Data Backup Approach using Software-defined Wide Area Network

Ahmed Attia, Nour Eldeen Khalifa[0000-0001-8614-9057], Amira Kotb
Information Technology Department
Faculty of Computers and Artificial Intelligence
Cairo University, Giza, Egypt

***Abstract*—Over the past several years, the traditional approaches of managing and utilizing hybrid Wide Area Network (WAN) connections, between sites across geographical regions, have posed many challenges to enterprises. Software-Defined Wide Area Network (SD-WAN) has emerged as a new paradigm that can overcome the traditional WAN challenges like the lack of visibility for WAN bandwidth utilization and the inefficient usage of expensive WAN resources. The flexibility and agility brought to WAN by applying the SD-WAN paradigm helped to improve the efficiency of bandwidth utilization and to address the surge of bandwidth demands. The SD-WAN capabilities become essential for meeting the heavy inter-data center's traffic exchange required for business continuity and disaster recovery operations. In this paper, a data backup approach is introduced using SD-WAN that makes the network centrally programmable. This will leverage the ability to make fine-grained traffic engineering for different data flows over WAN to optimize the bandwidth utilization of the expensive WAN resources by balancing the traffic load across network links between data centers and to minimize the time required to transfer backup data to disaster recovery sites. The proposed approach proved its efficiency according to the bandwidth utilization if it is compared to the other related works.**

Keywords*—Wide area networks; software defined network; software defined wide area network

I. Introduction

The Wide Area Network (WAN) is essential nowadays for large enterprises and businesses. It is a critical component to connect between data centers and different sites across geographies [1]. The first public WAN was deployed in early 1980 [2] and designed to connect between different sites using leased lines that had limited speeds and high cost.

In the previous decades, different WAN technologies have been developed to provide better service quality in terms of cost and speed such as VPN (Virtual Private Network), ATM (Asynchronous Transfer Mode), and MPLS (Multi-Protocol Label Switching). Despite the improvements in the bandwidth and speed of WAN networks and internet services, they are still congested with high traffic loads that cause data loss and jitters [3] which impact the performance and quality of the provided services.

During the COVID-19 pandemic, network traffic spikes have increased obviously and start to impact the services provided by different internet and cloud companies such as Amazon, YouTube, and Netflix [4]. To address these challenges and the surge in bandwidth demand, enterprises start to use technological innovations such as Software Defined Networks.

Software-Defined Networks (SDN) has emerged over the past years as a new promising model that simplifies network management. SDN aims to provide centralized management for multiple network devices like routers, switches, and firewalls by separating the control plane from the data plane [5]. The SDN framework consists of three layers. The lowest layer is the data plane layer that contains the network elements and devices responsible for packet forwarding. One layer above, the control plane is located where network intelligence is logically centralized to maintain a global view of the network.

The SDN controllers communicate with network devices to guide them in handling data packets and to execute network policies and rules. The application layer contains the network applications that introduce new network features and applications such as load balancing, network statistics monitoring, and security [6]. The application layer makes use of the holistic view of the network provided by controllers to get information about different data flows and give the appropriate guidance to the control layer for enhancing the network performance. Currently, there are several SDN controllers provided such as OpenDaylight [7], POX [8], NOX [9], and Beacon [10] that can simplify the management of network devices.

For the southbound interface, the OpenFlow protocol is the most used open protocol for the interaction between the SDN controllers in the control plane and SDN OpenFlow compliant switches in the data plane [6]. The OpenFlow SDN controller will insert flow entries on the OpenFlow compliant switches to instruct them for forwarding the incoming data packets. The OpenFlow switches will follow the basic packet forwarding mechanism to handle the incoming packets. They will check the packets header and match it to the entries inserted by the SDN controller in the flow table. In case no flow entry matches the incoming packet header, the OpenFlow compliant switch will notify the controller to identify the correct action for the packet and install the appropriate flow entries for it. The SDN controller can also interact with OpenFlow switches [11] to retrieve statistical information about data flows.

On the other side, the northbound interface is provided by the SDN controller to the application layer for network programmability. Applications neither have direct interaction

with data plane devices nor are aware of network topology [12]. They interact with controllers to make fine-grained traffic engineering, based on the data flow statistics controller can retrieve, and guide them with the path layout of the data flows [13].

The SDN paradigm is usually used in data centers local area networks (LAN) but recently it is used in wide area networks (WAN) to mitigate the challenges mentioned before. The Software-Defined Wide Area Networks (SD-WAN) is built on the concepts of SDN. The WAN networks will obtain many technical advantages by using the SDN concepts [14] that will help to leverage its performance, utilization, and efficiency to handle the massive data transmissions.

Multinational technology companies like Microsoft and Google used SDN concepts to deploy their SD-WAN solution for inter-data center connections. They have multiple data centers distributed across the planet that are exchanging data traffic extensively for communication between subsystems and data backup activities [5]. Gartner anticipates that by 2024 60% of enterprises will implement SD-WAN [15].

SD-WAN can play a big role in the enhancement of business continuity strategies. Data backup and disaster recovery mechanisms focus on what needs to be done to keep the business running when disasters hit. The long-distance WAN networks play a key role in disaster recovery operations that require a large amount of data to be exchanged over the long distance between primary and secondary sites. They are restricted by limited bandwidth and the high latency of WAN networks. SD-WAN can handle these issues [16] and cope with the increasing network demands to enhance the data backup and disaster recovery operations.

