy considers an application's or customer's different requirements such as QoS performance or security requirements. For example, a QoS policy may be set so voice packets are forwarded over any WAN as long as its QoS performance requirements, e.g., packet latency and loss, are met. SD-WAN provide secure, IP-based virtual overlay networks that may use a different underlay service/technology, e.g., Dedicated Internet Access, Broadband Internet (Cable, DSL or PON), Internet over LTE, MPLS over T1s, or MPLS over fiber. Because IP based SD-WANs are virtual overlay networks, no modifications have to be made to any of the underlay networks. SD-WANs also support any topology, e.g., full/partial mesh and hub & spoke [8].

A. SD-WAN architecture

As shown in the Figure 1 and Figure 2, SD-WAN architecture contains four main parts: (i) a mechanism to orchestrate connectivity (orchestrator), (ii) appliance to provide management capabilities for this environment (management), (iii) mechanism to enforce all different topologies and policy enforcements (control), and (iv) a carrier for applications and data (data).

Orchestrator, management and control parts are virtual entities and can reside on premise or within the cloud. Each of these entities will be responsible for providing specific functions and it will be consumed by a customer as a service. In data part are actual network elements such as actual physical or virtual router. When network device is on, it is already programmed according to the customer's needs and basic customer's configuration is installed. Then encrypted channel between network device and orchestrator is established. Through that channel, orchestrator checks authentication, authorization and certificates of the router. The main role of orchestrator is to validate identity of network devices in the data part. By now, router has permission to access network, but it does not have any configuration. Management entity has the role of configuring network devices.

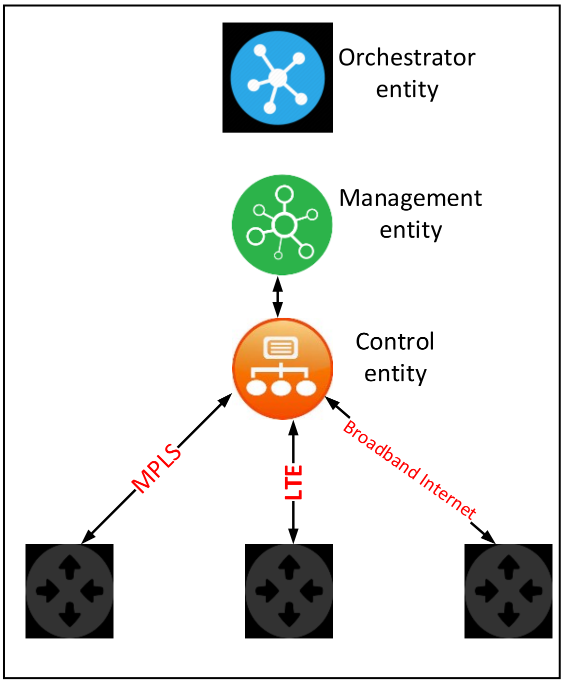


Figure 1 SD-WAN architecture

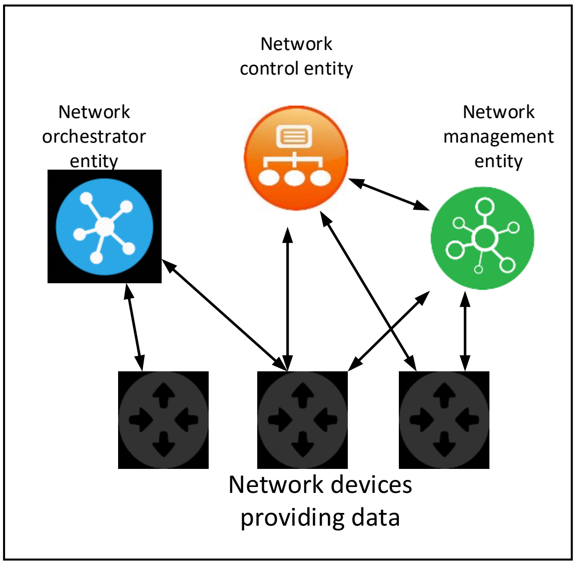

Network control entity
Network orchestrator entity
Network management entity
Network devices providing data

