Bare Metal Provisioning to OpenStack Using xCAT

Jun Xie
Network Information Center, BUPT, Beijing, China
Email: jeffreycohobupt@gmail.com

Yujie Su, Zhaowen Lin, Yan Ma and Junxue Liang
Network Information Center, BUPT, Beijing, China
Email: {suyj, linzw, mayan, liangjx}@bupt.edu.cn

***Abstract*—Cloud computing relies heavily on virtualization technologies. This also applies to OpenStack, which is currently the most popular IaaS platform; it mainly provides virtual machines to cloud end users. However, virtualization unavoidably brings some performance penalty, contrasted with bare metal provisioning. In this paper, we present our approach for extending OpenStack to support bare metal provisioning through xCAT(Extreme Cloud Administration Toolkit). As a result, the cloud platform could be deployed to provide both virtual machines and bare metal machines. This paper firstly introduces why bare metal machines are desirable in a cloud platform, then it describes OpenStack briefly and also the xCAT driver, which makes xCAT work with the rest of the OpenStack platform in order to provision bare metal machines to cloud end users. At the end, it presents a performance comparison between a virtualized and bare-metal environment**

***Index Terms*—cloud computing, IaaS, OpenStack, xCAT, bare metal provisioning**

I. BACKGROUND AND MOTIVATION

A. Background

OpenStack [1] is an Infrastructure as a Service(IaaS) cloud computing project started by Rackspace Cloud and NASA in 2010. Currently more than 200 companies have joined this project,including Intel, IBM, Dell and Red Hat. It is free open source software released under the Apache 2.0 license. It is included and released in both the Ubuntu and Red Hat Linux distributions. The OpenStack project aims to deliver solutions for all types of clouds by being simple to implement, massively scalable, and feature rich. It delivers various components for a cloud infrastructure solution, which makes it a somewhat open sourced Amazon Web Services(AWS), which is the most successful cloud computing platform at present.

In OpenStack, cloud providers most basically offer virtual machines. The virtual machines are run as guests by a hypervisor, such as Xen or KVM. Management of pools of hypervisors by the cloud operational support system leads to the ability to scale to support a large number of virtual machines. In this sense, OpenStack is mainly built on virtualization technologies (Xen, KVM, etc.).

B. Motivation

The adoption of virtualization in cloud environment has resulted in great benefits, however, this has not been without attendant problems,such as unresponsive virtualized systems, crashed virtualized servers, mis-configured virtual hosting platforms, performance tuning and erratic performance metrics, among others. Consequently, we might want to run the compute jobs on "bare metal": without a virtualization layer, Consider the following cases:

Case 1: Your OpenStack cloud environment contains boards with Tilera processors (non-X86), and Tilera does not currently support virtualization technologies like KVM. Obviously, a Tilera board cannot be provisioned to an end user using traditional virtualization technologies. Thus, we need to add a bare-metal provisioning driver in OpenStack in order to provision a Tilera board or other non-virtualizable boards in an OpenStack environment.

Case 2: You want to achieve higher performance in the cloud platform, as virtualization obviously brings some performance penalty.

Case 3: You have a lot of relatively old and low-performance machines (e.g. 1 CPU, 1G memory) in the data center. In this case, if you still use virtualization technologies and then provide virtual machines to end users, the performance penalty brought by virtualization is obviously too much for the machine. Bare metal provisioning is optimal in this scenario.

From these cases among others, we can conclude it would be great for OpenStack if it could support bare-metal provisioning, in addition to the traditional virtualization. This is because it can now offer to cloud end users a hybrid IaaS cloud platform with both choices of virtual machines and bare metal machines.

The intention is obviously to ultimately support different provisioning back-ends in order to support different bare-metal architectures. Several provisioning tools are available, such as Dell's crowbar [2], as an extension of opscode's Chef system [3], Argonne National Lab ' s Heckle [4], xCAT [5], Perceus [6], OSCAR [7], and ROCKS [8]. These tools provide different bare-metal provisioning, deployment, resource management, and authentication methods for different architectures. These tools use standard interfaces such as PXE (Preboot Execution Environment) [9] boot and IPMI (Intelligent Platform Management Interface) [10] power cycle management module. For boards that do not support PXE and IPMI, such as the TILEmpower board, specific back-ends must be written.

In this paper, we use xCAT to extend OpenStack, the most popular IaaS platform at present. xCAT (Extreme Cloud Administration Toolkit) is an open-sourced and distributed computing management tool. It is a wonderful tool to remotely control the machines and can also be used to network-install a machine, automated and unattended. xCAT supports many usual operating systems, including SLES, OpenSUSE, RHEL, CentOS, Windows, etc. As for the hardware, it supports X86, iDataplex, IBM's system X, system P, and most IPMI-based machines.

