OpenStack Platform and its Application in Big Data Processing

Cen Shao, Bo Liang*, Feng Wang, Hui Deng, Wei Dai, Shoulin Wei, Xiaoli Zhang and Zhi Yuan
Computer Technology Key Lab of Yunnan Province
Kunming University of Science and Technology
Kun'ming, China
Email: 516913794@qq.com, lb@cnlab.net

***Abstract*—During the past ten years, cloud computing gradually becomes mature which is a current research hotspot. OpenStack is currently one of the most popular open source cloud platforms that developed by Rackspace and NASA. It provides some toolkits to deploy cloud environment and implement cloud infrastructure like Amazon EC2. In this paper, we describe OpenStack architecture in detail and introduce its application in big data processing. This led us to the conclusion that OpenStack is a highly modular and extensible and platform and it is a good alternative if you want to hand large-scale data efficiently.**

***Keywords*-Cloud Computing; OpenStack; IaaS; Open Source; Big Data**

I. Introduction

With the rapid development of the Internet age, big data handling techniques have become an increasingly active topic in the IT scenarios. Big data concept refers to provide rich and complex data which beyond the average computers reach. Big data handling techniques are in high demand of every walk of life. We need a way to hand large-scale data efficiently. Open source cloud platform such as OpenStack, Hadoop, Eucalyptus, and AbiCloud have gained significant momentum in the last few years. Among them, OpenStack has been recognized by most users because of its reliability, elasticity and immense computing power.

OpenStack is an open source software [1] that can be used to build a cloud platform with virtual computing or storage service. OpenStack is also a community intended to provide an IaaS solution that comes with reliability robustness and availability for both public and private cloud infrastructures [2]. OpenStack not only can quickly deploy virtualization environment, but it also can create multiple interconnected virtual servers and enable users to quickly deploy applications on it. It controls large amounts of compute, storage and networking resources throughout a datacenter. Because it has maximum compatibility, nearly any type of cloud environment is supported and anyone can develop, test and run applications or services on it [3]. Since it was creating in 2010, over 200 companies have become involved in the project such as AMD, Canonical, NEC, Dell, Intel, HP, Red Hat [4]. Version Juno was officially published in October 2014 that is the 10th version since the establishment of OpenStack. Compared with the previous version, Juno put forward 342 new features and been supported by more than 133 organizations. This marks that OpenStack fast to the mature cloud platform advance and it has our full support. Now, OpenStack has been increasingly widely applied. OpenStack accounts for more than half of the worlds cloud computing platform use with the utilization of 64 percent, according to the investigation of usage of open source cloud computing. One classic example is uses OpenStack to deal with very large amount of data from all walks of life.

This paper is organized into two parts. Part one of this article will highlight some advantages by presenting three main modules for OpenStack. The second part describes the application of OpenStack in big data processing.

II. The Architecture Of OpenStack

In this section, we introduce the structure of OpenStack. OpenStack is a fully open source cloud platform management project. It consists primarily of three modules with well-defined APIs [5]: OpenStack Computing (Nova), OpenStack Storage (Swift) and OpenStack Image Service (Glance). The connection of three modules is shown in Figure 1. These modules can work cooperatively in a concurrent manner to provide a complete set of infrastructure services. They can also work independently to provide flexible, efficient and steady cloud services including virtualization, cloud storage and image services. It has modular and highly configurable features to ensure that it fits available hardware resources either they be professional or entry level [6]. We begin with an introduction to three main modules in detail, then we will introduce some important auxiliary modules.