In this paper, an approach for data backup and replication is proposed using the concepts of SD-WAN. The main objective of the proposed approach is to use SD-WAN capabilities to increase the efficiency of WAN network utilization and the adaptation to network demands. This should help to minimize the time needed for data backup between sites by shrinking the recovery point objective down to meet business continuity demands for enterprises.

In the proposed approach, SD-WAN centralized management and programmability are used to run applications that can check the network statistics of data flows like throughput across various links in real-time, and based on these measurements, decisions are taken to meet the demands of other data flows and guarantee the fair utilization of the available WAN links. The OpenNetMon open tool [17] is used in this research to get data flow statistics and to make fine-grained traffic engineering. This will be discussed in the methodology and the results sections.

The remainder of this paper is structured as follows: In Section II, related works to this research are discussed. In Section III, the methodology and algorithm used for balancing the traffic load across all available WAN links are presented. Sections IV and V summarizes the results after applying the proposed approach on the test topology. Finally, Section VI concludes this paper and presents the future works.

II. Related Work

SD-WAN shows great potential in improving the performance and utilization of WAN networks. Currently, many SD-WAN vendors in the market provide different mature products for traffic management and real-time analytics to improve network performance and availability such as Cisco SD-WAN (Viptela), VMware SD-WAN, and Silver Peak. In the past several years, many pieces of research were published showing the benefits of deploying SD-WAN and how it can improve the performance of businesses' workflows and services. Some approaches that aim to improve bandwidth utilization will be reviewed in this section.

Tech giants like Google and Microsoft who have multiple data centers across the planet deployed their SD-WAN solutions for interconnection between data centers. B4 is Google's software-defined inter-data center WAN solution. It uses OpenFlow protocol to manage individual network switches with centralized controllers that run traffic engineering applications. In [18], the authors discuss their experience with deploying B4 in production and achieving better bandwidth utilization for WAN links to perform large-scale data copies between sites.

The second deployment is SWAN from Microsoft that aims to achieve highly efficient and flexible WAN interconnection between data centers. It also improves the fairness to meet different traffic demands when WAN links are utilized with background traffic [19]. It is obvious that both B4 and SWAN, use the capabilities of SD-WAN like the controller's global vision of the network and the easiness to retrieve network statistics for traffic engineering, to maintain better utilization for the WAN expensive resources.

In [20], the authors introduced a software-defined traffic load balancing (SD-TLB) module that is embedded into the ingress nodes at the edge of carrier networks. It is built on the CPU of the physical ASIC-based (Application Specific Integrated Circuit) SDN switches. The SD-TLB performs per-flow scheduling in a round-robin manner over the optical paths besides detecting any path outage or failure using a global network controller based on per-port statistics.

This research showed by experiment that the SD-TLB module is better than the link aggregation group (LAG) for optimizing the bandwidth utilization for optical paths in carrier networks. However, it doesn't abstract completely the control plane and forwarding plane as SD-TLB is still running on the CPU of the edge network switches.

In [21], the authors proposed the implementation of a wide area network defined by software, which guarantees a predefined quality of service (QoS) and traffic prioritization between Software-Defined Datacenters.

The proposal deploys an experiment test scenario based on the concept of the living lab which measures the performance of the network when Voice Over IP (VOIP) services are provided and verified that an adequate level of QoS of Bandwidth can be guaranteed by providing traffic priority in an SD-WAN network between Software-defined data centers (SDDCs).

However, this approach focuses only on maintaining QOS by prioritizing traffic based on the source and destination IPs by a policy/rules introduced by the SD-WAN OpenFlow controller Floodlight (by using OpenFlow flow tables that help OpenFlow switches to steer traffic) and it doesn't introduce any approach that can balance traffic or steer for flows to another transport link in case of having poor bandwidth or high packet loss which can highly impact the quality of VOIP services.

In [22], the authors proposed a simple implementation for SD-WAN using open source tools like OpenDaylight SD-WAN controller and Open vSwitch as virtual switches. They developed applications to run on top of a centralized SD-WAN controller to manage and improve the quality of service (QoS) of the interconnecting WAN network between headquarter and branch offices. The two applications used in this implementation are for monitoring and path switching. The monitoring module will collect real-time statistics for network metrics like packet loss, packet delay, and jitters by injecting traffic into the network paths to evaluate the network links and confirm if they will meet the required QoS. After that, the path switching module will install the path that meets the enterprise QoS thresholds on network switches.

This implementation helped to guarantee the QoS requirements for enterprise applications and services while transferring data between sites. However, the usage of active monitoring and injecting traffic to measure network statistics in real-time may introduce additional network load that impacts the accuracy of measurements.

In this research, fine-grained traffic engineering is enabled in the proposed approach using the real-time measured network metrics for each data flow. The per-flow throughput is measured using an opensource tool OpenNetMon by querying per-flow counters from source and destination switches without injecting extra packets through the network between sites to avoid any overhead that may be introduced. The measured throughput metric will provide an advanced bandwidth utilization monitoring for all available links between sites which will enhance routing decisions for new data flows in a way that can improve the traffic load balancing and the efficiency of utilizing the available bandwidth.