Figure 2: SD-WAN components

Network device establishes encrypted channel between device and management entity. Through that channel, network device sends to management entity data about telemetry, availability, statistics etc. At the same time, management entity sends configuration to network device such as vlans, interfaces, routing protocols etc. After network device is configured, routing and policy updates from the system are needed. Network device create encrypted channel with control entity to get system updates. The control part of architecture is the most intelligent part, containing one or more controllers that are in charge of managing routing processes, creating tunnels, policy enforcement and so on. Network device exchange different information with controller such as: connected circuits (Internet, MPLS, LTE), VPNs or VRFs and all learned networks in specific VRF etc.

After identifying, getting configurations and all needed updates from system, network device is ready to communicate with other network devices in the network. Controller has visibility about all attributes and information in the network. This gives opportunity to a network provider to configure several connections over different circuits defining different policies.

IV. COMPARING CLASSICAL IP/MPLS AND SD-WAN SOLUTION

Policy-based routing is possible to configure in classical IP/MPLS networks using routing policies under Interior Gateway Protocol (IGP) or BGP. But thiese solutions have some limitations. In order to achieve optimal usage of WAN links, some requirements have to be satified: (i) Global view of the network, (ii) Constant measurment of link states, (iii) Dynamic rerouting traffic to links with better parameters. In this section, defined routing policies on WAN links are configured in two scenarios: (i) In classical IP/MPLS network and (ii) Using SDN controller.

As shown in the Figure 3, topology in both scenarios containes six routers connected with eight links. Due to insufficient number of physical test devices, configuration in both scenarios was implemented on virtual routers in Cisco's Cloud virtual environment [9]. Used routers are Cisco's ASR 9000 series with IOS XR. Each router is configured with loopback interface as an ID, where Router 1 is configured with 1.1.1.1/32, Router 2 with 2.2.2.2/32 and so on. On the routers IS-IS [10] is configured as a routing protocol, and MPLS TE process in enabled. In scenarios, router 6 is sending two additional prefixes in the network: 7.7.7.7/32 and 8.8.8.8/32. Communication from R1 to prefix 7.7.7.7/32 is latency-sensitive communication and it has higher priority than communication from R1 to 8.8.8.8/32. Also, in both scenarios, links R2 - R4 and R4-R5 are high latency links and R1-R3, R3-R5 are low latency links. Aim is to avoid high latency links when sending traffic from R1 to prefix 7.7.7.7, and use any other paths when sending traffic from R1 to prefix 8.8.8.8.

A. *Scenario 1 – Classical IP/MPLS network*

In this scenario classical IP/MPLS network is configured using BGP routing policies. As shown in the Figure 3, R6 is sending two prefixes via BGP: 7.7.7.7/32 and 8.8.8.8/32. For R1, prefix 7.7.7.7 has higher priority. On R6 route policy LOW_LATENCY is configured, marked with 100:100 community and attached to prefix 7.7.7.7/32 under BGP process. Under BGP process, also prefix 8.8.8.8/32 is configured without any attached routing policies. On R1 two TE attributes are defined: (i) LOW_LATENCY; (ii) ANY_OTHER. On routers R3 and R5 LOW_LATENCY attribute is defined, and on R2 and R4 HIGH_LATENCY attribute is defined. First attribute on R1 is for tunnels which have to avoid links with HIGH_LATENCY attribute (defined on R2 and R4), and second attribute is for other tunnels. When R1 receives prefixes from R6, it maps those prefixes to one of two attributes based on the 100:100 community presence. If community 100:100 is received, R1 mapps prefix on LOW_LATENCY attribute and send traffic to R3. With these routing policies, traffic from R1 to prefix 7.7.7.7/32 will avoid path with HIGH_LATENCY attribute defined, which are R2 and R4, and use path with LOW_LATENCY attribute defined, which are R3 and R5.