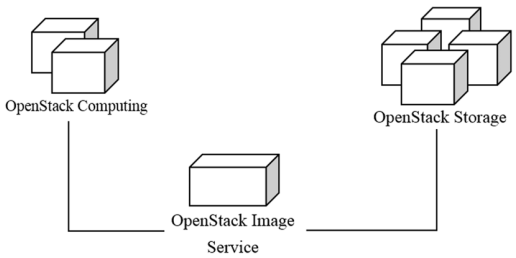

Figure 1. The connection of three modules

978-1-4673-8221-2/15 $31.00 © 2015 IEEE
DOI 10.1109/ICINIS.2015.45

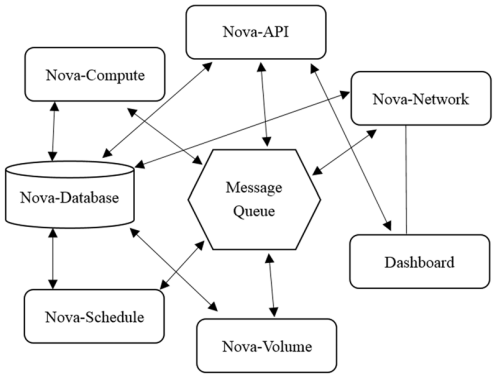

Figure 2. Components of Cloud Computing

A. OpenStack Computing

OpenStack Computing is mainly performed by Nova. It is a cloud computing middleware fabric controller [7], the major part of infrastructure system. Nova allows users to create their own virtual machines which provided us with the deployment and computing services. Nova supports all the life cycles of instances in the cloud and it is in charge of all the procedures required of an instance, including computational resources management, networking and authorization [8]. Nova is made up of six components [9] including Nova-API, Nova-Compute, Nova-Volume, Nova-Network, Nova-Scheduler and Message Queue. Each component can be installed on different server, it means that all components in the architecture follow a shared-nothing [10] policy. The relationships between all these components are shown in Figure 2.

Users interacted with OpenStack Computing through Nova API to manage other components and initialize the majority of deployment activities. OpenStack Computing provides two daemons for each component—one is WSCI application to accept API requests, and other one is a worker process which is responsible for deploying tasks. Individual daemons can exchange information via an AMQP (Advanced Message Queuing Protocol) broker [11] to perform the corresponding API request.

Here is a brief introduction of each component. (1)Nova-API: It is the core of OpenStack Computing which can accept and respond to user calls and control most orchestration activities [12]. It provides the endpoint for all API query and runs an instance and executes some polices. (2)Nova-Compute: It is a compute node that can provide a virtual server upon demand interacting with the hypervisors [13], supporting several kinds such as Xen, KVM, VMware or Hyper-V [2]. It can update the database resource consumption according to the virtual machine requests. It can also create or terminate a daemon. (3)Nova-Volume: It can create, append, cancel or delete a volume and offer persistent storage for an instance. (4)Nova-Network [14]: It controls the entire network by performing tasks received from the message queue. (5)Nova-Scheduler [15]: It provides the scheduling plan for the virtual machine creation, migration, resize, delete. It can also send a request to Nova-Compute through the message queue. (6)Message Queue [16]: It stores messages when communicating in all inter-process communication. (7)Nava-Database [17]: It keeps all kinds of state informations at compile time or run time.

B. OpenStack Storage

OpenStack Storage is mainly performed by Swift. Swift is composed of a Proxy Server, an Object Server, an Account Server, a Container Server and the Ring [18]. It mainly fulfills the function of storing distributed, consistent virtual objects [19] for OpenStack and also allows users to store or retrieve files [20]. A simple deployment of object storage is illustrated in Figure 3. Proxy node is responsible for providing proxy service and storage nodes are responsible for providing object service, account service and container service. It is similar to Amazons S3, we can provide a cloud storage service encapsulation [10]. Swift has the virtue of big storage capacity, scalability, lower redundancy and high fault-tolerance performance. Swift is a separate infrastructure that can be interacted by Object API. It can be used by Cinder to back up the VMs volumes.

Cinder affords OpenStack block storage which is attached to a virtual machine as a storage volume, it can be described as OpenStack Block Storage or Cinder Volumes [21]. It can provide users some services like disk storage. For example, it gives persistent block storage or volumes to guest virtual machines. Cinder can provide a faster storage service than Swift and allow users to use it just like any other regular file system.

