OpenStack Networking Technologies

Atharva Muraskar
Atharva.Muraskar33@gmail.com
Computer Engineering, Vishwaniketan iMEET, Khalapur, Maharashtra, India

Abstract-OpenStack is an open source cloud solution which aims at removing vendor locking by providing a virtualized environment in a production environment. OpenStack's networking module i.e. Neutron provides a centralized routing service, where L3 packets are redirected to a central network node. A single network node is incapable of handling overlapping IP addresses for multiple networks, which in turn, greatly decreases network bandwidth and throughput of production environments. This paper introduces two technologies that can alleviate the network performance issues as faced by Neutron. Furthermore, these two technologies namely OpenDaylight (ODL) and Distributed Virtual Routing (DVR) are then presented together with a set of benchmarks which showcase their performance in a production environment. The performance results show ODL and DVR out perform Neutron in every layer 3 case, making them as an upgrade for any traditional OpenStack based environment.

Index Terms— NFV, Cloud computing, Performance analysis, OpenStack, DVR, ODL

I. Introduction

Openstack provides a virtual layer on physical servers, decoupling underlying hardware from the workload and providing its users with virtualized computing, storage and networking resources. In OpenStack, networking component known as OpenStack Neutron, forward data packets internally and externally through networking components at the infrastructure level. This plugins-based, multi-tenancy supporting framework is argued to be technology-agnostic and modular. Comprised of elements such as Message Queue, DHCP Agent, L3 Agent etc., this service provides an API abstraction to allow cloud administrators to manage tenant/logical routers[2]. These logical artifacts can be used to connect logical networks to each other and to the outside world (e.g. the Internet), providing both security and NAT capabilities. Today, 84% of the Communication Service Providers rely on the capabilities of OpenStack services for efficient cloud management [1]. As the world is stepping into the realm of Network Function Virtualization, numerous telecom providers and enterprise leaders such as At&T, Bloomberg, Deutsche Telecom etc. are choosing to implement NFV with OpenStack. In NFV, dedicated network appliances such as routers, Domain Name Service (DNS) servers, and firewalls are replaced with virtual appliances running on commercial off-the-shelf servers (COTS). The advantage of using an NFV cloud is that service providers can more quickly roll out new services and VNFs using software rather than specialized hardware networks. The current efficiency provided by the routing capabilities of the established system do not go hand in hand with its said criticality. In order to alleviate these performance issues, two alternate strategies will be discussed. In the end there is a comparison between two technologies and advantages of using these technologies for a specific purpose.

II. Problem Formulation

Openstack that provides a centralized routing service[2], is well suited for a cloud management system but for NFV, it's not the best choice. Currently in Openstack, Neutron handles all the traffic of a virtual machine through a network

namespace router managed by L3 agent. Traffic from a virtual machine first passes through network node where the router resides. It uses Linux IP stack and iptables to perform L3 forwarding and NAT. Furthermore, a single router is incapable of handling overlapping IP addresses for multiple networks. To address this issue, the L3 agent of a network node configures routers in dedicated network namespaces to provide isolated packet forwarding. Having a centralized headquarter for handling traffic infuses latency in the networking process[2]. The logical architecture of OpenStack is shown in figure 1.For this paper, only those services are shown which are used in networking environment of Openstack.[2]

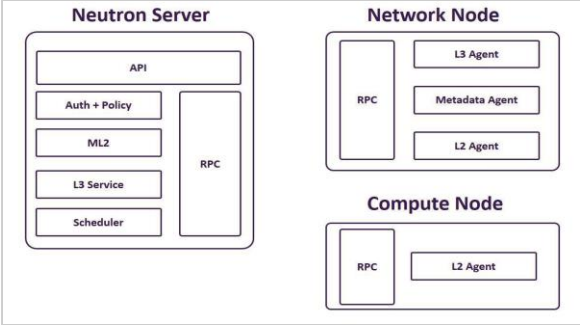


Figure 1: OpenStack Traditional Networking

VM traffic in Openstack is classified into four types end to end. As shown in figure 2:[6][7]