To support bare metal provisioning in OpenStack through xCAT, we implemented an xCAT driver.

II. OPENSTACK

OpenStack is a collection of open-sourced software components designed for building public and private clouds. OpenStack provides similar functionality to other open-source cloud projects such as Eucalyptus [11], OpenNebula [12], and Nimbus [13]. The main components include OpenStack Compute (codenamed Nova), OpenStack Image Service (codenamed Glance), and OpenStack Object Storage (codenamed Swift).

OpenStack is implemented as a set of Python services that communicate with each other via message queue and database. Fig. 1 shows a conceptual overview of the OpenStack architecture, with the OpenStack Compute components bolded and OpenStack Glance components (which stores and manages all the images) shown in a lighter color.

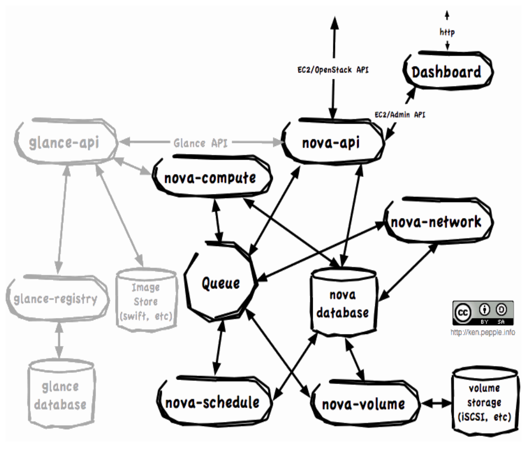


Figure 1. OpenStack Architecture. Credit: Ken Pepple [14]

The nova-api service is responsible for fielding resource requests from users. Currently, OpenStack implements two APIs: the Amazon Elastic Compute Cloud (EC2) API [15], as well as its own (OpenStack) API.

The nova-schedule service is mainly responsible for scheduling compute resource requests on the available compute nodes.

The nova-compute service is responsible for starting and stopping all the compute instances.

The nova-network service is responsible for managing IP addresses and virtual LANs for the instances.

The nova-volume service is responsible for managing network drives that can be mounted to running instances.

The queue is a message queue implemented on top of RabbitMQ [16] which is used to implement remote procedure calls as a communication mechanism among the services.

The dashboard implements a web-based user interface.

The database is a traditional relational database such as MySQL that is used to store persistent data that is shared across the components.

III. XCAT

xCAT (Extreme Cloud Administration Toolkit) is an open source scalable distributed computing management and provisioning tool that provides a unified interface for hardware control, discovery, and OS diskful/diskfree deployment. This robust toolkit can be used for the deployment and administration of huge clusters.

xCAT is a typical client/server architecture. The heart of its architecture is the xCAT daemon (xcatd) on the management node. This receives requests from the client, validates the requests, and then invokes the operation. The xcatd daemon also receives status and inventory information from the nodes as they are being discovered and installed/booted. Thus, the xcatd is the worker that really manages all the nodes in the cluster.

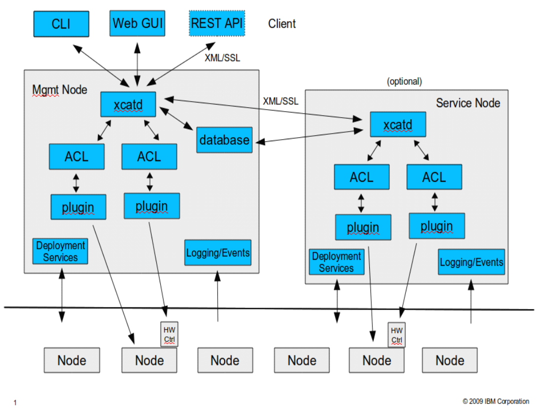


Figure 2. xCAT Architecture. Credit: xCAT[17]

IV. XCAT INTEGRATION INTO OPENSTACK

To support bare metal provisioning to OpenStack through xCAT, we implemented the xCAT driver, which receives requests from OpenStack Nova, deals with the messages and finally transfers the requests to xCAT.

