UnionPay’s Financial Cloud Achieves “Five-High” Production Requirements Using OpenStack

1. Summary

In 2012, UnionPay deployed an OpenStack-based financial cloud into operation - a first for the Chinese financial industry. Since then, we have been exploring and evolving OpenStack's technology and practice in the financial industry, while maintaining an orderly business migration to this technology. As a result, the UnionPay financial cloud supports expanding development of UnionPay's business through large-scale hosting of UnionPay's many applications.

The financial industry's IT architectures have five prominent demands and requirements, what we call the “Five-High” requirements: high performance, high reliability, high scalability, high manageability, and high security. The design and deployment of the UnionPay financial cloud has focused on these Five-High requirements, and we have gained significant OpenStack technical skills and experience in financial industry applications. The large-scale application of OpenStack at UnionPay has become a representative case and a benchmark model in the financial industry.

In this paper, We’ll start with a brief introduction to UnionPay, followed by a discussion of OpenStack's large-scale application scenarios. Finally we will introduce a practice scheme based on the financial Five-High requirements.

2. A Brief Introduction to UnionPay

UnionPay is a bankcard association established in March of 2002 under the approval of the State Council and the People’s Bank of China. UnionPay has become the central and pivotal part of China’s bankcard industry and plays an essential role in this industry’s development. As the bankcard association in China, UnionPay operates an inter-bank transaction settlement system through which the connection and switch between banking systems and the inter-bank, cross-region, and cross-border usages of bankcards (issued by associate banks) could be realized. Based on the system construction and operation, as well as the bankcard network, UnionPay actively collaborates with various industrial parties (such as commercial banks) to formulate

and extend united UnionPay card standards and regulations, create independent bankcard brand, promote the development and application of bankcard, maintain an orderly bankcard acceptance market, and prevent bankcard risks.

Thanks to the popularity of UnionPay cards, the UnionPay brand enjoys a high reputation in China. According to studies by A.C. Nelson, an international authoritative investigation institution, the UnionPay bankcard brand has nearly 100% domestic familiarity, much higher than any other famous bankcard brand in China. By the end of 2016, UnionPay issued a total of 6 billion cards worldwide, with a trading volume of 27.1 billion transactions and trading amount of 72.9 trillion RMB per year.

The UnionPay brand is fully recognized and trusted in China while also having strong world-wide influence and competitiveness. At the end of June 2017, the UnionPay acceptance network covered all the domestic urban and rural areas in China, extending to Asia, Europe, the Americas, Oceania, Africa, totaling more than 160 countries and regions, and with over 21 million overseas merchants and 1.49 million ATM machines. The standard UnionPay card, (beginning with 62xxx), has truly become the largest bank card branch with the amount of cards issued and volume of transactions.

UnionPay's rapid development comes from its strong technical innovation capability. UnionPay's success have also positively affected the development of Asian and international financial IT technology. Currently, UnionPay is one the makers and promoters of many financial standards in China. In the international arena, UnionPay's payment application systems have been exported to Laos, Thailand, and other regions

3. Overview of the UnionPay Financial Cloud

The financial industry in the Internet era needs the support of a more powerful cloud computing IT architecture. UnionPay predicted that financial cloud computing technology will be more widely based on open source technology in the future. Among them, OpenStack with its open source, scalability, flexibility, good ecological operation of the community and other advantages, has become the de facto open source cloud computing standard and the best choice for UnionPay financial cloud.

UnionPay was one of the first explorers and users of OpenStack in the financial

industry. In 2012, we deployed the first financial industry cloud in a production environment in China, powered by OpenStack Essex version, which has been in stable operation for over 1500 days. After several years of research and development, UnionPay has accumulated a great deal of practical experience and formed a dedicated team of around 50 people. The team focus on the design, development, deployment, and operation of the UnionPay financial cloud. Meanwhile, we collaborated with many other partners such as EasyStack, etc. What's more, UnionPay actively contributes to the OpenStack community, including establishing the OpenStack financial working group (recently approved by OpenStack User Committee) and participating in the Large Contributing OpenStack Operators (LCOO) working group.

UnionPay financial cloud provides power for UnionPay's business development while greatly reducing investment on IT resources. At the same time, the cloud provides massive computing power and support for rapid growth of trading volume and the number of users and merchants. In order to support more powerful and convenient management of cloud resource, UnionPay launched a plan for setting up a new cloud platform in 2016 and has already completed the deployment of financial cloud platform 2.0, with OpenStack Liberty in the second half of 2017. The new cloud platform will have many new capabilities such as Neutron networking virtualization, Ironic bare-metal management and Ceilometer telemetry. The platform also includes VxLAN networking, hybrid deployment of NAS and Ceph storage, management of hardware firewall and hardware load balancer, and integration with hardware SDN.

UnionPay has always attached great importance to the use and sharing of technology, especially the adoption and innovation of new technologies. We have close cooperative relationship with many financial partners, e.g. Bank of Shanghai, China Easter Airlines, and others, and we also encourage the adoption of OpenStack in the financial industry.

4. UnionPay Financial Cloud in Practice

4.1. Overview

This section focuses on UnionPay's financial cloud. First, it will briefly introduce the

physical deployment structure of UnionPay cloud, and then it will elaborate the design of UnionPay cloud from the five aspects of Five-High goals:

- High performance focuses on DPDK-related research work
- High reliability focuses on the reliability of compute nodes and virtual machines
- High scalability focuses on how to scale compute nodes horizontally
- High security focuses on how to integrate keystone with UnionPay's existing Single Sign On (SSO) system
- High manageability focuses on how UnionPay improves resource provisioning productivity based on group-based network policy.

4.2. Physical Topology

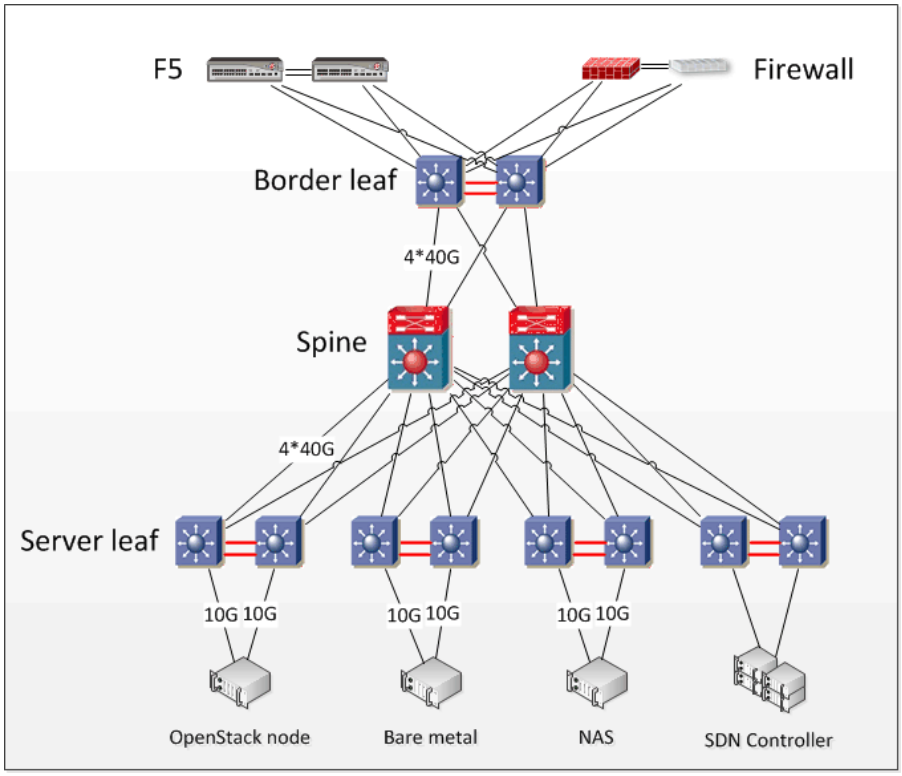

Figure 4-1 physical topology

Software Defined Networks (SDN) fabric with VxLAN networking is chosen to be the base of UnionPay's OpenStack cloud. OpenStack combines and triggers the SDN controller through the neutron's plugin system, greatly enhancing networking delivery

efficiency. This is the first time in China's financial industry that OpenStack, combined with hardware SDN, is put into production for critical businesses.

