Optimal Multi-Tenant Cloud Network Latency for Inter-Device Caching

Nader F. Mir
Dept. of Electrical Engineering
San Jose State University, California
San Jose, CA, 95192, USA
email: nader.mir@sjsu.edu

Pranav S. Paramel
Dept. of Electrical Engineering
San Jose State University, California
San Jose, CA, 95192, USA

Sandeep K. Sethumadhavan
Dept. of Electrical Engineering
San Jose State University, California
San Jose, CA, 95192, USA

***Abstract*—This paper presents an analysis on the latency over internal caching and computing using a multiple tenancy (multi-tenancy) structure of cloud networks. Enterprises accelerate their adoption of network virtualization leading to the deployment of multi-tenancy when Software-Defined Networking controls the clouds network. With this in mind, we considered in this paper the distributed storage and storage access that present everyone's challenges in the computer network domain. Ensuring availability and low latency is an important challenge that needs to be addressed in this scenario. In our analysis network testbeds, we deployed multi-tenant users, by using Virtual Local Area Networks (VLANs) in the structure of data centers, we notice communication latencies may vary. Our attempt is to catch the optimal latency that offers the best level of efficiency to clients who look for better support for growth. The end benefit out of this article's result helps plan and implement fault-tolerant and low latency networks.**

Keywords-Cloud Computing; Internal Caching; Multi-tenancy Virtualization; Software Defined Networking (SDN); VLAN; VxLAN, Mininet*; *Inter-VLAN Routing; Virtualization Technique.

I. INTRODUCTION

Cloud data centers handle and store large datasets, and there is a time lag between the collection and processing of data [1][2]. Cloud platforms are three main types of cloud hosting: private cloud, public cloud, and hybrid cloud. Private cloud models are solely dedicated to a single organization, whereas in the public cloud model, services are shared across the organizations over the public internet. A hybrid cloud is a combination of both private and public clouds. Virtualization in cloud systems enables cloud providers to make maximum use of their data center resources. Hardware resources can be utilized efficiently through the technique of virtualization and hence reduces the Internet Technology (IT) expenses in organizations [3]. As needed, computing resources are dynamically allocated for each virtual machine by a software called a hypervisor that separates out a Virtual Machine (VM) from the actual host [4].

Data centers with multi-tenancy facilities where enterprises can rent space to host their applications and data. Multi-tenancy provides the space for physical hardware and networking equipment to connect an organization to service providers at a minimal cost [5][6]. The ability to scale on-demand provides the business benefits of a data center without the high price. The advantage of outsourcing the resources to a multi-tenancy is that it helps in the rapid deployment of applications with maximum flexibility [7]. Strict isolation between tenants is required in a multi-tenant network environment since they share the same physical [8].

The challenge being faced by researchers is how expansive a multi-tenant network can be in cloud systems in order to experience a considerably low latency when clients of multi-tenant network are active [9]. This paper presents an analysis to tackle this challenge.

The rest of this paper is organized as follows. In Section II, we present testbed network architectures for various cloud scenarios. In Section III, we present some analytical results of our analysis, and in Section IV, a conclusive statement is presented.

II. TESTBED NETWORK ARCHITECTURES

Our testbed Data Center Network (DCN) includes two layers of switches called *core switches*, and *Top-of-Rack* (ToR) (or *access* or *edge*) *switches*. Access switches directly connect to end servers at server racks, and core switches directly connect to the Internet. The aggregate switches are the middle layer switches. Packets from outside find their destination server by traversing the access switches and server racks. The core layer switches in a data center network enforce network security and load balancing. In our analysis, we developed three types of multi-tenant architectures: (i) *multi-tenancy with a single multi-tenant database* - a single application and database; (ii) *multi-tenancy with a single multi-tenant database* - an individual database for each tenant, sharing a single application instance; and (iii) *standalone single-tenant application with single-tenant database* - each tenant has its own application instance and database instance.

In all testbeds used in this paper, we created multi-tenancy using virtual LANs (VLANs). Since VLANs are logical entities, they are adaptable in network management, administration, and reconfiguration. Trunking enables to carry information from multiple VLANs over a single link between the switches. VLAN in our study is a tool that acts as a separate physical switch, and the hosts on different VLANs cannot communicate with each other. Inter-VLAN routing refers to a routing technique where the network traffic can be routed between different VLANs.

