枯木文档系列之 OpenNebula 入门安装配置

OpenNebula4.4 入门

■ **制作**	枯木	■ **weibo**	枯木-Linux
■ **日期**	2013-11-28	■ **Blog**	http://kumu-linux.github.io

本手册基本根据官网文档配置，更详细的信息请参考官方 document **By 枯木**

OpenNebula 4.4 入门配置

■ 版本变更记录

时间	版本	说明	修改人
2013-11-28	1.0		枯木

目录

一. OpenNebula 简介

1.1 云管理平台的选择

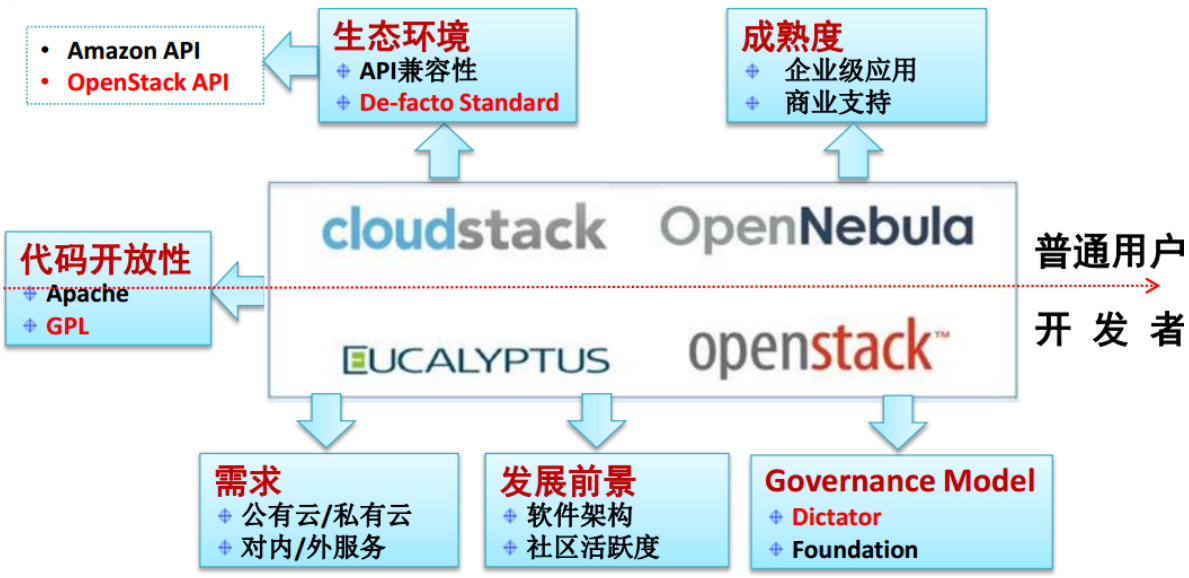

图 1.1 云管理平台对比图

OpenNebula 是一款为云计算而打造的开源工具箱。它允许你和 Xen、KVM 或 VMware ESX 一起建立和管理私有云，同时还提供 Deltacloud 适配器与 Amazon EC2 相配合来管理混合云。目前版本 4.4，可支持 XEN、KVM 和 VMware，以及实时存取 EC2 和 ElasticHosts，OpenNebula 可以构建私有云、混合云、公开云。官方网站 http://opennebula.org/ ，下图为几个使用 OpenNebula 的公司：

图 1.2 使用 OpenNebula 部分公司

1.2 OpenNebula 体系结构

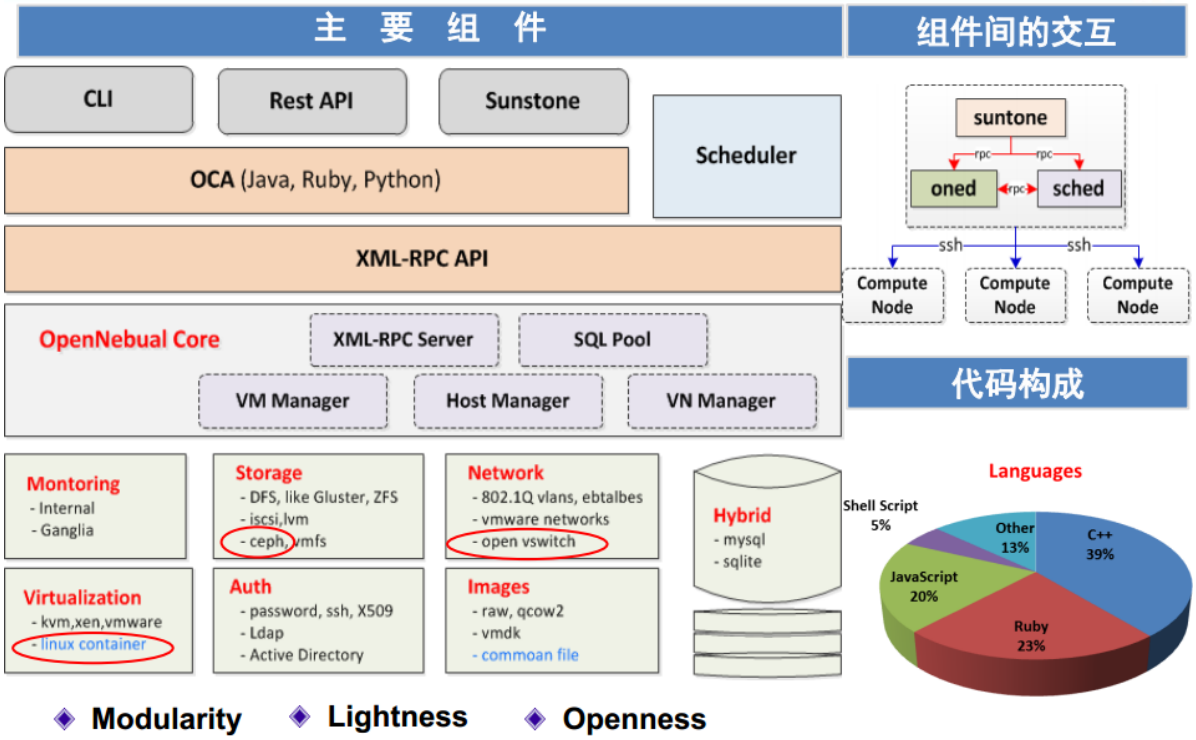

图 1.3 OpenNebula 体系结构

二. 环境说明

因为 CentOS6.4 虚拟化有很大的一个性能提升，特别是 CPU 的中断数，所以系统环境管理端和节点宿主机都采用 **CentOS6.4 x86_64**

三. OpenNebula 安装配置

3.1 软件包组成

从 OpenNebula 官网下载 CentOS/RHEL 6 对应软件包或者加入 OpenNebula 源，直接下载软件包这里不再赘述，添加 OpenNebula 源方法如下：

```
# cat << EOT > /etc/yum.repos.d/opennebula.repo
[opennebula]
name=opennebula
baseurl=http://downloads.opennebula.org/repo/CentOS/6/stable/\$basearch
enabled=1
gpgcheck=0
EOT
```

OpenNebula4.4 主要有以下几个软件组成：

```
# ls opennebula-*
opennebula-4.4.0-1.x86_64.rpm          //OpenNebula 命令行指令
opennebula-flow-4.4.0-1.x86_64.rpm     //管理 OpenNebula 服务
opennebula-java-4.4.0-1.x86_64.rpm     //OpenNebula Java Api
opennebula-ozones-4.4.0-1.x86_64.rpm   //OpenNebula 网页使用界面
opennebula-server-4.4.0-1.x86_64.rpm   //OpenNebula Server 守护进程
opennebula-common-4.4.0-1.x86_64.rpm   //基本依赖性组件
opennebula-gate-4.4.0-1.x86_64.rpm     //使虚拟机和 OpenNebula 之间的通信
opennebula-node-kvm-4.4.0-1.x86_64.rpm //元软件包，包括安装 oneadmin 用户、libvirt 和 kvm
opennebula-ruby-4.4.0-1.x86_64.rpm     //ruby 依赖性组件
opennebula-sunstone-4.4.0-1.x86_64.rpm //OpenNebula 网页使用界面
opennebula-context-4.4.0-1.x86_64.rpm  //context 组件
```

3.2 Server 端安装和配置

为解决一些依赖关系，安装之前可以激活 epel 源，因为测试为 CentOS6.4，因此激活方式如下：

```
# rpm -ivh http://dl.fedoraproject.org/pub/epel/6Server/x86_64/epel-release-6-8.noarch.rpm
```

如果下载的是 OpenNebula 软件包，则进入解压目录，安装方式如下 [以下安装为组成 Server 端最基本的软件]：

```
# yum localinstall opennebula-server-4.4.0-1.x86_64.rpm  \
```

```
opennebula-4.4.0-1.x86_64.rpm opennebula-common-4.4.0-1.x86_64.rpm \
opennebula-ruby-4.4.0-1.x86_64.rpm opennebula-sunstone-4.4.0-1.x86_64 -y
```