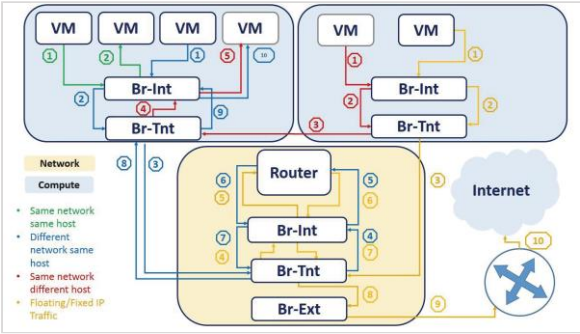


Figure 2: OpenStack VM traffic flow

Figure 2 shows different types of traffic as routed by the OpenStack Neutron. Each type of traffic is further explained below:

Same network same hostwhich is routed through br-int(integration bridge) and is directed to the specified VM in the same Network. **Different Network same host**different networks requires the vlan-tag to be changed, this change in tag is handled on the virtual router present in the network node. VM needs to make a connection with network node based virtual router for every communication. This exchange provides various degradation and bottlenecks during conversion to and from the physical link.

Same network different host-

VMs on different host, communication is done through Br-tnt(br-tenant) of both compute nodes. **Floating/Fixed IP traffic-** communication, VM traffic moves through the compute node and is sent to controller node. Controller node recives traffic through its br-tnt. Traffic moves to br-int where all the traffic converges and is moved to Router. Router changes the required fields and sends the traffic to br-ext(br-enternal) through br-int and br-tnt.

So in short, flaws of Openstack networking comes in the form of centralized Networking and its dependencies on multiple networking components for communication. Internetwork and Intranetwork VM traffic needs to be routed through the Network node. In production environment, Network node is deployed separately than compute node where VMs are run and becomes a bottleneck due to this centralized routing.This bottleneck leads to critical problems such as latency and performance degradation.

III. Alternate Strategies

To have a fully functional and robust application of NFV in an OpenStack environment, centralized L3 forwarding needs to be replaced with distributed routing. In a distributed model, routing capabilities that were previously held by the network node, are now allotted to all compute nodes. This strategy can be

implemented using the following two methods: 1-Neutron Distributed Virtual Routers.2-Neutron OpenDaylight.

Neutron DVR for North/South SNAT service while the North/South DNAT and East/West L3 forwarding capabilities have been distributed across all compute nodes. This distributed model bypasses the network node except in the case when VM needs to have access to the external network through a floating IP. Hence in order to do SNAT, the VM has to pass through the network/controller node where router for SNAT exists. If the compute node were to be assigned the SNAT service as well, there would be a high consumption of IPv4 external address space resulting in additional costs. The logical architecture of DVR is shown in figure 3.[2]

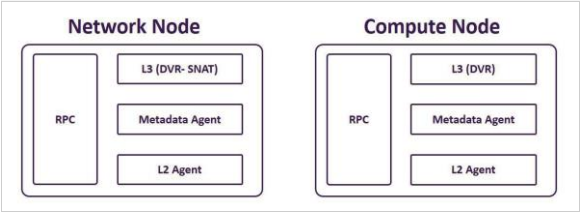


Figure 3: DVR with Neutron

From figure 6, both the Compute node and the Network node have the same system requirements to gain access to the network resources. This logical architecture assumes that the DHCP function is taken care of by a network element that is separate from both the Network node and the Compute node. [8]

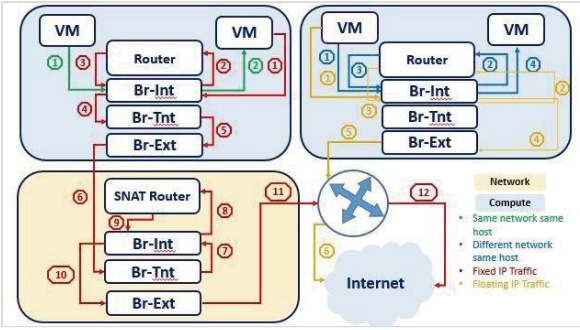


Figure 4: OpenStack DVR Traffic flow

Each compute node has its own router configured in a network namespace that caters for the packet transfer from VM to VM by passing it through the local routers. Only in case of fixed IP addresses for the VM, when SNAT is required to send traffic to external networks, the network node's services are required.