Copyright (c) IARIA, 2023. ISBN: 978-1-61208-993-5

Virtual Extensible LAN (VxLAN) is another factor we used in our study. VxLAN is a network virtualization technology that encapsulates MAC-based Layer 2 Ethernet frames within Layer 3 UDP packets. VxLAN tunnel endpoint (VTEP) that lies inside the hypervisor hosts is responsible for encapsulating and decapsulating packets. The multi-tenant architecture was built using two sets of tools:

1) **VMWare/Mininet**. This first setup is entirely on virtual machines that host Mininet and controller to establish a design of four tennants each having four virtual machines (VMs). We then created a topology using Python script. Nodes used in topology are OpenFlow switches and servers (hosts).
2) **GNS3**. The second setup is entirely on GNS3 tool with Open vSwitches and hosts, supporting networking utilities like Iperf and ping. Each virtual machine in this setup is refereed to as a PC.

Thus, in this paper, VM and PC are referred to interchangeably. To capture the performance of the edge caching/computing, we required Wireshark which is a cross-platform network protocol analyzer that lets analyze network traffic. It is often used for troubleshooting network issues as well and is a similar to tcpdump but with a graphical user interface. The application can inspect hundreds of protocols in real-time. Wireshark capture files can help us troubleshoot various issues involving latency issues, dropped packets, and other malicious activities on your network. Wireshark captures binary traffic on the links and converts that into a human-readable format with colored highlighting. Wireshark became the standard tool for packet analysis.

Figure 1 is a small-scale data center topology ises as a testbed in this paper. It is built using Mininet/VMWare and GNS3. The multi-tenancy architecture presented in this paper consists of an SDN controller, a core switch (S1), two aggregate switches (S2 & S3), and four access layer switches (S4, S5, S6 and S7). There are two access switches lying on each of the servers. The SDN controller is connected to the core and aggregation layer switches. These switches are in turn connected to VMs (PCs) deployed on these servers resulting in a total of 12 VM hosts.

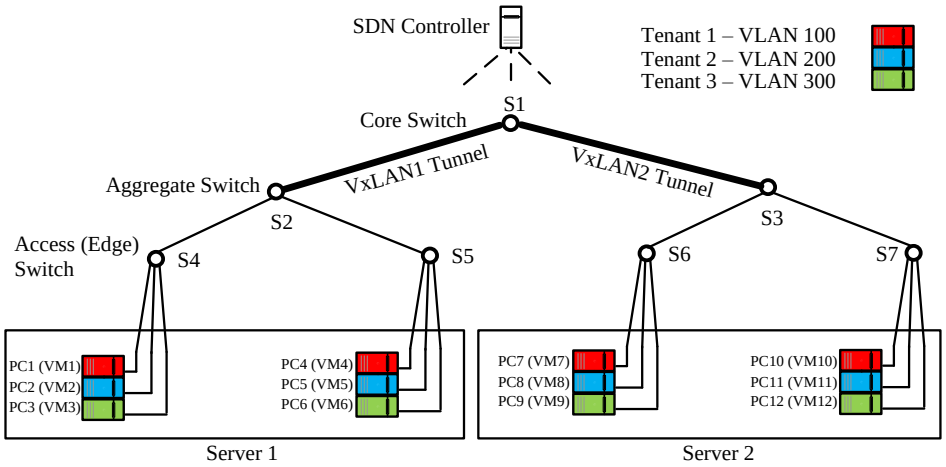

Figure 1. A multi-tenancy architecture used in both 1) VMWare/Mininet and 2) GNS3 tools.

Two VxLAN tunnels (VxLAN1 and VxLAN2) are created between the core switch and aggregate switches to execute the topology. All VMs lying on the same server (Server1 or Server 2) can communicate with each other. In addition, VMs in one server can communicate with the VMs in the other server through the tunnels created between the aggregate switches and the core switch. VMs are tagged to group them into multiple tenants, restricting them only to communicate between the VMs belonging to the same VLAN. In this architecture, tenants 1, 2 and 3, are created with the VLAN tags 100, 200 and 300, respectively.