There are many different types of equipment in the physical network environment: for example, an x86 server, top of rack (TOR) switch, spine switch, border switch, load balancer, firewall, and more. The top of rack (TOR) switch (which we call a "server leaf") is mainly used for connection with an x86 server. The spine switch (which we call simply the "spine") is used to connect all other server leaves and is specifically responsible for high-performance data forwarding. A border switch (which we call a "border leaf") is mainly used for connection with edge devices, such as a load balancer and firewall. The border leaf also connects the SDN fabric and other external network areas.

As shown in Figure 4-1：

a) Server leaf stack communicates through two 40GE interfaces, cross-linked to the spine through the 4*40GE interface, and connected to the x86 server through the 10GE interface.

b) Spine cross-connects server leaf and border leaf through the 4*40GE interface.

c) Border leaf stack communicates through two 40GE interfaces, cross-linked to the spine through the 4*40GE interface, connected to load balancer and firewall through the 2*10GE interface.

d) x86 server network interface bonding: Link Aggregation Control Protocol (LACP).

e) The SDN adopted a distributed VxLAN. The underlay routing of border leaf, spine and server leaf are using Open Shortest Path First (OSPF). The hosts routing is learning through Ethernet VPN (EVPN). Border leaf routing learning through (Border Gateway Protocol) BGP.

4.3. High Performance

The high performance requirements usually refer to the hardware processing capacity and how to use it wisely to enhance the overall processing capacity of the platform. For example:

- The system itself should use concurrent processing.
- The system should not have a single point of performance bottleneck.
- Use asynchronous calls instead of synchronous calls.
- Frequently visited parameters or configuration data should be kept in memory.
- Enable the use of load balancing technology to enhance the system's concurrent processing capabilities.

The following sections discuss performance considerations for hardware and software aspects of the cloud platform. The DPDK related research work is also introduced.

4.3.1. Hardware

In the networking hardware equipment level, we use TOR switches with 10G interface for x86 servers and 40G interface for the spine to fully guarantee the necessary business bandwidth, and we use a hardware firewall and hardware load balancer to achieve faster processing speeds. We also use X86 servers with a 10G network interface and dual-NIC binding to increase bandwidth and redundancy.

4.3.2. Software

At the software level, in order to support a stable large OpenStack cluster, we deploy nodes with the different roles of the OpenStack cloud platform on separate physical servers. This separation improves the overall cloud platform's processing capabilities and the efficiency of operation and maintenance management. The UnionPay cloud control plane consists of five kinds of servers: controller node, network node, MySQL node, RabbitMQ node, and compute node. The different types of the nodes and the main services running on them are shown in the following table:

Table 4-1 number of nodes and services running on it

Type	Description
Controller node	Mainly run portal and various OpenStack component API layer services
Network node	Mainly run neutron DHCP service and neutron metadata service
MySQL node	Mainly run database service
RabbitMQ node	Mainly run message queue service
Compute node	Mainly hosts virtual machines

4.3.3. DPDK

UnionPay continues studying and exploring cutting-edge technologies in pursuit of higher performance, while meeting the strict requirements of operation systems. For example, UnionPay and Intel have carried out cooperative research on DPDK and describes the relevant results below.

4.3.3.1. Overview

Data Plane Development Kit (DPDK) is an Open Source set of libraries and drivers for fast packet processing and is used, for example, for sending and receiving packets with the minimum number of cycles. UnionPay integrated DPDK with OpenStack's Open vSwitch (OVS) to speed up data processing and improve network packet throughput, enhance overall network performance, and avoid performance bottlenecks. We also built a multiple-SDN testing scenario environment, leveraging the multi-region deployment of OpenStack to test interoperability of heterogeneous SDN environments in multiple regions and to better explore and confirm results of our performance improvements. The testing architecture is shown below:

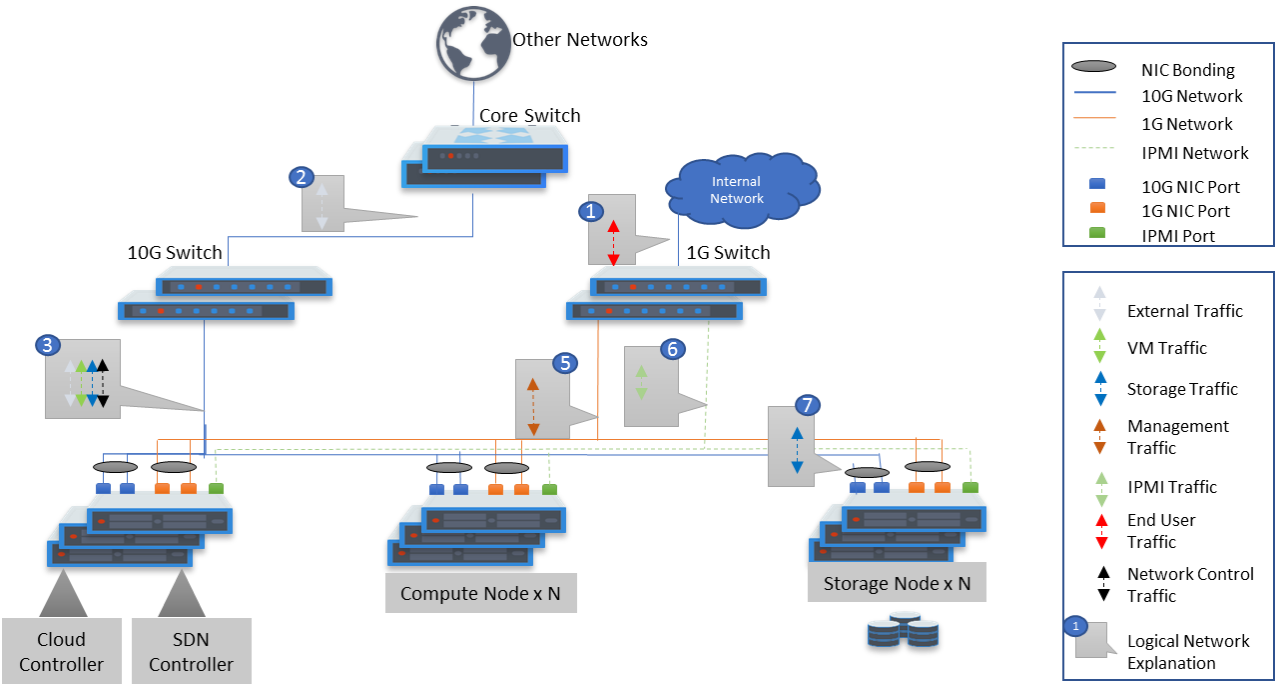

Figure 4-2 Overview of testing architecture

In Open Source solution scenario, the overall architecture of OpenStack + OpenDaylight (ODL) + OVS software-defined infrastructure is as follows:

- The controller node neutron server receives requests
- ML2 plugin processes requests
- the ODL mechanism driver sends requests to the ODL controller
- ODL generates the flow chart and sends it to the OVS of the computing node

The layer 2 switch, layer 3 routing and LBaaS functions of neutron are realized by the OVS flow chart, while DPDK is adopted to accelerate OVS and improve performance.