B. *Scenario 2 – Using SDN controller*

As mentioned before, SDN architecture containes three layers and two interfaces connecting those three layers. Starting from the bottom of architecture, on infrastructure layer same 6 routers from Scenario 1 are used. As a southbound protocols, BGP and PCEP [11], [12], [13] are used. On the controller layer, OpenDayLight (ODL) [14], [15], [16] controller is used. It is configured on Ubuntu virtual machine. On the application layer, due to lack of any SDN application, only basic scripts are used with RESTCONF APIs of the ODL controller as a northbound interface.

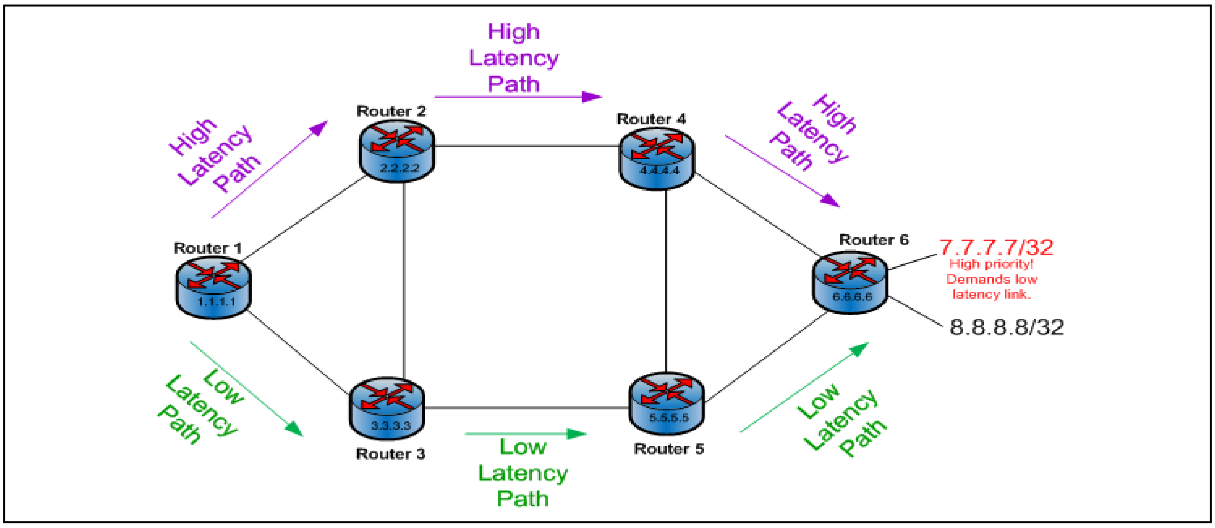
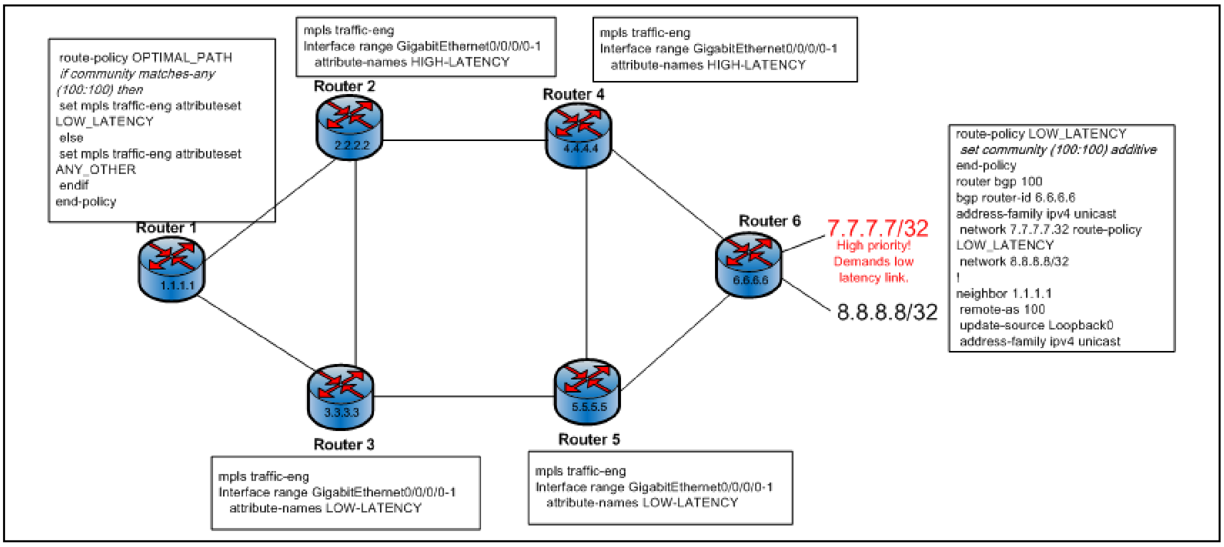


