**[RedHat源码安装OpenStack教程及预研云计算资源管理平台OpenStack支持RedHat](http://wiki.primeton.com/pages/viewpage.action?pageId=13402645)**

**一、RedHat源码安装OpenStack教程**

步骤一：安装操作系统

操作系统可以从公司内网<ftp://192.168.1.4/Linux/RHEL/rhel-server-6.4-x86_64-dvd.iso>下载

本次预研使用的操作系统是RHEL 6.4 Kernel版本为2.6.32

[root@rhel6 yum.repos.d]# cat /etc/issue

Red Hat Enterprise Linux Server release 6.4 (Santiago)

[root@rhel6 yum.repos.d]# cat /proc/version

Linux version 2.6.32-358.el6.x86\_64 (mockbuild@x86-022.build.eng.bos.redhat.com) (gcc version 4.4.7 20120313 (Red Hat 4.4.7-3) (GCC) ) #1 SMP Tue Jan 29 11:47:41 EST 2013

[root@rhel6 yum.repos.d]#

步骤二：安装光盘源

1、  创建光盘挂在目录

将光盘放入RedHat光驱

[root@rhel6 ~]# mkdir /media/cdrom

2、  挂载光盘

[root@rhel6 ~]# mount /dev/cdrom /media/cdrom/

mount: block device /dev/sr0 is write-protected, mounting read-only

[root@rhel6 ~]# ls /media/cdrom/

3、  创建源目录

[root@rhel6 ~]# vim /etc/yum.repos.d/cdrom.repo

加入以下内容

[Cdrom]

name=Red Hat Enterprise Linux $releasever - $basearch - Cluster

baseurl= [file:///media/cdrom](file:///\\media\cdrom)

enabled=1

gpgcheck=1

gpgkey=[file:///etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release](file:///\\etc\pki\rpm-gpg\RPM-GPG-KEY-redhat-release)

4、  清理yum缓存

[root@rhel6 ~]# yum clean all

Loaded plugins: product-id, refresh-packagekit, security, subscription-manager

This system is not registered to Red Hat Subscription Management. You can use subscription-manager to register.

Cleaning repos: Cdrom

Cleaning up Everything

步骤三：安装OpenStack nova-compute组件

OpenStack是一种开源的云计算解决方案，它是由许多项目组成的，云计算资源管理平台目前使用的是OpenStack E版本，E版本包括的项目有Nova、Keystone、Glance、Swift。存储方面云计算资源管理平台没有使用E版本的Swift，而加入的 OpenStack F版本的Cinder，针对银联的存储型号，针对性地开发了HDS NAS的Cinder 驱动，使得Cinder有效地管理NAS存储。在本次预研中将SUSE系统作为管理节点，将RedHat系统安装nova-compute组件作为计算节 点接入SUSE的管理节点。

OpenStack最重要的一个项目是Nova：

nova-api是Nova的中心。它为所有外部调用提供服务，除了提供OpenStack本身的API规范外，他还提供了兼容EC2的部分API，所以也可以用EC2的管理工具对nova进行日常管理。在本系统中使用开源的Jclouds进行对nova-api的调用。

nova-compute负责对虚拟机实例进行创建、终止、迁移、Resize的操作。工作原理可以简单描述为：从队列中接收请求，通过相关的系统命令执行他们，再更新数据库的状态。

nova-volume管理映射到虚拟机实例的卷的创建、附加和取消

nova-network从队列中接收网络任务，然后执行任务控制虚拟机的网络，比如创建桥接网络或改变iptables的规则。

nova-scheduler 提供调度，来决定在哪台资源空闲的机器上启动新的虚拟机实例

Queue为守护进程传递消息。只要支持AMQP协议的任何Message Queue Sever都可以，当前官方推荐用RabbitMQ。

SQL database存储云基础架构中的各种数据。包括了虚拟机实例数据，网络数据等。

user dashboard是一个可选的项目。它提供了一个web界面来给普通用户或者管理者来管理、配置他们的计算资源。

OpenStack的另外三大项目分别是Keystone、Glance、Swift。

Keystone是提供身份认证和授权的组件。任何系统，身份认证和授权，其实都比较复杂。尤其Openstack 那么庞大的项目，每个组件都需要使用统一认证和授权。

Glance是镜像管理。目前Glance的镜像存储，支持本地存储，NFS，swift，sheepdog和Ceph，基本是够用了。

Swift是对象存储的组件。对于大部分用户来说，swift不是必须的。只有存储数量到一定级别，而且是非结构化数据才有这样的需求。很多人都问一个相同的问题：是否可以把虚拟机的存储放在swift上。简单回答：不行。