如果使用 OpenNebula 的源，安装如下：

```
# yum install opennebula-server opennebula-sunstone -y
```

安装完成之后创建如下用户以及目录文件：

```
# grep oneadmin /etc/passwd
oneadmin:x:9869:9869::/var/lib/one:/bin/bash
# ls -ld /etc/one/     //OpenNebula 相关配置文件所在目录
drwxr-x---. 11 root oneadmin 4096 Aug 20 11:35 /etc/one/
# ls /etc/init.d/opennebula*
/etc/init.d/opennebula  /etc/init.d/opennebula-occi  /etc/init.d/opennebula-sunstone
# ls -ld /var/log/one/
drwxr-x---. 2 oneadmin oneadmin 4096 Jul 25 01:13 /var/log/one/
```

默认 OpenNebula 数据存储使用 sqlite，如果需要使用 MySQL，则需要做如下操作：

1. 创建相关数据库：

```
mysql> create database opennebula;
Query OK, 1 row affected (0.00 sec)

mysql> grant all privileges on opennebula.* to oneadmin@'localhost' identified by 'oneadmin';
Query OK, 0 rows affected (0.00 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec)
```

2. 修改配置文件如下 [用户、端口、密码、库名和实际情况对应修改]：

```
# vim /etc/one/oned.conf
… …
#DB = [ backend = "sqlite" ]
# Sample configuration for MySQL
DB = [ backend = "mysql",
       server  = "localhost",
       port    = 3306,
       user    = "oneadmin",
       passwd  = "oneadmin",
       db_name = "opennebula" ]
… …
```

修改 sunstone 默认监听 IP:

```
# grep ':host' /etc/one/sunstone-server.conf
:host: 127.0.0.1
# sed -i '/:host/s/127.0.0.1/192.168.80.130/g' /etc/one/sunstone-server.conf
# grep ':host' /etc/one/sunstone-server.conf
:host: 192.168.80.130
```

启动相关服务:

```
# /etc/init.d/opennebula start
# /etc/init.d/opennebula-sunstone start
# lsof -i:9869
COMMAND   PID     USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
ruby    22266 oneadmin    6u  IPv4 106746      0t0  TCP 192.168.80.130:9869 (LISTEN)
```

修改 datastore:

OpenNebula 默认用的是 Shared Transfer Driver，这种模式比较适合快速部署和热迁移，只是要配置网络文件系统。如果没有网络文件系统，不想做热迁移，那么可以换成 SSH Transfer Driver 测试部署。

```
$ onedatastore list
  ID NAME                SIZE AVAIL CLUSTER      IMAGES TYPE DS      TM
   0 system                OM -     -                 0 sys  -       shared
   1 default            98.4G 85%   -                 1 img  fs      shared
   2 files              98.4G 85%   -                 0 fil  fs      ssh
$ onedatastore update 1
CLONE_TARGET="SYSTEM"
DISK_TYPE="FILE"
DS_MAD="fs"
LN_TARGET="SYSTEM"
TM_MAD="ssh"
TYPE="IMAGE_DS"
$ onedatastore list
  ID NAME                SIZE AVAIL CLUSTER      IMAGES TYPE DS      TM
   0 system                OM -     -                 0 sys  -       shared
   1 default            98.4G 85%   -                 1 img  fs      ssh
   2 files              98.4G 85%   -                 0 fil  fs      ssh
```

修改过程产生如下**错误**:

```
$ onedatastore update 1
Editor not defined
```

这是因为如下原因，CentOS 默认 vi 位置是/bin/vi，添加相关链接即可

```
# grep -i editor_path= /usr/lib/one/ruby/cli/one_helper.rb
EDITOR_PATH='/usr/bin/vi'
# ln -s /bin/vi /usr/bin/vi
```

浏览器登陆测试：

图 1.4 OpenNebula web 登陆框

用户名和密码通过以下方式获得：

```
# cat /var/lib/one/.one/one_auth
oneadmin:cd24c3a59c9fd8a7ab853b10247e8147
```

注：测试过程中因为测试环境服务端**时间**不对，导致 cookie 被忽略，OpenNebula Sunstone 选择 ***Keep me logged in*** 一直登陆不上或者直接登陆很快退出，寻找原因花了很长时间，最后调整到正确时间，登陆显示 ok。P.S: 时间是一个非常容易被我们忽略的问题，切记切记!

3.3 节点端安装配置

软件包下载见 Server 端安装章节，节点只需要安装以下两个软件

- opennebula-node-kvm-4.4.0-1.x86_64.rpm
- opennebula-common-4.4.0-1.x86_64.rpm

yum 安装以上软件即可，安装过程同时会安装虚拟化相关组件，包括 bridge-utils、libvirt、qemu-kvm、qemu-img 等。

桥接网络：

```
# cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
TYPE=Ethernet
ONBOOT=yes
BRIDGE=br0
NAME="System eth0"
# cat /etc/sysconfig/network-scripts/ifcfg-br0
DEVICE=br0
ONBOOT=yes
TYPE=Bridge
BOOTPROTO=static
IPADDR=192.168.80.131
NETMASK=255.255.255.0
GATEWAY=192.168.80.2
```

修改之后，重启网络并查看确认：

```
# service network restart
# brctl show
bridge name bridge id       STP enabled     interfaces
br0         8000.000c2942e561     no         eth0
```

修改/etc/libvirt/qemu.conf 的相关配置：

```
user  = "oneadmin"
group = "oneadmin"
dynamic_ownership = 0
```

修改/etc/libvirt/libvirtd.conf 相关配置：

```
listen_tcp = 1              //OpenNebula 使用 libvirt 提供的 TCP 协议
listen_tls = 0
```

修改/etc/sysconfig/libvirtd 开启监听选项：

```
LIBVIRTD_ARGS="--listen"
```

启动 libvirtd 服务：

```
# /etc/init.d/libvirtd start
# netstat -tulnp | grep libvirt
tcp        0      0 0.0.0.0:16509               0.0.0.0:*                   LISTEN      2664/libvirtd
```

ssh 无密码登陆：

ssh 使用公钥认证无密码登陆这个比较简单，顺带也提一下，方法如下：

管理端

```
# su - oneadmin
$ cat ~/.ssh/config          //增加超时时间，不询问直接添加主机到 known_hosts 文件
ConnectTimeout 5
Host *
    StrictHostKeyChecking no
    UserKnownHostsFile /dev/null
```

节点端

```
# su - oneadmin
$ vim .ssh/authorized_keys              //把管理端 ssh 公钥加入节点.ssh/authorized_keys 文件
$ chmod 400 .ssh/authorized_keys
```