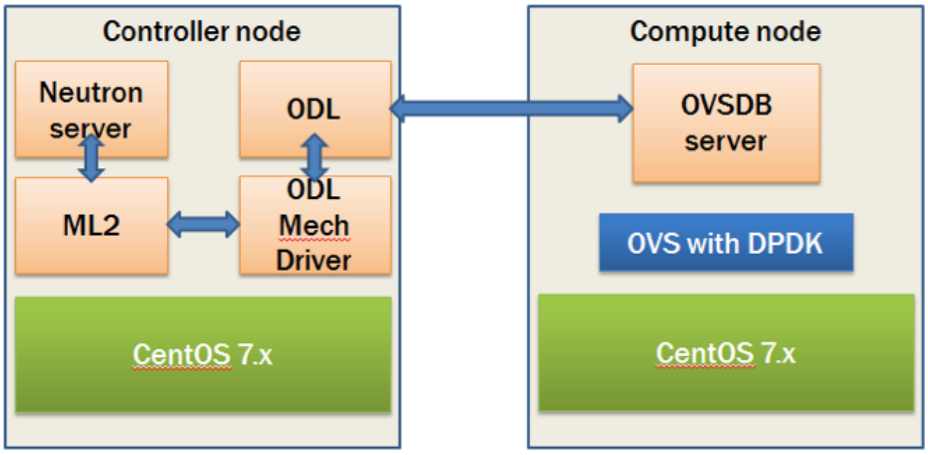

Figure 4-3 Architecture overview for open source solution scenario

Before conducting experiments on performance acceleration, we first did baseline experiments without DPDK technology to verify the value of introducing DPDK technology. Four layer 3 scenarios were built in the OpenStack environment to test the IPERF performance of VMs after layer 3 forwarding in VLAN mode. OpenStack community version L3 vRouter, vendor's vRouter, and the hardware switch are adopted for performance test comparison. The main configuration choices are as follows:

Software configuration:

- OS CentOS 7
- OpenStack Liberty
- OVS 2.4.0
- Libvirt 1.2.16
- QEMU 2.1

Hardware configuration:

- CPU: Intel® Xeon® CPU E5-2680 v2 @ 2.80GHz
- Network adapter: Intel 82599ES
- Memory: 128 GB

The following table shows the results of a contrast experiment. The test results show that the X86-based Open Source solution (solution 1) has a significant performance gap with hardware, and solution 2 (commercial) has certain improvements in performance. The main reason is that bottlenecks caused by the OVS Open Source module are eliminated on layer 2. Therefore, we change the server configuration and physical network adapter model and introduce DPDK technology for the optimization test.

Test Input				Solution 1 Open Source (Neutron L3)				Solution 2 Commercial SDN (Without SRIOV Accelerator)				Solution 3 Commercial SDN (With SRIOV Accelerator)				Solution 4 Hardware			
Test No	Protocol	Bandwidth (Mbps)	Packet (Bytes)	Bandwidth (Mbps)[*1]	Packet Lost	Ping Latency (ms)	Ping Packet Lost	Bandwidth (Mbps)	Packet Lost	Ping Latency (ms)	Ping Packet Lost	Bandwidth (Mbps)	Packet Lost	Ping Latency (ms)	Ping Packet Lost	Bandwidth (Mbps)	Packet Lost	Ping Latency (ms)	Ping Packet Lost
T001	PING	-	-	-	-	0.73	0%	-	-	1.05	0%	-	-	0.73	0%	-	-	0.60	0%
T002	TCP	-	-	3310	-	2.54	0%	5370	-	1.51	0%	6360	-	0.83	0%	8150	-	0.70	0%
T003	UDP	1000	1470	988	1.20%	0.46	0%	993	1%	1.96	1%	1000	0%	0.45	0%	986	1.40%	2.51	0%
T004	UDP	2000	1470	1270	41.00%	21.50	55%	1590	21%	18.83	18%	1800	1%	2.89	0%	1350	29.00%	10.71	28%
T005	UDP	3000	1470	978	60.00%	25.15	55%	1900	14%	17.69	16%	1610	0%	3.10	0%	1480	20.00%	9.54	26%
T006	UDP	10000	1470	988	57.00%	23.97	53%	1700	15%	16.11	14%	1780	2%	3.48	0%	1340	27.00%	8.66	38%
T007	UDP	1000	1024	999	0.74%	0.67	0%	953	5%	11.87	1%	1000	0%	0.52	0%	999	0.05%	0.38	0%
T008	UDP	2000	1024	1370	28.00%	18.40	30%	1160	34%	11.67	20%	1670	9%	8.69	0%	1580	3.90%	4.22	0%
T009	UDP	3000	1024	1390	26.00%	19.33	36%	1430	17%	13.33	21%	1600	12%	9.48	11%	1670	0.94%	3.10	0%
T010	UDP	10000	1024	1360	23.00%	19.64	30%	1420	19%	13.87	23%	1580	6%	6.17	6%	1610	0.64%	3.10	0%
T011	UDP	1000	120	190	8.40%	30.39	11%	171	18%	20.76	21%	189	7%	15.29	1%	200	0.18%	9.14	0%
T012	UDP	300	120	204	5.40%	29.66	8%	166	20%	20.35	21%	201	6%	12.91	1%	187	3.10%	13.22	6%
T013	UDP	1000	64	111	-	41.09	8%	113	-	25.05	15%	103	-	18.82	1%	108	-	14.83	1%

[1]: The L3 actual bandwidth from VM1 to VM2. Able to guarantee network quality when the load is below this level. Network quality will decrease when the load is above this level.

Figure 4-4: The results of experiments

We introduced DPDK technology into the experimental environment to further explore potential performance changes. For the new test architecture environment, we built two environments: an ordinary OVS, and a DPDK OVS. We adapted QEMU to start the VM on the two environments, and tested the IPERF performance between VMs under these four scenarios:

- Ordinary OVS + VLAN
- Ordinary OVS + VxLAN
- DPDK OVS + VLAN
- DPDK OVS + VxLAN

The software and hardware environment configuration in the new environment are:

Software configuration:

- OS CentOS 7
- OVS master (Commit ID:cd8747c542544fcbfcc91dfb983915812922b4c4)
- DPDK 16.04
- QEMU 2.5.1.1

Server hardware configuration:

- CPU: Intel Xeon CPU E5-2695 v3 @ 2.30GHz
- Network adapter: Intel X710 for 10GbE
- Memory: 160GB

4.3.3.2. DPDK PktGen Performance Test

We installed PktGen on server 1 and native/DPDK OVS on server 2. One port of server 1 sends packets with PktGen, and the traffic returns to another port of server 1 through OVS of server 2. We check the bandwidth of two OVS. See the architecture diagram below.