When this topology was created in GNS3 to execute inter-VLAN routing among the VMs, in the earlier testbed using Mininet, the VMs belonging to the same VLAN ID was only able to communicate with each other. All the ports of the access layer switches connected to each host were tagged with corresponding VLAN IDs. PCs belonging to each VLAN were configured to separate subnets as listed:

PC1, PC4, PC7, PC10 in VLAN100 - 10.0.10.1/24

PC2, PC5, PC8, PC11 in VLAN200 - 10.0.20.2/24

PC3, PC6, PC9, PC12 in VLAN300 - 10.0.30.3/24

The ports connecting the access layer switches and aggregation switches were configured as Trunk Ports. Similarly, configured the ports connecting the core switch and aggregation switches as Trunk Ports. By adding IP addresses to the VLAN interfaces created on access layer switches and enabling IPv4 packet forwarding between interfaces on the switch consoles, all the VMs were able to ping each other irrespective of their VLAN IDs/subnets.

Copyright (c) IARIA, 2023. ISBN: 978-1-61208-993-5

III. Analysis And Results

Our objective is to compare the effects of parameters like network latency, throughput, and jitter of the multi-tenancy topology in various scenarios. We consider both intra-VLAN communications and inter-VLAN communications.

A. Scenario I, Intra-VLAN Communication

In this scenario, the effects of packet traffic with the same VLAN are measured. The traffic is analyzed using Wireshark to capture the packets on the links between the nodes in the network. Various parameters like latency and throughput in the network are considered to deduce a relation with the network topology.

In order to find the latency in the traffic, ping tests were conducted for 10 seconds by giving a small-time gap between successive cases. In Figure 2, we can see that the latency is minimum when PC1 pings PC2 (assuming PC2 in this particular case belongs to VLAN1). But when the PC1 pings PC4 while these hosts are connected to the same aggregate switch S2, the latency increases significantly to the average 4.70 s. Here, hosts are at a minimal distance from each other. When PC1 pings either PC7 or PC10, which are farthest from the source, the traffic must traverse through the core switch. Hence, we have an increased hop count that results in greater latency with the averages 6.9 and 7.45 seconds.

The throughput in intra-VLAN communication scenario in VLAN 300 as an example is calculated. Therefore, PC3 acts as the client, and PC6, PC9, and PC12 act as servers. A TCP connection is established between each of the client-server combinations. The Iperf client uses a random port number to connect to the Iperf server on the TCP port 5201. The tests were conducted for 10 seconds by giving a small-time gap between successive cases. Throughput during multiple inter-VLAN communication is also measured as we set: PC1, PC4, PC7, and PC10 where PC1 and PC4 act as clients, and PC7, and PC10 act as servers. As shown in Figure 3, during interval A, only PC1 sends the TCP traffic to PC10. High throughput is observed during this time. During interval B, both PC1 and PC4 send the TCP traffic to PC7 and PC10, respectively. Since both of this traffic elements utilize the available bandwidth, the effective throughput decreases for both the traffic. During interval C, PC1 stops sending traffic to PC7, whereas PC4 continues to send traffic to PC10. Hence, the throughput between PC2 and PC11 again increases.

B. Scenario II, Inter-VLAN Communication

In this scenario, effects of packet traffic between different VLANs are measured. PC1, PC3, PC5, PC9, and PC12 were used to calculate the latency during multiple inter-VLAN communications. The tests were conducted for 10 seconds by giving a small-time gap between successive cases. As seen in Figure 4, the latency becomes minimum when PC1 pings PC3, hosts connected to the same access layer switch S4. When PC1 pings PC5, the traffic traverses through the aggregate switch S2, and thereby latency is found to be higher. When PC1 pings either PC9 or PC12 connected to a different aggregate layer switch S3, the traffic must traverse through the core switch, and a considerable increase in latency is observed.

To measure the latency during multiple inter-VLAN communication, we deployed caching applied to PC1, PC2, PC3, PC10, PC11, and PC12 considered with results shown in Figure 5. During interval A, there are three sets of traffic going through the network. Both PC1 and PC2 sending TCP traffic to PC10 and PC11, respectively. At the same time, ping requests were sent from PC3 to PC12. Therefore, in this interval, maximum latency is experienced, as shown by the bar graph. During interval B, PC1 stops sending traffic to PC10 (upper plot), whereas PC2 continues sending traffic to PC11 (red plot). Hence, latency slightly decreases. During interval C, we have no TCP traffic going on in the network, but only ICMP traffics between PC3 and PC12. Therefore, the latency further decreases.