如此，Server 端的 oneadmin 用户就可以无密码登陆节点 oneadmin 了。

四. 添加节点

节点如此安装软件和配置之后便可以在 Server 端添加了，可以使用 web 添加，也可以使用命令添加。

4.1 web 界面添加主机

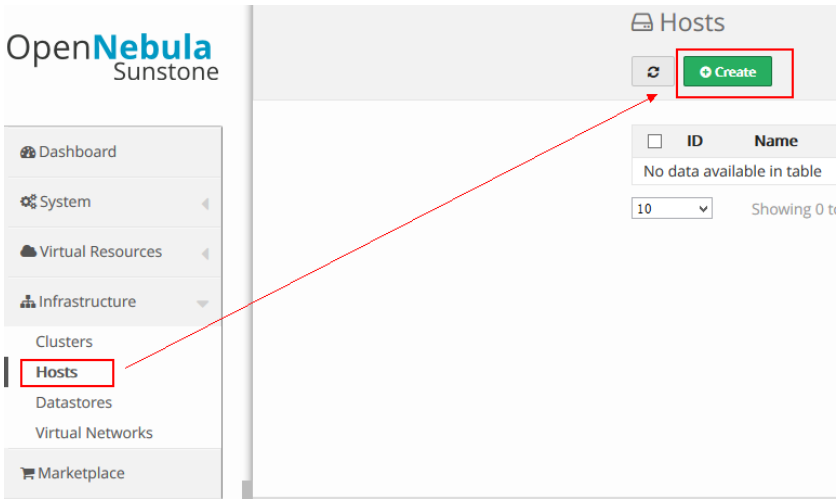

图 1.5 选择主机添加

添加主机，Hostname 一栏可以是 IP 也可以是可以解析的主机域名，这里使用的是 IP

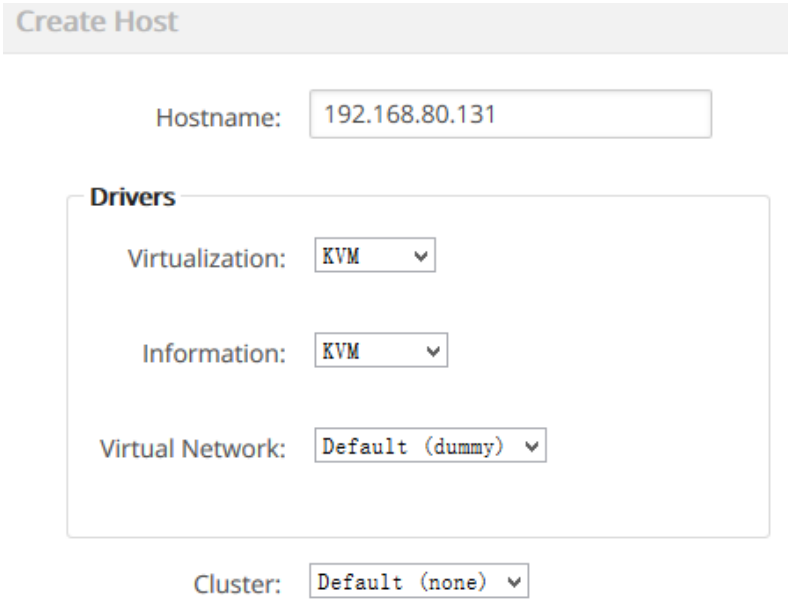

图 1.6 添加节点配置

如上图选择完之后，点击 Create 按钮就可以出现如下主机，初始状态为 INIT

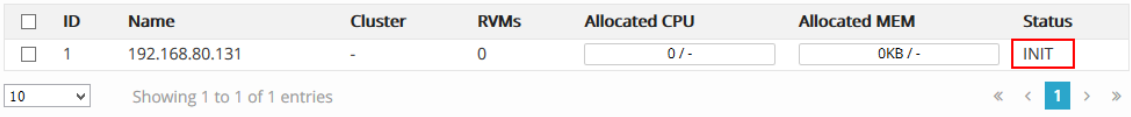

图 1.7 节点初始状态

等待片刻之后，如果一切正常，则状态就会变为 ON，也会显示 CPU 和内存信息，如下图：

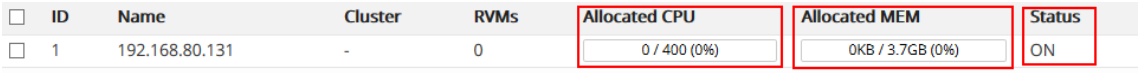

图 1.8 成功添加主机节点

以上 RVMs 表示当前运行的虚拟主机数，目前为 0。

点击目标主机之后，下方会显示详细的主机信息图，如下图

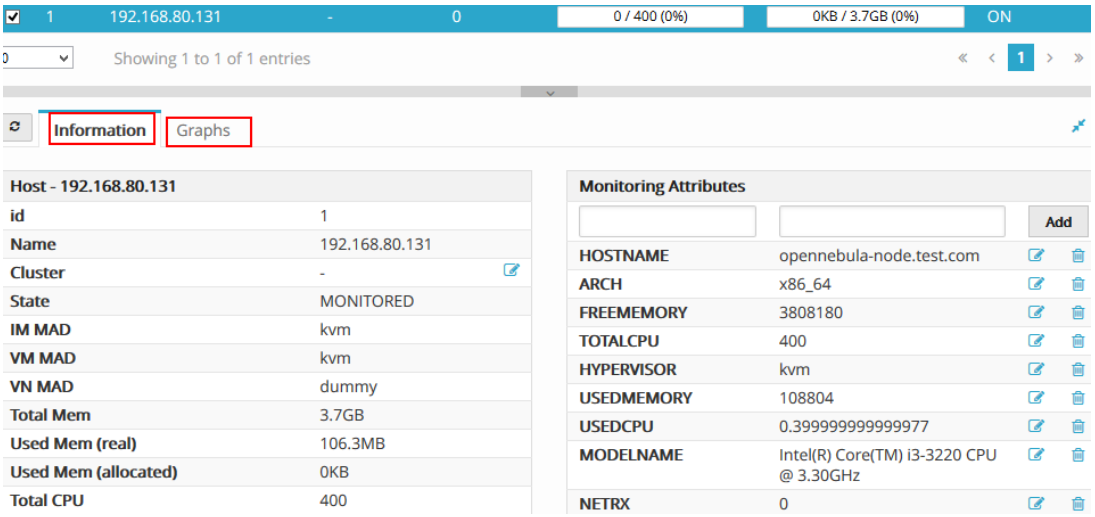

图 1.9 主机详细信息

删除则选择主机，点击删除按钮即可。

4.2 命令行添加主机

4.2.1 onehost 命令

使用命令行添加主机也比较简单，这里使用的命令是 onehost

使用 onehost 命令删除之前 web 创建的主机，如下：

```
$ su - oneadmin
$ onehost list
  ID NAME            CLUSTER    RVM      ALLOCATED_CPU      ALLOCATED_MEM STAT
```

```
   1 192.168.80.131  -           0       0 / 400 (0%)      0K / 3.7G (0%) on
$ onehost delete 1          //删除主机，可以是 ID 也可以是 NAME
$ onehost list
  ID NAME            CLUSTER   RVM       ALLOCATED_CPU       ALLOCATED_MEM STAT
```

当然删除之后我们还是需要再创建一遍，虽然很无聊，But 你懂的，如下

```
$ onehost create 192.168.80.131 --im kvm --vm kvm --net dummy
ID: 2
$ onehost list
  ID NAME            CLUSTER   RVM       ALLOCATED_CPU       ALLOCATED_MEM STAT
   2 192.168.80.131  -           0       0 / 400 (0%)      0K / 3.7G (0%) on
```

- --im/-i:信息管理 driver. 可选: kvm, xen, vmware, ec2, ganglia, dummy.
- --vm/-v: 虚拟化管理 driver. 可选: kvm, xen, vmware, ec2, dummy.
- --net/-n: 虚拟网络 driver. 可选: 802.1Q,dummy,ebtables,fw,ovswitch,vmware.

查看主机的详细信息 **onehost show**

```
$ onehost show 2
HOST 2 INFORMATION
ID                    : 2
NAME                  : 192.168.80.131
CLUSTER               : -
STATE                 : MONITORED
IM_MAD                : kvm
VM_MAD                : kvm
VN_MAD                : dummy
LAST MONITORING TIME  : 11/29 22:19:21

HOST SHARES
TOTAL MEM             : 3.7G
USED MEM (REAL)       : 111M
USED MEM (ALLOCATED)  : 0K
TOTAL CPU             : 400
USED CPU (REAL)       : 0
USED CPU (ALLOCATED)  : 0
RUNNING VMS           : 0

… …
```

通过-x 选项还可以以 xml 的格式显示主机相关信息

```
$ onehost show -x 2
```

```
<HOST>
  <ID>2</ID>
  <NAME>192.168.80.131</NAME>
  <STATE>2</STATE>
  <IM_MAD>kvm</IM_MAD>
  <VM_MAD>kvm</VM_MAD>
  <VN_MAD>dummy</VN_MAD>
  <LAST_MON_TIME>1385735001</LAST_MON_TIME>
  <CLUSTER_ID>-1</CLUSTER_ID>
  <CLUSTER/>
  <HOST_SHARE>
    <DISK_USAGE>0</DISK_USAGE>
    <MEM_USAGE>0</MEM_USAGE>
    <CPU_USAGE>0</CPU_USAGE>
    <MAX_DISK>0</MAX_DISK>
    <MAX_MEM>3916984</MAX_MEM>
    <MAX_CPU>400</MAX_CPU>
    <FREE_DISK>0</FREE_DISK>
    <FREE_MEM>3803128</FREE_MEM>
    <FREE_CPU>399</FREE_CPU>
    <USED_DISK>0</USED_DISK>
    <USED_MEM>113856</USED_MEM>
    <USED_CPU>0</USED_CPU>
    <RUNNING_VMS>0</RUNNING_VMS>
  </HOST_SHARE>
... ...
```

onehost 还有两个选项,**disable** 和 **enable**，disable 表示不再监控该物理主机，但是不影响正在运行的虚拟机，enable 表示开启监控

```
$ onehost disable 0
$ onehost enable 0
```

五. 制作系统镜像

qemu-img 和 qemu-kvm 命令是制作系统镜像的重要工具，在介绍这两个工具之前，稍微简单说明下虚拟机镜像格式，目前虚拟机有多种镜像格式可供选择，常见的有如 raw、vdi、qcow2、vmdk、qed、vhd 等格式。

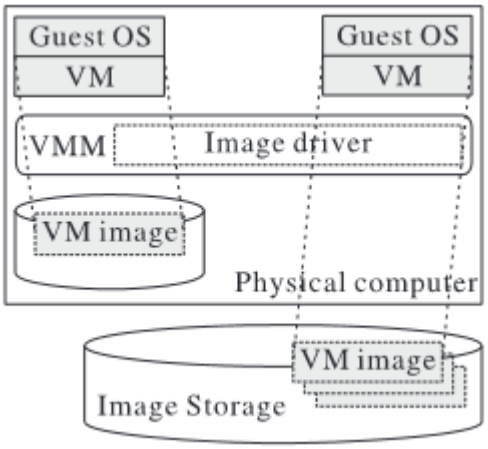

图 1.10 虚拟机镜像和服务器之间的关系

虚拟机	raw	qcow2	vmdk	qed	vdi	vhd
XEN	√	√	√			√
KVM	√	√	√	√	√	
VMware			√			
Virtualbox			√		√	√