Figure 3 Used topology in both scenarios



```
route-policy OPTIMAL_PATH
 if community matches-any
(100:100) then
 set mpls traffic-eng attributeset
LOW_LATENCY
 else
 set mpls traffic-eng attributeset
ANY_OTHER
 endif
end-policy
```

```
mpls traffic-eng
Interface range GigabitEthernet0/0/0/0-1
  attribute-names HIGH-LATENCY
```

```
mpls traffic-eng
Interface range GigabitEthernet0/0/0/0-1
  attribute-names HIGH-LATENCY
```

```
route-policy LOW_LATENCY
 set community (100:100) additive
end-policy
router bgp 100
bgp router-id 6.6.6.6
address-family ipv4 unicast
 network 7.7.7.32 route-policy
LOW_LATENCY
 network 8.8.8.8/32
!
neighbor 1.1.1.1
 remote-as 100
 update-source Loopback0
 address-family ipv4 unicast
```

```
mpls traffic-eng
Interface range GigabitEthernet0/0/0/0-1
  attribute-names LOW-LATENCY
```

```
mpls traffic-eng
Interface range GigabitEthernet0/0/0/0-1
  attribute-names LOW-LATENCY
```

Figure 4 Configuration in the first scenario

All scripts are based on a Newman [17], which is open source REST CLI client previously installed on the ODL server. BGP Link-State (BGP-LS), also known as BGP for Traffic Engineering or BGP-TE, is a new BGP address family allowed to carry link-state information. This link state information is acquired from the interior gateway protocol – IGP, which is IS-IS in this case. PCEP is protocol for communication between two elements: (i) Path Computation Element - PCE and (ii) Path Computation Client - PCC. Path Computation Element is a server from the perspective of PCEP. PCE has the global view of network topology and resources enabling centralized path computation and applying TE policies. On the other hand, Path Computation Clients are network devices that take instructions from PCE and make those instructions as local configurations. In this scenario, routers (R1-R6) are configured as clients – PCCs, and ODL controller is configured as PCE. Instructions from PCE to PCC are transferred via PCEP protocol. In this scenario, BGP-LS and PCEP work together as southbound protocols in this SD-WAN solution.

On the one hand, BGP-LS provides TED (Traffic Engineering Database) information on topology and link conditions (bandwidth, cost, existing LSPs, TE metrics, etc.), while PCEP communicates between PCE and PCC. After installing PCEP plugin on ODL, session between routers (PCCs) and ODL (PCE) has to be established.

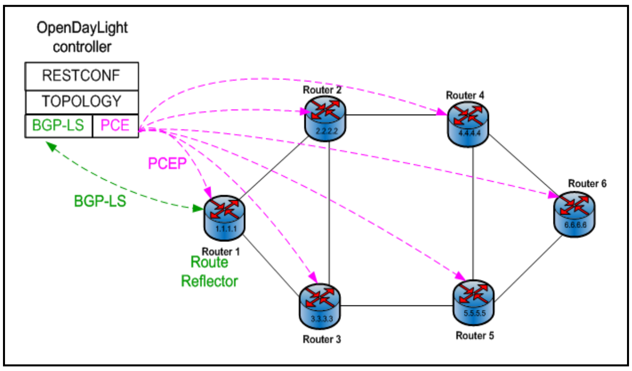
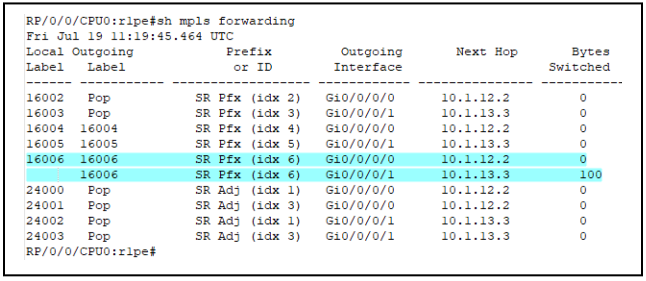


