Research on VLAN Division of Cloud Data Center Based on SDN

Suyang Huang
Tongji University
Shanghai, China
e-mail: 821028457@qq.com

Haihang Wang
Tongji University
Shanghai, China
e-mail: wanghh@tongji.edu.cn

***Abstract*—Network virtualization is an effective way to solve the stiffness, uncontrollability and integration of network, theoretical and applied research on which is the focus of computer science nowadays. In this paper, after exploring several key technologies of network virtualization, based on SDN (Software Defined Network), VM isolation and interoperability are achieved by dividing the VLAN with Open vSwitch. And the work is verified by VM migration.**

Keywords- network virtualization; SDN; Open vSwitch; VM isolation

I. INTRODUCTION

As the rapid development of cloud computing and virtualization, a more flexible, reliable and high-performance network is needed. Many users choose virtual machines (VMs), which can help save a lot of physical cost, make full use of the remaining available resources of physical servers, and provide easier resource management in the cloud. When an exception occurs in a physical server or the VMs run in it, or maintenance is required, VMs can be migrated freely. With the technology of migration, VMs can be backed up and recovered quickly. All of these require a network for cloud which can not only increase and reduce bandwidth when necessary, to support non-blocking, low latency forwarding of data, but also achieve barrier-free connection among physical servers, to support communication, migration and expansion of VMs [1], so as to ensure the continuity and high availability of service, reduce the cost of system maintenance and upgrades and improve load balancing and of fault tolerance of system.

In the cloud, to scheduling resources on demand, VM migration is needed, which means that migration of virtual resources in Layer 2 will be transparent. The key for resource scheduling in cloud is to build a large-scale Layer 2, in which switch ports corresponding to users' VMs can be divided by group, so that users can visit each other and do business migration here. In order to build a highly reliable network, Layer 2 is generally extended by ring network with Spanning Tree Protocol. However, with the expansion of network and the increase of VLAN numbers, the actual network topology is very complicated now.

The above requirements can be achieved through network virtualization, using which multiple user groups can be deployed on one physical machine. On the physical machine, users in one group access to the same physical network, and still with some degree of isolation logically. Currently, many research institutions and scholars has studied on network virtualization, proposing solutions such as NVGER [2], VXLAN [3] and STT [4], in which the flow of MAC layer is encapsulated in IP layer to isolate the virtual IP and the physical network. The emergence of these solutions is a big step forward a more dynamic and scalable cloud network, but none of them can be seen as a complete solution, since they are encapsulated protocols without control panel, and relying on some other network functions.

II. KEY TECHNOLOGIES

A. SDN

Software Defined Network is a new network architecture proposed by Clean Slate of Stanford, with which network is controlled by modules and devices are separated from data. In the architecture of SDN, control plane is partly abstracted to software, making network easier to control. [5] Briefly, network management is achieved with software, which contains the intelligent part of switches and routers. Thus, the hardware is just responsible for sending and receiving packets, leaving the work of thinking for software. Based on this architecture, logical network topology can be defined through software, with no need to concern for the underlying physical topology. [6]

The key technology of SDN, OpenFlow, is a project of ONF (Open Network Foundation). It is a network tool for virtualization, separating control plane and data plane [7], using new forwarding modes, and still supporting multi-tenancy, VM migration and scalability. Meanwhile, control plane is abstracted to a common platform, so that the data plane can be programmed through the control plane. With switches controlled, users can run different applications on them.

B. VLAN

To ensure on-demand extending and scalable scheduling of resources, transparent VM migration in Layer 2 is necessary. So the key of cloud resource scheduling is to build a large-scale Layer 2 to cover more resources. One facing problem is broadcast storm. With VLAN, it can be avoided, with broadcast domains isolated effectively. Dividing VLAN in Layer 2 can group switch ports corresponding to VMs by users, enabling Layer visits and business migration. Now, with the expansion of cloud data center services and the increase of users, VLAN needs to be expanded.

C. Online Migration

Online migration, also known as live migration, refers to a VM moving from one physical host to another [8], with the programs on it running normally and services not interrupted. Online migration shields the distribution and heterogeneity of underlying hardware [9] through remapping VMs to physical resources, which is the dynamic deployment of VM. The main online migration tools include QEMU-KVM/Libvirt, Microsoft Hyper-V, Citrix XenMotion and VMware Vmotion. These tools require a centralized shared external storage device between physical machines, or use block storage for online migration, and the main consideration is on the migration of the memory execution state of system when operating.

However, when migration is between different subnets or in WAN, if the network configuration remains unchanged, the VM may not be accessed from the outside; but if reconfigure the network according to the moved subnet, the transparency to applications and users will be destroyed. There are many solutions on the network configuration of VM migration currently, but few of them can keep complete transparent to applications or external access.