虚拟机和支持的镜像格式

5.1 qemu-img

qemu-img 是 QEMU 的磁盘管理工具，支持多种虚拟镜像格式

```
$ qemu-img -h | grep Supported
Supported formats: raw cow qcow vdi vmdk cloop dmg bochs vpc vvfat qcow2 qed parallels nbd
blkdebug host_cdrom host_floppy host_device file
```

qemu-img 默认创建的格式是 raw，man 手册中对几种格式也都有介绍。以下为对 raw 和 qcow2 镜像的详细介绍：

raw

原始的磁盘镜像格式，qemu-img 默认支持的格式，它的优势在于它非常简单而且非常容易移植到其他模拟器（emulator，QEMU 也是一个 emulator）上去使用。如果客户机文件系统（如 Linux 上的 ext2/ext3/ext4、Windows 的 NTFS）支持“空洞”（hole），那么镜像文件只有在被写有数据的扇区才会真正占用磁盘空间，从而有节省磁盘空间的作用。qemu-img 默认的 raw 格式的文件其实是**稀疏文件**（sparse file）[稀疏文件就是在文件中留有很多空余空间，留备将来插入数据使用。如果这些空余空间被 ASCII 码的 NULL 字符占据，并且这些空间相当大，那么，这个

文件就被称为稀疏文件，而且，并不分配相应的磁盘块。]，dd 命令创建的也是 raw 格式，不过 dd 一开始就让镜像实际占用了分配的空间，而没有使用稀疏文件的方式对待空洞而节省磁盘空间。尽管一开始就实际占用磁盘空间的方式没有节省磁盘的效果，不过它在写入新的数据时不需要宿主机从现有磁盘空间中分配，从而在第一次写入数据时性能会比稀疏文件的方式更好一点。简单来说，raw 有以下几个特点：

- 寻址简单，访问效率高
- 可以通过格式转换工具方便地转换为其它格式
- 格式实现简单，不支持压缩、快照和加密
- 能够直接被宿主机挂载，不用开虚拟机即可在宿主和虚拟机间进行数据传输

qcow2

qcow2 是 qcow 的一种改进，是 Qemu 实现的一种虚拟机镜像格式。更小的虚拟硬盘空间（尤其是宿主分区不支持 hole 的情况下），支持压缩、加密、快照功能，磁盘读写性能较 raw 差。

关于 raw 和 qcow2 的性能对比可以参考以下几篇测试文章：

- ✓ qcow2 Performance
- ✓ KVM 性能测试报告
- ✓ KVM I/O slowness on RHEL 6
- ✓ 几种虚拟机镜像格式及其性能测评

qemu-img 它支持的命令分为如下几种：

（1）check [-f *fmt*] *filename*

对磁盘镜像文件进行一致性检查，查找镜像文件中的错误，目前仅支持对“qcow2”、“qed”、“vdi”格式文件的检查。其中，qcow2 是 QEMU 0.8.3 版本引入的镜像文件格式，也是目前使用最广泛的格式。qed（QEMU enhanced disk）是从 QEMU 0.14 版开始加入的增强磁盘文件格式，为了避免 qcow2 格式的一些缺点，也为了提高性能，不过目前还不够成熟。而 vdi（Virtual Disk Image）是 Oracle 的 VirtualBox 虚拟机中的存储格式。参数-f *fmt* 是指定文件的格式，如果不指定格式 qemu-img 会自动检测，*filename* 是磁盘镜像文件的名称（包括路径）。

```
$ qemu-img check CentOS6.4-x86_64.qcow2
No errors were found on the image.
```

（2）create [-f *fmt*] *filename* [*size*]

创建一个格式为 *fmt* 大小为 *size* 文件名为 *filename* 的**镜像文件**。

```
$ qemu-img create -f qcow2 test.qcow2 10G
Formatting 'test.qcow2', fmt=qcow2 size=10737418240 encryption=off cluster_size=65536
$ qemu-img create -f qcow2 test.raw 10G
```

```
Formatting 'test.raw', fmt=qcow2 size=10737418240 encryption=off cluster_size=65536
```

注意这里的 qcow2 后缀只是为了便于自己区分格式方便，如果不加后缀也可以通过 qemu-img 来获取镜像的格式。

（3）info [-f *fmt*] *filename*

显示 *filename* 镜像文件的信息。如果文件是使用稀疏文件的存储方式，也会显示出它的本来分配的大小以及实际已占用的磁盘空间大小。如果文件中存放有客户机快照，快照的信息也会被显示出来。

```
$ qemu-img info test.qcow2
image: test.qcow2
file format: qcow2
virtual size: 10G (10737418240 bytes)
disk size: 136K
cluster_size: 65536
$ qemu-img info test.raw    //qemu-img 生成 raw 格式镜像也是采用稀疏文件方式存储的
image: test.raw
file format: qcow2
virtual size: 10G (10737418240 bytes)
disk size: 136K
cluster_size: 65536
$ dd </dev/zero >test.dd bs=1MB count=1000
1000+0 records in
1000+0 records out
1000000000 bytes (1.0 GB) copied, 1.80597 s, 554 MB/s
$ qemu-img info test.dd     //可以看到 dd 产生的格式也是 raw 格式的，并且没有用到稀疏存储方式
image: test.dd
file format: raw
virtual size: 954M (1000000000 bytes)
disk size: 954M
```

（4） convert [-c] [-f *fmt*] [-O *output_fmt*] [-o *options*] *filename* [*filename2* [...]] *output_filename*

镜像格式转换，将 *fmt* 格式的 *filename* 镜像文件根据 *options* 选项转换为格式为 *output_fmt* 的名为 *output_filename* 的镜像文件。它支持不同格式的镜像文件之间的转换，比如可以用VMware用的vmdk格式文件转换为qcow2文件，这对从其他虚拟化方案转移到KVM上的用户非常有用。一般来说，输入文件格式 *fmt* 由 qemu-img 工具自动检测到，而输出文件格式 *output_fmt* 根据自己需要来指定，默认会被转换为与 raw 文件格式（且默认使用稀疏文件的方式存储以节省存储空间）。

其中，“-c”参数是对输出的镜像文件进行压缩，不过只有 qcow2 和 qcow 格式的镜像文件才支持压缩，而且这种压缩是只读的，如果压缩的扇区被重写，则会被重写为未压缩的数据。同样可以使用“-o *options*” 来指定各种选项，如：后端镜像、文件大小、是否加密等等。使用 backing_file 选项来指定后端镜像，让生成的文件是 copy-on-write 的增量文件，这时必须让转换命令中指定的后端镜像与输入文件的后端镜像的内容是相同的，尽管它们各自后端镜像的目录、格式可能不同。

如果使用 qcow2、qcow、cow 等作为输出文件格式来转换 raw 格式的镜像文件（非稀疏文件格式），镜像转换还可以起到将镜像文件转化为更小的镜像，因为它可以将空的扇区删除使之在生成的输出文件中并不存在。

```
$ qemu-img info test.dd
image: test.dd
file format: raw
virtual size: 954M (1000000000 bytes)
disk size: 954M
$ qemu-img convert -O qcow2  test.dd test_qcow2.qcow2
$ qemu-img info test_qcow2.qcow2
image: test_qcow2.qcow2
file format: qcow2
virtual size: 954M (1000000000 bytes)
disk size: 136K
cluster_size: 65536
```

以上介绍了 qemu-img 的基本使用方法之后，关于 qemu-img 的更多高级用法可以参考 man 手册，或者参考 qemu-img 命令详解，关于 qemu-img 我也是基本引用这篇博文，案例稍作修改。

5.2 qemu-kvm

新建测试镜像，因为 qcow2 的一些特性，这里采用 qcow2 格式制作系统镜像

```
# qemu-img create -f qcow2 CentOS6.4-x86_64-tpl.qcow2 8G
# chown oneadmin:oneadmin CentOS6.4-x86_64-tpl.qcow2
Formatting 'CentOS6.4-x86_64-tpl.qcow2', fmt=qcow2 size=8589934592 encryption=off
cluster_size=65536
```

安装系统

```
# /usr/libexec/qemu-kvm -m 1024 -cdrom /data/images/CentOS-6.4-x86_64-bin-DVD1.iso -drive \
file=/data/images/CentOS6.4-x86_64-tpl.qcow2,if=virtio -net nic,model=virtio \
-net tap,script=no  -boot d -nographic -vnc :0
```

