Deployment and Performance Evaluation of Virtual Network based on OpenStack

Shaoka Zhao[1,2],Liyao Li[1]
1.Mathematics and Computer Science Department
Fuqing Branch of Fujian Normal University
Fuqing ,China
zska@cernet.edu.cn,
llywck@139.com

Jiahai Yang[2],Cong Xu[2],Xiao Ling[2]
2.Institute for Network Sciences and Cyberspace
Tsinghua University
Beijing,China
yang@cernet.edu.cn,
xucong10@mails.tsinghua.edu.cn,
lingxiao2011@yeah.net

Shuxiao Huang[3,2]
3.School of Software Engineering
Beijing University of Posts and Telecommunications
Beijing,China
hsx0905@gmail.com

***Abstract*—In the context of the ever-developing virtual network, OpenStack officially launches its new network component-Quantum, but the comparative performance between single-host and multi-host deployment is yet unknown. The paper introduces detailed deployment strategies of single-host and multi-host after analyzing Quantum work mechanism and the construction polices of virtual networks. And then, we probe into the influence of various deployments on the reliability of network service, and conduct connectivity and performance tests on different deployment plans. Experiment results show that our virtual network multi-host deployment plan improves virtual network service reliability and efficiency compared with single-node node.**

Keywords-Quantum; Single-host; Multi-host; Agent Scheduling

I. INTRODUCTION

With the booming of cloud platforms, the traditional network model is no longer able to meet the needs of cloud services in terms of scale, performance and automation. Therefore, the introduction of advanced network virtualization technologies is quite necessary as they can not only improve network utilization but also make the network more extensive and manageable. Virtual network allows user groups with different needs who are logically isolated from each other, to a certain degree, to access the same physical network. With network virtualization, a couple of closed user groups can be deployed on single physical infrastructure, and the entire network will maintain high-security, extensibility, manageability and feasibility.

At present, various platform software used to build cloud-computing environment have emerged, for example, OpenStack, Eucalyptus, OpenNebula and CloudStack, etc. This paper chooses OpenStack, which has a great advantage in layered architecture, SOA, componentization, open sourcing, etc., as the experiment platform to build the virtual network.

In Essex, the earlier version of OpenStack, the network function of virtual machines was run by the compute component, Nova. In the latest versions of Folsom and Grizzly, however, the network component, Quantum, which possesses rich network APIs and enables users to create network topology through network and sub-network configuration, is separated from Nova. Quantum is highly flexible and practical.

Currently, domestic and foreign research literature on Quantum, the OpenStack-based virtual network component, is relatively rare, and most of it centers on engineering document. Refs [1]-[5] describe background, approaches and application status of the latest network virtualization technology. Ref [6] presents a measurement study to characterize the impact of virtualization on the networking performance. As an important component of OpenStack, Quantum could make network virtualization in cloud environment possible. Refs [7] and [8] depict the various constituents of Quantum and how it works; refs [9] and [10] tell us the major usage scenarios and the simple and common deployment plans, however, they fail to provide us with composition scenarios which are more applicable and fully realize the deployment strategies based on various specific scenarios. Refs [11] and [12] put forward some improvement ideas to solve the problem of Single Point of Failure (SPoF), which is common in deployment. Refs [13] and [14] give some simple assumptions on future deployment, yet neither of them illustrates the highly available deployment for more complex scenarios or analyzes or evaluates the performance of the deployment.

From the above analysis, it is easy to see that multi-host virtualization routing is a completely new subject on the latest released version OpenStack Grizzly. In addition, the performance of the virtual network deployment is still uncertain, and there is hardly any further research which concentrates on this topic. Therefore, based on this novel platform, we will design a set of effective deployment scenarios, and conduct connectivity testing and performance evaluation at the same time.

II. QUANTUM COMPONENT

Quantum mainly includes Quantum-server, plugin and agent, which collaboratively provide network services. The Quantum-server first receives the API request and then passes it to the plugin for further processing. Later, according to the API request, the plugin conveys the

message to the agents（L3 agent，DHCP agent，etc.），which, in turn, pass it to the corresponding process for performance. Finally, the agent returns the request message.