Neutron OpenDaylight Controller, developed under Linux Foundation. It integrates in Neutron as a ML2 plugin and centrally controls each virtual switch on all nodes. ODL pushes all flows into OVS that takes the decision regarding packet forwarding. OpenDayLight uses conntrack based SNAT with OVS netfilter integration where netfilter maintains the network address translations. This translation is managed at a switch designated as a NAPT switch on the network node, and performs the centralized translation role. All the other switches send the packet to centralized NAPT for SNAT. If a NAPT switch goes down an alternate switch is selected for the translations however, the existing translations will be lost on a failover. For North/South SNAT, traffic has to still pass through the network node. OpenDayLight addresses network latency by routing packets through a virtual switch and hence removes the necessity of passing the packet to network namespaces.

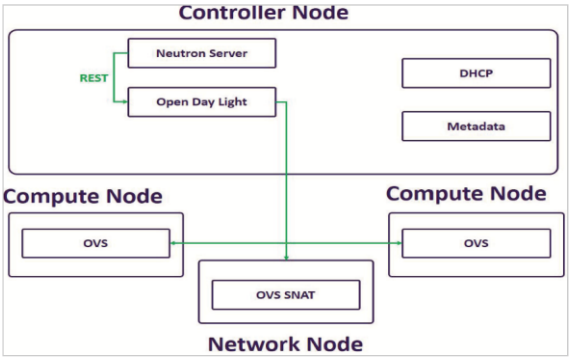


Figure 5: ODL with OpenStack

As can be seen in the diagram, Neutron sends RESTful call to OpenDayLight which then uses OVSDB and OpenFlow to configure each virtual switch across all nodes. OpenDayLight pushes flows in the virtual switch based on which decisions are made to forward packets to the destination VM. OpenDayLight can also act as a

DHCP server but it is recommended to use Neutron's DHCP due to its high availability. [9][10]

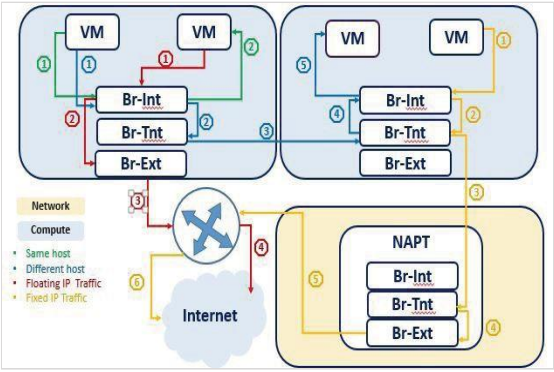


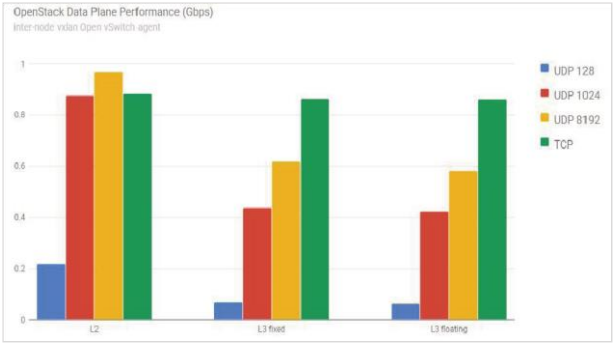
Figure 6: ODL VM traffic

III. Alternate Strategies

Test Bed- routing creates a bottleneck, performance analysis report of default Openstack has been generated using VMTP[3] over 1G links. Note that the same environment has been set up for all future tests carried out.Two VMs were created, Both were placed, first on different compute nodes then, on the same compute node. Depending on the test case they were configured for the following networking schemes: 1-VM to VM Same virtual Network.2- VM to VM Different virtual Network (L3 fixed).3-VM to VM Different virtual Network (Floating IP).Both UDP and TCP packets of varying lengths i.e. UDP 128 bits, UDP 1024 bits, UDP 8192 bits, and TCP 65535 bits were sent between the two VMs. And graphs were plotted through the VMTP Plugin[3].UDP 128 generates the lowest number of packets due to the increased computational power required by the VMs. Similarly, UDP 8192 and TCP 65535 generates plethra of packets due to decrease in the computational power required to send these packets. Given below are the results obtained by using the above mentioned parameters.