上面命令参数解释如下：

✓	-m	指定内存大小
✓	-cdrom	指定系统 iso 镜像
✓	-drive file=xx,if=xx	指定硬盘镜像,file=镜像文件名,if=镜像格式类型
✓	**-net nic**,model=xx	表示网卡配置,model=模拟网卡类型,默认 rt18139
✓	**-net tap**,script=no	虚拟设备，桥接网络,script 表启动虚拟机自动执行网络配置脚本，如果不需要启动，写 no 即可
✓	-boot d	系统启动顺序,d 表示表示 cdrom
✓	-nographic	关闭图形输出
✓	-vnc :0	开启 vnc 监听

详细的关于 qemu-kvm 的参数使用说明请参考 man 手册。

输入以上命令之后，通过 VNC 客户端连接按照正常的安装流程安装系统即可，

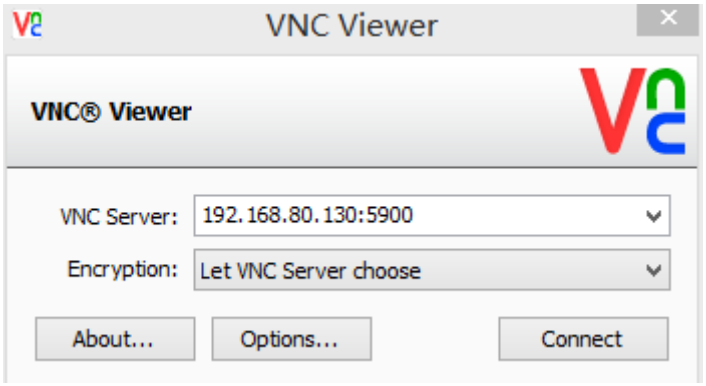

图 1.11 VNC View 连接安装系统

如果虚拟机需要和外界通信，则需要创建桥接网络，之前介绍 qemu-kvm 安装时提到**-net tap**,script=no 选项，默认只是创建桥接虚拟网络，并没有启用，只有启用了才可以设置对应网络配置和外界通信。

5.2.1 手动桥接

```
# ip link show dev tap0          //使用如上方式默认创建虚拟网卡 tap0，状态为 DOWN
37: tap0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN qlen 500
    link/ether d2:b0:af:7b:23:0f brd ff:ff:ff:ff:ff:ff
# brctl show br0                 //查看桥接列表，没有 tap0
```

```
bridge name   bridge id       STP enabled interfaces
br0       8000.b8975a626020   no      eth0
```

通过以下方式桥接网络

```
# ip link set tap0 up            //使 tap0 状态变为 up
# brctl addif br0 tap0           //桥接 tap0 到 br0
# brctl show br0                 //显示 tap0 已经加入到桥接列表
bridge name   bridge id       STP enabled interfaces
br0       8000.b8975a626020   no      eth0
                                      tap0
```

如此，配置好虚拟机的网络就可以和外界通信了。

`brctl delif br0 tap0` 删除桥接网络，qemu-kvm 工具在客户机关闭时会自动解除 TAP 设备的 bridge 绑定，所以这一步无需操作。

5.2.2 脚本实现

如果不想每次都手动操作，则可以通过脚本自动化实现虚拟网卡的桥接。使用选项`-net tap,script=/tmp/qemu-ifup.sh` 把之前的 no 替换为需要执行的脚本，以下为 qemu-ifup.sh 脚本内容

```
# cat /tmp/qemu-ifup.sh
#!/bin/bash

# 桥接网络设备
switch=br0                  //设置桥接网卡

if [ -n $1 ]; then          //$1 为 qemu-kvm 传递值，这里是 tap
    ip link set $1 up
    brctl addif ${switch} $1
    exit 0
else
   echo "no interface!"
   exit 1
fi
```

5.3 系统相关优化

完成系统安装配置之后，需要对镜像模板系统做如下一系列优化操作：

selinux、iptables、服务、文件描述符设置

```
# 关闭 SELINUX
sed -i -c 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/sysconfig/selinux

# iptables 根据相关需求配置

# 关闭系统其它额外的服务
service=`chkconfig --list | grep '3:on' | awk '{print $1}'`
for i in $service
do
 case $i in
      acpid | crond | irqbalance |  messagebus | network | sshd | rsyslog | udev-post)
           chkconfig --level 2345 $i on
      ;;
      *)
           chkconfig --level 2345 $i off
      ;;
 esac
done

# 文件描述符相关配置
cat >>/etc/security/limits.conf <<EOF
*               soft    nofile          65535
*               hard    nofile          65535
*               soft    nproc           10240
*               hard    nproc           10240
EOF
sed -i '/1024/s/1024/65535/' /etc/security/limits.d/90-nproc.conf
```

注：经测试 **acpid 服务**必须安装且在虚拟机系统中**开启**，否则 OpenNebula web 端和 shell 终端发送关机命令无效。

修改/etc/fstab

默认 fstab 使用 uuid 挂载文件，作为系统母盘镜像肯定是不能使用 uuid 的，这里个人推荐使用卷标，使用 **e2label** 命令添加磁盘分区卷标，修改内容如下所示：

```
LABEL=/                 /                       ext4    defaults        1 1
LABEL=/boot             /boot                   ext4    defaults        1 2
```

关于母盘的配置根据实际需求修改，这里介绍一些最基本的配置。之后包括 IP、DNS 等等相关的配置都可以放到以后配置，这里先小小的卖个关子，继续前行，不要走开，后面更加精彩。

六. 导入镜像

第五章我们创建了系统镜像，这一章节就要介绍把镜像导入 OpenNebula。就如之前所说添加主机可以使用 web 页面添加，也可以使用命令添加，导入镜像 web 和命令也都可以实现。

6.1 web 界面导入映像

这里说明一下，OpenNebula4.4 是支持简体中文页面的，如果你不习惯英文页面，可以修改如下

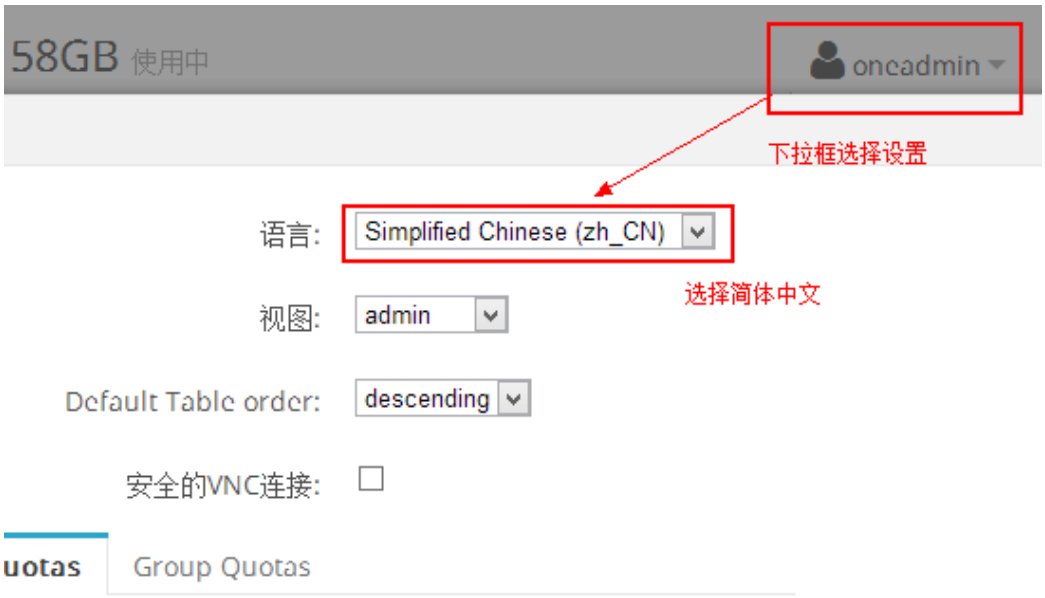

图 1.12 修改 web 语言

注：Opennebula4.2 简体中文界面 VNC 有 bug，在 4.4 版本中已得到修复。

导入镜像方法如下：

1、选择映像→创建

图 1.13　创建映像

2、点击创建之后出现如下弹框，根据实际填写如下

图 1.14　映像导入选项

持久性：如果您在虚拟机模板中使用了持久性的磁盘映像，你只能够基于该模板创建一个正在运行的虚拟机。这是因为一个持久型的磁盘映像只能够同时被应用到一台正在运行的

虚拟机上。如果您在虚拟机模板中使用的是临时性的磁盘映像，则可以同时基于该模板创建多个虚拟机。

导入后初始状态为 LOCKED

所有者	群组	名称	数据仓库	类型	状态	VM数量
oneadmin	oneadmin	centos_test	default	OS	LOCKED	0

图 1.15 镜像 LOCKED 状态

导入完成之后就会显示 READY 状态

所有者	群组	名称	数据仓库	类型	状态	VM数量
oneadmin	oneadmin	centos_test	default	OS	READY	0

图 1.16 镜像 READY 状态

如果已经被虚拟机使用，则状态为 USED，下表显示了所有的镜像状态

Short state	State	Meaning
lock	LOCKED	The image file is being copied or created in the Datastore.
rdy	READY	Image ready to be used.
used	USED	Non-persistent Image used by at least one VM. It can still be used by other VMs.
used	USED_PERS	Persistent Image is use by a VM. It cannot be used by new VMs.
disa	DISABLED	Image disabled by the owner, it cannot be used by new VMs.
err	ERROR	Error state, a FS operation failed. See the Image information with oneimage show for an error message.
dele	DELETE	The image is being deleted from the Datastore.