III. Methodology

SDN and SD-WAN are based on the same methodology of separating the control plane from the data plane. Both are using controllers and OpenFlow switches, the main difference is that SDN controllers manage every network device including switches and core switches in LAN networks while SD-WAN controllers manage and interact only with edge devices [23].

In this research, topology in Fig. 1 is suggested to simulate the WAN connection between two sites using different paths (simulating WAN links with different speeds/bandwidth).

Edge devices act as gateways that allow data centers to communicate with other sites through WAN links that can be served by different WAN providers. SD-WAN controllers can't manage or interact with intermediate devices that are under the control of WAN service providers but still can be programmable to interact with sites' edge devices to prioritize and control WAN traffic between different locations.

OpenNetMon software is used in the proposed approach, an open-source software that runs on SD-WAN controller for network monitoring. The OpenNetMon monitoring software will interact with edge OpenFlow switches to monitor TCP flows in real-time and gather statistical info about them. The real-time monitored information will leverage the ability to make fine-grained traffic engineering and hence make more efficient routing decisions. This will enhance data transfer operations through WAN between different sites which is essential for WAN-based backup.

OpenNetMon software is composed of two modules, the forwarding module, and the monitoring module. The forwarding module will discover paths between source-destination pairs when an un-matched TCP flow connection or packet arrives and install path on switches instructing them to route packets for this TCP flow. The monitoring module will monitor flow metrics (throughput/delay/packet loss) between source-destination pairs.

In this proposed approach, the per-flow monitored throughput metric is used for traffic engineering to enhance the bandwidth utilization and balance the traffic load across WAN links between sites while transferring data for WAN-based data backup operations.

Pox controller eel branch is used on Ubuntu Linux 3.13.0-24-generic virtual machine with 8 processor cores and 4GB RAM to run the OpenNetMon software, and mininet version 2.2.1 is used to build testing topology.

The TCP flow throughput metric is periodically measured by querying flow statistics via OpenFlow protocol. The SD-WAN controller will interact with edge Openflow switches to get individual flow statistics on a timely basis to measure the TCP flow throughput in real-time. Equation (1) is used to measure the TCP flow throughput in a given time [24].

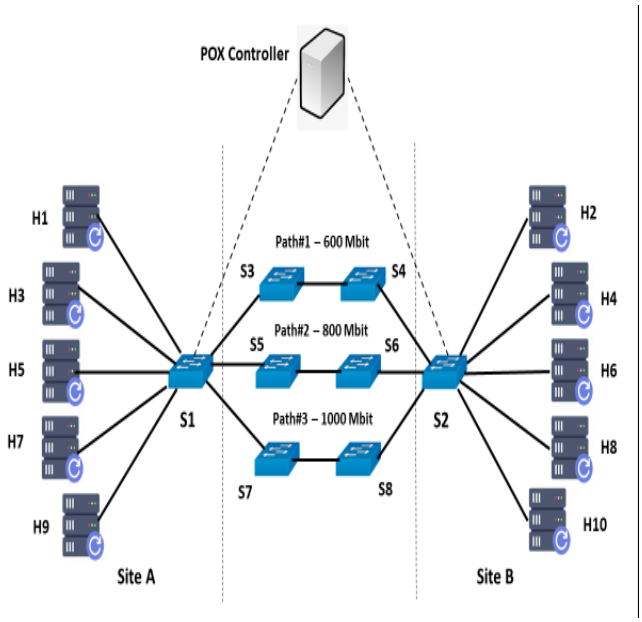


Fig. 1. Test Network Topology.

$$\text{Throughput (tp)} = \frac{\sum_{j=1}^{t} r_j}{t} \quad (1)$$

r_j Number of bytes processed for the TCP flow

t Time

The measured throughput metric of each TCP flow is used to find out the bandwidth utilization of each WAN link and based on this SD-WAN controller will select the least utilized link to be used by the newly initiated TCP flows for transferring data.

By calculating the average throughput of all TCP flows, it can be predicted if the WAN link used by each flow is highly loaded or has room for a new TCP stream. Equation (2) is used to measure the average throughput where tp donates to the measured throughput records for a specific TCP flow and n donates to the number of readings [24].

$$\text{Average throughput} = \frac{\sum_{i=1}^{n} tp_i}{n} \quad (2)$$

tp_i Represents measured TCP flow throughput at a specific interval.

n Represents the number of measured throughput records.

The Flow chart in Fig. 2 explains the proposed approach algorithm to select the least utilized path when new TCP flow starts:

In a nutshell, the following is a step-by-step implementation methodology for the proposed approach:

- Emulate the testing network topology in Fig. 1 on mininet for two sites with edge OpenFlow switches that have three WAN links each one has different Bandwidth (Path#1 600 Mbit – Path#2 800 Mbit – Path#3 1000 Mbit).
- Start running the OpenNetMon software on Pox Controller.
- Initiate three TCP flows at the same time using iperf tool between source-destination pairs H1-H2, H3-H4, and H5-H6 to transfer 10 Gig of Data.
- The three TCP flows will utilize the three WAN Links between the two sites.
- The forwarding module in OpenNetMon software was modified to discover all available paths between source-destination pairs.
- When the OpenNetMon software starts running on the POX controller, all paths between the two sites are in the ideal state. An index variable (i = 0) is used and will increment every time a new TCP flow starts.
- If 'i' is smaller than the number of available paths between source-destination pair, the controller will select one of the ideal paths to the new TCP flow.
- The aim of this is to generate traffic and flow congestion on all available paths.
- The monitoring module will keep a record of throughput for TCP flows on each path and calculate the average throughput for each link between 2 sites.
- After 30 seconds, a new TCP flow was initiated between source-destination pair H7-H8 to transfer 5Gig data using iperf.
- After 60 Seconds new TCP flow was initiated between source-destination pair H9-H10 to transfer 5 Gig of data using iperf.
- In the proposed approach, the forwarding module will select the path with the largest average throughput that will be the least utilized path and will have room to transfer more data.