III. DEPLOYMENT STRATEGIES

A. *Single-host Deployment*

The emergence of Quantum has brought changes to both virtual networks and physical architecture. Figure 1 displays a typical single-host deployment plan, through which we see three types of physical nodes, namely, control node, network node and compute node. Communication and networking mainly involves Management network, Data network, External network and API network.

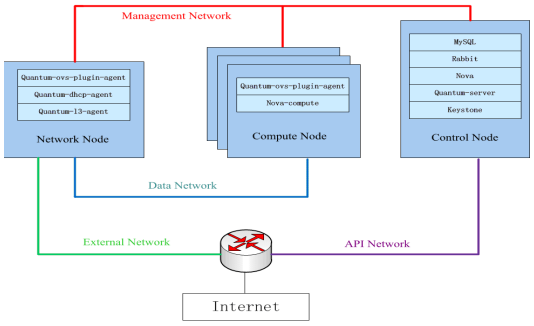

Figure 1. Typical single-host deployment plan of OpenStack

In real-world deployment, networks can be configured and merged according to different experimental environment. Regarding single-host deployment, this paper does not draw strict distinctions between management network and data network, as both of them are small and medium sized internal networks. Therefore, we merge the two types of networks into one.

In the process of single-host deployment, there are generally several compute nodes and one network node. The proper functioning of network services—the core of cloud services—is of great significance, but single network nodes are prone to huge security and failure risks. On one hand, SPoF might occur in a certain node, resulting in the malfunction of the network services of the whole cloud platform as well as that of other services. On the other hand, if heavy network traffic exists on the cloud platform, single network nodes will be limited by bandwidth, becoming the bottleneck of QoS, or even break down.

In order to improve the reliability of the network services on the platform of OpenStack, Multi-host deployment, which provides network services on all compute nodes to properly solve the problem of SPoF, is designed in our deployment plan. Such a framework could enhance the performance of network services, because it equally distributes network traffic on all compute nodes, which are mutually independent.

B. *Multi-host Deployment*

To improve the reliability of virtual networks, Multi-host deployment strategy of OpenStack is introduced. With a control node and three compute nodes (as shown in Table I), we first and foremost set the parameters of the network environment. The management network and data network are merged into one in the deployment process.

TABLE I. NETWORK CARD SETTING OF MULTI-HOST DEPLOYMENT

node	eth0	eth1
control node	166.111.XXX. XXX	172.16.0.51
compute node 01	166.111. XXX. XXX	172.16.0.52
compute node 02	166.111.XXX. XXX	172.16.0.53
compute node 03	166.111. XXX. XXX	172.16.0.54

The architecture of Multi-host deployment in Grizzly is shown in Figure 2. During the process of Multi-host deployment, parameters should be configured to the configuration files of L3 and DHCP agents on each compute node:

enable_multi_host = True;

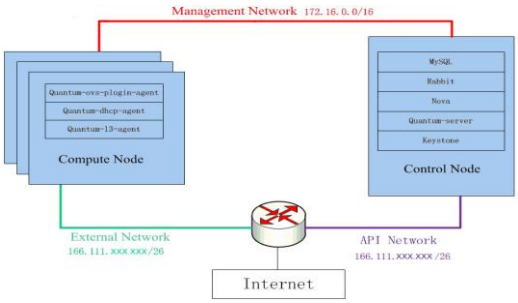

Figure 2. Multi-host deployment

The remaining configuration is almost the same as that in Single-host deployment.

In the deployment scenario of Figure 2, each compute node equals to a network node, and the L3 and DHCP agents, which are deployed on certain node, are only responsible for the network communication of the virtual machines in its own node. In this way, if the L3 or DHCP agent of a certain node fails, the influence will be limited to the virtual machines running on the specific node. Therefore, SPoF can be avoided to some extent, improving the quality of network services.

