Abstract—With the growth of the Internet, virtualization is an important component in hyper-scale data centers to achieve fast deployment and scalability with small time-to-market values. Virtualization is a software representation of the physical hardware. This creates an abstraction of the underlying hardware components and simplifies the process of network functions deployment and network management using software technologies. Network virtualization relies on similar principles to implement multiple network functions in software. Deployment of network virtualization requires capital investment and operational expertise to enable micro-segmentation of networks and most importantly, avoid hair-pinning in data center networks. It reduces the OPEX required to deploy and maintain large-scale virtual networks, but this approach results in overhead due to multiple layers of encapsulation and thus reducing the performance and causing more resource utilization of the underlying network. This research paper focuses on implementing various tests with different virtualization technologies to observe and record the trade-offs between the advantages of virtualization and the network overheads. Furthermore, it also analyses the encapsulation layers added by different network virtualization technologies such as VXLAN, GRE, SSL VPN, etc. in different networking settings such as an inter-server and an intra-server topology. The overhead per packet is studied in both the traditional and the virtualized network models to obtain a clear perspective on the difference between the two implementations.

Index Terms – Virtual network, overhead, VMware NSX, OpenStack, VXLAN, GRE, SSL VPN

TABLE OF CONTENTS
1. INTRODUCTION……………………………………1
2. RESEARCH QUESTION…………………………2
3. RELATED WORK…………………………………2
4. TEST SYSTEM SETUP AND DESCRIPTION……2
5. TEST RESULTS……………………………………4
6. CONCLUSIONS……………………………………5
7. FUTURE WORK……………………………………6

1. INTRODUCTION

The success of server virtualization in the past few years has led to the adoption of virtualization in the networking domain. We live [1] in a world where this is becoming a requirement to achieve large-scale cost benefits. Many of the telecom service providers such as AT&T, NTT, etc. [2] have already deployed network virtualization architecture in their data centers, whereas others are in the process of implementing the same. This technology creates network service components in a software container and this process is analogous to server virtualization wherein the traits of a physical server(s) are recreated in a software.

In virtual networks, the functionality of the hardware is separated and is simulated as logical instances [3]. The advantage of such networks is that a single hardware platform can be utilized to support multiple logical network functions (VNFs). These VNFs can be spun up or down easily as per one’s needs. This reduces network provisioning time and makes the network more scalable and portable. Additionally, the
workloads can be placed and moved independently of the physical topology. In a survey conducted by SDxCentral [4], 77% of respondents felt that network virtualization was important for flexibility purposes, 68% for scalability, 52% for operational cost saving and 31% for capital cost saving.

2. RESEARCH QUESTION

For exchanging traffic between two virtual machines (VMs) in different networks, the packet must traverse via the physical adapter of the host to the switch and ultimately to the physical router for getting a routing decision. The packet must exit the host to go to the physical network and reach the destination VM after getting a routing decision. This non-optimal flow of traffic is referred to as “hair-pinning” [5][6]. Network virtualization eliminates the use of a physical router to perform the routing functionality and it uses a logical router that performs routing between the various logical interfaces consisting of multiple networks. But, this process requires extra layers of processing. The question that then arises is to what extent virtualization performs well with virtual machines hosting different applications [7].

Figure 2 – Hair-Pinning in traditional and virtual network

The important research question that our paper seeks to answer is “How much overhead is introduced in virtual networks due to additional encapsulation layers created using virtualization technologies and how much financial impact is caused by the network overheads? Our research work focuses on how extra encapsulation layers are directly influencing the cost required to maintain the network required to handle such overhead containing data packets. We implement different applications such as VoIP & video with network virtualization technologies such as VXLAN, SSL VPN, etc. to understand the actual overhead?”

3. RELATED WORK

Research shows that virtualization simplifies networks and creates cost-effective solutions but at the cost of additional overheads due to the increase in the layers of encapsulation and thus increasing the packet size. Since the maximum size is limited by the MTU of the routers between the source and the destination, the size of the data payload is reduced for a single transmission and thus requires multiple transmissions to transfer the complete data. Our research focuses on the packet overhead or inefficiencies created by virtualization whereas most of the research currently available is focused on measuring the CPU and memory utilization overheads created by virtualization technologies. Similar research work performed by Menon et al. [15] measured the degradation of the performance for network I/O virtualization. This research analyses the transmit and the receiving bandwidth to measure the overhead in processing the packets [15]. Gulati et al. [14] measured the workload characteristics and performance to analyze the overhead in a consolidated environment. These research works show the performance overhead added due to the shift from traditional to a virtualized networking environment.