i. VMs on different compute host

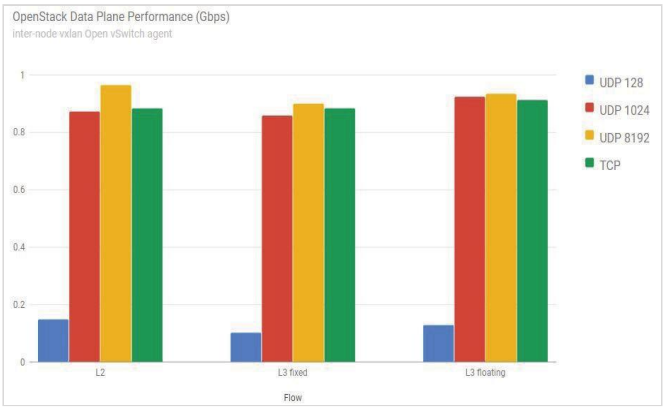
Neutron OpenStack



Graph 1: OpenStack Network performance on different Computes

Graph 1 shows VMs communicating within the same network (L2 communication) gives better throughput then L3 routing for both floating and fixed IP addresses.

OpenStack with DVR

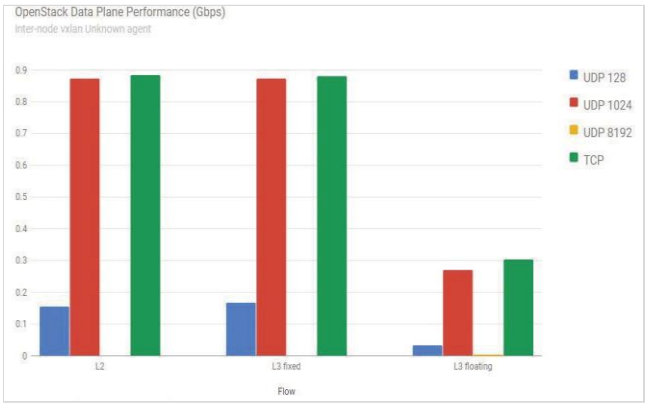


Graph 2: Neutron DVR different Computes

Graph 2 shows throughput when DVR is being used for L3 routing instead of Neutron

improving the traffic flow of Fixed and floating IP addresses.

OpenStackwithODL

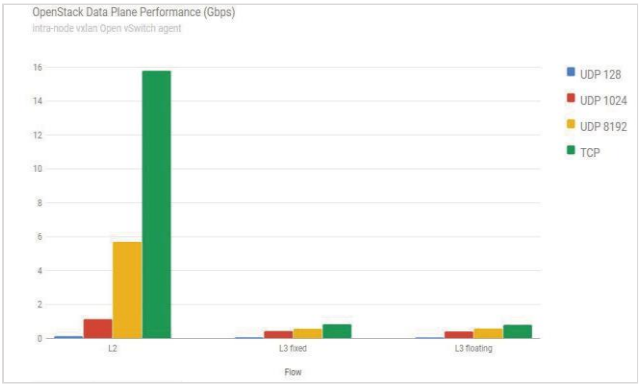


Graph 3: OpenStack with ODL different Computes

Graph 3 shows, L3 floating is affected by NAT translations handled by OVS, resulting in reduced bandwidth. This, however, is still higher than that of Openstack's default routing, however much slower than that of Neutron DVR's.

ii. VMs on same compute host

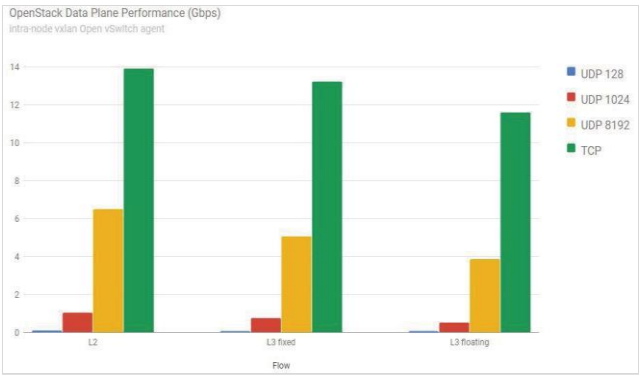
Neutron OpenStack



Graph 4: OpenStack Neutron Same Compute

As shown in graph 4 in case of VM on the same Compute host, L2 traffic yields significantly better performance than L3 routing.