In the next section, results for the proposed approach are highlighted regarding the throughput and the completion time needed for each TCP flow. Also, a comparison was made between the proposed approach and the Round-Robin approach [25].

Round-Robin: TCP flows are distributed among available paths in a sequential manner. In other words, the chosen path for the new TCP flows for source-destination pairs H7-H8 and H9-H10 is always the path of the next round in the available paths.

Proposed approach: The controller will direct TCP flows for source-destination pairs H7-H8 and H9-H10 to the least utilized path the moment a new flow is initiated based on real-time monitoring.

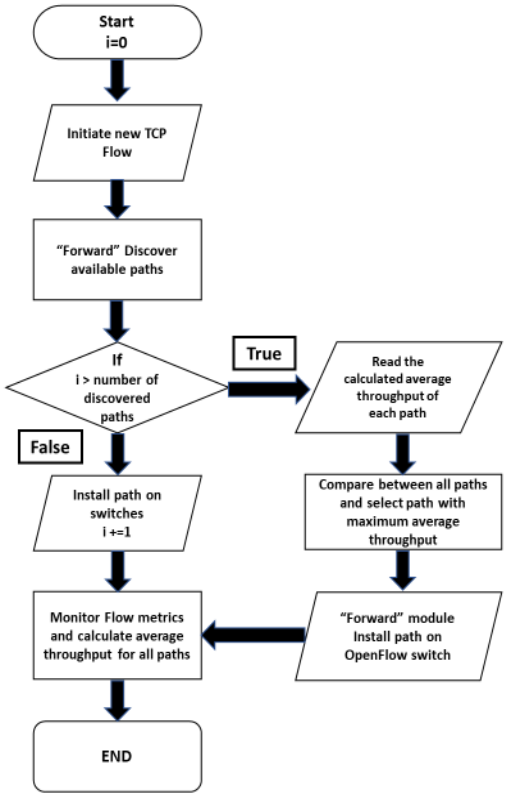


Fig. 2. Flow Chart for the Algorithm used in the Proposed Approach.

IV. Experimental Results

The two approaches were tested on the topology in Fig. 1 while transferring data (simulating data backup operations) between two sites.

The first approach used is Round-Robin. This approach will use all available paths with different Bandwidth (that simulates WAN links) in a sequential manner. Data was transferred between hosts/backup servers from site A to site B using iperf tool and Table I shows the path used by each source-destination pair in the Round-Robin approach.

Fig. 3 shows the throughput of all TCP flows on the 3 paths between source-destination host pairs. As it can be observed from the Round-Robin approach, the TCP flows from source-destination pairs H7-H8 and H9-H10 use path#1 and path#2 to transfer data, however, Path#3 has higher bandwidth, and this can be seen in the high throughput of H5-H6 TCP flow that utilizes this path in the graph (c). The usage of a single path by multiple TCP flows will impact the throughput share of each flow as shown in graphs (a) and (b).

The second approach tested is the proposed approach. Table II shows the path used by each source-destination pair while testing. In Fig. 4, the TCP flow for source-destination pair H7-H8 (that starts after 30 Seconds) uses path#3 based on the average throughput calculation done by the controller for the three paths that are utilized by TCP flows for source-destination pairs H1-H2, H3-H4, and H5-H6.

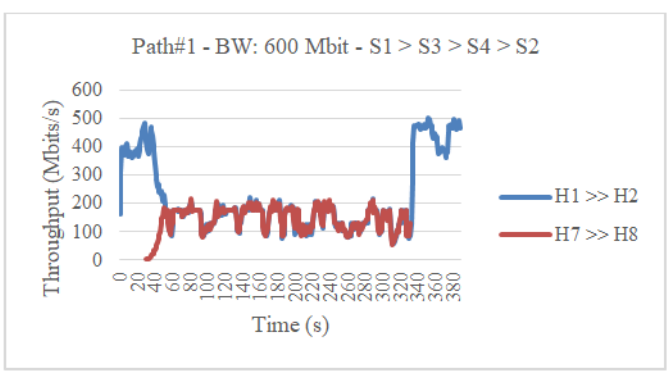
After 60 Seconds when a new TCP flow was initiated for H9-H10, path#2 was selected by the controller for it. The average throughput for path#2 shows better real-time results the moment this flow was started compared to path#3 that was already utilized by 2 TCP flows for H5-H6 and H7-H8 and path#1 that has the least bandwidth.

