SD-WAN: an Open-Source Implementation for Enterprise Networking Services

Sebastian Troia, Ligia M. Moreira Zorello, Alvin J. Maralit, Guido Maier
Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milan 20133, Italy

ABSTRACT
A reliable Wide Area Network (WAN) has become a necessity for businesses enterprises to transmit critical data between multiple branches and to increase their revenues. Software-Defined Wide Area Networking (SD-WAN) is an emerging paradigm that introduces the advantages of Software Defined Networking (SDN) into Enterprise Networking (EN). SD-WAN can support differentiated services over public WAN by dynamically changing the flow forwarding rules over an overlay network based on monitoring data and service requirements. This paper proposes an early implementation of SD-WAN based on open source components, such as OpenDaylight as SDN controller, OpenvSwitch (OvS) and a set of services for network monitoring and policy-based path selection. We present a demo-test in a simple emulated but realistic network environment, showing new features and advantages for the enterprise in terms of resource optimization.
Keywords: Software-Defined Wide Area Networking (SD-WAN), Software Defined Networking (SDN), Enterprise Networking (EN), monitoring.

1. INTRODUCTION

A Wide Area Network (WAN), is a telecommunication network interconnecting multiple access nodes distributed over different geographic areas. Enterprises use WANs to connect to their different branches and to reach cloud services that are provisioned from a cloud computing provider. In the case of an enterprise, the access nodes are called Customer Premesis Equipment (CPE), and there is usually one CPE per enterprise branch. The rapid evolution of Enterprise Networking (EN) and Information Technologies (IT) increases consistently the demand of higher capacity and higher quality WANs [1]. The first public WANs were deployed in the early 1980s and underwent continuous development. Initially, wide area connections were designed to connect two remote sites by using leased lines that had high costs and limited speeds [2]. As a result, many different technologies have been proposed to enhance WANs in terms of cost and quality, such as Asynchronous Transfer Mode (ATM), Frame Relay (FR) and Multi-Protocol Label Switching (MPLS). Considering their advantages and disadvantages, we can refer to MPLS as the milestone of EN.

Since enterprises use critical services as video conferencing, VoIP calls, access to private and public cloud services, the quality of the WAN interconnection service is a mandatory requirement. MPLS provides full mesh connectivity, high reliability and separation of private traffic flows from the rest of IP traffic. Instead of hop-by-hop based IP routing, MPLS uses label-based forwarding and turns IP networks from connection less to connection oriented by ensuring Quality of Service (QoS). Although MPLS guarantees QoS, it presents some challenges, such as: 1) High bandwidth cost, MPLS is more expensive than Internet services which only supports best-effort mechanism; 2) Configuration overhead, zero-touch deployments are not possible, each device is configured separately; 3) Time required to transition/upgrade: operation time is directly related with the number of branch offices and edge devices. The high cost of MPLS is pushing companies to use Internet/broadband services.

By adapting WAN solutions in form of software-defined, enterprises have many advantages, such as better network performance, automation on networking deployment, cost reduction, expedited service delivery [3]. SD-WAN provides an overlay architecture that is much easier to manage than older WANs and provides a network structure that moves the control and management layers into the cloud over a centralized controller. As SDN, SD-WAN separates network functions into two part. The control and application planes are responsible for routing decisions, system configuration and monitoring; while the data plane forwards user and application data.

Nowadays, SD-WAN is gaining momentum as a connectivity solution across the enterprise landscape. Currently, there are around 30 SD-WAN vendors in the market with varying level of product maturity. Notable those with mature product offerings include VeloCloud (owned by VMware), Viptela (owned by Cisco) and Silver Peak. SD-WAN market is projected to surpass USD 17 billion by 2025[1]. Given the uprising commercial success of this technology, we believe SD-WAN should be also studied in a scientific context as a novel networking topic. In this work, we present an early implementation of a simple SD-WAN solution made by open-source software tools, such as OpenDaylight [4] as SDN controller and OpenvSwitch (OvS) [5] as virtual switches. Exploiting this implementation to perform experimental measurements, we aim at evaluating what is the degree of availability achievable by an SD-WAN system, to compare it against the availability guaranteed by more expensive WAN technologies, such as MPLS. We point out that our interest in such a study is obviously not confuting or confirming

[1] Software-Defined Wide Area Network (SD-WAN) Market 2019 In-Depth Analysis of Industry Share, Size, Growth Outlook up to 2025. White paper, Market Study Report LLC, USA, 2020.

the performance achievements claimed by the vendors of commercial SD-WAN products. Our primary goal is instead to develop a methodology to solve an interesting engineering problem of availability evaluation. In order to treat the problem on an experimental basis, we built an SD-WAN open-source solution, by means of virtualization tools such as Virtualbox[2], that can be tested in a realistic network scenario.