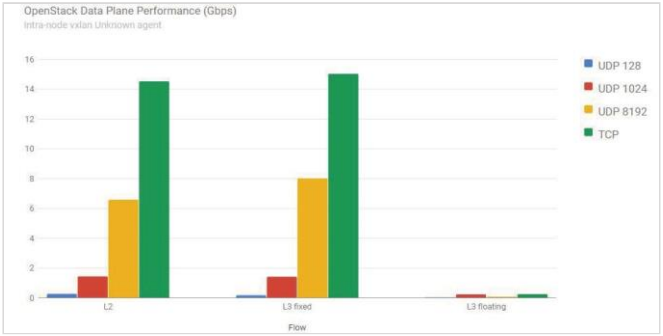
OpenStack with DVR



Graph 5: OpenStack DVR same Compute

Graph 5 depicts traffic communication between two different VMs on the same compute host. It can be seen L3 communication has significantly improved whencomparingwithGraph4'sL3 Network communication.

OpenStack with ODL



Graph 6: OpenStack ODL same compute

Graph 6 presents layer 3 communication for static IPs has improved when comparing with Graph 4.

IV. Comparison Matrices

Same Compute host

	OpenStack Centralized Routing	Neutron Distributed Virtual Routing	OpenDayLight
L2	16Gbps	14Gbps	15Gbps
L3 Fixed IP	0.86Gbps	13Gbps	15Gbps

Address			
L3 Floating IP Address	0.82Gbps	12Gbps	0.3Gbps

Different Compute Host

	OpenStack Centralized Routing	**Neutron Distributed Virtual Routing**	**OpenDayLight**
L2	0.88Gbps	0.89Gbps	0.89Gbps
L3 Fixed IP Address	0.86Gbps	0.88Gbps	0.89Gbps
L3 Floating IP Address	0.86Gbps	0.91Gbps	0.3Gbps

V. Conclusion

Network Function Virtualization has the ability to scale rapidly and extensively. With scaling comes geographical and topological distances, which in turn are bound to affect NFV's performance if our reliance on centralized routing continues to be. Neutron DVR and OpenDayLight are promising solutions that support distributed routing that significantly resolves the latency issues of the former technique. The performance comparison clearly highlight OpenDayLight and Neutron DVR as the optimal solutions as opposed to OpenStack's default routing. For L3 forwarding between virtual machines with floating IP addresses, Neutron DVR handles all address translations through routers configured in the network namespaces. On the other hand, OpenDayLight handles the translations through OVS which is much slower than network namespaces. This makes Neutron DVR a better choice for L3 forwarding in case of floating IP addresses. In case of fixed IP addresses however, OpenDayLight shows a better performance as all the routing decisions are being made at OVS itself and does not requires to enter a network namespace for the decision. This removal of extra hop saves time. For Neutron DVR however, packet still has to pass through network namespace which although resides on the same compute host, but creates a latency as compared to OpenDayLight.

VI. References

[1]https://www.openstack.org/telecoms-and-nfv/

[2]https://wiki.openstack.org/wiki/Neutron/DVR

[3]https://github.com/openstack/vmtp

[4]https://www.zdnet.com/article/telecoms-love-th e-openstack-cloud/

[5]https://www.networkworld.com/article/3253118/virtualization/what-is-nfv-and-what-are-its-cost-performance-and-scaling-benefits.html

[6]https://docs.openstack.org/security-guide/networking/architecture.html

[7]http://docs.ocselected.org/openstack-manuals/kilo/networking-guide/content/under_the_hood_linuxbridge_scenario1_compute.html

[8]https://docs.openstack.org/liberty/networking-guide/scenario-dvr-ovs.html

[9]https://docs.opendaylight.org/en/stable-oxygen/opendaylight-with-openstack/openstack-with-vtn.html

[10]https://access.redhat.com/documentation/en-us/red_hat_openstack_platform/10/html-single/opendaylight_and_red_hat_openstack_installation_and_configuration_guide/index