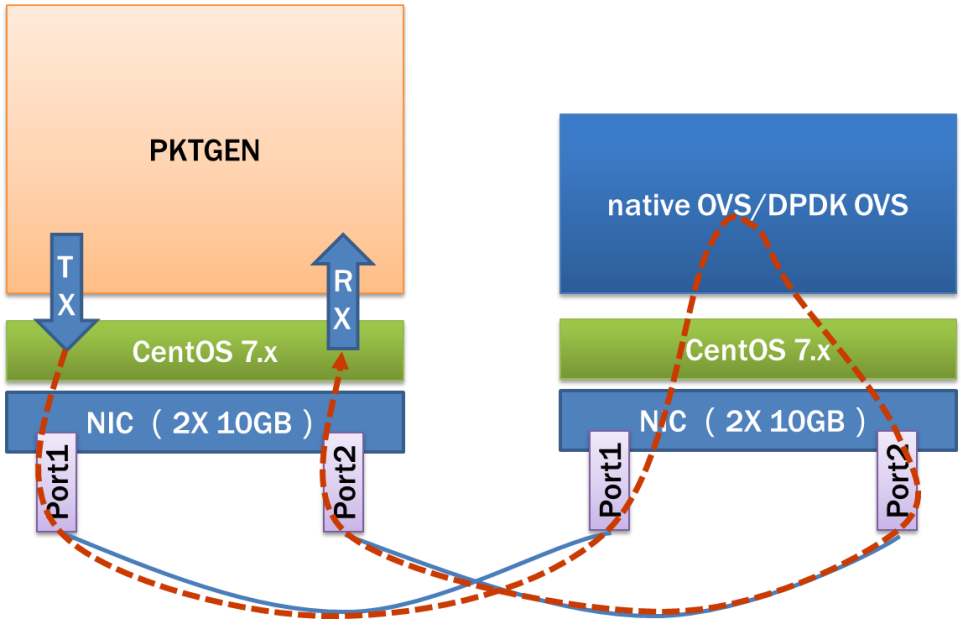

Figure 4-5: The architecture of DPDK PktGen Performance Test

The figure shows an experimental architecture designed to test the basic performance

of DPDK data layer forwarding. The length of the test packages were 64 bytes and 128 bytes.

The follow table shows the results of the DPDK PKtGen performance test. The figure shows that DPDK OVS has clear advantages in small packet performance (up to 10 Mpps), while ordinary OVS has obvious bottlenecks in packets-per-second of OVS (with maximum performance of around 1.5 Mpps).

Table 4-2 The results of DPDK PKtGen performance test (high values are better)

Configuration	Throughput (Gbps)
Ordinary OVS 64-byte packet length	0.877
Ordinary OVS 128-byte packet length	1.531
DPDK OVS 64-byte packet length	5.403
DPDK OVS 128-byte packet length	9.716

4.3.3.3. VM Network Performance Test

For this VM network test, we installed ordinary or DPDK OVS on two physical machines. Then we inserted a physical network adapter to DPDK and started OVS. We added a bridge (creating a tunnel by VxLAN mode). We used QEMU to start VMs and connect to OVS on two VMs, and we use IPERF to test the performance. See the architecture below:

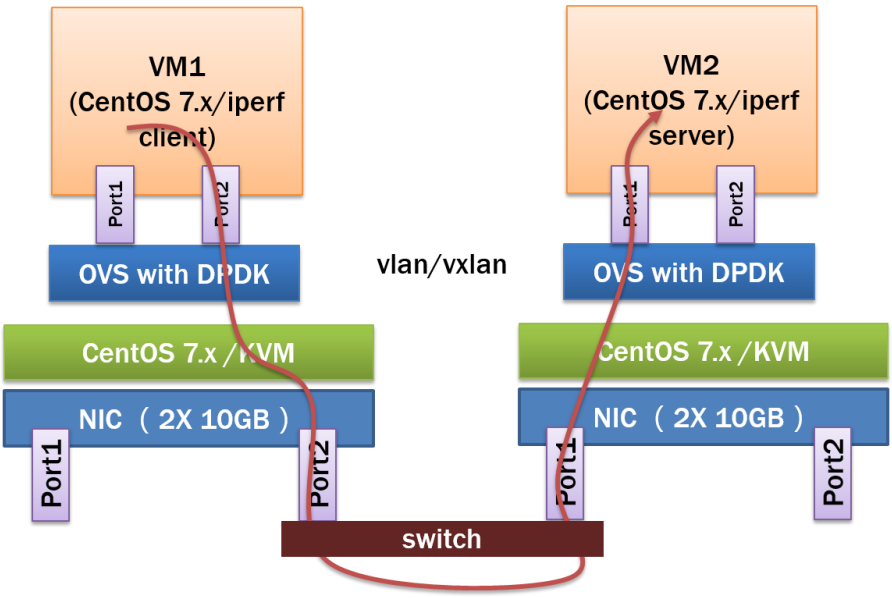

Figure 4-6 The architecture of VM Network Performance Test

Through data comparison, we observed the following:

- DPDK OVS has significant optimization on transmission latency, reducing the latency of almost all data to over 10% of ordinary OVS.
- **Bandwidth of single pair VM IPERF**: ordinary OVS is near linear speed in VLAN mode as packets are large in such scenario, and TSO and GRO of ordinary OVS are activated by default. The bandwidth of DPDK OVS is about 7.5G, as GRO is not activated among codes of DPDK OVS, and TSO and GRO has significant influence on performance in VM scenario. In VxLAN mode, the bandwidth of DPDK OVS is improved by around 40% compared with ordinary OVS.
- **Bandwidth of two pairs of VM IPERF**: ordinary OVS and DPDK OVS are near linear speed in VLAN mode. The bandwidth of DPDK OVS in VxLAN mode is slightly higher than ordinary OVS. See the following table for details.

Table 4-3 The results of VM Network Performance Test

	Throughput of single pair of VM (Gbps)	**Latency of single pair of VM (ms)**	**Pairs of VM at maximum throughput**	**Total throughput (Gbps)**	**Average latency (ms)**
VLAN+OVS	9.27	1.334	1	9.27	1.334
VLAN+DPDK -OVS	7.49	0.168	2	9.32	0.330
VxLAN+OVS	3.53	1.802	The third pair untested	The third pair untested; total throughput of 2 pairs: 7.69	1.627
VxLAN+DPDK -OVS	4.93	0.061	2	8.55	0.097

4.3.3.4. Performance Testing Conclusion

According to the network performance pressure test results show above, DPDK OVS offers a large improvement on transmission latency compared to the original OVS, as well as significant performance improvement in VxLAN scenario. This improved performance addresses UnionPay's requirements for VxLAN use scenarios. In the VLAN scenario (in consideration of multiple VMs deployed in the same physical server in the actual production environment), both DPDK OVS and original OVS can basically achieve the maximum transmission performance of the physical network adapter.