Nevertheless, a new issue is raised: although the malfunction of a compute node does not affect other nodes, the virtual machines on the faulty node cannot communicate with the virtual machines on other nodes or external networks. Since several network nodes are allowed to coexist in one system, we could make efforts to design the agent scheduling mechanism on Multi-host deployment, unbinding the network agent service of a certain node from that node. In this way, the network service of one node could also reach others. The advantage of the modified mechanism is that even though all virtual machines on a single node fail to

acquire network services because of network faults on the node, we can still access external networks with the services provided by other nodes. Therefore, the reliability of network services is further improved.

C. Agent Scheduling Mechanism Design

Agent scheduling mechanism on Multi-host deployment requires the cooperation of several agents. We can let a couple of such agents provide the same routing function, but only one of them serves as the default agent, with others being considered as coordinating agents. The default agent uses the IP of the subnet gateway as the port IP of the interface, while coordinating agents create new ports on the subnet as the interface port. This section mainly discusses a scheduling framework based on the above deployment plan.

A network environment, which includes three networks Net1，Net2 and Net3 with the corresponding subnets of subnet1，subnet2 and subnet3, has been designed as in Figure 3. Net3 is an external network, while Net1 and Net2 are internal networks of the tenants.

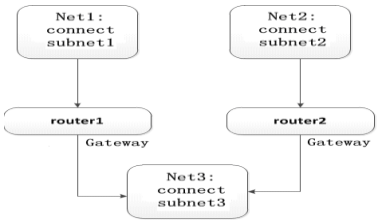

Figure 3. Network environment setting

Figure 4 shows a default scenario: there are three hosts altogether, namely, Host A, Host B, and Host C. The virtual machine VM_1_A which belongs to subnet1 runs on Host A, and the L3 agent in operation is the default agent of router1. The virtual machine VM_1_B of subnet1 and the virtual machineVM_2_B of subnet2 run on Host B. The DHCP agents of Host A and Host B are collaboratively responsible for Net1 and Net2. The L3 agent runs on Host C is the default agent of router2. In such a scenario, the L3 agent of Host A is in charge of the communication with the VMs of subnet1, and the L3 agent of Host C is responsible for the communication with the VMs of subnet2.

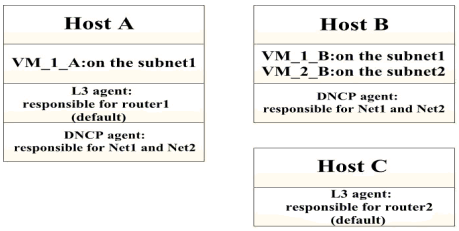

Figure 4. Default scenario

Figure 5 shows the scenario of cooperative routing: the L3 agent of Host C is responsible for router1 and router2, and the L3 agent of Host A is also in charge of router1, which means that the virtual machine VM_1_A of Host A is in coordination model. Therefore, the default communication of VM_1_A on Host A is the responsibility of the L3 agent of Host A; when L3 agent on Host A is out of work, VM_1_A will turn to L3 agent on Host C for routing service. Since there is no L3 agent running on Host B, the communication of the virtual machine on Host B is in the charge of the default router, namely, the L3 agent of Host C.

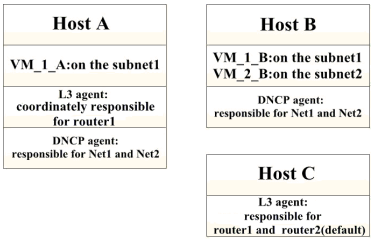

Figure 5. Scenario of cooperative routing

To make the virtual machine of Host B in coordination model like that of Host A, the L3 agent needs to be run on Host B, cooperatively taking charge of router1 and router2. Bellow is a scheduling framework of Multi-host deployment plan based on agent cooperation:

- If each router acquires services from only one L3 agent, which is fundamental in network communication of the virtual machines, it is the default router.
- If the default router and the cooperating router coexist in a network, and one of them (either the default router or the cooperating router) is running on the virtual machine node, external network will be accessed through the router on the node preferentially. If no default router or cooperating router exists on the node of the network, external network will be accessed through the default router located on another physical server.