C. OpenStack Image Service

OpenStack Image Service design philosophy is to support as many back-end storages and registry databases as possi-

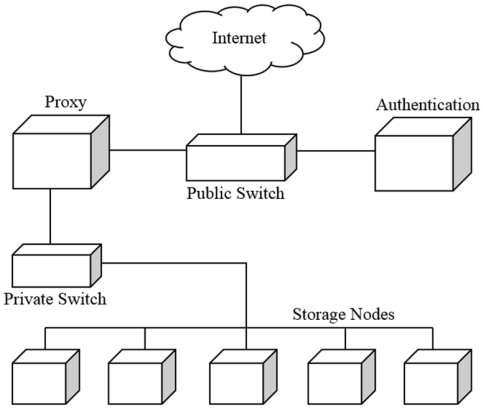

Figure 3. Object Storage Architecture

ble. It is mainly performed by Glance. Glance is an essential service for a basic cloud architecture implementation. It is a system that is responsible for discovering, retrieving and storing the mirror in virtual machine when required by the Nova service [22]. These images can be stored on the local file system and the host where Glance is running, or stored in Swift. It is mainly consist of two parts, API Server and Registry Server. API Server works on interchange of information, including the communication with users and other components. User authentication for image access is performed through keystone [23]. Registry Server is responsible for metadata registries.

The back-end storages supported by OpenStack Image Service are include:

- HTTP: OpenStack Image Service can read as much as VM images available through HTTP on the Internet. The mode is read-only.
- S3: OpenStack Image Service can be stored in Amazon S3.
- File System: It is the default back-end file system of OpenStack Image Service and it will write the image files to its local system.
- OpenStack Object Storage: To store a high-availability data object of OpenStack.

D. Other Modules

In addition to these three core modules, it also contains other modules such as Identity (Keystone) and Dashboard (Horizon). Keystone [18] is the central identity which provides a common authentication and authorization layer and an OpenStack service catalog. It provides compliance between services consequently the cloud solutions can be available for end users. Horizon is a modular web application that provides a graphical interface to administrators and users [24], makes it easier to access compute, storage and various resources required by virtual machines [25].

All the modules of OpenStack are developed by Python. Compared with other programming languages such as Java and c#, Python has a higher parallel processing capabilities and a lower rate of system resources occupation [18]. It is truly a highlight that gives the OpenStack its competitive edge.

III. The Application In Big Data Processing

As the cloud computing technology matured, big data processing can then pass from virtual to real. Then we can see the various data processing tools flowing and flowing fast. Among them, OpenStack has attracted people attention because of its security, reliability, flexibility, high-efficiency etc. OpenStack could hand large-scale data efficiently. In this section, the application of OpenStack in big data computing and storing will be introduced in detail.

A. Application in Computing

Now there are some drawbacks exist in traditional big data computing tools such as the relatively low facilities use rate of storage and computing device, difficult to manage centralized, long time of processing [26]. It could solve the problem with cloud computing management platform like OpenStack. OpenStack Compute module could largely integrate all sorts of software and hardware resources which will enhance the computing efficiency [27]. We can use some normal computers act as compute nodes to hand big data.

Mirantis, which is the well-known company of OpenStack community, had officially declared the open source project Sahara in 2013. Sahara could provide a BDaaS (Big Data as a Service) solution which based on OpenStack. This marks that a dedicated module will be created in the future by OpenStack for big data services.

B. Application in Storage

Compared with the application of OpenStack in computing, there is a further wide of use in storage. The Swift module can provide storage services with large capacity and provide the highest efficiency by lowering resource redundancy [28]. It will make it easy for users to access various resources. Users should create account for themselves before use. There are several containers in each account and multiple objects can be stored in each account. This is called the account-container-object mode [29]. It can satisfy the growing data requirements by providing several extensible storage nodes. Swift can be deployed on a separate server to store big data exclusively. We need to use the Nova during the deployment process which could create instance to store some SQL database [30].

