Learning From Real Practice of Providing Highly Available Hybrid Cloud Service with OpenStack Neutron

Kazuhiro MIYASHITA
FUJITSU LIMITED
PLATFORM SOFTWARE BUSINESS UNIT
PLATFORM SOFTWARE DIV.I
DEVELOPMENT DEPT.III

* Note: Information in this document is subject to change without notice
* Note: Please refrain from making audio recordings
Agenda

- Introduction to FUJITSU’s new cloud platform
- Our challenge about OpenStack Neutron
- Contribution to OSS community
- Summary
Digital Business Platform MetaArc

- Supports migration of traditional core system SoR (Systems of Record) onto the cloud
- Supports SoE (Systems of Engagement) using new technologies such as IoT and AI

Traditional information system (SoR) → Digital transformation in business (SoE)

Cloud (K5)

Mobile  IoT  Analytics  AI  Security

AI : Artificial intelligence
SoR : Systems of Record (Systems for business processing and recording)
SoE : Systems of Engagement (Systems to engage with people and assets)
The New Core Cloud Platform "K5"

K = Knowledge, 5 = 5 continents

- New cloud service combining FUJITSU's know-how and open source technology
- Provision of IaaS/PaaS functions that support SoR and SoE

SoR

Fujitsu's know-how
- development know-how
- Company-wide application

FUJITSU Cloud Service K5
- IaaS
- PaaS

SoE

Open source technology
- OpenStack
- Cloud Foundry

Copyright 2016 FUJITSU LIMITED
Why we chose OpenStack

- Openness
  - used by numerous companies
  - providing tremendous advantages
  - interop of APIs between OpenStack clouds

- Incorporating advance of technology
  - Engineers from all over the world are always adding function, stability

- Hybrid Cloud
  - The combination of Private and public cloud is the best model
  - same API should be used in public cloud and private cloud
Challenges Regarding K5

**K5 Requirements**

- set up high availability business systems to multiple sites in preparation for large-scale disasters

**Challenges for OpenStack/Neutron**

- **[Availability]** continuing business when a data center is damaged

  OpenStack supports Availability zone (AZ), but OpenStack controller including Neutron can not be isolated in each AZs

  **Challenge (1): mechanism that isolates OpenStack controller in each AZs and manages them**

- **[Support]** taking action promptly in case of trouble

  checking OpenStack logs requires time and effort (distributed to many nodes)

  **Challenge (2): Troubleshooting tools that support the distributed architecture of OpenStack**
Region and Availability Zone

- **Region**
  - Computer equipment which are located in a certain geographic range

- **Availability zone**
  - Units to share computer equipment and control plane, facilities for deliver our cloud service

### Japan East Region

- **AZ1**
  - server
  - Network
  - storage

- **AZ2**
  - Open Stack
Functions Added to standard Neutron

**Availability:** mechanism that isolates OpenStack controller in each AZs and manages them

- A manager to manage multiple AZs
- Mechanisms for connection and sharing of resources between AZs
  - Network between AZs
  - Security Groups
- LBaaS straddling AZs (LB-like AWS is desired)

**Support:** Troubleshooting mechanism that support the distributed architecture of OpenStack

- Automatic retrieval of troubleshooting data from multiple nodes
- Mapping of physical and logical networks
- Improving logging
Challenge (1): mechanism that isolates OpenStack controller in each AZs and manages them
Even if an AZ is down, other AZs continue K5 service

- System/User Resources for K5 service are distributed over multiple AZs.
- But, OpenStack resource management functions is limited to an AZ
Sharing resources between AZs using M-AZ manager

AZs not connected

AZs connected

- When connections between AZs are not possible...
  - As user resources are managed separately in each AZ, this restricts usability
    - security groups, auto-scaling

- Sharing and sync of various user resource setting between AZs using M-AZ manager
  - Users do not need to consider the locations of resources
Modeling about inter-AZ connection

- **Network Connector**
  - Logical resource that abstracts various different network connection between multi sites

- **Network Connector Endpoint**
  - Logical resource representing endpoint of network connector
  - Abstracting various connection methods
    - Inter AZ connection (closed in K5)
    - Connects between K5 AZs and customer’s network for hybrid cloud

![Diagram of Network Connector and Endpoints]
The Flow for Connecting AZs – extends Neutron API -

1. Create a net connector
   - POST $NET/v2.0/network_connectors

2. Create a net connector endpoint (AZ1)
   - POST $NET/v2.0/network_connector_endpoints

3. Connect a port to the net connector endpoint of AZ 1
   - PUT $NET/v2.0/network_connector_endpoints/$NC_EP_ID/connect

4. Create a net connector endpoint (AZ 2)
   - POST $NET/v2.0/network_connector_endpoints

5. Connect a port to the net connector endpoint of AZ 1 (AZ 2)
   - PUT $NET/v2.0/network_connector_endpoints/$NC_EP_ID/connect
Improvements to connections between AZs

■ Invisible backend connection between AZs
  ▪ Connect the Neutron virtual router with the physical router. Connection complexity are hidden

■ status monitoring for communications between AZs
  ▪ Introduce a mechanism for monitoring communication errors (ex: bit errors) to immediately switch the route
Load Balancer Service on multi AZs

Create an LB. Deploy VMs in AZ 1 and AZ 2

Network A

- LB
- LB-VM1
- LB-VM2
- VM 1
- VM 2
- VM 3
- VM 4

Network B

- Connect the network between AZs using Network Connector
- Users' system using LB can continue even though AZ downs
- Supports scaling out backend VMs and LB itself
Architecture of LB Services

- LB Manager operates the OpenStack APIs, creates LB-VMs, and sets Security Groups
- Integrate HAProxy based on Nova-VM to provide the LB function in each subnet
  - Neutron’s LBaaS(v1,V2) didn’t fit our customers requests

Resource monitoring

Ceilometer

Operation Applications/Services

Add or delete VMs in each subnet

LB VM

HAProxy

LB Manager

VM Instance

VM Instance
Points of Architecture (Why VM?)