D. Virtual Ethernet Bridge

What Virtual Ethernet Bridge (software such as VMware, XEN, etc.) does now is to make a virtual switch inside the physical server for the communication of VMs. Compared to traditional physical switch, a virtual switch has many advantages. First, the configuration is more flexible since there may be multiple virtual switches on one single server and the number of ports is easy to choose. Second, cost is lower by using virtual switching, but still achieving the performance of expensive physical switches. Virtual switching provides fine routing and management for multi-tenant network of cloud. In a multi-tenant network, user data and applications are stored in separate sections divided by virtual network. That is to say, real VLAN is expanded to virtual VLANs for every user to avoiding confounding.

As a virtualization software switch following the Apache 2.0 protocol and implemented by portable C, Open vSwitch is applicable for VM. It can make the realization of large-scale network automation, as well as support standard management interfaces and protocols. [10] With Open vSwitch, functions of most commercial switches can be achieved, such as creating cluster-level network configuration for multiple servers, eliminating network configuration for every single VM and physical host.

In a traditional data center, network of every physical machine is visible for manager, and can be configured through policy of a port on the switch, controlling the network access and configuration of physical machine. However, in a cloud platform, without the support of virtualization technology, data can't be distinguished among VMs to which it belongs, let alone network isolation and flow monitoring and other needs. Using the software switches created by Open vSwitch, VMs on one physical host can be assigned to different VLANs for different user groups, in order to achieve isolation in Layer 2. Because of the dynamic configuration of interface brought by Open vSwitch, VM network can be controlled and managed easily, and VM migration will also be very easy between NICs. Meanwhile, different speeds and bandwidth can be given to VMs through the ports of software switches, to ensure the performance of VMs with core business. The architecture of Open vSwitch is shown in Fig. 1.

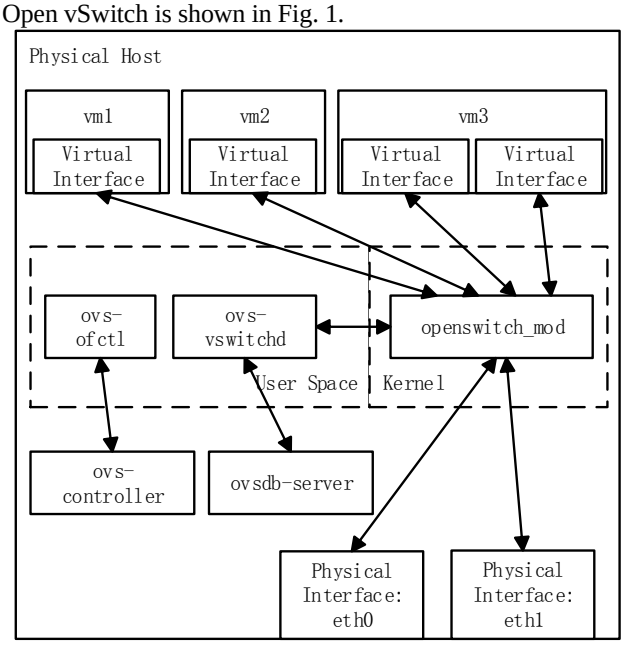

Figure 1. Architecture of Open vSwitch

*1) Forwarding plane:*Located in the user space, the slow forwarding plane ovs-vswitchd makes the basic forwarding logic, including address learning, mirror configuration, 802.1Q, VLAN, LACP protocol, binding with external physical ports and load balancing based on source MAC and TCP. It can be configured and managed remotely through OpenFlow, after which the data will be sent to ovsdb-server process for storage and management.

The kernel forwarding plane openvswitch_mod.ko is in the kernel space, for inquiry, modification and forwarding of packets, tunnel encapsulation and underlying forwarding tables maintenance.

*2) Ovsdb_server:*Ovsdb_server is a lightweight database server that ovs-vswitchd queries to obtain its configuration.

*3) Ovs-brcompatd:*Ovs-brcompatd is a Linux kernel module that makes ovs-vswitchd working as a Linux bridge.

*4) User tools:*Ovs-vsctl: viewing and modifying ovs-vswitchd configuration;

ovs-dpctl: kernel module configuration of switch;

ovs-appctl: sending command messages;

ovsdbmonitor: GUI tool that can remotely access the OVS databases and OpenFlow flow tables;

*5) OpenFlow:*OpenFlow plays the role of control plane in Open vSwitch. Each data forwarding keeps a link to OpenFlow so that when a data flow first goes through the software switch, it will be forwarded to the control platform. OpenFlow defines policies of forwarding, discarding,

modifying and queuing according to feature matching of information of Layer 2-4. When a data flow is forwarded again, it will be processed by kernel forwarding module directly, thus speeding up the subsequent data processing.