Figure 5 Topology in Scenario 2 using SDN controller

When PCEP session is established, using script with HTTP GET request and corresponding REST API URL ODL gets whole topology of network. On routers, all control over routing and creating tunnels is delegated to PCE element. With previously written JSON scripts for managing routing or traffic engineering process only command to run a specific script using ODL RESTCONF API is needed. Controller can be programmed to send IP SLA probes in order to examine loss, latency and jitter on the network. Also, different dynamical routing policies can be programmed on the controller. These policies can define different types of traffic and ways of treating different types of traffic. In our example, traffic from R1 do 7.7.7.7/32 has higher priority than R1-8.8.8.8/32. Further on, communication from Router 1 to prefix 7.7.7.7/32 is presented as Communication A, and communication from Router 1 to prefix 8.8.8.8/32 is presented as Communication B. Network management entity policy route for Communication A is configured to transfer packets over better link, and at the same time continuously examines congestions, loss, and latency on links. If quality level on link R2-R4, R4-5 ever become better, prioritized traffic is rerouted automatically on that link. Controller automatically manages different parameters of network based on configured policies. In order to test the second scenario, we started ping from R1 to 7.7.7.7/32 and we analyzed MPLS forwarding table in two cases: (i) when R1-R3-R5-R6 is low-latency link, (ii) when R1-R2-R4-R6 is low latency link. R1 is connected to R2 via Gi0/0/0/0, and with R3 via Gi0/0/0/1 port. In order to easier analyse labels and outgoing interfaces, we manually configured that label on R1 to R2 is 16002, R1 to R3 16003 and so on. Since prefix 7.7.7.7/32 is directly connected on R6, analyzed label is label number 16006. In the first case, when analyzing the output of show mpls forwarding command, label 16006 has two outgoing interfaces (Gi0/0/0/0, which connects R1 and R2; Gi0/0/0/1 which connects R1 and R3), but only Gi0/0/0/1 is used. We can conclude that because we can see that 100 bytes is switched through Gi0/0/0/1 and 0 bytes is switched through Gi0/0/0/0. That means that communication is going through R1-R3-R5-R6 link. Output of show mpls forwarding command is presented in Figure 6. In the second case, we increased latency on R1-R3 link by manually decreasing the bandwidth. After increasing the latency on link and clearing MPLS forwarding counters, we started ping from R1 to 7.7.7.7/32 and analyzed the output of show mpls forwarding command again. This time, 100 bytes switched through Gi0/0/0/0 and 0 bytes through Gi0/0/0/1, which means that R1-R2-R4-R6 link is used. Output of show mpls forwarding command in the second case is shown on the Figure 7.