Network virtualization is used to implement multiple virtual machines sharing common physical infrastructure. All the virtual machines share the same I/O interface present on the common underlay infrastructure. The logical networking may cause some issues to latency sensitive applications due to packet loss and performance degradation when compared to native or traditional networking [17]. A research project at VMware was performed to identify the overhead using the response-time for a particular request. Additional layers of encapsulation due to virtualization increase the processing time of a packet thus increasing the total time. Overhead is classified into constant overhead and variable overhead. Constant overhead is caused by network virtualization, I/O, and CPU while variable overhead is caused by Memory Management Unit [7].

Another research work [16] evaluates the performance of various type-1 and type-2 hypervisors such as VMware ESXi, QEMU, etc. This research paper analyses the CPU, bandwidth, and I/O operation after implementing virtualization. The above research work focuses on performance overheads created by virtualization and less on packet overhead due to multiple layers of encapsulation created as a result of virtualization.

The CLONER project focuses on evaluating packet inefficiencies resulting in more packet transmissions due to MTU restriction and comparing them to the traditional network model. Different combinations of encapsulation technologies will be used to understand the inefficiency and the encapsulation technology. Applications such as voice, video, etc. will be used to generate traffic and to simulate real-world implementation.

4. TEST SYSTEM SETUP AND DESCRIPTION

This research project begins with the selection of an appropriate network virtualization platform. VMware vSphere and OpenStack were selected out of all the available platforms in the market due to features such as compatibility with various network overlay features and good documentation.
A. VMware vSphere

VMware’s network virtualization platform, NSX, forms a key product in a Software Defined Data Center (SDDC) [8]. It can programmatically create, manage, snapshot, delete and restore virtual networks that are software based. All the network functionalities ranging from Layer 2 to Layer 7 such as routing, switching, access control list, QoS, firewalling, VPNs, etc. are reproduced in a software in NSX. VMware vSphere 6 Enterprise plus package was selected for compatibility purposes by making use of VMware Product Interoperability Matrices. It includes vCenter Server, vSphere Web Client, and the VMware ESX.

The VMware virtualization architecture is built by deploying ESXi on two separate servers. ESXi is VMware’s server virtualization platform which enables abstraction of physical hardware resources so that multiple virtual machines can reside on a single server. An instance of vCenter Server Appliance (VCSA) is deployed which resides on one of the ESXi servers. The VCSA is an orchestration platform to create, modify and manage the resources of multiple ESXi servers from a single web interface. The VCSA enables the creation of VMs, allocating resources, deploying virtual overlay switches and most importantly, the NSX virtualization platform.

A cluster is created by placing the ESXi hosts into one application domain after which multiple VMs are added on them. A virtual Distributed Switch (vDS) is created to logically separate the VMs into different broadcast domains. The vDS has several features such as configuring which physical network adapters are used for traffic communication, configuring VLAN tags, traffic failover & load balancing, permitting packet sniffing, etc. The management traffic and data traffic for VMs is placed on different broadcast domains - called as ‘port groups’ in vDS.

Some of the benefits of the VDS are provisioning east-west and north-south communication by maintaining isolation between tenants, reducing the use of VLAN IDs, the creation of flexible L2 overlay over existing IP networks, support to network overlay features such as VXLAN and other features like port mirroring, QoS, etc. For every host cluster participating in NSX, it was verified that hosts within a cluster are attached to a common VDS.

Finally, NSX was deployed where an NSX manager instance maintains information about all hosts with regard to logical switching, logical routing and logical firewalling. Three controller nodes that form a control cluster were deployed for redundancy purposes as per VMware’s recommendation [11]. Tests to exchange packets between two VMs on two different ESX hosts were performed. The results from these experimental tests observed some overhead due to network virtualization technologies which are explained later in Section 5.