A. The Logical Architecture

In OpenStack, it uses libvirt library to manage different virtualization technologies. For bare metal provisioning, we need a bare-metal driver, as an alternative to the libvirt driver. For a compute node with bare metal driver, we set compute_driver to nova.virt.baremetal.driver.BareMetalDriver in its nova configuration file (/etc/nova/nova.conf by default). As depicted in Fig. 3,when an end user requests a bare-metal machine (e.g.,a System X machine),a scheduler chooses a nova compute node,whose compute_driver is nova.virt.baremetal.driver.BareMetalDriver and at the same time baremetal_driver is xcat. And then the xCAT bare-metal driver will take care of everything and finally start the system with the specified operating system. If what the user requested is a Tilera machine, then the scheduler will choose a nova compute node who manages Tilera boards.

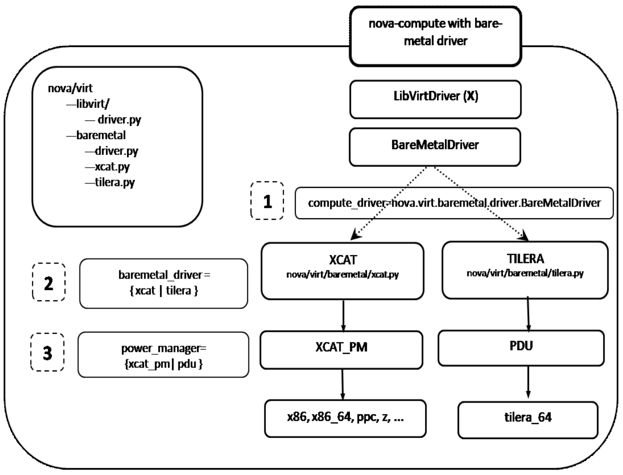

nova-compute with bare-metal driver
nova/virt
—libvirt/
— driver.py
—baremetal
—driver.py
—xcat.py
—tilera.py
LibVirtDriver (X)
BareMetalDriver
1
compute_driver=nova.virt.baremetal.driver.BareMetalDriver
2
baremetal_driver = {xcat | tilera }
XCAT nova/virt/baremetal/xcat.py
TILERA nova/virt/baremetal/tilera.py
3
power_manager= {xcat_pm| pdu }
XCAT_PM
PDU
x86, x86_64, ppc, z, ...
tilera_64

Figure 3. Logical Architecture

B. The Physical Architecture and Our Solution

As depicted in Fig. 4, In our environment, we have a compute node with xCAT driver in SERVER1.Besides nova-compute, xCAT client is also installed in it. We installed Glance (the Image Service) and xCAT server in SERVER2.In addition, a cluster of bare metal machines (in our case, System x3650) are also available.

Other OpenStack components like nova-scheduler and keystone (Identity Service) are not shown in the figure; they can just be in any machine if they are configured correctly according to OpenStack Installation procedure.

As shown in Fig. 4, when a request comes asking for a bare metal machine (if not for tilera), nova-compute will invoke xCAT driver, who will later communicates with xCAT client. And xCAT client will communicates with xCAT server to finally complete the request (start/reboot/shutdown/terminate the instance).

As for the virtual machines, there is a data base for the bare metal machines. This is mainly to record all the information about the registered bare-metal machines, with the mac address and BMC address (In our case, IMM address) being some of the most important information.

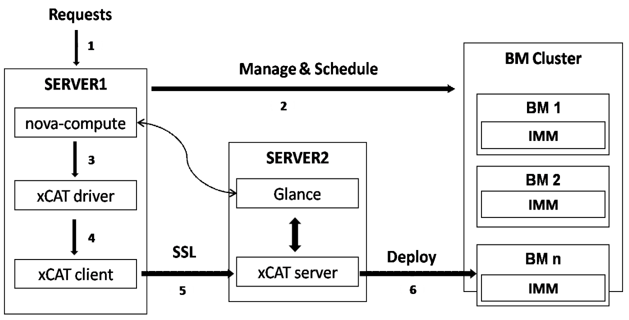


Figure 4. Physical Architecture

C. The Test Workflow

(1) we used devstack to install all the nova services in SERVER1, which runs an ubuntu12.04 operating system. Of course, we have to put our xCAT driver in this nova compute node. Also, the xCAT client is installed in this node, to be invoked by our xCAT driver. Make sure you have set XCATHOST environment variable to mgmt: 3001 in /etc/profile.d/xcat.sh, where mgmt is the IP address of xCAT server.

(2) The Image Service is installed in SERVER2 along with xCAT server. This is to avoid having to copy the image from the Glance node to the xCAT server, if we put them in different places. In this step, we need to make all the necessary configurations for both Glance and xcatd.