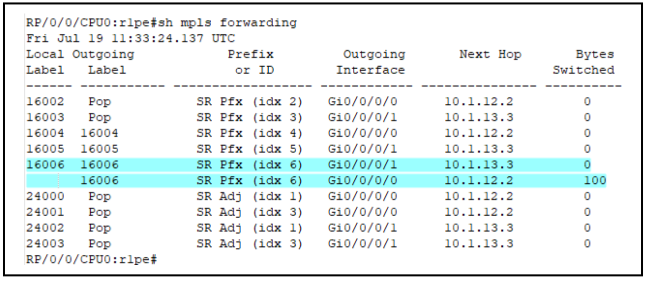
V. CONCLUSION

In this paper, routing policy on WAN links in classical IP/MPLS network is compared to scenario where SDN controller is added to a network in order to analyze benefits of using SDN in WAN. Both scenarios contain same number of routers and links, and have the same requested routing policy: Communication from Router 1 to prefix 7.7.7.7/32 (communication A) request low latency and has higher priority than communication from Router 1 to prefix 8.8.8.8/32 (communication B). In order to satisfy low-latency requirement for Communication A, tasks in the network are related to route Communication A over better links. In first scenario, routing policy is configured using BGP SR-TE routing policies. Each communication is manually marked with specific BGP community and sent over specific path. Measurments of the network are previously done, and results showed that R1-R3, R3-R5 are low latency links, so Communication A is sent through that links using 100:100 BGP community. In the second scenario, SDN controller is programmed to constantly measure states on link and dynamically route Communication over better link. If quality level on link R1-R2-R4-R5 ever become better, prioritized traffic is rerouted automatically on that link, which is impossible to achieve in the first scenario. In the first scenario, this automatization is impossible because we manually had to configure routing policies over high-latency or low-latency paths. Communication A will always use R1-R3-R5-R6, even if latency on that link increases. Comparing these two scenarios, the benefits of using SDN in WAN are obvious. SDN controller allows much more possibilities when configuring routing policies than BGP. In the first scenario, there is no network element that has global view of the network. Each router makes decisions independently for itself without being aware of the other network routers' requirements. SDN controller, on the other hand, has global view of whole network topology and all WAN links. Also, in the first scenario, static route policy under BGP process is manually added.



```
RP/0/0/CPU0:rlpe#sh mpls forwarding
Fri Jul 19 11:19:45.464 UTC
Local Outgoing          Prefix         Outgoing        Next Hop        Bytes
Label   Label           or ID         Interface                      Switched
------ ----------- ------------------ ------------ --------------- -----------
16002   Pop          SR Pfx (idx 2)   Gi0/0/0/0      10.1.12.2           0
16003   Pop          SR Pfx (idx 3)   Gi0/0/0/1      10.1.13.3           0
16004  16004         SR Pfx (idx 4)   Gi0/0/0/0      10.1.12.2           0
16005  16005         SR Pfx (idx 5)   Gi0/0/0/1      10.1.13.3           0
16006  16006         SR Pfx (idx 6)   Gi0/0/0/0      10.1.12.2           0
       16006         SR Pfx (idx 6)   Gi0/0/0/1      10.1.13.3           100
24000   Pop          SR Adj (idx 1)   Gi0/0/0/0      10.1.12.2           0
24001   Pop          SR Adj (idx 3)   Gi0/0/0/0      10.1.12.2           0
24002   Pop          SR Adj (idx 1)   Gi0/0/0/1      10.1.13.3           0
24003   Pop          SR Adj (idx 3)   Gi0/0/0/1      10.1.13.3           0
RP/0/0/CPU0:rlpe#
```

Figure 6 Output of show mpls forwarding command-Case I



```
RP/0/0/CPU0:rlpe#sh mpls forwarding
Fri Jul 19 11:33:24.137 UTC
Local Outgoing          Prefix         Outgoing        Next Hop        Bytes
Label   Label           or ID         Interface                      Switched
------ ----------- ------------------ ------------ --------------- ----------
16002   Pop          SR Pfx (idx 2)   Gi0/0/0/0      10.1.12.2           0
16003   Pop          SR Pfx (idx 3)   Gi0/0/0/1      10.1.13.3           0
16004  16004         SR Pfx (idx 4)   Gi0/0/0/0      10.1.12.2           0
16005  16005         SR Pfx (idx 5)   Gi0/0/0/1      10.1.13.3           0
16006  16006         SR Pfx (idx 6)   Gi0/0/0/1      10.1.13.3           0
       16006         SR Pfx (idx 6)   Gi0/0/0/0      10.1.12.2           100
24000   Pop          SR Adj (idx 1)   Gi0/0/0/0      10.1.12.2           0
24001   Pop          SR Adj (idx 3)   Gi0/0/0/0      10.1.12.2           0
24002   Pop          SR Adj (idx 1)   Gi0/0/0/1      10.1.13.3           0
24003   Pop          SR Adj (idx 3)   Gi0/0/0/1      10.1.13.3           0
RP/0/0/CPU0:rlpe#
```