6.2 命令行导入映像

6.2.1 oneimg 命令

镜像的管理使用 oneimg 命令，如下显示刚刚 web 导入的系统映像

```
# su - oneadmin
$ oneimage list
  ID USER       GROUP      NAME            DATASTORE     SIZE TYPE PER STAT RVMS
   1 oneadmin   oneadmin   centos_test     default         8G OS    No rdy     0
```

下面我们使用 oneimg **导入系统映像**

```
$ vim centos_test1.one
$ cat centos_test1.one      //创建相关配置文件
NAME          = "centos_test1"
PATH          = /data/images/CentOS6.4-x86_64-tpl.qcow2
TYPE          = OS
DESCRIPTION   = "centos test1."
DRIVER        = qcow2
$ onedatastore list         //选择数据仓库，这里选择 default
  ID NAME                SIZE AVAIL CLUSTER      IMAGES TYPE DS       TM
   0 system              98.4G 90%   -                 0 sys  -        shared
   1 default             98.4G 81%   -                 3 img  fs       ssh
   2 files               98.4G 81%   -                 0 fil  fs       ssh
$ oneimage create centos_test1.one --datastore default
ID: 2
$ oneimage list             //初始状态 lock
  ID USER       GROUP      NAME            DATASTORE     SIZE TYPE PER STAT RVMS
   1 oneadmin   oneadmin   centos_test     default         8G OS    No rdy     0
   2 oneadmin   oneadmin   centos_test1    default         8G OS    No lock    0
$ oneimage list             //稍等片刻状态变为 rdy
  ID USER       GROUP      NAME            DATASTORE     SIZE TYPE PER STAT RVMS
   1 oneadmin   oneadmin   centos_test     default         8G OS    No rdy     0
   2 oneadmin   oneadmin   centos_test1    default         8G OS    No rdy     0
```

其中 centos_test1.one 文件内容可以直接添加到 web 页面创建映像的**高级模式**中

图 1.17 创建磁盘映像高级模式

克隆镜像 oneimage clone

有时需要修改某个镜像，在修改之前我们可以通过克隆的方式备份这个镜像

```
$ oneimage clone centos_test1 centos_test2        //克隆 centos_test1 为 centos_test2
```

查看详细信息 oneimage show

```
$ oneimage show centos_test
IMAGE 1 INFORMATION
ID              : 1
NAME            : centos_test
USER            : oneadmin
GROUP           : oneadmin
DATASTORE       : default
TYPE            : OS
REGISTER TIME   : 12/04 21:17:49
PERSISTENT      : No
SOURCE          : /var/lib/one//datastores/1/a266f6dbeb3779ca3b2f944e4e42a1ba
PATH            : /data/images/CentOS6.4-x86_64-tpl.qcow2
SIZE            : 8G
STATE           : rdy
RUNNING_VMS     : 0

PERMISSIONS
OWNER           : um-
GROUP           : ---
OTHER           : ---

IMAGE TEMPLATE
DEV_PREFIX="hd"
DRIVER="qcow2"

VIRTUAL MACHINES
```

删除镜像 oneimage delete [注意，状态为 used 使用的镜像是不能被直接删除的]

```
$ oneimage delete centos_test
```

基本常用的先介绍这些，详细的可以参考 man 手册和 Opennebula 官方文档的介绍。

七. 虚拟网络

在虚拟机中一个主机会通过相应的桥连接一个或多个网络，Opennebula 允许创建虚拟网络来对应映射物理网卡。这个章节会简单介绍如何定义和使用虚拟网络，假设这些虚拟主机他通过两个物理网卡通信：

- ✧ 一个私有网络，通过桥接 br0
- ✧ 一个公网，通过桥接 br1

7.1 web 界面添加虚拟网络

图 1.18 创建虚拟网络

图 1.19 创建选项

注意添加的时候填写正确的网桥，如果加入 MAC，则表示只租用给指定 MAC 的虚拟主机。另外虚拟网络有固定式网络和范围式网络，固定的则是必须是一个个添加固定的 ip，而范式则可以添加一个范围的网络 ip，范围式网络较固定的要相对方便一点，如下图：

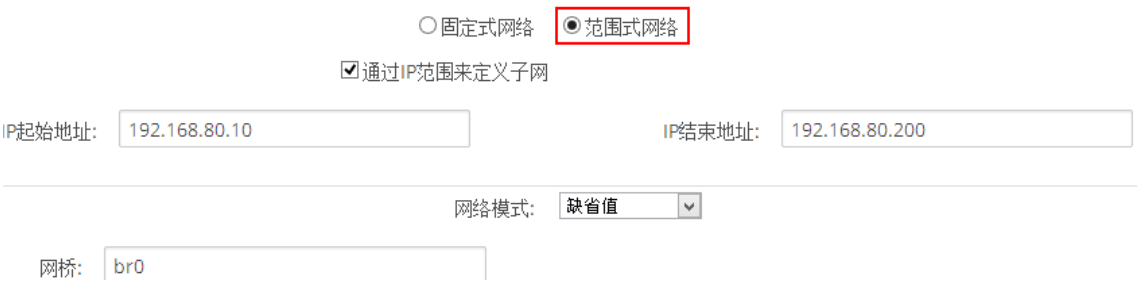

图 1.20 范围式网络

注意：这些虚拟网络中 IP 并不会被虚拟机所使用，只是一个 IP 给定一个虚拟机，相当于一个映射对应的关系，后面会介绍如何在部署虚拟机的时候指定虚拟机 IP。

下图是已被虚拟机占用的虚拟 IP [虚拟网络→租用管理 可以查看]，后续还会介绍

图 1.21 虚拟网络 ip 租用相关信息

7.2 命令行添加虚拟网络

7.2.1 onevnet 命令

列出虚拟网络 **onevnet list**

```
$ onevnet list          //显示当前的虚拟网络
  ID USER         GROUP        NAME            CLUSTER      TYPE BRIDGE   LEASES
   0 oneadmin     oneadmin     test-network    -               R br0           0
```

添加虚拟网络 **onevnet create**

```
$ cat test1.net
NAME    = "test1 LAN"
TYPE    = RANGED

BRIDGE  = br0

NETWORK_ADDRESS = 192.168.0.0/24
IP_START        = 192.168.0.3
```

```
GATEWAY = 192.168.0.1
DNS     = 192.168.0.1
$ cat test2.net
NAME    = "test2 LAN"
TYPE    = FIXED

# We have to bind this network to ''br1'' for Internet Access
BRIDGE  = br1

LEASES  = [IP=130.10.0.1]
LEASES  = [IP=130.10.0.2, MAC=50:20:20:20:20:21]
LEASES  = [IP=130.10.0.3]
LEASES  = [IP=130.10.0.4]

GATEWAY = 130.10.0.1
DNS     = 130.10.0.1
$ onevnet create test1.net
ID: 1
$ onevnet create test2.net
ID: 2
$ onevnet list
  ID USER         GROUP        NAME            CLUSTER      TYPE BRIDGE   LEASES
   0 oneadmin     oneadmin     test-network    -               R br0           2
   1 oneadmin     oneadmin     test1 LAN       -               R br0           0
   2 oneadmin     oneadmin     test2 LAN       -               F br1           0
```

以上.net 文件中的内容可以直接输入到 web 创建虚拟网络的高级模式中创建。

删除虚拟网络 **onevnet delete**

虚拟网络在使用过程中也是可以删除的，不像镜像，如果有虚拟主机在使用是删除不了的，使用 onevnet delete id 可以删除指定的网卡

```
$ onevnet delete 0
```

添加移除虚拟网络指定 ip **onevnet addleases/ rmleases**

```
$ onevnet addleases 0 130.10.0.10
$ onevnet addleases 0 130.10.0.11 50:20:20:20:20:31
$ onevnet rmleases 0 130.10.0.3
```

查看虚拟网络详细信息 **onevnet show**

```
$ onevnet show 1
```

```
    VIRTUAL NETWORK 1 INFORMATION
    ID             : 1
    NAME           : test1 LAN
    USER           : oneadmin
    GROUP          : oneadmin
    CLUSTER        : -
    TYPE           : RANGED
    BRIDGE         : br0
    … …

    VIRTUAL NETWORK TEMPLATE
    DNS="192.168.0.1"
    GATEWAY="192.168.0.1"
    NETWORK_ADDRESS="192.168.0.0/24"
    NETWORK_MASK="255.255.255.0"

    RANGE
    IP_START       : 192.168.0.3
    IP_END         : 192.168.0.254

    USED LEASES
    LEASE=[ MAC="02:00:c0:a8:00:03", IP="192.168.0.3", IP6_LINK="fe80::400:c0ff:fea8:3",
USED="1", VID="13" ]

    VIRTUAL MACHINES

        ID USER     GROUP    NAME            STAT UCPU    UMEM HOST             TIME
    … …
```

