A loosely coupled scalable cloud infrastructure

T Cui, Y D Cheng, Z J Cheng, X W Jiang
IHEP, 19B Yuquan Road, Beijing, 100049 China
Cuit@ihep.ac.cn

Abstract. Large-scale virtual computing system requires a loosely coupled virtual resource management solution. The solution should be able to flexibly increase or decrease virtual resources and to smoothly upgrade the cloud platform. OpenStack provides large-scale virtualization solution such as Cells v2.0 and Tricircle/Trio2o. But it’s so complex and difficult to be deployed and maintain for small cloud computing teams. We discuss a loosely coupled cloud infrastructure. It is based on a central database and plugin to achieve information collection, virtual machine scheduling and controlling and uses a simplified method to achieve unified management of network and images. It can flexibly expand or reduce cluster members, deploy different versions of OpenStack and even a heterogeneous cloud platform. The test bed works well at IHEP. We will show and analyse the infrastructure.

1. Introduction
IHEPCloud has been in production since 2014. The platform is dedicated to achieve virtualization of physical computing resources and to provide virtualized computing resources for the HPC computing environment. The IHEPCloud is built based on the OpenStack platform and uses GlusterFS [1] storage system to be the backend storage. The virtualized computing system consists of HTCondor [2] scheduler and VManager middleware, CVMFS:AFS:EOS [3] storage system and OpenStack and so on. VManager is a self-developed middleware and dynamically provides HTCondor with virtual resources based on the status of job accumulation in the queue. At the end of July 2017, Physics resources increased to 2K CPU cores. The size of EOS storage system is about 1.34PB. It serves more than 100 users and 4 scientific projects (BESIII, JUNO, LHAASO and CEPC).

As the development of local and cross-regional science projects at IHEP, local computing resources and remote computing resources have grown rapidly. Remote resources had been connected to IHEP through limited network bandwidth. We urgently need an expansion solution for cloud infrastructure. The solution should solve the problems of large-scale resources management, and cross-regional resources integration which are connected through limited network bandwidth. The solution should have a hierarchical and loosely coupled architecture. It also should be easy operation and maintenance.

Cells v2.0 [4] and Tricircle/Trio2o [5] [6] are large-scale virtualization solutions provided by OpenStack. Cells are better than Tricircle/Trio2o because of successful deployment experience at CERN. But it's also difficult to be deployed. First, Cells is complex and a tightly coupled solution. The cloud architecture isn't flexible. It is difficult to be deployed and maintain by a small cloud computing team. Second, RabbitMQ clusters are very sensitive to latency. It's unsuitable for network of limited bandwidth and unstable latency. Cells would be unstable for cross-regional cloud. Third, the performance of OpenStack goes down as computing resources increasing and frequent scheduling. A single OpenStack causes some problems, such as the risk of increasing, reducing or addition physical nodes, ceilometer component performance degradation, single point failures, etc. These problems may affect the operation of Cells.
We discuss a hierarchical architecture of loosely coupled OpenStack cluster. The core control layer consists of a central database and plugin. The plugin installed on the OpenStack sites weakly connect to central database through REST API interface. The architecture could simply and conveniently extend or shrink cloud platform by adding or reducing OpenStack unit. We will discuss the detail in this paper.

2. The VPManager middleware

The VPManager middleware [7] consists of five components, VCondor, VQuota, VMCtrl, NETDB and Image Mgmt. The VCondor and VQuota are responsible for virtual machine scheduling and quota controlling. The VMCtrl and NETDB are responsible for virtual machines controlling and environment configuration. Image Mgmt provides images and is responsible for image updating. The architecture is like Figure 1.

![Figure 1. The VPManager Architecture](image)

Figure 1. The VPManager Architecture

The running experience of the VPManager has shown us the feasibility to build an OpenStack cluster which controlled by central database. The central database just focuses on global data such as physical resources and virtual resources in this architecture. It doesn’t focus on the OpenStack itself. The NETDB component is a bridge which connects central database and OpenStack and at the same time isolating them. If the NETDB plugin can collect virtual machine’s information and execute operation such as launching or destroying virtual machine, it will be the only connection between central database and OpenStack. This is a loosely coupled relationship. We can build a Hierarchical, loosely coupled system architecture based on a central database and the NETDB plugin. Figure 2 shows the model.

![Figure 2. A Hierarchical, Loosely Coupled Model](image)
In this model, a timestamp will be set to check each OpenStack site is alive or not. The NETDB plugin must refresh the timestamp to show an activity of the site. To start the NETDB plugin means an OpenStack site add to the cloud cluster. To terminate the NETDB means that the corresponding OpenStack site exits the cloud cluster. When the site exits cloud cluster, virtual machine on the site still running until the job is end. This design can prevent the problem that a site exits the cluster caused by link quality problems. Thus the controlling layer only focuses on centralized data and the global view of resources, the executing layer is responsible for resource controlling. It’s as same as Cells v2.0 in terms of centralized data and resource controlling, but our model is more flexible and convenient and suitable for small team doing operation and maintenance.

The model we designed solves the problem of resources scheduling. To build a loosely coupled cloud infrastructure, a complete solution includes a core controlling component, network, OpenStack automatic configuration, image management and security etc. It always includes shared storage supported, VCondor/VQuota virtual scheduler, VMCtrl and so on. Figure 3 shows the infrastructure. We will discuss the infrastructure in detail below.