1、安装git工具

[root@rhel6 ~]# yum search git

[root@rhel6 ~]# yum install git

Git下载nova源码

<https://github.com/openstack/nova.git>

2、使用git下载OpenStack代码

[root@rhel6 ~]# mkdir openstack

[root@rhel6 ~]# cd openstack/

[root@rhel6 openstack]# git clone <https://github.com/openstack/nova.git>

3、安装OpenStack nova组件

RHEL6.4默认已经安装Python，Python版本为2.6.6

[root@rhel6 openstack]# cd nova/

[root@rhel6 nova]# python setup.py install

剩下的就是按照提示查找依赖包，可以通过yum search xxx 查找光盘中是否有所需要的依赖包

……

此处省略1W字

经过耗时又耗力地查找依赖包------------------安装依赖包…终于安装完二十多个依赖包

步骤四：启动nova-compute服务

手动创建nova配置文件/etc/nova/nova.conf，并使用下列命令启动nova-compute服务。

/usr/bin/python /usr/bin/nova-compute --flagfile=/etc/nova/nova.conf --logfile=/var/log/nova/compute.log

**二、预研云计算资源管理平台OpenStack支持RedHat**

本次预研重在比较云计算资源管理平台SUSE平台的OpenStack是否支持在RedHat上运行，并且是否支持创建RedHat系统的虚拟机？

初步系统差异比较：

|  |  |  |
| --- | --- | --- |
| 操作系统 | SUSE11sp2 | RHEL6.4 |
| python版本 | python-2.6.0 | python-2.6.6 |
| 虚拟化技术 | XEN | KVM |
| 安装包类型 | Rpm包 | Rpm包 |

预研问题1：云计算资源管理平台OpenStack版本是否支持在RedHat6.4上运行？

预研问题1结果：支持在RedHat6.4上运行

所需变更操作：

1、变更libvirt.xml.template模板文件

OpenStack创建虚拟机的过程中需要通过libvirt.xml.template模板文件给每一个虚拟机生成libvirt.xml文件，用来给虚拟机指定设备的数量与设备类型等信息。

经过预研后,针对原来的模板文件需要修改以下4处地方（红色标示）才能在RedHat上使用。

<domain type='$

Unknown macro: {type}

'>

    <uuid>$

Unknown macro: {uuid}

</uuid>

    <name>$

Unknown macro: {name}

</name>

    <memory>$

Unknown macro: {memory\_kb}

</memory>

    <os>

#if $type == 'lxc'

        <type>exe</type>

        <init>/sbin/init</init>

        <cmdline>console=ttyS0</cmdline>

#else if $type == 'uml'

    #set $root\_disk\_bus = 'uml'

    #set $ephemeral\_disk\_bus = 'uml'

        <type>uml</type>

        <kernel>/usr/bin/linux</kernel>

        #set $root\_device\_name = $getVar('root\_device\_name', '/dev/ubda')

        <root>$

Unknown macro: {root\_device\_name}

</root>

#else

    #if $type == 'xen'

        #set $root\_disk\_bus = 'scsi'

        #set $ephemeral\_disk\_bus = 'scsi'

    <type>hvm</type>

        <loader>/usr/lib/xen/boot/hvmloader</loader>

    #else

        #set $ephemeral\_disk\_bus = 'virtio'

        #if $root\_device\_type == 'cdrom'

            #set $root\_disk\_bus = 'ide'

        #else

            #set $root\_disk\_bus = 'virtio'

        #end if

            <type>hvm</type>

    #end if

    #if $getVar('rescue', False)

        <kernel>$

Unknown macro: {basepath}

/kernel.rescue</kernel>

        <initrd>$

/ramdisk.rescue</initrd>

    #else

        #if $getVar('kernel', None)

            <kernel>$

Unknown macro: {kernel}

</kernel>

            #if $type == 'xen'

                <cmdline>ro</cmdline>

            #else

                #set $root\_device\_name = $getVar('root\_device\_name', '/dev/vda')

                <cmdline>root=$

console=ttyS0</cmdline>

            #end if

            #if $getVar('ramdisk', None)

                <initrd>$

Unknown macro: {ramdisk}

</initrd>

            #end if

        #else

            <boot dev="hd" />

        #end if

    #end if

#end if

    </os>

#if $type != 'lxc' and $type != 'uml'

    <features>

        <acpi/>

    </features>

#end if

    <vcpu>$

Unknown macro: {vcpus}

</vcpu>

    <devices>

#if $type == 'lxc'

        <filesystem type='mount'>

            <source dir='$

Unknown macro: {basepath}

/rootfs'/>

            <target dir='/'/>

        </filesystem>

#else

    #if $getVar('rescue', False)