Modules introduced: OpenFlow switch ovs-openflowd, OpenFlow controller ovs-controller, OpenFlow command line configuration interfaces ovs-ofctl and patch for ovs-pki and tcpdump.

The flow control is mostly achieved through OpenFlow switching protocols since OpenFlow enables a software controller accessing to the data path of a switch or router through network. Besides, the Flow Table + Controller architecture introduced provides an open platform for new services and protocols.

III. ARCHITECTURE DESIGN AND EXPERIMENTS

In this paper, VMs are managed and migrated by Libvirt, and VM isolation and interoperability are achieved by VLANs divided with Open vSwitch. In order to ensure the network configuration unchanged around live migration, dynamic network configuration and port management via Open vSwitch are needed.

There are two physical hosts Host1 and Host2, with Open vSwitch running on both. VMs vm1 and vm2 are in Host1, vm3 and vm4 in Host2. Each VM has its separate virtual Linux Device Interface on its physical host, numbered sequentially from vnet0 by default.

A. VLAN Division with Open vSwitch

1) Installation and configuration of Libvirt and Open vSwitch. Do the following on both Host1 and Host2:

Install packets needed:

#apt-get install aptitude kvm libvirt-bin virtinst virt-viewer openvswitch-controller openvswitch-brcompat openvswitch-switch openvswitch-datapath-source

Delete the default bridge of Libvirt:

#virsh net-destroy default

#virsh net-autostart --disable default

Start Brcompat module of Open vSwitch:

#vim /etc/default/openvswitch-switch

Modify: BRCOMPAT=yes

Compile openvswitch-datapath：

module-assistant auto-install openvswitch-datapath

Restart the system:

reboot

2) Network Configuration. Create ovs bridges on Host1 and Host2:

Create bridge ovsbr0：

ovs-vsctl add-br ovsbr0

Create port eth0 for Host1 and eth1 for Host2:

ovs-vsctl add-port br0 eth0(1)

Modify network configuration:

vim /etc/network/interfaces

Edit the file as following:

auto eth0

iface eth0 inet manual

up ifconfig $IFACE 0.0.0.0 up

down ifconfig $IFACE down

auto ovsbr0

iface ovsbr0 inet static

address 192.168.2.100

netmask 255.255.255.0

network 192.168.2.0

broadcast 192.168.2.255

gateway 192.168.2.1

dns-nameservers 192.168.2.1

bridge-ports eth0

3) VM Creation. Create vm1 and vm2 on Host1, with configuration files vm1.xml and vm2.xml (See Appendix contents of configuration files)

virsh define vm1.xml

virsh start vm1

virsh define vm2.xml

virsh start vm2

Create vm3 and vm4 on Host2, with configuration files vm3.xml and vm4.xml

virsh define vm3.xml

virsh start vm3

virsh define vm4.xml

virsh start vm4

4) VLAN Division. Divide VLANs to isolate VMs, with vm1 and vm3 in VLAN1, vm2 and vm4 in VLAN2, shown in Fig. 2.

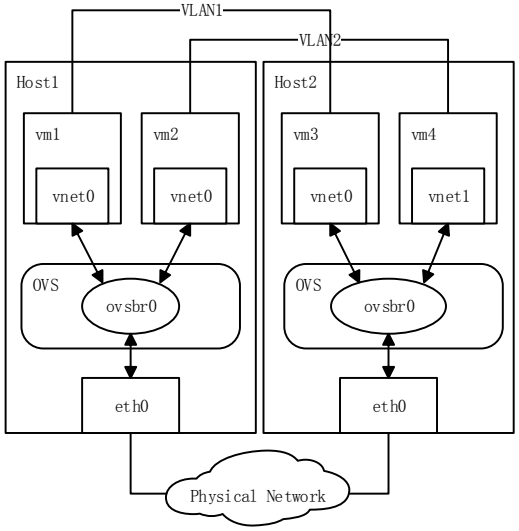

Figure 2. VLAN Division

Configured as follows:

On Host1:

Add vm1 to VLAN1:

ovs-vsctl add-port br0 vnet0 tag=1

Add vm2 to VLAN2:

ovs-vsctl add-port br0 vnet1 tag=2

On Host2:

Add vm3 to VLAN1:

ovs-vsctl add-port br0 vnet0 tag=1

Add vm4 to VLAN2:

ovs-vsctl add-port br0 vnet1 tag=2

5) Tests. Ping other VMs on the four VMs separately to test if they are connected. Results are shown in Table I:

TABLE I. CONNECTION OF VMS

	vm1	**vm2**	**vm3**	**vm4**
vm1		Timeout	Connected	Timeout
vm2	Timeout		Timeout	Connected
vm3	Connected	Timeout		Timeout
vm4	Timeout	Connected	Timeout	

From Table I, it is known that vm1 and vm3 are in one VLAN, and vm2 and vm4 in another.