2. AN OPEN-SOURCE IMPLEMENTATION

The key idea of the proposed implementation is to improve wide-area network performance by deploying an SD-WAN open-source solution based on a path-switching application and a monitoring module. Thus, we implement an SD-WAN architecture in which an enterprise has to connect two branch offices to its headquarter through two Internet/Broadband connections.

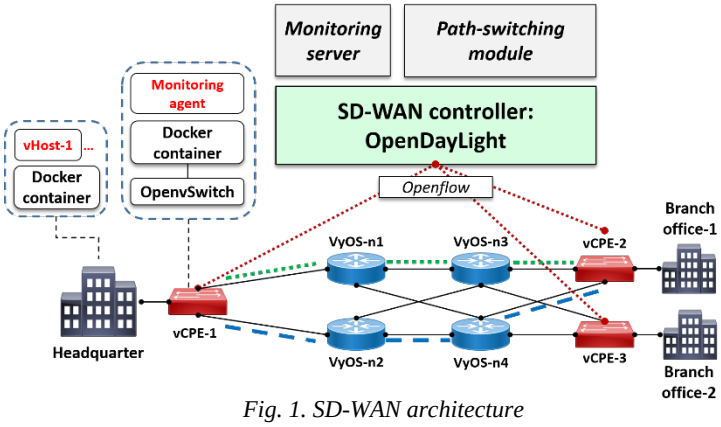

Fig. 1. SD-WAN architecture

In Fig. 1, we show the proposed SD-WAN implementation following a typical SDN architecture made by data, control and application plane. The data plane is composed by different tools to emulate: hosts, CPE's, WAN and tunnels. We used VyOS [6] to build an emulated WAN made by four virtual routers. VyOS is an opensource Linux-based network operating system that provides software-based network routing, firewall and VPN functionalities. Once they are installed and configured, we enabled the Routing Information Protocol (RIP) [7]. Then, we setup a virtual machine for each virtual CPE (vCPE) by installing: OpenvSwitch (OvS) [5], Docker [8] and D-ITG [9] software tools. OvS is a virtual switch that we exploit to configure the overlay network; while Docker and D-ITG were used as tools to program monitoring agents responsible for observing the status of the tunnels. In detail, Docker is a software tool that can package an application and its dependencies in a virtual container that can run on any Linux server. We deploy three docker containers, one for each vCPE of the test-bed, and installed D-ITG. The last is a platform capable of generating traffic at packet level accurately replicating appropriate stochastic processes for both Inter Departure Time (IDT) and Packet Size (PS). We used D-ITG to program our monitoring agents in order to compute the packet delay, jitter and loss and transmits this information to the monitoring module in the application plane. Furthermore, we have instantiated three additional docker containers, running virtual Linux-based Hosts (vHosts), to emulate SD-WAN services at the headquarters, branch office-1 and branch office-2, such as VoIP services and video streaming.

We applied the Generic Routing Encapsulation (GRE) tunnelling protocol [10] to realize two tunnels between the vCPE's through the WAN, see Tunnel-1 and Tunnel-2 in Fig. 1. GRE is a protocol for encapsulating any network layer. The advantage of GRE over other tunnelling protocols, is that it can include unicast and multicast traffic (multicast streaming or routing protocols), or other non-IP protocols. GRE packets are protected using Internet Protocol Security (IPSec), providing the integrity of the tunnel. The control plane is a logical entity that receives instructions or requirements from the application layer and relays them to the networking components. In this work, we implemented OpenDaylight [4] as SD-WAN controller. It is an open source platform for SDN that uses open protocols to provide centralized, programmatic control and network device monitoring.

Finally, we developed two applications on top of the SD-WAN controller in order to manage and improve QoS at the edge of the enterprise network. The first one is a monitoring module responsible for getting real-time network statistics from the monitoring agents, such as: packet delay, jitter and loss rate. The second is the path-switching module. According to the required network performance constraints, i.e delay, jitter or packet loss thresholds, it is in charge of changing the used tunnels by updating the OvS flow tables as a reaction to performance degradation, as explained in the next section.

2.1 Monitoring module

By monitoring we mean the collection of measurements of specific network parameters that allow to have an insight on the state of links and equipment. Thanks to these measurements, we can understand if the service

[2] Website: https://www.virtualbox.org/ (last access: 29/04/2020)

provided by the network is meeting the QoS agreed with the customer. The mostly common monitored parameters are: packet loss, packet delay and packet jitter.