4.4. High Reliability

The requirements for high reliability are that the system must run for a long time without

failure and should automatically recover when various failures occur, minimizing the impact of the failure on the business process. For example, there is redundancy in the hardware equipment at all levels of the system, and there are no single points of failure. The internal components of each device are redundant. The system has strong fault detection and active and passive isolation mechanism, and the system has automatic fault recovery capability.

The following sections discuss aspects of hardware high reliability, software high reliability, compute node high reliability, virtual machine high reliability, and whole business high reliability.

4.4.1. Hardware High Reliability

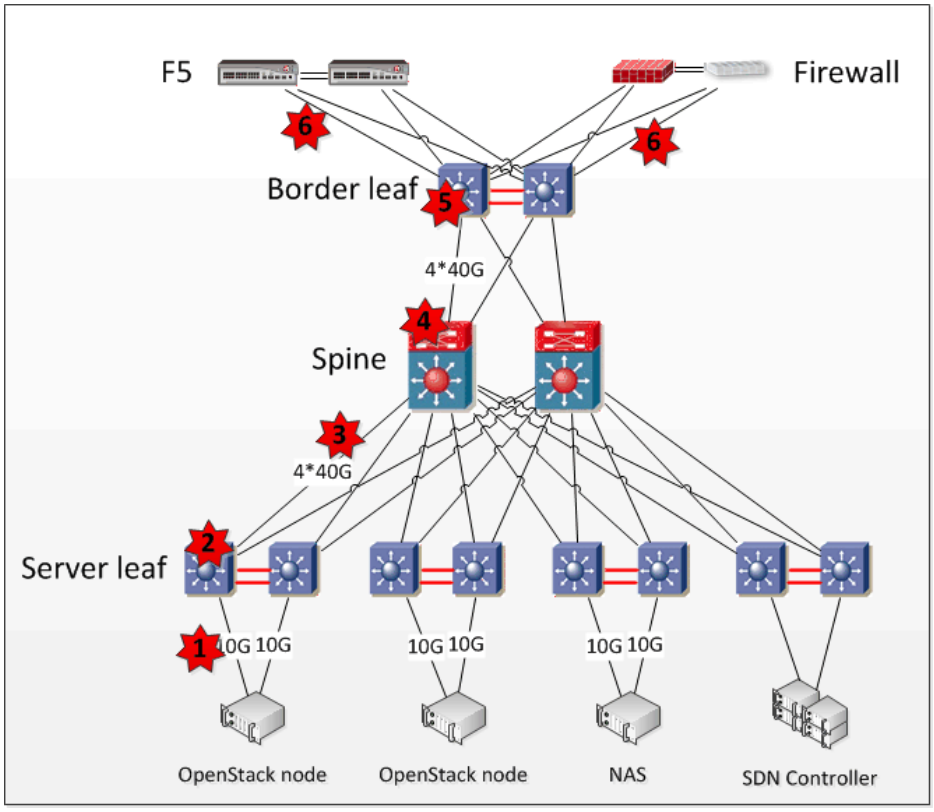

Figure 4-7 Physical Topology

As shown in Figure 4-7：

- If a link fails, the system automatically switches to the backup link; the business is not affected。

- If a server leaf (TOR) fails, the system automatically switches to another server leaf; the business is not affected。
- There are multiple links between each server leaf and the spine. If an uplink fails, it automatically switches to the redundant link. The service is not affected.
- Spine and server leaves use ECMP. If the spine equipment fails, traffic from another aggregation device uplink is used; business is not affected.
- If the link fails, dual-master detection can avoid equipment dual master. The system will down the backup device business port to avoid network loops.
- F5 and firewall dual master detection can avoid the need of active and standby switches.

4.4.2. Software High Reliability

Software high reliability is achieved through the use of clusters and redundant software services that allow functionality to continue (with less performance capacity) while the system recovers from failures:

- Stateless services on the controller node, such as nova-api, neutron-api, cinder-api, ironic-api, manila-api, keystone-api, glance-api, and ceilometer-api, are backends of HAProxy to achieve high availability. If a controller node fails, the cloud platform is not affected because HAProxy has multiple backends.
- For stateful services, such as HAProxy, pacemaker and Corosync are used through the VIP to ensure that HAProxy has high availability. If the current node is not available, pacemaker and Corosync will set up VIP on another node to continue to provide services.
- Three MySQL nodes with Galera as a cluster. If one of the nodes is not available, it does not affect the overall function of MySQL.
- Three RabbitMQ nodes using its own clustering mechanism with Mirror Queue. If one of the nodes is not available, it does not affect the overall RabbitMQ function.

4.4.3. Compute Node High Reliability

By deploying the Host-HA daemon on the controller node to periodically detect unexpected downtime in the compute node, we can avoid accidental downtime.

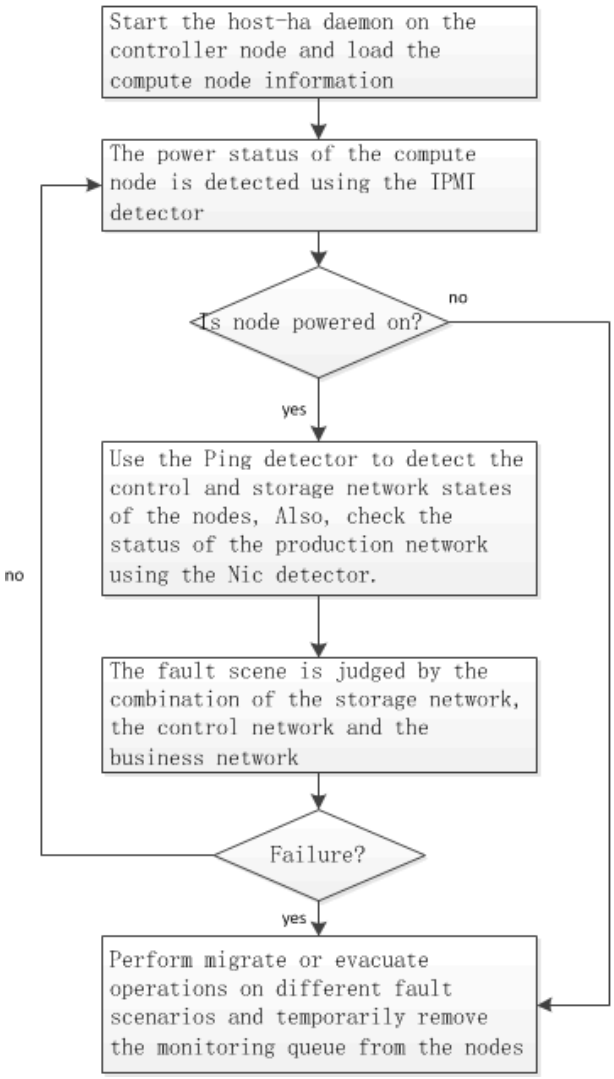

Figure 4-8 Host-HA flow diagram

The specific principle is that the Host-HA process periodically checks the status of all compute nodes in the availability zone. The detection methods include obtaining the power state of the computing node through IPMI, pinging the node control network IP, testing network connectivity, and SSH login to the compute node to check the network interface status corresponding to the service network. If power is detected to be off,

then the node is deemed to have failed. If the node control network IP or the storage network IP is not available for a threshold time, or if the service network interface is down, the control network plane, the storage network plane and the business network plane of the compute node are deemed to be failed. Several test results are arranged in a combination of different fault scenes. For a different fault scene, we might migrate or evacuate the virtual machine running on the compute node in the same availability zone through a predetermined policy to migrate it to the normal node. After the migration is complete, you can see the migration path of the virtual machine on the platform. When the faulty node is restored, the administrator can easily reactivate the compute node through the management platform.