八. 制作模板

制作模板这章是重点，在这章就要说到之前卖的那个关子，如如何实现自动化配置网络、主机名等等，这就是 OpenNebula 闪耀的 Context 功能。

8.1 制作模板

模板就是包括创建虚拟机时使用到的系统镜像、以及各种配置的一个定义，所以制作模板是整个过程中非常重要的一个环节。

8.1.1 web 界面制作模板

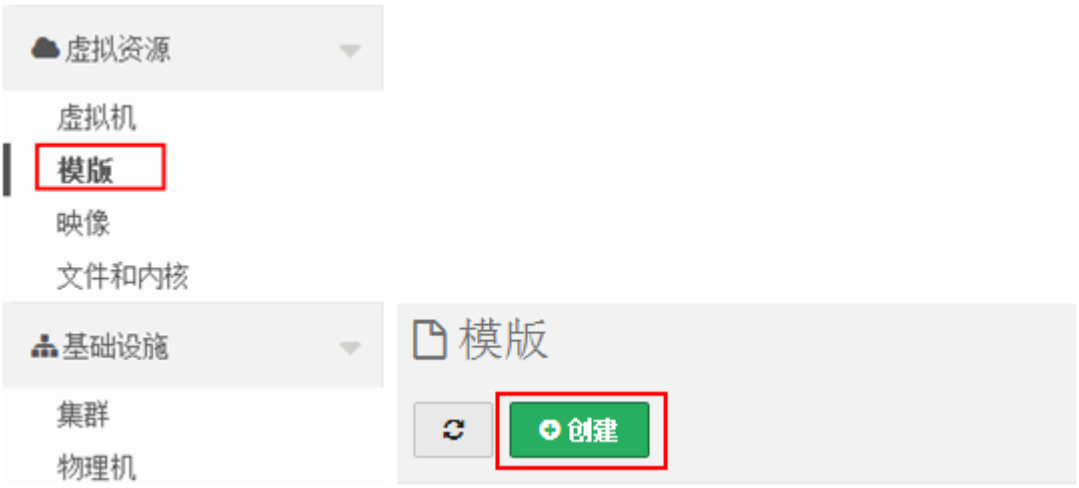

图 1.22 创建模板

下面为创建模板的选项截图，这几后我会逐一介绍各个选项的含义

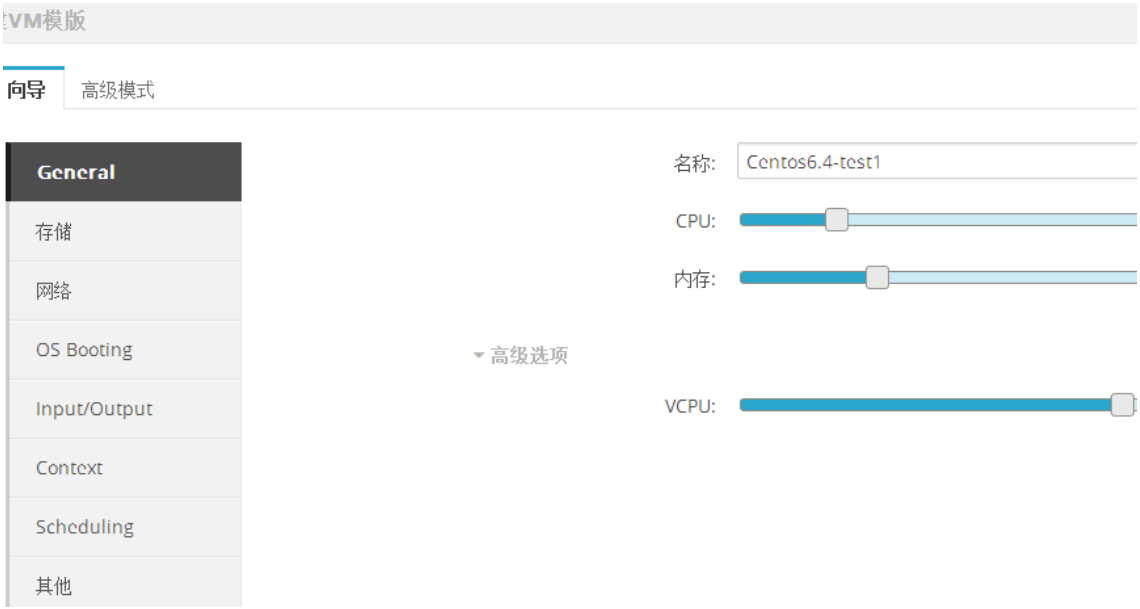

图 1.23 创建模板选项

8.1.1.1CPU 和 VCPU

上图中有个关于 CPU 和 VCPU 的选项，顾名思义 CPU 就是物理机真实的 CPU，VCPU 就是虚拟机中看到的虚拟 CPU 个数。这里的 CPU 数量表示 VM 在宿主机上实际占用的 CPU，

精确到 0.1。在 KVM 环境中，每个客户机都是一个标准的 Linux 进程（QEMU 进程），而每一个 vcpu 在宿主虚拟机中是 QEMU 进程派生的一个普通线程。

8.1.1.2存储

存储相关主要有一些这些选项

图 1.24 存储选项

- 首先是选择采用的映像，填写映像 ID 即可
- 驱动这里填写映像类型，kvm 选项就是 qcow2 和 raw
- DEV_PREFIX 指定设备以何种方式挂载，之前我们制作时使用的是 virtio，所以这里填写 vd，另外还有 sd 和 hd 方式
- CACHE，设置宿主机对块设备访问中的 cache 情况，可用选项为 none、writeback、writethrough 等，none 表示关闭缓存，writeback 读写性能较好。

另外推荐数据存储使用 raw，性能较 qcow2 要好，为数据安全性考虑，这里 CACHE 采用 none。点击 web 添加另外一块磁盘，这里只需要添加即可，后文会介绍创建虚拟机时如何格式化并挂载分区。