B. OpenStack

The basic building blocks of OpenStack are Neutron, the networking component, Keystone, the identity management component, Glance, the image component, Nova, the compute component, Cinder, the block storage management component, Horizon, the web interface component, Swift, the object storage management component, Heat, the orchestration component and Ceilometer, the telemetry component. The KeyStone component provides authentication and identity services for users in OpenStack. Nova is the Compute component which provides on-demand access to computing resources. The image management component, Glance acts as an image repository for different operating systems (images for virtual machines). The networking component, Neutron is responsible to manage all the virtual networking services. Neutron makes use of a plugin known as ml2 to configure type drivers and network types needed in the OpenStack environment. All the instances managed by Nova require Neutron even for basic networking services such as obtaining an IPv4 or IPv6 address. Cinder and Swift come with APIs and provide storage for the end users.

OpenStack can be installed in a computing environment using DevStack, PackStack or Ansible. In this project, OpenStack was deployed using Devstack, which is basically a series of scripts grouped into a single package. DevStack installation requires a minimum of 6 GB RAM, 40 GB disk space and connectivity to the internet (using a NAT network adapter). With DevStack, OpenStack architectural components like
Neutron, Glance, Keystone, Nova, Horizon, and Cinder can be easily installed and configured.

In our project setup, two Dell servers with Ubuntu 16.04 LTS operating system were used. OpenStack multi-node architecture was deployed since most organizations use this kind of setup in their data centers. One server was configured to be OpenStack Controller/Compute node (Server 1) and the other to be just a Compute node (Server 2). The servers played the role of hypervisors for OpenStack instances (which were created later). Server 1 served the role of Control, Compute, Network and Storage while Server 2 had the role of Compute. In the DevStack configuration file (local.conf) of Server 2, Controller node's IPv4 address was specified so that it would use Server 1 as its Service host. The Nova service in Compute nodes has a feature called cells which is used to arrange the compute nodes (along with a database and a queue) into smaller groups. A multi-node deployment with one or more Compute nodes requires compute services to be mapped to a cell. After Server 2 was stacked, its existence and proper configuration was ensured using the command `nova service-list` --binary nova-compute. All compute hosts were mapped to a single cell using `tools/discover_hosts.sh` script (which is a part of DevStack package). An external plugin known as TAP as a Service (TAPaaS) was installed to enable port mirroring.

The OpenStack dashboard could be accessed using the IPv4 address of the server 1.

5. TEST RESULTS

VXLAN
VXLAN is used to create an overlay virtualized layer 2 networks on top of the hardware underlay network. VXLAN is also a good tunneling mechanism to communicate between physical and virtual machines where the physical machines will terminate a VXLAN and will be capable of sending and receiving VXLAN tagged packets from the virtual hosts. When virtual hosts add VXLAN tags, an extra encapsulation layer is added which will then increase the packet size and if the packet size is more than the MTU of the routers between the source and the destination then the actual packet would be split into smaller sized packets to avoid fragmentation in the network which would cause additional problems such as more retransmissions. VXLAN technology is responsible for adding overhead and this technology is used in our project to understand the effect of multiple overheads on the performance of the network.

GRE and VPN Tunnels
Tunnels can be used extend a network and connect isolated locations. They encapsulate the original packet with an external layer to abstract the original packet protocol and to carry the packet through intermediary networks which do not understand the protocol of the original packet. When tunnels are implemented, the extra encapsulation layer will increase the packet size thus creating packet overhead.

Packet Capture and Sample Analysis
An OpenStack setup with VMs in two servers was used for a sample overhead analysis. The payload size was varied for each packet and the overhead associated with payload per packet was determined. The experiment was conducted with VXLAN and without VXLAN and the percentage of overhead was plotted against the payload.

The setup used for the experiment is shown below. Virtual machines on the same subnet were created. Using proper host-to-node mapping the VMs were made to reside both in OpenStack server 1 and OpenStack server 2. SPAN (Switch Port Analyzer) was configured on the physical switch between the servers to capture packets transferred between the VMs in the servers. This helped us to see the VXLAN headers added by the overlay tunnel endpoints. TAP as a Service was used to analyze traffic between VMs on the same server. TAPaaS enabled this by mirroring the traffic to a port represented by an object called Tap Service. The ports on the virtual switch where the VMs were connected (the source ports) were mapped to an object called TapFlow.
The packet captures corresponding to the setup with VXLAN and the setup without VXLAN are shown below.