- Easy Maintenance
  - For LBaaS(v1, using network namespace), the entire kernel of the network node must be updated
  - For a VM type, update can be performed for each VM

- Easy Upgrade
  - Upgrade can basically be performed simply by providing a VM image of the upgraded version

- Easy to Follow Upgrades of OpenStack
  - Example: When upgrading from icehouse to kilo, simply perform live migration of VMs from the compute node of icehouse to that of kilo

When providing network services in which OpenStack has been extended, it is recommended to use the VM (or container) method
Challenge (2): Troubleshooting mechanism that support the distributed architecture of OpenStack
I. When a communication error occurred, the Neutron team was deluged with requests for troubleshooting
   - Even after the network was virtualized using Neutron, as network components and routers were not changed, there were many inquiries from users who lack detailed knowledge

II. The following mapping process is difficult
   - Which node are the virtual network resources of Neutron (router/DHCP/port, etc.) deployed?
   - Which layer did the communication error occur on? (L2/L3/L4...)

III. Lack of logging
   - service controller (ex: DBaaS) uses LB service and Neutron’s firewall, Security Group internally. admin needs investigate if network packets are dropped. But, Neutron’s function don’t supports logging

We solved I and II with “dump viewer”, solved III with improving logging.
Dump Viewer

- Collects and integrates the information retrieved from the Neutron DB, the Compute/Network Nodes
- Entering a resource ID from the Web screen displays the connection relationships of resources
- In failures, the impact on customer can immediately be understood
### Configuration Validators

Detects whether the configuration is correct by integrating the Neutron DB information and compute/network node information.

<table>
<thead>
<tr>
<th>Name</th>
<th>Result</th>
<th>Progress</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network not double hosted checker</td>
<td>Validate Method with Error</td>
<td>0 %</td>
</tr>
<tr>
<td>IP Address Field with Invalid char checker</td>
<td>OK</td>
<td>100 %</td>
</tr>
<tr>
<td>NetworkConnectorEndpointProfile existence checker</td>
<td>OK</td>
<td>100 %</td>
</tr>
<tr>
<td>Some checker 1</td>
<td>Validate Method with Error</td>
<td>10/987</td>
</tr>
<tr>
<td>nexthop of routes not contain in destination checker</td>
<td>Detect ERROR</td>
<td>100 %</td>
</tr>
<tr>
<td>DHCP Port per Network checker</td>
<td>Detect ERROR</td>
<td>100 %</td>
</tr>
<tr>
<td>LNetdev must exist for Router with external_gateway_info</td>
<td>Detect ERROR</td>
<td>100 %</td>
</tr>
<tr>
<td>Some checker 2</td>
<td>Detect ERROR</td>
<td>100 %</td>
</tr>
</tbody>
</table>
Improving logging

- We use standard Neutron’s firewall and security group. And the “iptables” is used to them
- But, standard implementation don’t record traffic logs
- We implemented log mechanism into Neutron

UTC-19
Our upstream activity about Neutron on the topic of this presentation

- **Multi AZs management**
  - Key technology: **Multi AZs Security Group (SG)**
    (aiming that the user doesn’t need to consider AZ boundary)
    - FUJITSU has already proposed and works on it.
      - [https://bugs.launchpad.net/neutron/+bug/1534458](https://bugs.launchpad.net/neutron/+bug/1534458)
      - [https://bugs.launchpad.net/neutron/+bug/1586352](https://bugs.launchpad.net/neutron/+bug/1586352)
    - We focus on performance improvement for more large scale cloud

- **FW/SG logging**
  - Logging which packet is passed or dropped.
    - For troubleshooting and security audit.
  - FUJITSU has proposed and leads this function.
    - [https://bugs.launchpad.net/neutron/+bug/1468366](https://bugs.launchpad.net/neutron/+bug/1468366)
Summary

Based on our experiences in K5, here we introduce our approaches to the challenges of OpenStack/Neutron

- **Availability**: mechanism that isolates OpenStack controller in each AZs and manages them
  - Multi-AZ Manager
    - Sharing of connections and resources between AZs (Network Connectors, Security Groups, AutoScale, etc.)
  - Load balancers educating ability of AZs

- **Support**: Troubleshooting mechanism that support the distributed architecture of OpenStack
  - Dump Viewer
    - Automatic retrieval of troubleshooting data from multiple nodes
  - Firewall and Security Group logging improvement

Fujitsu will continue to contribute to the community in the domain of SDN that uses OSS such as Openstack
About Fujitsu Booth

Fujitsu Booth

Fujitsu shares the emerging technology and trends. Please come and experience the future innovation of technology with us. Find out what Fujitsu delivers you today.

Cloud Monitoring software for OpenStack (based on Monasca)
- Fujitsu Software ServerView Cloud Monitoring Manager

Cloud Service Management software (Open Source Software)
- Fujitsu Software Enterprise Service Catalog Manager

Fujitsu booth is at corner of the foyer. We are looking forward to see you in our booth.
Fujitsu

shaping tomorrow with you