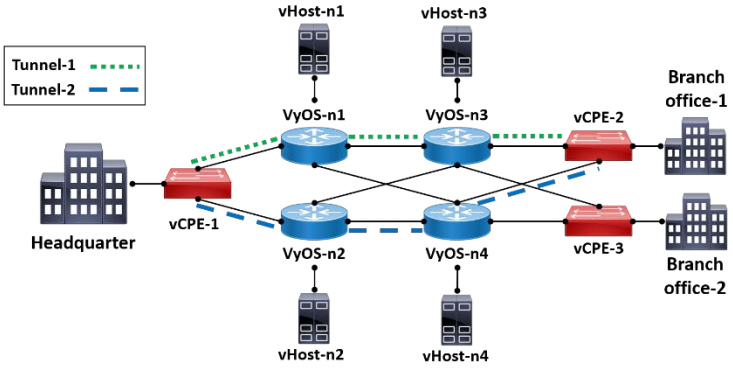

Fig. 2. SD-WAN test-bed setup

In this work, we exploit D-ITG features to implement an active monitoring module. Active monitoring is done by injecting additional packets into the network and monitoring their behaviour. The core features of D-ITG are provided by four modules: 1) *ITGSend* is the component responsible for generating traffic flows toward *ITGRecv*; 2) *ITGRecv* is responsible for receiving multiple parallel traffic flows generated by one or more *ITGSend* instances; 3) *ITGLog* is responsible for receiving and storing monitoring data information possibly sent by *ITGSend* and *ITGRecv*; 4) *ITGDec* is in charge of analyzing the monitoring data in order to extract performance metrics related to the traffic flows. We generate traffic between the headquarter and the two branch offices, and two monitoring flows for each site by using different ports. Monitoring flows are routed into Tunnel-1 and Tunnel-2 (see Fig. 1) through their port number by matching different output ports in the OvS flow table. Then, the receiver computes the packet delay and loss and transmits this information to the monitoring module in the application plane. The initialization of the monitoring application is composed by three phases: 1) the first concerns the installation of the flow rules of the monitoring traffic in the OvS switches; 2) the second step activates the monitoring flows using the UDP protocol; 3) finally, the monitoring data is received by the monitoring module. The monitoring module checks every second the information gathered by the receiver. When there is an update, the module checks if the collected metrics are still compliant to the QoS thresholds. If the collected metrics exceed the threshold of service policies, the monitoring module triggers the path switching module which will select the best path for each specific service.

2.2 Path-switching module

The path switching module aims at selecting the best tunnel compliant with QoS thresholds. An enterprise can define multiple QoS requirements, each one with specific threshold values. The path switching module is responsible of deciding which tunnel to use for each specific service by comparing the status of the tunnels against the QoS thresholds. When the monitoring module detects the collected metrics are exceeding the threshold of a specific service, it triggers the path switching module which will check the best tunnel according to the measured metrics where to send the flow of that service. This module operates the change of tunnel by modifying the flow tables of the OpenvSwitches for the specific entries that are related to that flow. As a consequence, the switches will steer the packets of the flow on the selected tunnel.

3. EVALUATIONS

In order to test the performance of the proposed SD-WAN implementation, we generate constant traffic flows between the headquarter and the branch offices and monitor the tunnels' performance in presence of link congestion, both when the path-switching module is enabled and when it is disabled. In the following experiment, we considered two performance metrics: 1) out of service time; 2) percentage of packet lost. We generate a video streaming flow (one constant UDP traffic flow at 1 Mbps) in which we setup the delay threshold to 10 ms and the packet loss threshold to 2 packets per second (pps). At a certain time, we generate a congestion in a WAN link by injecting a large amount of traffic. This artifice will lead to an increase in the delay which will exceed the defined threshold and eventually to a loss of packets. In Fig. 3, we show the delay of the UDP flow as a function of time. Between second 10 and 23, we inject about 400 Mbps between router VyOS-n1 and VyOS-n2 through vHost1-n1 and vHost1-n2 (see Fig. 2). Since the path-switching module is disabled, the UDP flow delay is increased by more than 10 ms causing an out of service equal to the total congestion time, i.e. 13 seconds. The previous considerations apply to Fig. 4 as well, in which we obtain a total of 227 packet lost (21.9 %). In Fig. 5 and Fig. 6, we show the delay and packet loss of the UDP flow considering two congestion events: one at second 10 (Tunnel-1) and the other at second 40 (Tunnel-2). Considering Fig. 5, we immediately notice how the trend of the delay changes if compared to Fig. 3. The path-switching module, now enabled, triggers the change of the tunnel as soon as the monitoring module detects the exceeding of the threshold. As a result, we obtain an out of service of just 2 seconds

over the total period of the experiment, i.e. 60 seconds. The previous considerations apply to Fig. 6 as well, in which we obtain 23 packet lost (3.3 %).