4.4.4. Virtual Machine High Reliability

In the cloud platform, some business cannot be achieved through clustering and other means of high availability, so we need to monitor every single virtual machine in order to provide high reliability of VMs. If a failure, such as a virtual machine internal crash or accidental shutdown occurs, a pre-defined strategy is used to restore the failed virtual machine.

The principle is to run the VM-HA daemon process on the compute node, execute the timed task, call the OpenStack API to collect the information of virtual machines running on the compute node or retrieve the virtual machine log by calling Libvirt and other methods to detect whether the virtual machine is abnormal. If an exception occurs according to the configuration strategy, appropriate action will be performed on the virtual machines to restore them to normal.

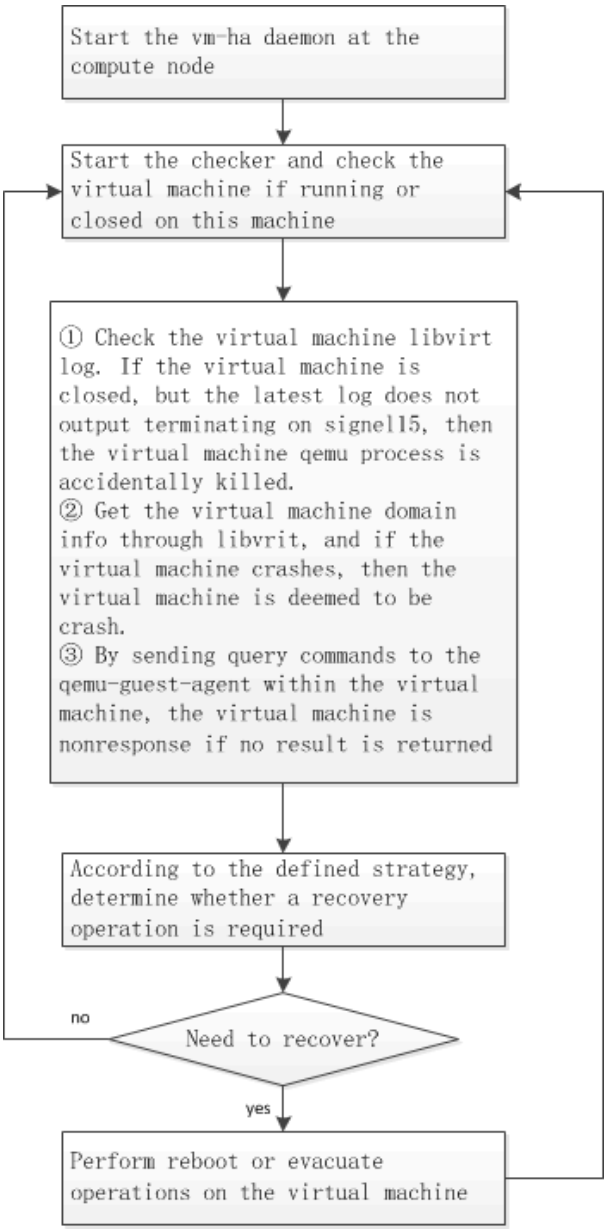

Figure 4-9: VM-HA Flow Diagram

4.4.5. Business High Reliability

To avoid the monolithic single point of failure of commercial NAS storage, UnionPay uses a multi-availability zone decentralized deployment of business systems.

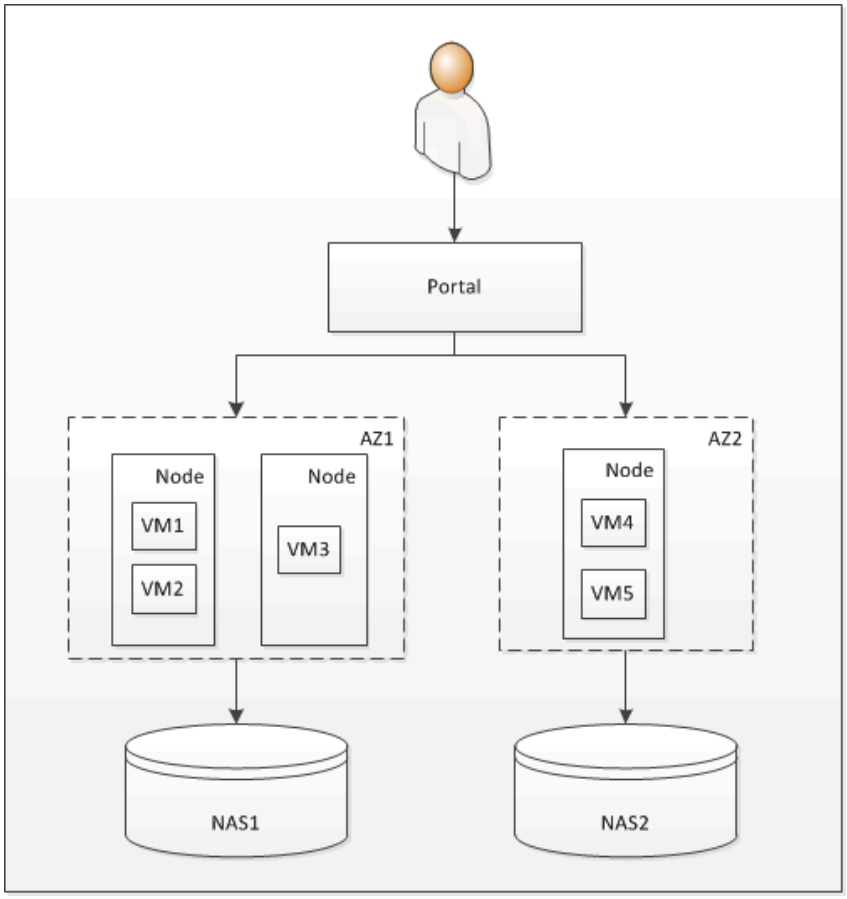

Figure 4-10: Logical View of Business-HA

The data center has multiple sets of independent commercial NAS storage and Ceph distributed storage. Each storage system corresponds to a specific nova availability zone. When the operator creates the virtual machine in batches, the portal automatically scatters multiple virtual machines to different availability zones. Since all instances of a business virtual machine are attached as a backend to a front-end load balancer, it does not affect the overall availability of the business system even if a single NAS storage system is corrupted.

4.5. High Scalability

The requirement for high scalability means that the system can easily increase the overall processing power and throughput by increasing hardware resources. The system should have the flexibility to adapt to business changes and minimize the impact of business changes on the system architecture. For example, horizontal scalability, vertical scalability, support for scaling of compute node pool, and the use of

decoupled interface are all options for expanding the computing resource pool.