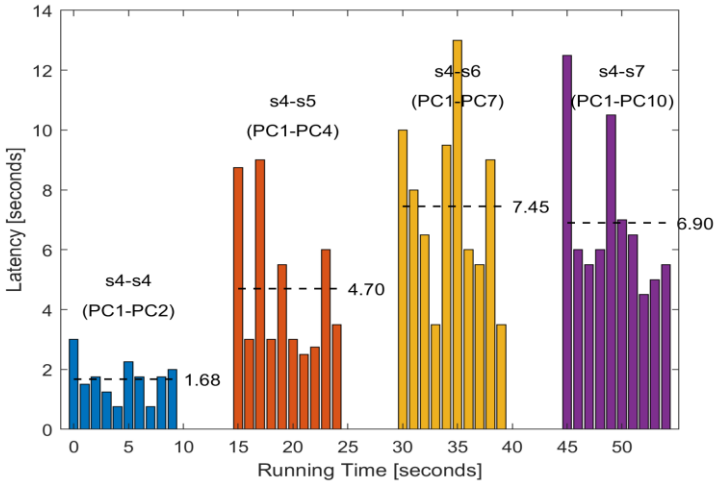

Figure 2. Latency analysis for the intra-VLAN communications

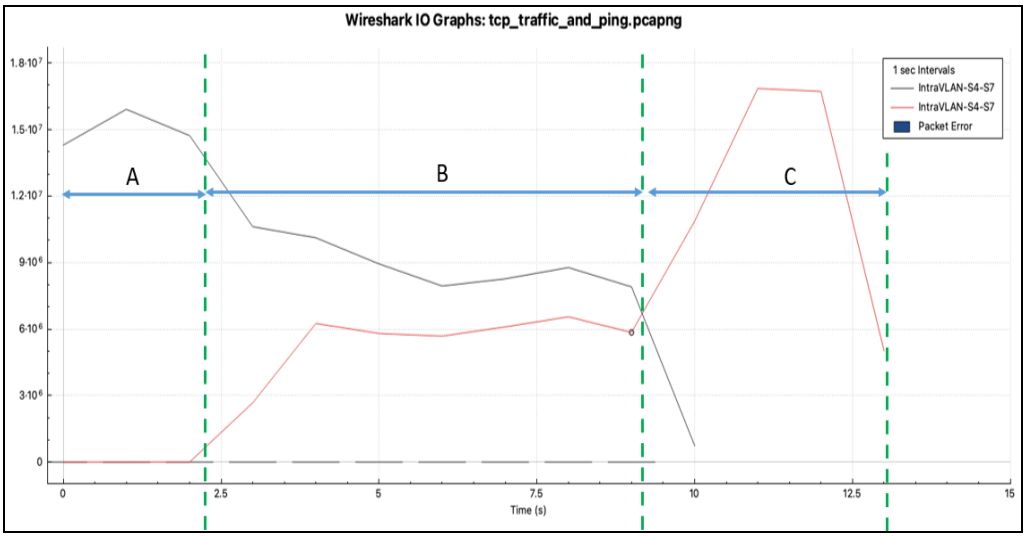

Figure 3. Throughput during multiple traffic in the network

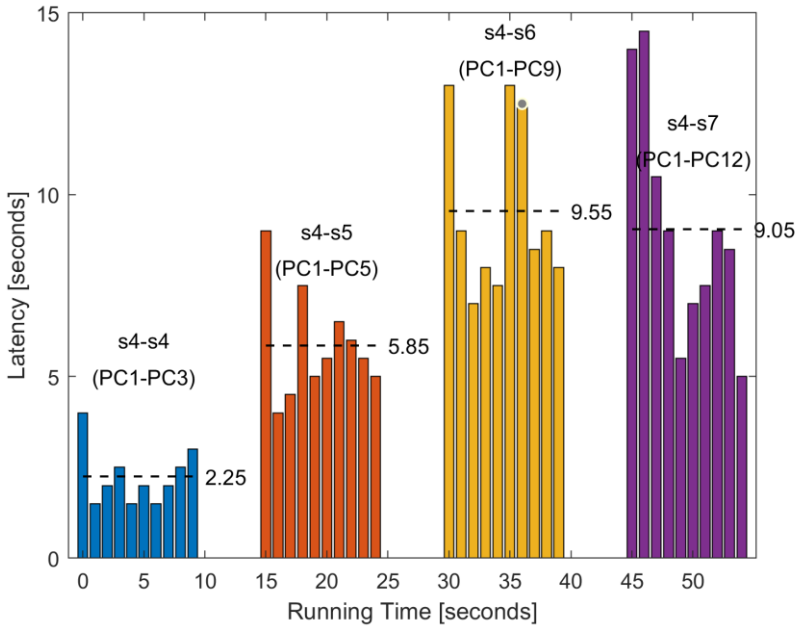

Figure 4. Latency in inter-VLAN communications

To analyze the throughput in the inter-VLAN communication, PC1, PC3, PC5, PC9, and PC12 are considered. PC1 acts as the client, and the rest act as servers. The maximum bandwidth is utilized when PC1 sends traffic to PC3, hosts connected to the same access layer switch. The graph shows that the throughput decreases drastically when the traffic traverses through aggregate/core layer switches.

PC1, PC2, PC6, PC8, and PC12 are considered to conduct this test. PC1 acts as the client, and PC2, PC6, PC8, PC12 as the server. This time UDP packets were sent between the client and servers. From Figure 6, it is observed that the jitter is very low when PC1 sends traffic to PC2, which is connected to the same access layer switch S4. When PC1 sends traffic to PC6 connected to the access switch S5, the packet travels over the aggregate switch, increasing the hop count. As a result, the jitter increases. When PC1 sends traffic either to PC8 or PC12, there is a considerable difference in jitter compared to that of the traffic between hosts lying under the same access layer switch.