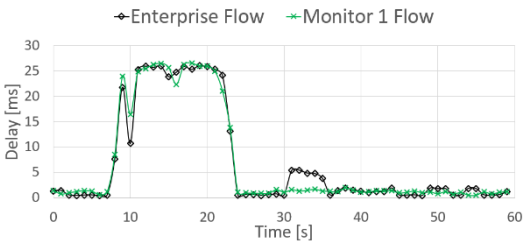

Fig. 3. Delay with switching module disabled

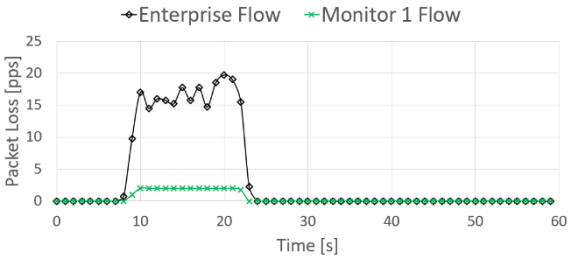

Fig. 4. Packet loss with switching module disabled

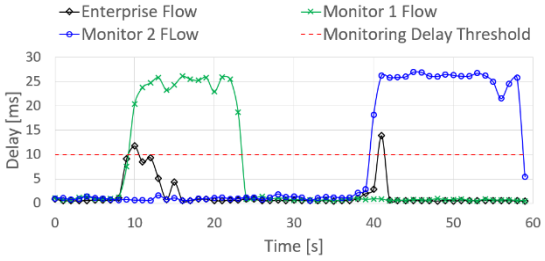

Fig. 5. Delay with switching module enabled

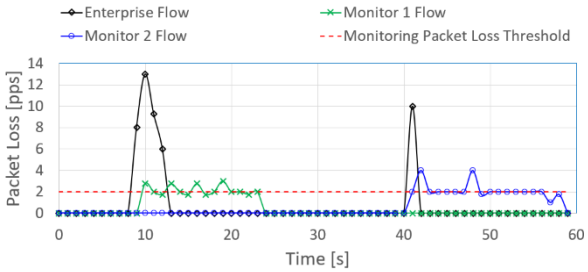

Fig. 6. Packet loss with switching module enabled

4. CONCLUSIONS

Over the past decade, as enterprises have gradually embraced the cloud as mean for their business, the traditional enterprise WAN architecture has had to evolve to address the new requirements of the digital era. SD-WAN expands the possibilities for enterprises to migrate to cloud applications in a secure manner and provides a flexible deployment model. In this work we aim at developing an experimental SD-WAN test-bed as an enterprise network to deliver service flows with certain QoS thresholds over broadband Internet by using software-defined capabilities. In order to measure the performance of the proposed solution, we developed two applications: monitoring and path-switching module. The monitoring module can provide delay and packet loss metrics by sending and capturing probe packets to the network by using D-ITG traffic generator. The path-switching module manages flow forwarding decisions according to the output taken by the monitoring modules. This work represents a first open-source implementation of this emerging technology demonstrating the advantages that its adoption entails.

ACKNOWLEDGEMENTS

The work leading to these results has been supported by the European Community under grant agreement no. 761727 Metro-Haul project.

REFERENCES

[1] Oracle group, “Five ways sd-wan is transforming cloud connectivity,” 2019. [Online].
[2] R. Graziani and B. Vachon, Cisco Networking Academy: Connecting Networks Companion Guide. Cisco Press, 2014.
[3] S. Uppal, S. Woo, and D. Pitt, “Software defined wan for dummies,” 2015.
[4] Medved, Jan, et al. "Opendaylight: Towards a model-driven sdn controller architecture." Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014.
[5] OpenvSwitch. Web: http://www.openvswitch.org/
[6] VyOS, an opensource linux-based operating system for routers and firewalls. Web: https://vyos.io/
[7] Gary Scott Malkin, RIP Version 2, STD 56, RFC Editor, 1998.
[8] Docker container. Web: https://www.docker.com/whydocker
[9] A. Botta, A. Dainotti, and A. Pescapé, “A tool for the generation of realistic network workload for emerging networking scenarios,” Computer Networks, vol. 56, no. 15, pp. 3531–3547, 2012.
[10] RFC 2784, generic routing encapsulation (GRE), IETF tools.