4.5.1. Horizontal Expansion of Computing Resources

With the continuous development of new business, if the current computing resources are not sufficient to accommodate business needs, rapid expansion of the existing computing resource pool is required. This means a standard process is needed to expand the resource pool including: installing operating system, host name settings, IP address settings, software installation and configuration, host security reinforcement, and more. Expanding computing resources must not negatively impact the existing production environment, so we prepare new resources offline for bulk installation and configuration and then put them into production. Specific expansion steps are listed below:

1) Brand new computing servers arrives.
2) Put the new computing servers into a rack and connect the power supply.
3) Set the IPMI address, IPMI user name, IPMI password one by one.
4) Access a temporary switch to configure the network to ensure that it can relay DHCP packets.
5) Collect information about existing online computing resources, such as IP segments, in advance.
6) Connect operating machine, which is cobbler server ready, to the temporary switch.
7) Use the ipmitool command to bulk restart the computing servers, booting from network, so that the new operating system can be properly installed from the Cobbler.
8) Execute the Ansible playbook on the operating machine to start the host name setting, perform IP address modification, install and configure the related OpenStack components, do host security reinforcement, and start various service, etc.
9) Switch the temporary network to the production network, and the new computing resources are automatically found.

4.5.2. Virtual Machine Live Resize

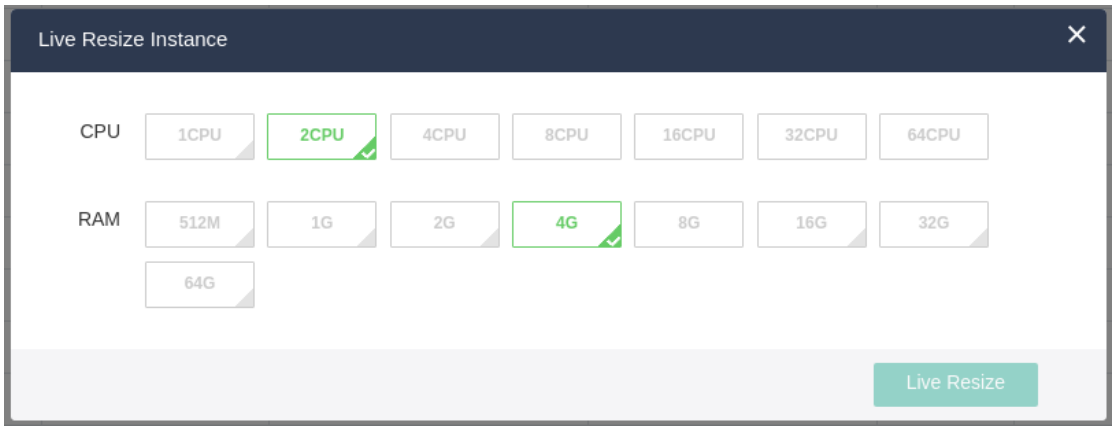

Figure 4-11 Live resize UI

The financial core business system requires non-stop reliability and stability. Vertical expansion of the business virtual machine must minimize downtime exposure. Since OpenStack nova does not support live resize for virtual machines, UnionPay has implemented this functionality by extending the nova API to upgrade a virtual machine to increase CPU and memory with no downtime.

4.6. High Security

The requirement for high security means that the system must resist external and unintentional attacks. For example, the system management operations should enforce levels of security. The system should prevent attacks from the inside and the external Internet and have enforced audit functions. The system data should be safe and protected through effective backup and recovery strategies.

The following sections describe the division of the network plane and QoS, followed by a description of keystone and UnionPay's existing Single Sign On (SSO) system.

4.6.1. Network Plane Division

The different types of network traffic are divided into a number of different data planes for effective isolation, including: management network plane, control network plane, storage network plane, business network plane, and IPMI network plane, as shown in

the table below:

Table 4-4 network planes

Network Plane	Description
Management network plane	Mainly used for ssh management traffic and other device control traffic.
Control network plane	Mainly used for the Rest API call to OpenStack components and the RabbitMQ message traffic between processes.
Storage network plane	Mainly used for virtual machine access to storage device.
Business network plane	Mainly used for virtual machine business traffic.
IPMI network plane	Mainly used for IPMI out of band management.

4.6.2. Quality of Service: QoS

Because of the individual network plane traffic needs, the cloud platform setup uses a different QoS strategy for each plane, fully ensuring the different types of network traffic will not impact one other. From our years of operating experience with UnionPay business, we configure the total single server 20 Gb bandwidth management as:

- network plane: 1 Gb
- control network plane: 1 Gb
- storage network plane: 8 Gb
- business network plane: 10 Gb (due to the use of a 10 Gb stack)

The cloud platform also supports setting specific QoS policies for a single virtual machine. Each virtual machine has two virtual network cards, one for business traffic, and the other for management traffic. In this way, traffic for managing the virtual machine does not affect the business.

4.6.3. Keystone and SSO

Most enterprises have an existing user authentication system, such as Single Sign On (SSO) system; UnionPay is no exception. When performing an OpenStack deployment in the enterprise, integrating keystone (OpenStack's identity service) with an existing SSO system is a problem which must be solved. UnionPay cloud completed this integration through custom development.

The default authentication mechanism provided by keystone is to issue a token. After the user logs in with a username and password, the token is obtained. Subsequent HTTP request headers carry this token id, and the requested component goes to keystone to verify the user token and obtain the user's role for authentication.

Keystone is not only responsible for authentication, but also for authorization. If you want to integrate with the SSO certification system, you need to transfer keystone's original token verification work to the SSO for processing. This will lead to each resource request operation on the OpenStack platform going to SSO for authentication. To avoid putting too much demand on SSO, we need to keep the original keystone token authentication mechanism on the OpenStack platform. Users can use the TGT authentication ticket to get the token before replacing the token with the username and password and then use the token in the OpenStack Platform for resource requests and keystone token validation.

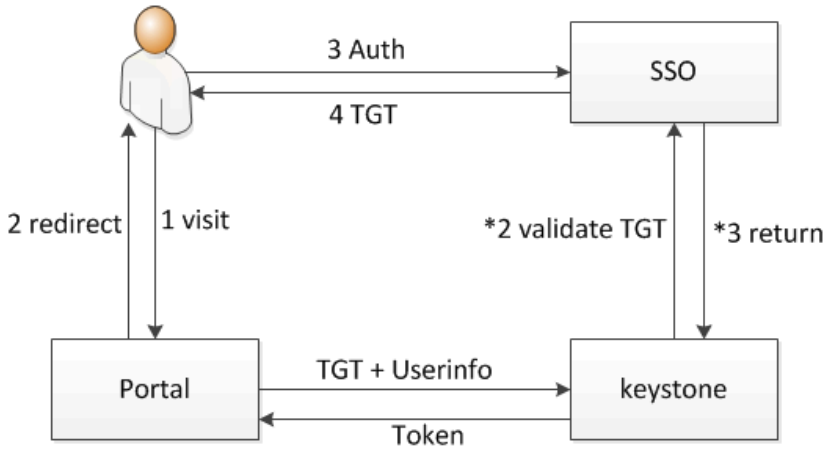

Figure 4-12 Authentication Process between Keystone and SSO

As shown in Figure 4-12, the steps are as follows：

1) When the request arrives at the portal, if the request does not carry a TGT authentication ticket, it redirects the user's browser to the SSO login page (steps 1 and 2 in the figure). After passing the SSO authentication, it returns a TGT to the client browser (steps 3 and 4 in the figure), and the page is redirected to the original URL. When the request again arrives at the portal, it will now be carrying the ticket and TGT (the ticket will no longer be used for local authentication). The portal sends the TGT to keystone to get a token, and keystone sends the TGT to verify the SSO. Keystone returns the token after verification.