Copyright (c) IARIA, 2023. ISBN: 978-1-61208-993-5

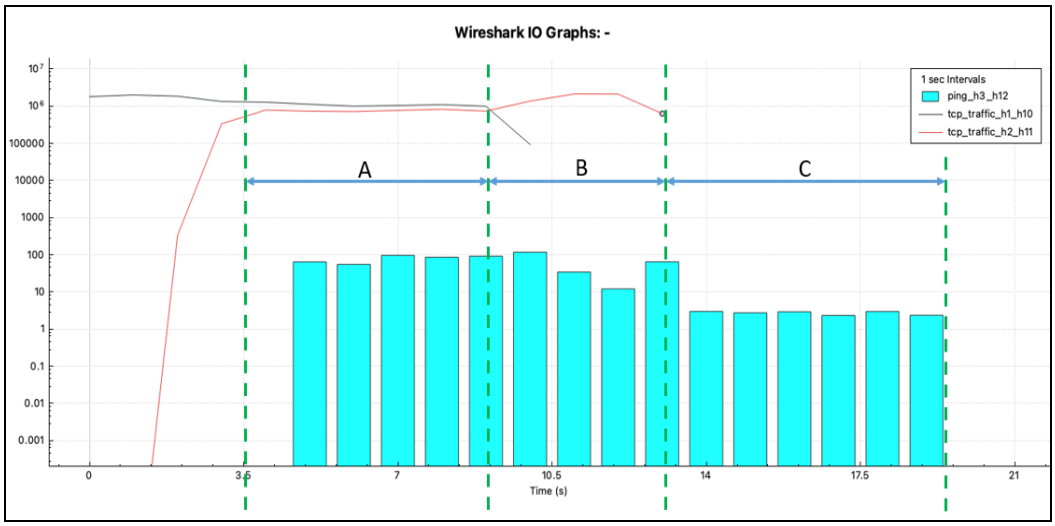

Figure 5. Latency during multiple caching traffic in the network

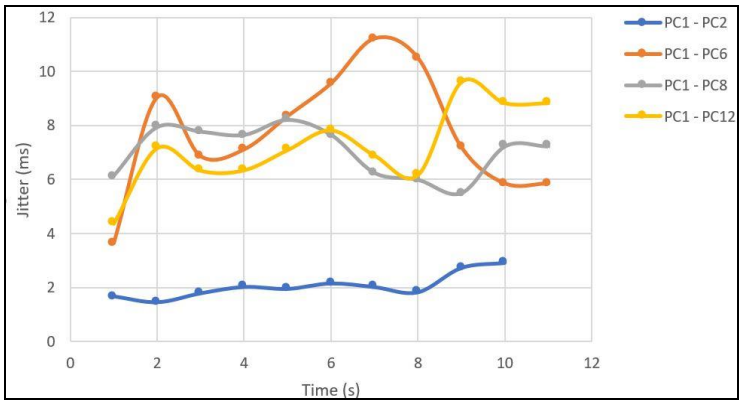

Figure 6. Jitter during multiple caching traffics in the network.

Our study shows that if we could temporarily move VM7 (PC7) of Server 2 to Server 1 for use, then in spite of a processing time for this move, the benefits are noticeable. As observed in Table 1, with this move, the latency average is improved from 7.45s to 6.35s and throughput is increased from 3.5 Mb/s to 4.75 Mb/s.

TABLE 1. THE IMPACT OF MOVING VM (PC) FROM OUTSIDE OF VLAN TO INSIDE.

	PC1-PC7 (Intra-VLAN)	**PC1-PC7 (If PC7 is moved to Server1)**	**PC1-PC9 (Inter-VLAN)**
Average Latency	7.45 s	6.35 s	9.55 s
Average Throughput	3.5 Mb/s	4.75 Mb/s	1.75 Mb/s

IV. CONCLUSION

The paper initially presented the construction of a testbed network for a small-scale data center using caching. As the first step of the analytical process, we created a basic topology with multiple VMs running on top of the hypervisor. To enable communication between the caching hosts on different VMs, the VxLAN tunneling technique was adopted. Then multiple tenants were created by assigning the hosts to different VLANs. Hosts belonging to a same VLAN were able to communicate with each other. Commands to enable Inter-VLAN routing was configured on the Open vSwitches. The findings from both the Wireshark capture and simulation results, multiple graphs showing how the inter/intra VLAN communication affects the throughput, latency, and jitter in a DCN were plotted. It is found that these parameters depend on the number of hop-counts during the network traffic.