B. *Live Migration*

Vm3 will be migrated to Host1, but remaining in VLAN1.

1) Live migration. On Host2:

virsh migrate vm3 qemu+ssh://[IP of Host1]/system --live

Vm3 is in Host1 now, view the VM list of Host1:

virsh list --all

Add vm3 to VLAN1:

ovs-vsctl add-port br0 vnet2 tag=1

2) Tests. Ping other VMs on the four VMs separately to test if they are connected. Results are shown in Table II:

TABLE II. CONNECTION OF VMS

	vm1	**vm2**	**vm3**	**vm4**
vm1		Timeout	Connected	Timeout
vm2	Timeout		Timeout	Connected
vm3	Connected	Timeout		Timeout
vm4	Timeout	Connected	Timeout	

From Table II, it is known that vm1 and vm3 are still in one VLAN, and vm2 and vm4 in another.

IV. CONCLUSION

In this paper, studies have been done on several key technologies of network virtualization and the thinking of SDN. With the software switch Open vSwitch, based on OpenFlow, which is the realization protocols of SDN, VLANs are divided, thus achieving VM isolation and interoperability. VMs are managed and migrated by Libvirt. The work is verified by testing if two VMs in different VLANs are connected.

SDN presents a brand new thinking and direction for building of virtual network, meeting the requirements of cloud data centers. However, the research for SDN is in development, with no uniform model for the time being, so there are still some problems:

- Lack of theoretical models. Each has put forward their own implementations, but theoretical studies have yet to be perfected.
- Unable to upgrade smoothly. Since existing equipment doesn't support SDN, it's necessary to upgrading. The cost may be high for a large-scale implementation of new programs, especially with the completely new subversive technology SDN.
- Performance degradation brought by flexible configuration. Network virtualization solves many problems in cloud, but it's not sure whether it will influence the performance. When handling complex data flow in a large network node, there may be a performance bottleneck.
- More secure and stable factors from more complex and flexible management mode, may making safety and reliability deteriorated.

All these problems need to be studied and solved, leading for future development. Among them, there are some urgent directions for in-depth study:

- More flexible and centralized control.
- More independent of control and data plane.
- Centralized control will eventually evolve into network operating system, controlling traffic, rules and equipment and provide user control interfaces.
- Monopoly of kernel network will be broken and protocol format will be more flexible and personalized, thus making the protocol standardization issues more prominent, also bringing higher requirements for compatibility of protocols.

REFERENCES

[1] M.Bari, R.Boutaba, R.Esteves, L.Granville, M.Podlesny, M.Rabbani, Q.Zhang, M.Zhani.Data Center Network Virtualization: A Survey[J].Communications Surveys & Tutorials, IEEE,2012,99:1-20.

[2] A.Greenberg, et al. NVGRE: Network Virtualization using Generic Routing Encapsulation. draft-sridharan-virtualization-nvgre-01(work in progress), 2012,7.

[3] M. Mahalingam, et al. VXLAN: A Framework for Overlaying Virtualized Layer 2 Networks overLayer 3 Networks. draft-mahalingam-dutt-dcops-vxlan-02(work in progress), 2012,8.

[4] B.Davie, J.Gross. A Stateless Transport Tunneling Protocol for Network Virtualization. draft-davie-stt-02, 2012,8.

[5] http://en.wikipedia.org/wiki/Software_Defined_Networking.

[6] J.Rubio-Loyola, A.Galis, A.Astorga, J.Serrat, L.Lefevre, A.Fischer, A.Paler, H.Meer. Scalable service deployment on software-defined networks[J]. Communications Magazine, IEEE, 2011,49(12);84-93.

[7] Nick McKeown et al. (2008-04). "OpenFlow: Enabling innovation in campus networks". ACM Communications Review. Retrieved 2009-11-02.

[8] http://www.zdnet.com.cn/wiki-virtualization_onlinemigration.

[9] Minxiong Wen, Qin Li. Network connection redirection technology in virtual machine migration system [J]. Application Research of Computers,2009,(05):1839-1843.

[10] http://openvswitch.org/cgi-bin/gitweb.cgi?p=openvswitch;a=blob_plain;f=README;hb=HEAD.