        <disk type='file'>

            <driver type='$

Unknown macro: {driver\_type}

' cache='$

Unknown macro: {cachemode}

'/>

            <source file='$

/disk.rescue'/>

            <target dev='$

Unknown macro: {disk\_prefix}

a' bus='$

Unknown macro: {ephemeral\_disk\_bus}

'/>

        </disk>

        <disk type='file'>

            <driver type='$

Unknown macro: {driver\_type}

' cache='$

Unknown macro: {cachemode}

'/>

            <source file='$

Unknown macro: {basepath}

/disk'/>

            <target dev='$

b' bus='$

Unknown macro: {ephemeral\_disk\_bus}

'/>

        </disk>

    #else

        #if not ($getVar('ebs\_root', False))

        <disk type='file' device='$

Unknown macro: {root\_device\_type}

'>

            <driver name='**qemu**' type='qcow2'/>**（SUSE平台为tap）**

            <source file='$

Unknown macro: {basepath}

/disk'/>

            <target dev='hda'/>

        </disk>

        #end if

        #if $getVar('ephemeral\_device', False)

            <disk type='file'>

                <driver name='**qemu**' type='$

Unknown macro: {driver\_type}

' cache='$

Unknown macro: {cachemode}

'/>

                <source file='$

Unknown macro: {datapath}

/disk.local'/>

                <target dev='$

Unknown macro: {ephemeral\_device}

' bus='$

'/>

            </disk>

        #end if

        #for $eph in $ephemerals

         <disk type='block'>

                <driver type='$

Unknown macro: {driver\_type}

' cache='$

Unknown macro: {cachemode}

'/>

                <source dev='$

Unknown macro: {basepath}

/$

Unknown macro: {eph.device\_path}

'/>

                <target dev='$

Unknown macro: {eph.device}

' bus='$

Unknown macro: {ephemeral\_disk\_bus}

'/>

         </disk>

        #end for

        #for $ext\_disk in $ext\_disks

         <disk type='file' device='disk'>

                <driver name='tap' type='qcow2'/>

                <source file='$

Unknown macro: {datapath}

/$

Unknown macro: {ext\_disk.device\_path}

'/>

                <target dev='$

Unknown macro: {ext\_disk.device}

' bus='scsi'/>

         </disk>

        #end for

        #if $getVar('swap\_device', False)

            <disk type='file'>

                <driver type='$

' cache='$

Unknown macro: {cachemode}

'/>

                <source file='$

Unknown macro: {basepath}

/disk.swap'/>

                <target dev='$

Unknown macro: {swap\_device}

' bus='$

Unknown macro: {ephemeral\_disk\_bus}

'/>

            </disk>

        #end if

        #for $vol in $volumes

            $

Unknown macro: {vol}

        #end for

    #end if

    #if $getVar('config\_drive', False)

        <disk type='file'>

            <driver type='raw' cache='$

'/>

            <source file='$

Unknown macro: {basepath}

/disk.config' />

            <target dev='$

Unknown macro: {disk\_prefix}

z' bus='$

Unknown macro: {ephemeral\_disk\_bus}

' />

        </disk>

    #end if

#end if

#for $nic in $nics

    #if $vif\_type == 'ethernet'

        <interface type='ethernet'>

            <target dev='$

Unknown macro: {nic.name}

' />

            <mac address='$

Unknown macro: {nic.mac\_address}

' />

            <script path='$

Unknown macro: {nic.script}

' />

        </interface>

    #else if $vif\_type == '802.1Qbh'

       <interface type='direct'>

            <mac address='$

'/>

            <source dev='$

Unknown macro: {nic.device\_name}

' mode='private'/>

            <virtualport type='802.1Qbh'>

                <parameters profileid='$

Unknown macro: {nic.profile\_name}

'/>

            </virtualport>

            <model type='virtio'/>

       </interface>

    #else if $vif\_type == 'ovs\_virtualport'

        <interface type='bridge'>

            <source bridge='$

Unknown macro: {nic.bridge\_name}

'/>

            <mac address='$

Unknown macro: {nic.mac\_address}

'/>

            <virtualport>

                <parameters interfaceid='$

Unknown macro: {nic.ovs\_interfaceid}

'/>

            </virtualport>

        </interface>

    #else

        <interface type='bridge'>

            <source bridge='$

'/>

            <mac address='$

Unknown macro: {nic.mac\_address}

'/>

#if $getVar('use\_virtio\_for\_bridges', True)