In the modified Multi-host deployment plan, several L3 agents (redundant L3) provide routing services for the same VMs so that the network communication of the VMs on the faulty node will not be affected when SPoF occurs. In this way, the reliability of network can be guaranteed.

IV. EXPERIMENTS

A. Test configuration

After the single-host and multi-host deployments are completed, in order to check whether it's successful or not, connectivity and performance tests of network services should be run on Quantum components based on the same use-case scenario.

Figure 6 and Figure 7 show the single-host test scenario. In this scenario, each private network provides routers for the tenants, which means every tenant has his (her) own private network and router. Each having their own routers, namely, router_1 and router_2, the two tenants of pro_1 and pro_2 are connected to the external network, with the port IP of 166.111.xxx.4 and 166.111.xxx.5. Tenant pro_1 possesses

one network called net_1, with the segment of 10.10.10.0/24. Tenant pro_2 has two networks, namely, net_1 and net_2, the segments of which are 10.10.20.0/24 and 10.10.30.0/24 respectively. Both of the two networks are connected to the tenant's own router—router_2. Four virtual machines are running in the networks of the two tenants, among which virtual machines vm_1 and vm_2 of pro_1 belongs to the same segment, while virtual machines vm_1 and vm_4 of pro_2 belongs to different segments.

Deployment architectures of Multi-host and Single-host are similar, except that the route of Multi-host is selected under the designed scheduling mechanism from a segment selection pool of External Network. Meanwhile, the virtual network agents within each host (Figure 8) finish work in accordance with our scheduling mechanism designed before.

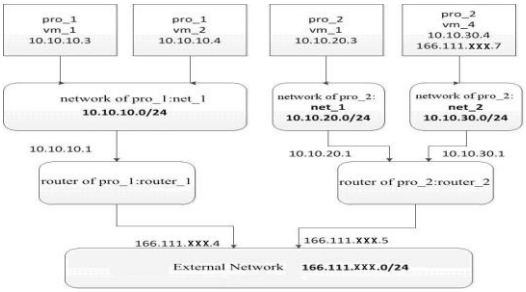

Figure 6. Single-host deployment scenario(1)

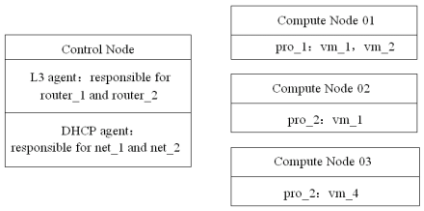

Figure 7. Single-host deployment scenario(2)

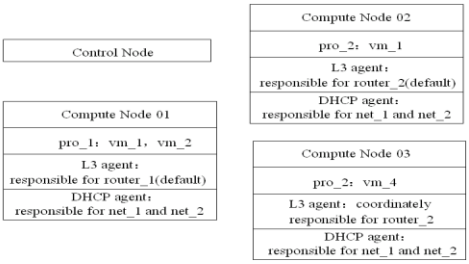

Figure 8. Multi-host deployment scenario

B. *Test Result and Analysis*

Below are the tests on network connectivity within different VMs on the same compute node, between VMs on different computing nodes, as well as VMs in the cloud and external hosts outside the cloud based on single-host and multi-host deployment.

In testing network connectivity within different VMs on the same compute node, we use vm_1 to ping the address of vm_2 —10.10.10.4 to find out whether vm_1 and vm_2 of pro_1 could communicate normally. It is very easy to see the result is positive. When testing network connectivity between VMs on different computing nodes, an external IP address should be allocated to the virtual machine in the internal network, as the tenants have set up their own isolated internal networks through routers. After allocating the external IP address to vm_4, we use vm_2 on node 01 to ping the external IP address of vm_4 on node 03, and the result is also positive. In testing network connectivity between VMs in the cloud and external hosts outside the cloud, we use vm_2 on node 01 to ping external IP address, and the result is satisfactory.

From the above connectivity tests, it's not hard to conclude that the deployment of the virtual network of OpenStack has gained initial success.