TABLE I. Paths used by each Flow in the Round-Robin Approach

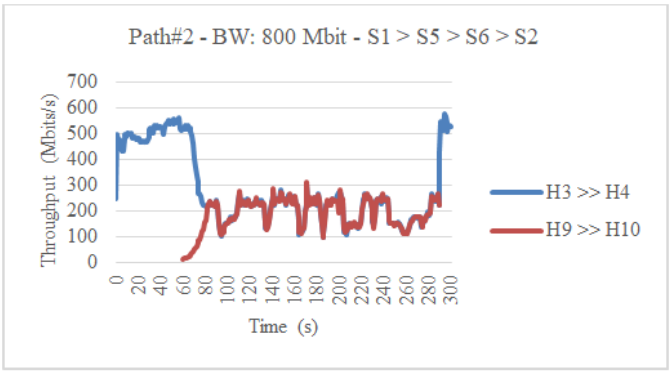
Src-Dst pair	Data Transferred	Path	Path Bandwidth
H1-H2	10 Gig	Path#1 - S1 > S3 > S4 > S2	600 Mbit
H3-H4	10 Gig	Path#2 - S1 > S5 > S6 > S2	800 Mbit
H5-H6	10 Gig	Path#3 - S1 > S7 > S8 > S2	1000 Mbit
H7-H8	5 Gig	Path#1 - S1 > S3 > S4 > S2	600 Mbit
H9-H10	5 Gig	Path#2 - S1 > S5 > S6 > S2	800 Mbit

TABLE II. Paths used by each Flow in the Proposed Approach

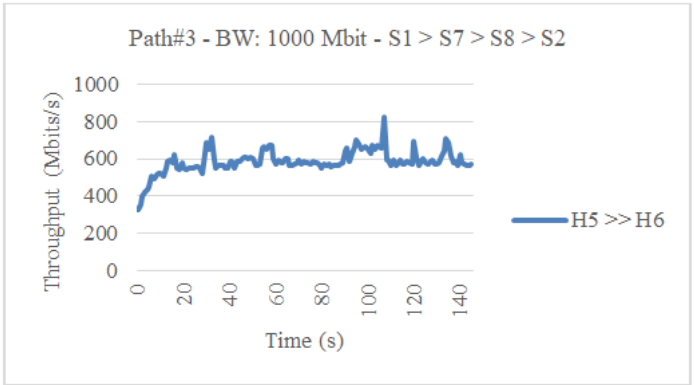
Src-Dst pair	Data Transferred	Path	Path Bandwidth
H1-H2	10 Gig	Path#1 - S1 > S3 > S4 > S2	600 Mbit
H3-H4	10 Gig	Path#2 - S1 > S5 > S6 > S2	800 Mbit
H5-H6	10 Gig	Path#3 - S1 > S7 > S8 > S2	1000 Mbit
H7-H8	5 Gig	Path#3 - S1 > S7 > S8 > S2	1000 Mbit
H9-H10	5 Gig	Path#2 - S1 > S5 > S6 > S2	800 Mbit



(a)



(b)



(c)

Fig. 3. (a) Network throughput for H1-H2 and H7-H8 TCP Flows on Path#1 using the Round-Robin Approach, (b) Network throughput for H3-H4 and H9-H10 TCP Flows on Path#2 using the Round-Robin Approach, and (c) Network throughput for H5-H6 TCP flow on Path#3 using the Round-Robin Approach.