            <model type='virtio'/>

#end if

**<!--**            <filterref filter="nova-instance-$

-$

Unknown macro: {nic.id}

">**-->**

                <parameter value="$

Unknown macro: {nic.ip\_address}

" />

                <parameter value="$

Unknown macro: {nic.dhcp\_server}

" />

#if $getVar('nic.extra\_params', False)

                $

Unknown macro: {nic.extra\_params}

#end if

#if $getVar('nic.gateway\_v6', False)

                <parameter value="$

Unknown macro: {nic.gateway\_v6}

" />

#end if

**<!--**            </filterref>**-->**

        </interface>

    #end if

#end for

#if $type == 'qemu' or $type == 'kvm'

        <!-- The QEMU 'pty' driver throws away any data if no

         client app is connected. Thus we can't get away

         with a single console. Instead we have

         to configure two separate consoles. -->

        <serial type='file'>

            <source path='$

Unknown macro: {basepath}

/console.log'/>

        </serial>

        <serial type='pty'/>

#else

        <console type='pty'/>

#end if

#if $getVar('use\_usb\_tablet', True) and $type != 'lxc' and $type != 'xen'

        <input type='tablet' bus='usb'/>

#end if

#if $getVar('vncserver\_listen', False)

        <graphics type='vnc' port='-1' autoport='yes' keymap='$

Unknown macro: {vnc\_keymap}

' listen='$

Unknown macro: {vncserver\_listen}

'/>

#end if

    </devices>

</domain>

2、变更OpenStack配置文件

云计算资源管理平台OpenStack前期所选环境为SUSE+XEN的组合，而在RedHat平台官方默认使用KVM的虚拟化技术，那么针对现在的OpenStack 配置文件/etc/nova/nova.conf进行以下修改

1】  修改虚拟化技术

修改nova.conf中的libvirt\_type参数

libvirt\_type=xen ==> libvirt\_type=**kvm**

2】修改挂载disk盘的方式

对于qcow2 格式的镜像，只能使用guestfs或nbd设备配合qemu-nbd进行注入。但是使用guestfs挂载镜像的速度较慢。因此在SUSE中云计算资源 管理平台使用qemu-nbd方案，遗憾的是CentOS/RedHat中，内核默认没有编译nbd模块, qemu默认也没有编译qemu-nbd程序。

解决方案有2种，1）手工编译nbd模块和qemu-nbd程序、2）使用速度较慢的guestfs 2种方案在本次预研中都有测试，通过查看OpenStack操作日志可以比较得出2种方案在时间上相差10至20倍。

|  |  |  |
| --- | --- | --- |
|  | qemu-nbd | guestfs |
| 挂载时间 | 2~3s | 30~40s |

方案1实现起来开销较大，因此建议使用方案2。

修改nova.conf中的img\_handlers参数

img\_handlers=nbd ==> img\_handlers=**guestfs**

预研问题2：云计算资源管理平台OpenStack版本是否支持RedHat系统作为镜像模板？

预研问题2结果：目前云计算资源管理平台OpenStack版本暂不支持RedHat系统镜像模板的注入，需要修改代码中注入路径及内容才能满足支持RedHat镜像的需求。

OpenStack默认只支持基于Debian的Linux发行版本，比如Ubuntu。注入主机名、注入默认路由、注入DNS都是云计算资源管理平台进行二次开发的新功能。

针对SUSE与RedHat注入的路径与内容不同点见下表。因此采用现有OpenStack 代码无法在RedHat镜像模板注入以上相关性息。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | SUSE注入路径 | 注入内容 | RedHat注入路径 | 注入内容 | 结果 |
| 网络配置注入 | /etc/sysconfig/network/ifcfg-eth0 | BOOTPROTO=static IPADDR=192.168.2.120 NETMASK=255.255.252.0 STARTMODE=auto USERCONTROL=no | /etc/sysconfig/network-scripts/ifcfg-eth0 | ONBOOT=yes DEVICE=eth0 BOOTPROTO=static IPADDR=192.168.2.120 NETMASK=255.255.252.0 STARTMODE=auto USERCONTROL=no | 默认注入路径与内容不可用，增加ONBOOT、DEVICE后可用 |
| 默认路由注入 | /etc/sysconfig/network/routes | default 192.168.2.73 - - | /etc/sysconfig/network | GATEWAY=192.168.2.73 | 默认注入路径与内容不可用 |
| DNS注入 | /etc/resolv.conf | nameserver 114.114.114.114 | /etc/resolv.conf | nameserver 114.114.114.114 | 可用 |
| 主机名的注入 | /etc/HOSTNAME | XXX | /etc/sysconfig/network | HOSTNAME=XXX | 默认注入路径与内容不可用 |