2) When a request with TGT is reached, the token is requested directly from the keystone. Keystone forwards the TGT to the SSO for authentication (steps *2 and *3 in the figure) and returns the token to the portal after authentication.

3) With keystone integrated SSO certification, the user can use the verified token to further operate the OpenStack platform resources until the token expires after the use of TGT re-get token. If the TGT has expired, it will jump to the SSO login page for recertification.

4.7. High Manageability

High manageability requires the system to accept various management and maintenance operations and minimize the impact of management operations on system architecture and business processing. For example, the system should provide a clear and concise management interface that can actively command, manage, and monitor data interfaces, actively respond to alarms, and import and export reports. The following sections elaborate on these areas from the perspective of the network policy group.

4.7.1. Concise Portal

UnionPay cloud provides a unified, clear, and simple management interface. The administrator can easily view and manage computing resources, storage resources, network resources, resource orchestration, management, global management, policy management, and more.

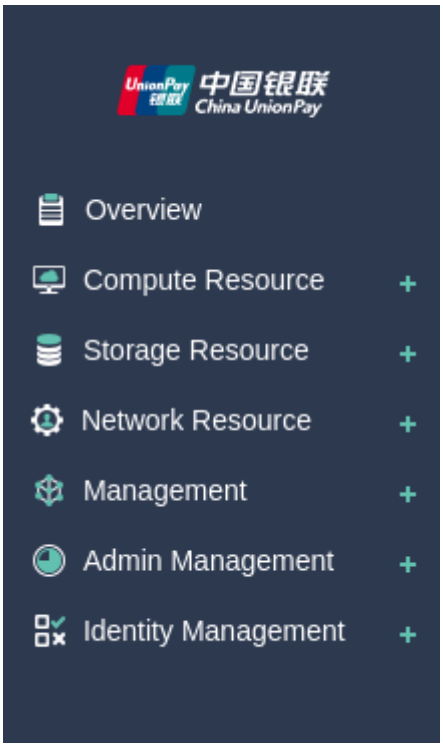

Figure 4-13 Management Portal UI

4.7.2. Network Policy Group

UnionPay cloud uses the group-based network policy to meet the network control management needs of virtual machines. Administrators can easily apply the same network control policy on multiple virtual machines within the same policy group and specify network access rules between multiple groups, which in turn apply to all virtual machines within the group. As shown in Figure 4-14, the main function is composed of network container, subnet, network group, group rules, and so on.

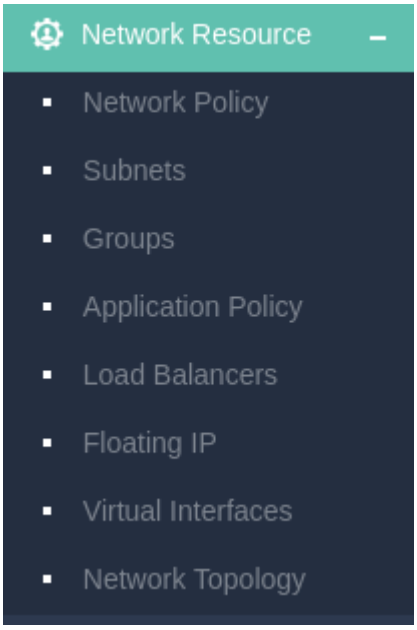

Figure 4-14 network resource UI

We can examine the “Create Rule Application” as an example of applying a group rule using this interface:

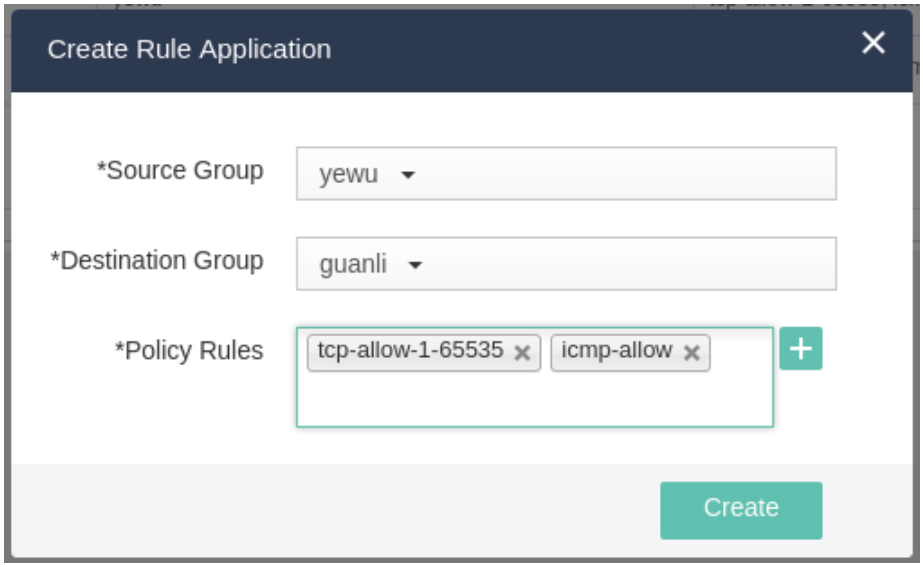

Figure 4-15 group rule apply UI

Here are the steps an administrator would use:

1) Create a new routing domain in the network container, which is isolated from other routing domains by default.

2) Create a broadcast domain in the routing domain in the network container, instructing the platform to allocate the corresponding Layer 2 VLAN number.

3) Plan and create subnets under the broadcast domain, including network segment information, DHCP information, etc.

4) Create a new network group and bind the relationship between the network group and the corresponding subnet.

5) Apply the specific access rules between network policy groups, including protocols, ports, etc.

At this point, the network rules have been completed. When creating new virtual machines, you will only need to put the virtual machine into the corresponding network policy group, and it will use that group's network rules attributes, thus implementing virtual machine Access control rules between virtual machines.

5. Conclusion

This paper introduces the construction and application of the UnionPay financial cloud powered by OpenStack. Since its deployment in 2012, UnionPay financial cloud has been stably running and supporting UnionPay's business, meanwhile it has saved IT resource costs and improved our ability to innovate.

In five years, UnionPay has accumulated experience and ability with OpenStack, openly sharing learnings with financial partners such as Bank of Shanghai and expanding influence of OpenStack in financial industry. With the evolution of OpenStack community technology, UnionPay co-operate with many partners to expand the services of UnionPay financial cloud, and provides better services for users, businesses, financial industries and other partners. UnionPay also actively participates in and contributes to the OpenStack community. We welcome partners from all walks to discuss OpenStack with us and promotes its application in the financial industry and in the production environment of other industries.

6. Copyright Information

All materials, including trademarks, design, text, image and any other information, contained in this white paper are copyright by UnionPay, unless otherwise indicated.

Other names and brands may be claimed as the property of others. All rights reserved. This white paper is not to be used for commercial purposes without permission of the copyright holder.