A classic example is the corporate information sharing platform. The platform is built upon information resources, but it has often a lot of various informations which goes against uniform management. The cloud storage offered by OpenStack Storage module can effectively resolve this issue. It can build a storage cluster by using multiple normal computers to memorize massive data efficiently [31]. Meanwhile this service can satisfy the needs of some complex data and information handing, including counting, querying, altering and so on. In this way, we can not only save time in computing, but also can reduce the equipment cost to save the company. OpenStack will be a good choice to provide specific applications for enterprises.

C. Application in Some Practical Fields

In order to show the big data processing based on OpenStack in some practical fields, three examples are given here. A first example is a forensic framework for cloud in OpenStack cloud platform. This framework [32] can obtain live snapshots, image evidences, packet captures and log evidences and do analysis on it. Another example is a solution based on OpenStack about emerging collaboration

issues [33] in an era of software crisis. It presents a new concept of open-coopetition theory in order to better understand on how rival-firms collaborate with competitors by open-source manners. The final example is an OpenStack cloud framework which accelerates research development in the biomedical field [34]. It can help solve the problem of complex biomedical data from different sources demands large storage and high performance computing.

IV. Conclusion And Future Work

In this paper, we describe the architecture of OpenStack and its application in big data processing. The introduction of architecture illustrates the advantage such as security, reliability, flexibility, high-efficiency, expandability of using OpenStack. The introduction of usage in big data processing exhibits the wide applications of OpenStack in every field. In the future, OpenStack is likely to become the standard on the cloud computing platform and the prospect of OpenStack development will be wider.

Of course, OpenStack does pose some problems such as the web interface is short of integrity, insufficient commercial toolkit, excessive request for small businesses. We may be able to blend OpenStack and other cloud tools, which can reach the best effect. For example, we can use the management tools of CloudStack interface to remedy the deficient aspects of OpenStack interface. These are challenges of OpenStack in the future.

Acknowledgment

We are grateful to the anonymous referee for constructive suggestions. Our work is supported by the National Natural Science Foundation of China (grant Nos.11203011, U1231205, 11163004, 11103005, 11403009), Natural Science Foundation of Yunnan Province, China (grant Nos.2013FA013, 2013FA032, 2013FZ018) and Educational Commission of Yunnan Province, China (grant Nos.2013Y320).

References

[1] F. Callegati, W. Cerroni, C. Contoli, et al, "Performance of Network Virtualization in cloud computing infrastructures: The OpenStack case," pp. 132-137.

[2] T. Rosado, and J. Bernardino, "An overview of openstack architecture," in Proceedings of the 18th IDEAS, Porto, Portugal, 2014, pp. 366-367.

[3] E. Qevani, M. Panagopoulou, et al, "What Can OpenStack Adopt from a Ganeti-Based Open-Source IaaS?," pp. 833-840.

[4] G. Robles, J. M. González-Barahona, and C. Cervigón, "Estimating development effort in Free/Open source software projects by mining software repositories: a case study of OpenStack," in Proceedings of the 11th IMSR, Hyderabad, India, 2014, pp. 222-231.

[5] F. Wuhib, R. Stadler, and H. Lindgren, "Dynamic resource allocation with management objectives: implementation for an OpenStack cloud," in Proceedings of the 8th CNSM, Las Vegas, Nevada, 2013, pp. 309-315.

[6] H. Bin, and Y. Hong, "Research of Scheduling Strategy on OpenStack," pp. 191-196.

[7] R. H. Khan, J. Ylitalo, and A. S. Ahmed, "OpenID authentication as a service in OpenStack," pp. 372-377.

[8] L. Beernaert, M. Matos, R. Vilaça, and R. Oliveira, "Automatic elasticity in OpenStack," in Proceedings of the ISDMCMM, Montreal, Quebec, Canada, 2012, pp. 1-6.

[9] R. Nasim, and A. J. Kassler, "Deploying OpenStack: Virtual Infrastructure or Dedicated Hardware," pp. 84-89.