Copyright (c) IARIA, 2023. ISBN: 978-1-61208-993-5

REFERENCES

[1] "Virtualization Technology & Virtual Machine Software: What is Virtualization?" VMware, 16-Sep-2020, Available from: https://www.vmware.com/solutions/virtualization.html, Aug. 2022.

[2] "Toward Performance Optimization with CP Offloading for Virtualized Multi-Tenant Data Center Networks," Available from: http://libaccess.sjlibrary.org/login?url=https://ieeexplore.ieee.org/document/5507957, May/June 2022.

[3] "SaaS: Single Tenant vs Multi-Tenant - What's the Difference?" Available from: https://digitalguardian.com/blog/saas-single-tenant-vs-multi-tenant-whats-difference, July, 2022.

[4] "Getting Started with a Multi-tenant Application on Node.js," Available from: https://blog.lftechnology.com/designing-a-secure-and-scalable-multi- tenant-application-on-node-js-15ae13dda778, Jan, 2022.

[5] M. F. Bari et al., "Data Center Network Virtualization: A Survey," IEEE Communications Surveys and Tutorials, vol. 15, no. 2, pp. 909-928, Second Quarter 2013.

[6] "What is VXLAN?" Juniper Networks, Available from: https://www.juniper.net/us/en/products-services/what-is/vxlan/. Oct 2021.

[7] N. M. Mosharaf, K. Chowdhury and R. Boutaba, "Network virtualization: state of the art and research challenges," in IEEE Communications Magazine, vol. 47, no. 7, pp. 20-26, July 2009.

[8] "NVGRE vs VXLAN: What's the Difference?" Available from: https://community.fs.com/blog/nvgre-vs-vxlan-whats-the-difference.html, Jan. 2020.

[9] "SaaS: Single Tenant vs Multi-Tenant - What's the Difference?" Available from: https://digitalguardian.com/blog/saas-single-tenant-vs-multi-tenant-whats-difference, July 2022.

Copyright (c) IARIA, 2023. ISBN: 978-1-61208-993-5