Figure 7 Output of show mpls forwarding command-Case II

If any quality changes on path R1-R3-R5 occurs, communication A will not use better path until network engineer changes routing policy manually. On the other hand, PCE on ODL can be programmed to monitor links and dynamically route different types of traffic in accordance with requirements. Advancing MPLS with SD-WAN has many benefits. First of all, since SD-WAN is agnostic to the underlying physical transport, implementation is easy to deploy without replacing and changing the existing MPLS network. Also, SD-WAN is less expensive solution, improves application performance, simplifies the network by automating site deployments, configurations and operations. SD-WAN improves application performance through a combination of WAN optimization techniques and its ability to dynamically shift traffic to links with bandwidth sufficient enough to accommodate each application's requirements.

REFERENCES

[1] "MPLS VPNs." [Online]. Available: https://en.wikipedia.org/wiki/MPLS_VPN.

[2] Juniper, "MPLS VPN Overview." [Online]. Available: https://www.juniper.net/documentation/en_US/junos/topics/concept/mpls-security-vpn-overview.html.

[3] Packet Design, "Understanding and Managing IP / MPLS Mobile Backbone and Backhaul Networks White Paper." 2015. Available: https://www.blueplanet.com/?src=packetdesign

[4] Expereo, "Reasons to reconsider mpls." . Available: https://www.expereo.com/9-reasons-make-switch-mpls/

[5] I. Šeremet and S. Čaušević, "Evolving IP / MPLS network in order to meet 5G requirements," no. March, pp. 20–22, 2019., Available: https://infoteh.etf.ues.rs.ba/zbornik/2019/radovi/KST-1/KST-1-2.pdf

[6] T. Ryan, *Programming and Automating Cisco Networks*. Cisco Press, 2017.

[7] S. W. Sanjay Uppal, S. Woo and D. Pitt, *Software-Defined WAN SD-WAN*. ISBN: 978-1-119-10148-2

[8] MEF, "Understanding SD-WAN Managed Services," no. July, p. 15, 2017., Available: https://www.mef.net/resources/download?id=45&fileid=file1

[9] "Cisco dCloud Virtual Environment." [Online]. Available: https://dcloud2-lon.cisco.com/content/demo/246986.

[10] "Implementing IS-IS on Cisco ASR 9000 Series Routers," *Cisco*. [Online]. Available: https://www.cisco.com/c/en/us/td/docs/routers/asr9000/software/routing/configuration/guide/rcasr9kisis.html.

[11] J. A. A. Farrel, J.-P. Vasseur, "A Path Computation Element (PCE)-Based Architecture." p. 40, 2006.

[12] F. Paolucci, F. Cugini, A. Giorgetti, N. Sambo, and P. Castoldi, "A survey on the path computation element (pce) architecture," *IEEE Commun. Surv. Tutorials*, vol. 15, no. 4, pp. 1819–1841, 2013.

[13] Cisco, "Dynamic Path Computation Client," *Cisco*. [Online]. Available: https://www.cisco.com/c/en/us/td/docs/routers/asr920/configuration/guide/segment-routing/16-6-1/segment-routing-book/seg-routing-dynamic-pcc.html.

[14] J. Medved, R. Varga, A. Tkacik, and K. Gray, "OpenDaylight: Towards a model-driven SDN controller architecture," *Proceeding IEEE Int. Symp. a World Wireless, Mob. Multimed. Networks 2014, WoWMoM 2014*, 2014.

[15] OpenDayLight, "Downloading and installing OpenDaylight," *OpenDaylight Project*, 2016. [Online]. Available: https://test-odl-docs.readthedocs.io/en/latest/getting-started-guide/installing_opendaylight.html.

[16] OpenDayLight, "OpenDaylight User Interface (DLUX)," *OpenDaylight Project*, 2016. [Online]. Available: https://docs.opendaylight.org/en/stable-nitrogen/getting-started-guide/common-features/dlux.html.

[17] "Command line integration with Newman," *Postman learning center*. [Online]. Available: https://learning.getpostman.com/docs/postman/collection_runs/command_line_integration_with_newman/.