(3) Create the database for the bare metal machines and register them. We wrote a python script bm_db_sync to create the database tables. Of course, add one line to point to this database in the nova.conf file. For example, add"baremetal_sql_connection=mysql://nova_bm:passwd @127.0.0.1/nova_bm" to the nova.conf file. After this, we register the available bare metal machines into the nova_bm database. We have another python script bm_node_create to accomplish the registration job. Now Nova knows you have these machines and when a request comes, it will choose the one which is the best candidate.

(4) As for the virtual machines, we need instance types (or flavor) for bare metal machines. We use nova-manage to create a new instance type.

```
$ nova-manage instance_type create --name=bm.small --
cpu=4 --memory=4096 --root_gb=20 --ephemeral_gb=30
--flavor=6 --swap=1024 --rxtx_factor=1
```

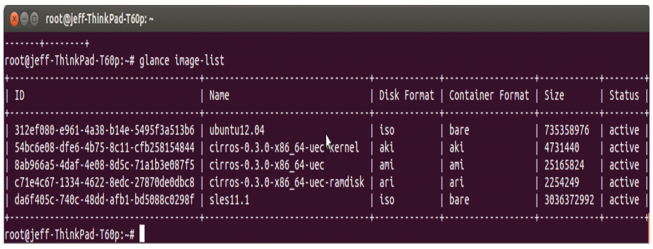
Then we set bare metal extra_spec to the instance type:

```
$ nova-manage instance_type set_key --name=bm.small
--key cpu_arch --value 'x86_64'
```

In this example, we have an instance type bm.small and its id is 6.This adds a record to instance_type table in nova database. In fact, we now can use the following command to see if we have added the instance type successfully.

```
$ nova-manage instance_type list
```

(5) Now we start all the necessary services (nova, glance, keystone, xcatd) and launch a bare metal instance.



```
root@jeff-ThinkPad-T60p: ~
-------+--------+
root@jeff-ThinkPad-T60p:~# glance image-list
+--------------------------------------+-------------------------------+-------------+------------------+------------+--------+
| ID                                   | Name                          | Disk Format | Container Format | Size       | Status |
+--------------------------------------+-------------------------------+-------------+------------------+------------+--------+
| 312ef080-e961-4a38-b14e-5495f3a513b6 | ubuntu12.04                   | iso         | bare             | 735358976  | active |
| 54bc6e08-dfe6-4b75-8c11-cfb258154844 | cirros-0.3.0-x86_64-uec-kernel | aki        | aki              | 4731440    | active |
| 8ab966a5-4daf-4e08-8d5c-71a1b3e087f5 | cirros-0.3.0-x86_64-uec       | ami         | ami              | 25165824   | active |
| c71e4c67-1334-4622-8edc-27870de0dbc8 | cirros-0.3.0-x86_64-uec-ramdisk | ari       | ari              | 2254249    | active |
| da6f405c-740c-48dd-afb1-bd5088c0298f | sles11.1                      | iso         | bare             | 3036372992 | active |
+--------------------------------------+-------------------------------+-------------+------------------+------------+--------+
root@jeff-ThinkPad-T60p:~#
```

```
nova boot --image da6f405c-740c-48dd-afb1-bd5088c0298f --flavor 6 instance_jeff
```

In this example, we use the nova command line to start the new instance instance_jeff. As we can see, sles11.1 is used here and the flavor is 6, which is the bare-metal instance type we added before.

Sometime later, we can use "nova list" command to list all the active instances and you will find the new instance is available. And if you use the username/password provided by xCAT, you can login to your new system and do your job. As an administrator, you can also use xCAT to monitor all these bare metal nodes and use xCAT's other monitoring and management functions to ease your admin work.

V. Performance Evaluations

Last,we evaluate the performance of the Oracle database both in a virtualized and bare metal environment. We present the experimental results here to show the different performances in the two testing environments so that the cloud administrators and end users could make better choices in the OpenStack platform,when faced with the choices between virtualization technologies and bare metal provisioning.

A. Test Environment

The same physical computer is used in both environments. For the bare-metal part,we installed Redhat5.5 as the operating system,and Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 as the testing application.For the virtualized environment,we used VMware vSphere as the hypervisor,on which we created a virtual machine. Here,we installed the same set of software in the virtual machine and bare-metal machine.

B. Test Description and Experiment Results

we use Swingbench v2.3 in this simple experiment to measure the maximum database transactions per minute(tpm) in a bare-metal machine and virtual machine, with the work load set to 30 active users.Table I shows the database maximum transaction rates and the standard deviation values.The standard deviation values show a 12.42% performance gains by the bare-metal machine, compared to the virtual machine.The results lead us to a conclusion that the bare-metal machines could avoid some performance penalty,thus prove the usefulness of the bare-metal machines provisioning in the cloud platform.