A similar setup in VMware vSphere environment yielded the same results.

The headers found in each case (with and without VXLAN) is depicted below. Here the IPv4 addresses given are specific to the OpenStack setup.

The percentage of overhead per packet for data specific to ICMP connectivity was plotted. Here the difference in overhead for small payloads was found to be significant.

6. CONCLUSIONS

In this project, we have conducted various tests to compare traffic patterns from several kinds of applications running in a virtualized architecture versus a traditional environment. The project has enabled us to make a cost-benefit analysis to evaluate if the benefits provided by virtualization compensate for the high capital investment and high operational complexity of running the virtualized architecture.

We also show that virtualization overhead can cause more resource utilization from the underlying physical device than actually necessary.

Each network in large organizations has a different architecture and traffic patterns depending on the applications. This research work enables network architects in organizations to understand the details of traffic in virtual networks and the future application of the data as a starting point to evaluate the overheads in their networks. Our research analysis of cost to transition from traditional to virtual networks [12] compares two NFV virtualization models with a traditional appliance-based architecture. According to [12] three types of costs have to be considered: migration cost, project cost and education cost. Migration cost includes the cost incurred on replacing a traditional networking node with an NFV node [12] and this can be thought of as the Capex. Project cost includes money spent in setting up and making improvements on the virtual networking platform [12], which can also be termed as OPEX. Education cost is the cost of providing training [12] to the employees in the organization, to acquaint them with virtualization technologies. The paper [12] asserts that there are
only short-term benefits or cost savings with both the NFV virtualization models.

Further, the operational costs can be added which are in the form of human resource, that is, special engineers required to regularly maintain and modify the architecture as required [12]; and in the form of training which needs to be provided to engineers to help them understand how to use the platform to perform regular tasks [12]. The cost for this will also have to be taken into consideration.

7. FUTURE WORK

Next, this project will be expanded to take into consideration the emerging technologies in the networking field which will potentially be deployed on a wide-scale in the next ten years. Various organizations have started deployment of Internet Protocol version 6 (IPv6) and Software-Defined Networks.

8. ACKNOWLEDGEMENTS

For IPv6, the IP Address size is of 128 bits which four times that of IPv4. With this bigger size of address, the percentage of data to encapsulation decreases further even in traditional networks. IPv6 also has the capability to add optional sub headers if required. In a virtualized environment, the VXLAN header consists of an extra IP layer. With IPv6, that extra layer will be further bigger in size. When it is deployed in virtualized networks the amount of overhead will increase further thus reducing the channel utilization and protocol efficiency to a higher extent.

We extend our sincere thanks to our advisors Prof. Jose Santos for his detailed insights, guidance and providing us the required test hardware in the lab; and Mr. John Griffith for his advice throughout this research project. We also extend our gratitude to Dr. Kevin Gifford for his support at different stages of the project and also in getting an academic license for VMware’s suite of products.
REFERENCES


BIography

Rishid Adusumilli is a Network Engineering graduate student at University of Colorado Boulder. He has experience in large-scale system administration, network operations and network automation using scripting languages. His areas of interest include network engineering, operations, and automation.

Pratik Lotia is a Network Engineering graduate student at University of Colorado Boulder. He has experience in service provider networks, security analysis and incident response. He has a strong understanding of packet analysis and tools such as NetFlow and Protectwise. He has
received recognition from Lockheed Martin and Govt. of India.

Pranit Yadav is a Network Engineering graduate student at University of Colorado Boulder. He has experience in setting up an enterprise network architecture, Software-Defined Networking and cloud computing. His interests also include automation, troubleshooting enterprise and service provider networks and working as a system administrator.

Sruthi Sneha Jose is a Network Engineering graduate student at University of Colorado Boulder. Her areas of interests include traditional networking, cloud computing, Software-Defined Networking and network virtualization technologies. She has experience in server and desktop virtualization technologies, Python, and PowerShell scripting.