Next is for other performance tests. Based on design scenarios of Figure 6 to Figure 8, we make the estimation of virtual network time delay and packet loss rate under the way of two deployment modes, i. e., the Single-host and Multi-host. The experiment uses a software package D-ITG [15], adopting different approaches to select the sender and recipient for sending packets, meanwhile gradually increase the number of packages to simulate large file transfers. We consider the following three test cases: 1) between different VMs on the same compute node, 2) between VMs on different computing nodes, 3) VMs in the cloud and external hosts outside the cloud. Just as the connectivity test, for the first case, we conduct the performance test between vm_1 and vm_2 in tenant pro_1. For test case 2, we conduct performance test between vm_1 of pro_1 and the vm_1 and vm_4 of pro_2 simultaneously. The third test uses vm_1 of pro_1 and vm_1 of pro_2 in the cloud in parallel to send packets to the external host, and then the measurement results plotted in a line graph showing comparison display. The horizontal axis of the plot stands for test file sizes, and the vertical axis represents the time delay (Figure 9) and packet loss rate (Figure 10).

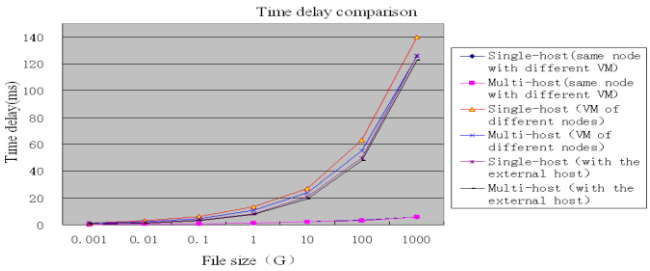

Figure 9. Time delay curve

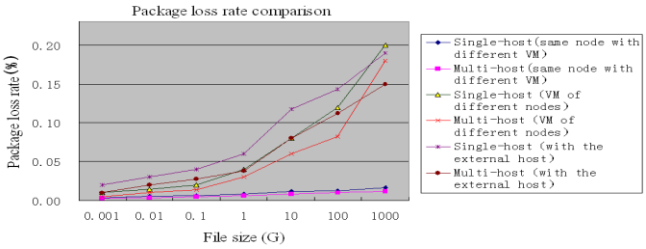

Figure 10. Package loss rate curve

In Figure 9, it is easy to see, as the file size sent increases, the time delay on the whole shows a gradual increasing trend. As in the same node, communication between different VMs is with no involvement of agents, the difference between single host and multi-host deployment delays are not significant.

While in multi-host deployment, if the sent package is very small, the time delay between different VMs located in different compute nodes inside the cloud is roughly twice that between the cloud VMs and external hosts. The reason is that the packages go through twice the number of agents during the transmission between cloud VMs compared with that between cloud VMs and external hosts (the external hosts are located closely to the router).

With the communication increases, between the virtual machines, cloud virtual machines and external host of different computing nodes, owing to the test of parallel communication, there is competition for resources and agent scheduling. Thus Multi-host deployment manner highlights the advantages, which is specifically showed by the performance of its slower growth trend curve. Overall, when in Multi-host scenario, every time the communication volume increases 1,000 times, the three scenarios' (between different virtual machines of the same node, between virtual machines of different computing nodes, between virtual machines in the cloud and hosts outside the cloud) growth of time delay is about 5 times, 12 times and 16 times, respectively. Figure 10 gives a comprehensive reflection that the packet loss rate and the time delay curve trend are roughly the same. Experiments show that Multi-host deployment takes certain advantages than Single-host on the whole.

V. CONCLUSION

This paper introduces the single-host and multi-host deployment of OpenStack, qualitatively analyzes the reliability of network services. Taking advantage of the new features of Quantum component, we emphasize on Multi-host deployment and design a coordination mode of agents. According to the connectivity and performance evaluation results, our deployment plan avoids SPoF and improves the reliabilities of virtual network services.