3. **A loosely coupled OpenStack cluster**

3.1. *Core controlling component*

The core controlling component is designed according to the loosely coupled model and it includes a central database, a REST API interface and a plugin named NETDB. There are four types of data in this database. The first is the data of OpenStack environment such as flavor, network and image. This information will help the system sync OpenStack configuration when a new OpenStack is deployment. The second is resources data which include physical nodes and all virtual machines. It’s collects by NETDB and provides service for scheduler. The third is scheduling data which added by resources scheduler (VCondor) and executed by NETDB to create or destroy virtual machine. The last is security data for security control and account management. The core controlling component provides a REST API interface to achieve data uploading or downloading. The web portal is based on HTTPS service and PHP script.

NETDB component is developed by C language. It is multi-versioned to accommodate the corresponding version of OpenStack. It’s deployed on the OpenStack controller node and responsible for refreshing timestamp, collecting data such as number and state of virtual machines, physical and virtualized resources amount and free resources amount, and sending all information to central
database. It also executes virtual resource operations such as launching or destroying virtual machine when the system gets task from central database.

3.2. Network design
A simplified network topology is deployed in IHEPCloud. 802.1Q protocol and linuxbridge mode are configured on neutron component and physical network switch. All rules on security-group are opened and subnets directly connect to physical network. The subnet gateway is configured in physical switch. These operations ensure smallest loss of network transmission efficiency. Virtual machine’s IP address is valid local address. VMs can be accessed as same as physics hosts. Job-scheduler (HTCondor) can directly access all of the virtual machines. Cloud node interconnects each other through only one NIC. We continue to use this network design in multiple OpenStack cluster environment. IP address utilization should be fully planed and configured to central database. Each OpenStack gets theirs own subnets configuration from central database and maintain theirs DHCP service and broadcast area. New infrastructure does not focus on private network of tenants because it services HPC system. Virtual machines (UI and WN) are a part of the HPC system. But testing virtual machines should be isolated from HPC environment for security.

3.3. OpenStack automatic configuration
We design automatic configuration mechanism in OpenStack cluster environment to avoid manually configured errors. It is based on the central database and applied by the NETDB component. When an OpenStack is installed, NETDB will be running to register this site, download configuration information from central database and finished the configuration. Any modification will be sync on schedule. The configurations include flavor, network and subnet, images information and account information.

3.4. Image management
Automatic configuration and maintenance are based on the Puppet system in our HPC environment. The computing nodes need to update or put some patches for security and running problems almost every week. This is difficult for virtualized system because virtual machine is dynamical scheduled. Puppet can’t be supported in virtual system because of the sync’s efficiency problem. So we design a web images release system to service the OpenStack cluster environment and other users who build a private cloud. The image models are synced by puppet system as same as physical computing work node on schedule. NETDB synchronizes images to individual OpenStack unit when images are updated. Some urgent updating or patching will be done through the VMCtrl component during two image updating. Another problem of image management is instance’s baking files. This is a very big row image located directory /var/lib/nova/instance/_base/. Normally, one image has multiple copies of backing files in the OpenStack cluster environment based on shared storage. The only difference of copies is name. This is great waste for unified Shared storage. To specify same image ID in the OpenStack cluster environment can solve this problem. Kilo and above version of OpenStack support this operation.

3.5. Security and Accounting
In the loosely coupled infrastructure, security protection includes three aspects. The first aspect is central database and REST API interface access control. The second aspect is OpenStack security protection. The third aspect is account authentication. There are four measures to build security protection in the loosely coupled infrastructure which is shown in Table 1.
measures | description
---|---
Iptables | Isolating services such as MySQL, SSHD etc. Open access for each OpenStack controller.
REST API interface | Hypertext Transfer Protocol Secure and tokens will be used for access security. The REST API interface is open only to the authorized source IP address.
Central database | Local IP address access limitation.
Account authentication | To support unified authentication system.

Table 1. Security measures

4. Summary
The loosely coupled cloud infrastructure aims to solve extension problem on the cloud computing system through a simple, weakly connected system architecture and design. It provides a simple mechanism to control resources and integrate OpenStack clusters. It is different from Cells v2.0 or Tricircle/ Trio2o. It’s more simple and convenient. The loose coupling connection makes the system more convenience to extension or shrink. The hierarchical architecture ensures the separation of control and execution, each layer just focus on itself. Centralized virtual resource view makes global resource-based scheduling possible. This may be something like Cell v2.0. But in this infrastructure, the scheduling algorithm just focuses on the resources themselves but not OpenStack sites. The feature makes cloud system more flexible and reasonable. Now the system is still in the phase of test-bed and the running of test-bed proves that it can work well. In future, we'd like to do more research on the scheduling algorithm such as green energy-saving scheduling, regional scheduling, etc. We have a lot of work to do to further improve the design.

References
[1] GlusterFS Documentation, https://docs.gluster.org/en/latest/
[2] UW-Madison, HTConder, https://research.cs.wisc.edu/htcondor/manual/
[3] CERN IT data storage group, EOS, http://eos-docs.web.cern.ch/eos-docs/intro.html
[4] OpenStack Nova Cells v2, http://docs.openstack.org/developer/nova/devref/cells.html
[5] OpenStack Tricircle, https://wiki.openstack.org/wiki/Tricircle
[6] OpenStack Trio2o, https://wiki.openstack.org/wiki/Trio2o
[7] T Cui, Y D Cheng, The advances in IHEP Cloud facility, https://indico.cern.ch/event/531810/contributions/2331773/, Oct 2016