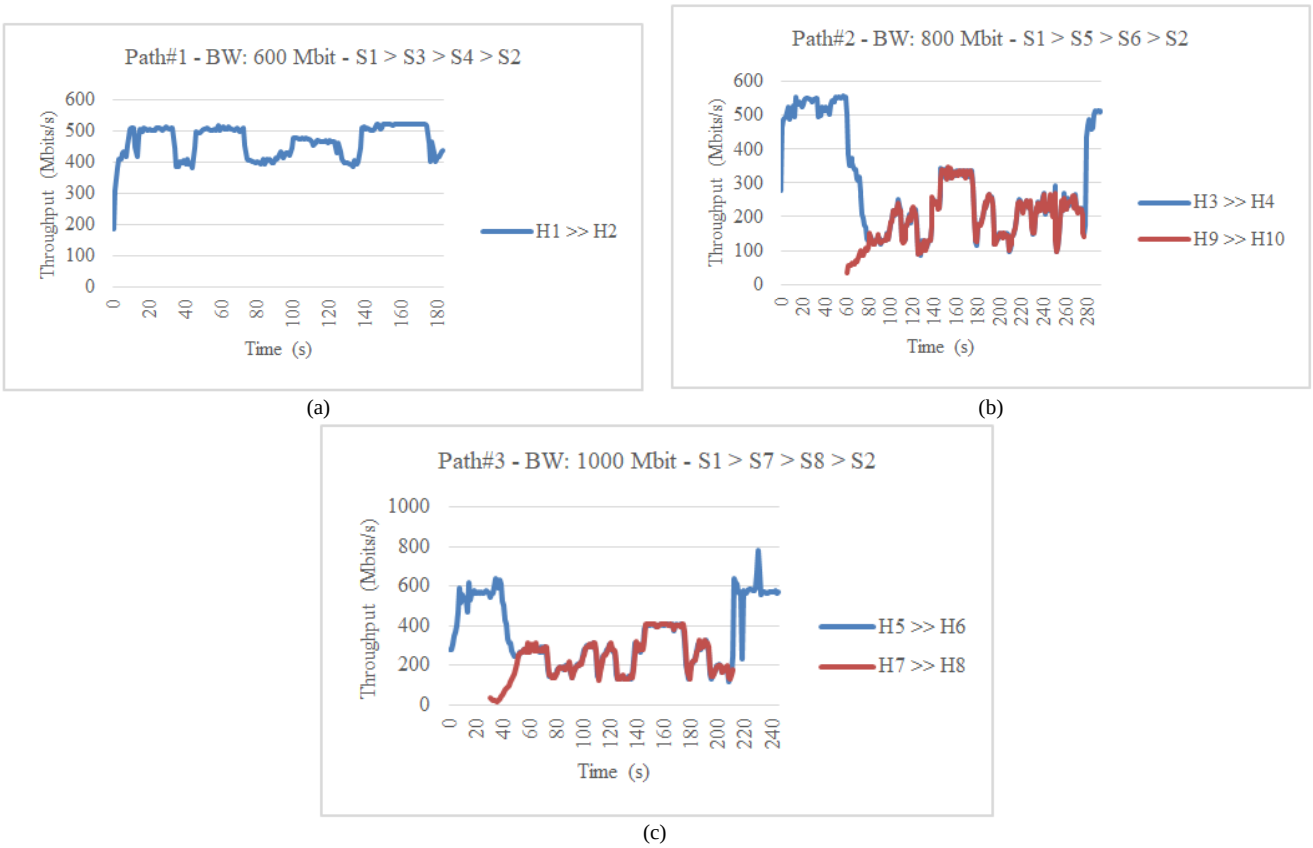


Fig. 4. (a) Network throughput for H1-H2 TCP Flow on Path#1 using the Proposed Approach, (b) Network throughput for H3-H4 and H9-H10 TCP Flows on Path#2 using the Proposed Approach, and (c) Network throughput for H5-H6 and H7-H8 TCP Flows on Path#3 using the Proposed Approach.

V. Results and Discussion

As it can be observed in the comparison between the 2 approaches in Fig. 5, for the average throughput and time taken by each TCP flow for completing data transfer, the proposed approach shows that a fair share and better utilization of the bandwidth resources can be achieved in challenging network conditions beside reducing the time needed to transfer backup data between sites.

In Fig. 5(a) it is clear that the total time required for the 5 data flows that are initiated between the 5 source-destination pairs to transfer 40Gig of data, has been reduced significantly using the proposed approach compared to Round-Robin. The total time required for transferring 40Gig using the Round-Robin approach is 389.3 seconds across the 3 available links used to connect between the two sites A and B. While in the proposed approach, using traffic engineering based on SD-WAN capabilities to balance the traffic load across network links, the time required to complete data transfer is reduced to 293 seconds.

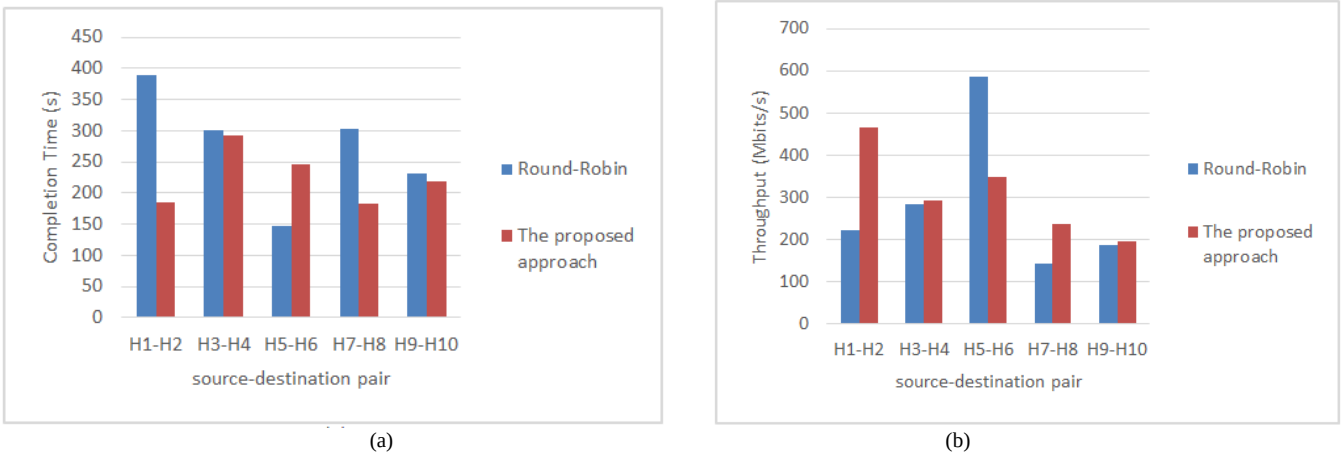


Fig. 5. Comparison between the Two Approaches for (a) Completion Time and (b) Average throughput of each TCP Flow.