TABLE I.
DATABASE MAXIMUM TRANSACTION RATE FOR 30 USERS:

Bare-metal machine	Virtual machine
6088	5640
6007	5662
5746	5573
6020	5345
5966	5386
5701	5360
Standard deviation:158.28/min	Standard deviation:140.79/min

VI. Conclusion

Cloud computing is quickly becoming a dominant model for cloud end-users to access centrally managed computational resources. OpenStack, as well as other IaaS platforms, should provide cloud end users both choices of virtual machines and bare metal machines. Through our work in extending OpenStack, we used xCAT to accomplish the capacity of bare metal support.

The initial work to date has finished the basics of the xCAT driver. In the next phase, we will run more tests and also re-factor the code to finally contribute the code back to the OpenStack community. Moreover, we will focus more on the scheduling algorithm when it tries to pick up a nova compute node or a bare metal machine as the best candidate for an instance request from cloud end-users.

Acknowledgment

This paper was supported by the National High Technology Research and Development Program of China(No. 2011AA010704), the Fundamental Research Funds for the Central Universities (No. 2011RC0508) and IBM Shared University Research Project (JSA_CDL_BUPT_201207). The authors wish to thank all those people from BUPTNIC for their longtime help and support.

References

[1] http://en.wikipedia.org/wiki/OpenStack 9/1/2012
[2] https://blueprints.launchpad.net/openstack-common/+spec/installer-crowbar. 9/3/2012

[3] http://www.opscode.com/ 9/3/2012
[4] http://trac.mcs.anl.gov/projects/Heckle/ 9/3/2012
[5] http://xcat.sourceforge.net/ 9/5/2012
[6] http://www.perceus.org/ 9/3/2012
[7] M. J. Brim, T. G. Mattson, and S. L. Scott, "OSCAR: open source cluster application resources," Proceedings of the 3rd Annual Linux Symposium, 2001
[8] P. M. Papadopoulos, M. J. Katz, and G. Bruno, "NPACI rocks: tools and techniques for easily deploying manageable linux clusters," Concurrency and Computation: Practice and Experience, vol.15, no.7-8, pp.707–725, 2003
[9] http://en.wikipedia.org/wiki/PrebootExecutionEnvironment 9/1/2012
[10] http://en.wikipedia.org/wiki/IPMI 9/1/2012
[11] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, "The Eucalyptus open-source cloud-computing system, " in Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, ser. CCGRID '09. Washington, DC, USA: IEEE Computer Society, pp. 124–131,2009.
[12] B. Sotomayor, R. S. Montero, I. M. Llorente, and I. Foster, "Capacity leasing in cloud systems using the OpenNebula engine," in Proceedings of the 2008 Workshop on Cloud Computing and its Applications (CCA08), October 2008
[13] K. Keahey and T. Freeman, "Contextualization: Providing one-click virtual clusters," in eScience, 2008. eScience '08. IEEE Fourth International Conference on, pp. 301-308, Dec. 2008.
[14] http://ken.pepple.info/openstack/2011/04/22/openstack-nova-architecture 9/1/2012
[15] http://aws.amazon.com/ec2/ 9/1/2012
[16] http://www.rabbitmq.com/ 9/1/2012
[17] http://sourceforge.net/apps/mediawiki/xcat/index.php?title =XCAT_2_Architecture 9/1/2012

Jun Xie Hunan province,China,1987. Bachelor of Science, School of Software Engineering,Beijing University of Posts and Telecommunications,Beijing,China,2010; Master of Science, Institute of Network Technology, Beijing University of Posts and Telecommunications,Beijing,China,2013.

He will graduate from BUPT in March,2013.During the past few years,he was intern at the department of ICTO-DS at T-System P.R.China Ltd in 2010.Then in the year 2012,he was an intern of the Extreme Blue Program at International Business Machine(IBM).

Yujie Su Engineer at Network Information Center, Beijing University of Posts and Telecommunications.

Zhaowen Lin Associate Professor at Beijing University of Posts and Telecommunications.

Yan Ma Professor and director of the Network Information Center of Beijing University of Posts and Telecommunications, he is an Executive Committee member of APNIC, Executive Committee member of China Education and Research Network (CERNET) since 1994, convenor of APEC TEL HRDSG (Human Resource Development Steering Group) and DSG (ICT Development SG).

Junxue Liang Ph. D. Candidate at Network Information Center, Beijing University of Posts and Telecommunications