[10] W. Xiaolong, G. Genqiang, L. Qingchun, et al, "Comparison of open-source cloud management platforms: OpenStack and OpenNebula," pp. 2457-2461.

[11] J. Á. L. d. Castillo, K. Mallichan, et al, "OpenStack Federation in Experimentation Multi-cloud Testbeds," pp. 51-56, 2013.

[12] A. F. Thaha, M. Singh, A. H. M. Amin, et al, "Hadoop in OpenStack: Data-location-aware cluster provisioning," pp. 296-301.

[13] F. Callegati, W. Cerroni, C. Contoli, and G. Santandrea, "Performance of multi-tenant virtual networks in OpenStack-based cloud infrastructures," pp. 81-85.

[14] O. Litvinski, and A. Gherbi, "Openstack scheduler evaluation using design of experiment approach," pp. 1-7.

[15] H. Yu, W. Shi, and T. Bai, "An OpenStack-Based Resource Optimization Scheduling Framework," pp. 261-264.

[16] A. TaheriMonfared, and M. G. Jaatun, "As Strong as the Weakest Link: Handling Compromised Components in OpenStack," pp. 189-196.

[17] L. Foschini, A. Pernafini, A. Corradi, et al, "A performance evaluation of TopHat RNA sequences alignment tool on openstack-based cloud environments," pp. 358-365.

[18] OpenStack Home Page.http://www.openstack.org/.

[19] M. Bist, M. Wariya, and A. Agarwal, "Comparing delta, open stack and Xen Cloud Platforms: A survey on open source IaaS," pp. 96-100.

[20] D. Sitaram, P. Hallymysore, S. Harwalkar, et al, "OpenSim: A Simulator of OpenStack Services," pp. 90-96.

[21] Y. Zhang, R. Krishnan, et al, "Secure Information and Resource Sharing in Cloud Infrastructure as a Service," in Proceedings of the 2014 ACM WISCS, Scottsdale, Arizona, USA, 2014, pp. 81-90.

[22] S. Ristov, M. Gusev, and A. Donevski, "Security Vulnerability Assessment of OpenStack Cloud," pp. 95-100.

[23] C. A. Lee, and N. Desai, "Approaches for Virtual Organization Support in OpenStack," pp. 432-438.

[24] X. Ju, L. Soares, K. G. Shin, K. D. Ryu, et al, "On fault resilience of OpenStack," in Proceedings of the 4th annual SOCC, Santa Clara, California, 2013, pp. 1-16.

[25] S. S. Sahasrabudhe, and S. S. Sonawani, "Comparing openstack and VMware," pp. 1-4.

[26] J. Teixeira, "Developing a cloud computing platform for Big Data: The OpenStack Nova case." pp. 67-69.

[27] S. Bonner, C. Pulley, I. Kureshi, et al, "Using OpenStack to improve student experience in an H.E. environment," pp. 888-893.

[28] A. Y. S. Tan, R. K. L. Ko, and G. P. Y. Ng, "OpenStack Café: A Novel Time-Based User-centric Resource Management Framework in the Cloud," pp. 422-429.

[29] X. Qifeng, and Y. Jie, "A Study on Service Performance Evaluation of Openstack," pp. 590-593.

[30] A. Paradowski, L. Lu, and Y. Bo, "Benchmarking the Performance of OpenStack and CloudStack," pp. 405-412.

[31] S. A. Baset, "Open source cloud technologies," in Proceedings of the Third ACM SOCC, San Jose, California, 2012, pp. 1-2.

[32] S. Saibharath, and G. Geethakumari, "Design and Implementation of a forensic framework for Cloud in OpenStack cloud platform," pp. 645-650.

[33] J. Teixeira, "Understanding Coopetition in the Open-Source Arena: The Cases of WebKit and OpenStack," in Proceedings of the OPENSYM, Berlin, Germany, 2014, pp. 1-5.

[34] C. Mazurek, J. Pukacki, M. Kosiedowski, et al, "Federated clouds for biomedical research: Integrating OpenStack for ICTBioMed," pp. 294-299.