Two problems remain to be solved in the next phase regarding to the deployment plan in this paper: first, how to figure out the exact number of redundant routers. Indeed, the L3 agent of each compute node could take charge of all routers, but such a deployment is no doubt ineffective and wasteful, especially in large scale cloud networks. Second, how to design a more effective agent scheduling algorithm with the purpose of realizing rational allocation of resources and effective implementation of the highly-reliable deployment plan, which are composed of part of our future works.

ACKNOWLEDGMENT

We would like to thank anonymous reviewers for valuable comments. This research is supported by Tsinghua University-Cisco Joint Laboratory Research Fund Project, Ministry of Education - China Mobile Research Fund (No.MCM20123041)， Fujian Provincial Department of Education Science and Technology Project (No.JA13343), Fuqing Branch of Fujian Normal University Research Fund (No.KY2013001).

REFERENCES

[1] Gregor Schaffrath,Christoph Werle,Panagiotis Papadimitriou, et al,"Network virtualization architecture: proposal and initial prototype," ACM SIGCOMM Conference, Barcelona, Spain, pp. 63-72, August 2009.

[2] N.M.Chowdhury,and Raouf Boutaba,"A survey of network virtualization,"Computer Networks, vol. 54, no. 5, pp. 862-876, April 2010.

[3] Nicira Inc,"Nicira Network Virtualization Platform(NVP) ,"http://nicira.com/en/network-virtualization-platform, 2012.

[4] Piotr Rygielski,and Samuel Kounev,"Network Virtualization for QoS-Aware Resource Management in Cloud Data Centers,"PIK , Volume 36, Issue 1, pp. 55–64, Feb.2013.

[5] Rafael Esteves, Lisandro Zambenedetti Granville, and Raouf Boutaba, "On the Management of Virtual Networks,"IEEE Communications Magazine.to appear.

[6] Wang G, Ng T S E,"The impact of virtualization on network performance of amazon EC2 data center,"INFOCOM, 2010 Proceedings IEEE. IEEE, pp. 1-9. March 2010.

[7] Dan Wendlandt,"Introduction to OpenStack Quantum for Cloud Operators,"Folsom Conference, April. 19, 2012.

[8] Salvatore Orlando,"Quantum: Virtual Networks for OpenStack," QCon,May 2012.

[9] OpenStack,"OpenStack Quantum Administration Guide,"http://docs.openstack.org/trunk/openstack-network/admin/bk-quantum-admin-guide-trunk.pdf, April 2012.

[10] Shake Chen,"Ubuntu12.04 OpenStack Folsom installation based on VLAN," http://www.chenshake.com/openstack-folsom-install-guide-vlan-mode," November 2012.

[11] Quantum multi-host. https://docs.google.com/document/d/1Y41g-POd3DLmtFnD6JfKFDvZJikbUvsL0wIi8LFD35U/edit,2013.

[12] Quantum scheduler, https://docs.google.com/document/d/1TJlW0_tMpeENA_ia38fvRu7ioKRt9fsWXBjivwd1mMw/edit#heading=h.v5d86yy8opif, 2013.

[13] Yongsheng Gong,"Architecture of Quantum Grizzly Release," https://docs.google.com/viewer?a=v&q=cache:tU30-8LnNbwJ:www.valleytalk.org/wp-content/uploads/2012/12/quantum-technical-archicture.ppt+&hl=zh-CN&gl=cn&pid=bl&srcid=ADGEESg7xw0r-V1WDu8pcp4xIkNTFGPEBKWAWOVOUEgp28S5TYHCvFZZ5NhD76P6x2ejHFO7kIllTN-ZEWn3hmKeWLK1gtk7LRIFX0U4U5d1qC8hoaYCAvceY7E6XvpqRuUTCr8QVX5j&sig=AHIEtbRjIoOPWr7n-WlDHZN9P2PTAy0dwQ, 2013.

[14] Kirill Ishanov,"OpenStack Super Bootcamp,"OpenStack Summit, San Diego, Oct.15-18,2012.

[15] D-ITG :Distributed Internet Traffic Generator – v. 2.3 Reference Manual(last update),http://www.grid.unina.it/software/ITG, April, 2004.