These numbers reflect the bandwidth utilization enhancement between the data flows as shown in Fig. 5(b). The traffic load balancing mechanism introduced in this research, based on real-time measured network throughput, helps to improve the fairness in bandwidth utilization between the different data flows. The focus on background traffic initiated by the data flows (H1-H2, H3-H4, H5-H6) that initially utilized the three available network links, helps to make better routing decisions for newly initiated flows (H7-H8 and H9-H10). The average throughput for H7-H8 data flows while using the proposed approach is 236 Mbit/sec compared to round-robin 142 Mbit/sec. This enhancement occurs because of a better routing decision to use network Path#3 (bandwidth 1000 Mbit) and to share it with data flow (H5-H6) instead of sharing Path#1 (that has the least bandwidth 600 Mbit) with data flow (H1-H2). On the other hand, data flow (H1-H2) average throughput has improved to be 466 Mbit/sec, instead of 221 Mbit/sec in the round-robin case, after utilizing the limited bandwidth resources of Path#1 (bandwidth 600 Mbit) without having another competing data flow that tries to share resources on this network link.

It is obvious that the average throughput of H5-H6 data flow (transferring 10Gig of data) has been reduced from 587 Mbit/sec to 349 Mbit/sec after using the proposed approach for balancing data. This reduction is acceptable in the interest of improving fairness between all data flows and meeting the traffic bandwidth demands. This flow was using individually Path#3 (bandwidth 1000 Mbit) to transfer 10Gig of data in the round-robin approach, leaving the other four data flows to share the remaining two available network links Path#1 (bandwidth 600 Mbit) and Path#2 (bandwidth 800 Mbit) for transferring 30 Gig of data. After applying the proposed approach to balance traffic load, Path#3 is shared between two data flows (H5-H6 and H7-H8) to transfer 15 Gig of data.

The average throughput results for data flows H9-H10 and H3-H4 are nearly similar in the two approaches. The reason is that both data flows have shared the same network link Path#2 (bandwidth 800 Mbit) when applying the round-robin approach and the proposed approach as shown in Table I and Table II.

The results show that the aim of this research has been achieved by deploying fine-grained traffic engineering using SD-WAN to balance the traffic load across all available WAN links between data centers in a way that improves the bandwidth utilization and minimizes the time required for transferring data. The centralized traffic engineering used in the proposed approach helps to overcome the poor efficiency and sharing capabilities of WAN. This was achieved by considering the current state of the available WAN links before initiating new flows. The average throughput measurements are used to identify the reliability of each WAN link to handle new TCP flow without impacting the current running flows. From the test results, it is obvious that Round-Robin can cause an unplanned over subscription for some WAN links which degrades the performance of the competing TCP flows using them while other links are underutilized. On the other hand, the proposed approach helps to overcome the under and over subscription swinging state for available WAN links by enhancing the routing decisions for new data flows in a way that improves the fairness in sharing WAN resources and the utilization of bandwidth.

These enhancements are blessings for data backup and replication operations that are not a basic one-time function and they need to overcome many network limitations like latency for transferring data over long-distance networks. By minimizing the time required to transfer data through the network to the disaster recovery sites, the efficiency of data protection and business continuity will improve, and enterprises will be able to minimize their downtime.

VI. Conclusion and Future Work

In this work, the Software-Defined Network paradigm has been applied to achieve traffic load balancing across the hybrid WAN links available between data centers. The centralized programmable interface introduced by SD-WAN leverages the ability to make advanced network measurements for data flows that are propagating across network WAN links. An open-source network monitoring tool OpenNetMon is used to measure network throughput metrics for data flows based on SD-WAN capabilities. The measured network throughput metrics were used to balance the traffic load across available network links and to improve the efficiency of bandwidth utilization while transferring data between data centers.

The aim of this proposed approach is not to confirm or confute the performance improvement that can be achieved by other approaches that are used for traffic load balancing or to improve the bandwidth utilization, but the primary goal instead is to develop a new approach based on an open-source tool that was introduced in previous research to monitor per-flow metrics in real-time and provide the network statistical information essential for traffic engineering purposes.

By testing the proposed approach on a network topology simulated using mininet and compared it to the round-robin approach, the results showed that the approach developed in this research has shown an improvement in utilizing the bandwidth resources and helped to minimize the time required for transferring data between two sites. This will improve the efficiency of business continuity applications and helps the data backup operations to overcome network latency limitations that impact the time required to backup data between production and disaster recovery sites.

In the future work, it is planned to develop the used algorithm to add other network metrics like packet loss and latency to measure the reliability of available paths and to ensure meeting adequate quality of service policies before making routing decisions for new data flows. It is intended to extend the proposed approach by integrating algorithms to apply fault tolerance. The aim of fault tolerance is to failover data flows to use another WAN link in case of link failure or severe degradation in the link performance and throughput.

References

[1] B. Shin, *A Practical Introduction to Enterprise Network and Security Management*. Auerbach Publications, 2021.

[2] R. Graziani and B. Vachon, *Cisco Networking Academy: Connecting Networks Companion Guide*. Cisco Press, 2014.

[3] X. Tao, K. Ota, M. Dong, W. Borjigin, H. Qi, and K. Li, 'Congestion-aware Traffic Allocation for Geo-distributed Data Centers', *IEEE Trans. Cloud Comput.*, 2020.

[4] M. Said Elsayed, N.-A. Le-Khac, and A. D. Jurcut, 'Dealing With COVID-19 Network Traffic Spikes [Cybercrime and Forensics]', *IEEE Secur. Priv.*, vol. 19, no. 1, pp. 90–94, Jan. 2021, doi: 10.1109/MSEC.2020.3037448.

[5] O. Michel and E. Keller, 'SDN in Wide-Area Networks: A Survey', p. 6, 2017.

[6] G. Pujolle, *Software Networks: Virtualization, SDN, 5G, and Security*. John Wiley & Sons, 2020.

[7] A. Eftimie and E. Borcoci, 'SDN controller implementation using OpenDaylight: experiments', in *2020 13th International Conference on Communications (COMM)*, 2020, pp. 477–481.

[8] H. M. Noman and M. N. Jasim, 'POX controller and open flow performance evaluation in software defined networks (SDN) using mininet emulator', in *IOP conference series: materials science and engineering*, 2020, vol. 881, no. 1, p. 012102.

[9] V. S. Raju, 'SDN controllers comparison', 2018.

[10] M. Paliwal, D. Shrimankar, and O. Tembhurne, 'Controllers in SDN: A Review Report', *IEEE Access*, vol. 6, pp. 36256–36270, 2018, doi: 10.1109/ACCESS.2018.2846236.

[11] D. J. Hamad, K. G. Yalda, and I. T. Okumus, 'Getting traffic statistics from network devices in an SDN environment using OpenFlow', *Inf. Technol. Syst.*, vol. 2015, pp. 951–956, 2015.

[12] F. A. Lopes, M. Santos, R. Fidalgo, and S. Fernandes, 'A software engineering perspective on SDN programmability', *IEEE Commun. Surv. Tutor.*, vol. 18, no. 2, pp. 1255–1272, 2015.

[13] C. Trois, M. D. Del Fabro, L. C. de Bona, and M. Martinello, 'A survey on SDN programming languages: Toward a taxonomy', *IEEE Commun. Surv. Tutor.*, vol. 18, no. 4, pp. 2687–2712, 2016.

[14] J. Zhao, Z. Hu, B. Xiong, and M. Zheng, 'Performance Modelling and Optimization of Controller Cluster Deployments in Software-Defined WAN', in *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, Zhangjiajie, China, Aug. 2019, pp. 106–113. doi: 10.1109/HPCC/SmartCity/DSS.2019.00030.

[15] 'Gartner Dell - Transforming Networking for the Cloud Era with SD-WAN'. Dell Technologies, 2020. [Online]. Available: https://www.delltechnologies.com/asset/en-eg/products/networking/briefs-summaries/gartner-dell-newsletter-transforming-networking-for-the-cloud-era-with-sd-wan.pdf

[16] P. Kokkinos, D. Kalogeras, A. Levin, and E. Varvarigos, 'Survey: Live Migration and Disaster Recovery over Long-Distance Networks', *ACM Comput. Surv.*, vol. 49, no. 2, pp. 1–36, Nov. 2016, doi: 10.1145/2940295.

[17] N. L. M. van Adrichem, C. Doerr, and F. A. Kuipers, 'OpenNetMon: Network monitoring in OpenFlow Software-Defined Networks', in *2014 IEEE Network Operations and Management Symposium (NOMS)*, Krakow, Poland, May 2014, pp. 1–8. doi: 10.1109/NOMS.2014.6838228.

[18] S. Jain *et al.*, 'B4: Experience with a globally-deployed software defined WAN', *ACM SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 3–14, 2013.

[19] C.-Y. Hong *et al.*, 'Achieving high utilization with software-driven WAN', in *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, 2013, pp. 15–26.

[20] Y.-J. Kim, J. E. Simsarian, and M. Thottan, 'Software-defined traffic load balancing for cost-effective data center interconnection service', in *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, Lisbon, Portugal, May 2017, pp. 255–262. doi: 10.23919/INM.2017.7987287.

[21] R. E. Mora-Huiracocha, P. L. Gallegos-Segovia, P. E. Vintimilla-Tapia, J. F. Bravo-Torres, E. J. Cedillo-Elias, and V. M. Larios-Rosillo, 'Implementation of a SD-WAN for the interconnection of two software defined data centers', in *2019 IEEE Colombian Conference on Communications and Computing (COLCOM)*, Barranquilla, Colombia, Jun. 2019, pp. 1–6. doi: 10.1109/ColComCon.2019.8809153.

[22] S. Troia, L. M. M. Zorello, A. J. Maralit, and G. Maier, 'SD-WAN: An Open-Source Implementation for Enterprise Networking Services', in *2020 22nd International Conference on Transparent Optical Networks (ICTON)*, Bari, Italy, Jul. 2020, pp. 1–4. doi: 10.1109/ICTON51198.2020.9203058.

[23] S. Troia, L. M. M. Zorello, and G. Maier, 'SD-WAN: how the control of the network can be shifted from core to edge', in *2021 International Conference on Optical Network Design and Modeling (ONDM)*, 2021, pp. 1–3.

[24] V. Malagar, M. Kumar, S. M. V. Devi, and S. Gupta, 'Performance Evaluation of VANETs for Evaluating Node Stability in Dynamic Scenarios', *Int J Comput Appl Technol Res*, vol. 7, no. 9, pp. 376–385, 2018.

[25] A. Al-Najjar, S. Layeghy, and M. Portmann, 'Pushing SDN to the end-host, network load balancing using OpenFlow', in *2016 IEEE international conference on pervasive computing and communication workshops (percom workshops)*, 2016, pp. 1–6.