图 1.25 添加临时性磁盘

8.1.1.3网络

图 1.26 网络选项

网络部分选择对应网络即可，这里比较简单，也可以做一些安全选项。

8.1.1.4启动设置

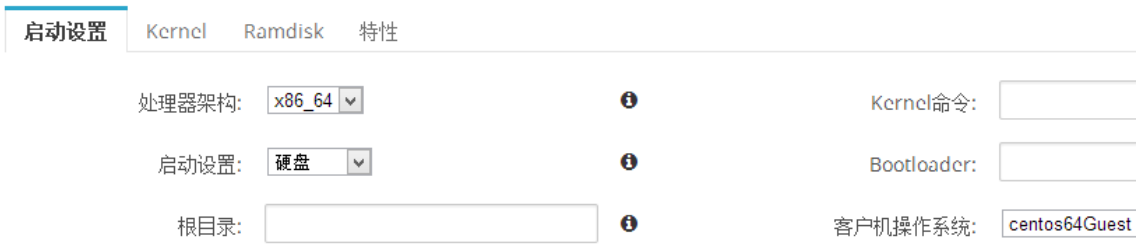

图 1.27 启动设置

启动设置包括处理器架构的选项，启动顺序，客户机操作系统选择以及 kernel 的启动参数设置。另外一个注意的地方是开启 ACPI 选项，如下图

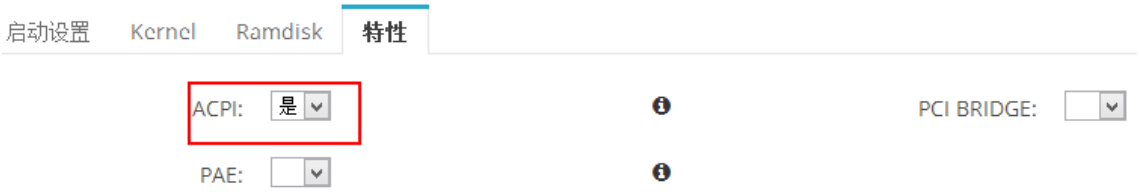

图 1.28 开启 ACPI

8.1.1.5输入输出配置

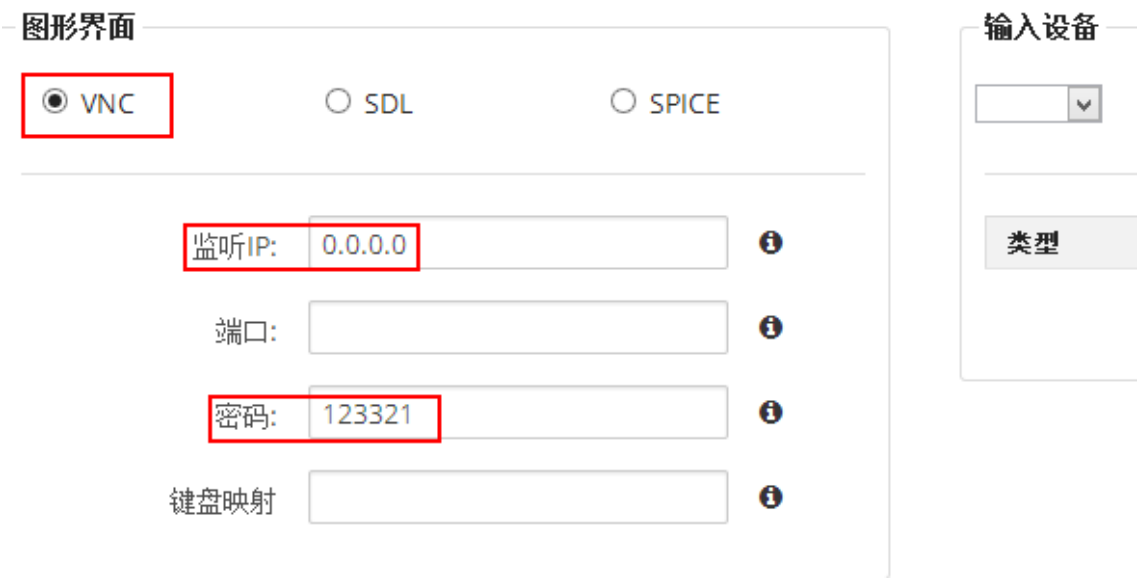

图 1.29 VNC 配置

Sunstone 自带 VNC 连接界面，设置相应监听 IP 和密码即可，端口不用填写。

8.1.1.6Context 初始化

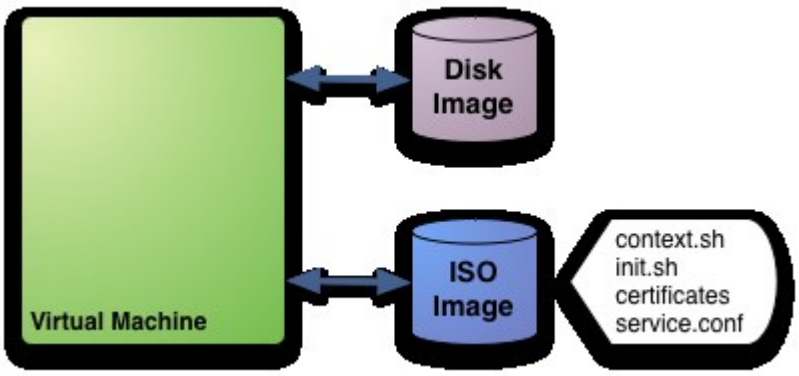

图 1.30 Context 原理图

Context 通过 ISO 镜像的方式配置新启动的虚拟机，即创建虚拟机的时候 Server 管理端会把相关的配置文件和脚本打包生成到一个 ISO 文件中，新启动的虚拟机会挂载该 ISO 镜像并执行其中的脚本以达到自动初始化虚拟机的目的。关于 context 有很多可以折腾的东西，这里讲解一些相对简单的配置。

8.1.1.7修改母盘镜像

这里我直接修改之前创建的母盘镜像，通过 qemu-kvm 结合 vnc 启动之前创建的系统镜像：

```
$ oneimage show centos-test1 | grep SOURCE        //获取之前创建镜像位置
SOURCE          : /var/lib/one//datastores/1/58c23e0a78405911f7591e21e33b0b9e
# su - root
# /usr/libexec/qemu-kvm -m 1024 -drive \          //开启系统镜像主机
file=/var/lib/one/datastores/1/58c23e0a78405911f7591e21e33b0b9e,if=virtio \
-net nic,model=virtio -net tap,script=no  -nographic -vnc :0
```

开启之后，通过 VNC View 连接：

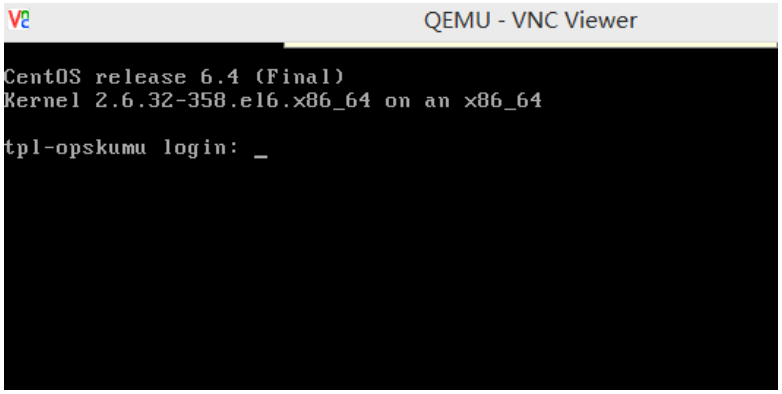

图 1.31 VNC 连接镜像系统

登陆系统，新建 context 脚本，脚本内容如下：

```
# chmod +x /etc/init.d/vmcontext
# cat /etc/init.d/vmcontext
#!/bin/bash

if [ ! -d /mnt/cdrom ]; then
    mkdir /mnt/cdrom
fi

mount /dev/cdrom /mnt/cdrom

if [ -f /mnt/cdrom/context.sh ]; then
    bash /mnt/cdrom/context.sh
fi
```

加入开机启动执行：

```
# ln -s /etc/init.d/vmcontext /etc/rc3.d/S01vmcontext
```

以上操作完成之后，保存退出系统即可。

8.1.1.8Context 脚本配置

Server 管理端需要安装 **genisoimage** 以生成 iso，一般系统默认已经安装，检查 Server 管理端是否安装，如果没有安装，yum 安装即可。以下为 context.sh 的一个简单实例，可根据实际需求定制相关内容：

```
# cat context.sh
```

```
#!/bin/bash
rm -f /etc/udev.d/

PATH="/sbin:/usr/sbin:/bin:/usr/bin"
export PATH

if [ -f /mnt/cdrom/context.sh ]; then
    . /mnt/cdrom/context.sh
fi

IP=`ifconfig eth0 | grep 'inet addr:' | cut -d: -f2 | awk '{print $1}'`
if [ "$IP" != "" ]; then
    rm -f /etc/rc3.d/S01vmcontext
    exit 0
fi

# hostname configuration
hostname $HOSTNAME
sed -i "/HOSTNAME=/s/=.*$/=$HOSTNAME/" /etc/sysconfig/network

# network configuration
IP=`echo $HOSTNAME | awk -F'-' '{print "192.168."$4"."$5}'`
GATEWAY="192.168.0.1"
MAC=`ifconfig eth0 | grep HWaddr | awk '{print $5}'`

ifconfig eth0 $IP netmask 255.255.255.0 up
route add default gw $GATEWAY

mkdir /root/.ssh
chmod 600 /root/.ssh
cat >> /root/.ssh/authorized_keys <<EOF
ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAQEAqOWjLRgobGCH5es/9d1yKSOxjkQ5fQx3CQI4sVugfrBH8N15RLo9qyPROO+Hvm9WZ+
v6aC1eCEKgjEzMWjSs4p59rWWcp5979wNjpQmXOva+K6gdiIBE7JOG8/i2mgEpXkUsX8uUBg4ZZSxouqpP2T7giQHSVhma
LdGKF+UOftEDmRvsMgkwOBao6CgxHxhM6T/EnVQZ348XsD1QVxYTsOb4sXfImGGkNOV9x6OIWuTRo9hwYND4I+5wi5v9Fb
kTCNI7+uD95EUwKt4uW2X3uxTuQ18ngMz1KqDoAOpdvej1SsIOAmtLQpRja2c0ShlacutaG/HSMUwf6Lij9fPlCQ==
root@kumu-ops
EOF
chmod 400 /root/.ssh/authorized_keys

echo -n > /etc/sysconfig/network-scripts/ifcfg-eth0
cat >> /etc/sysconfig/network-scripts/ifcfg-eth0 <<EOF
```

```
DEVICE=eth0
BOOTPROTO=static
ONBOOT=yes
TYPE=Ethernet
HWADDR=$MAC
NETMASK=255.255.255.192
IPADDR=$IP
GATEWAY=$GATEWAY
EOF

# the end
rm -f /etc/rc3.d/S01vmcontext

# rhel6 or centos6 need del the file 70-persistent-net.rules and reboot
# if not, the eth0 interface will rename to eth1
ifconfig eth0 >/dev/null 2>1
if [ "$?" != "0" ]; then
 rm -f /etc/udev/rules.d/70-persistent-net.rules
 reboot
fi
```

8.1.1.9添加 Context 自定义参数

Context 选项我们选择自定义参数，添加需要设置的参数

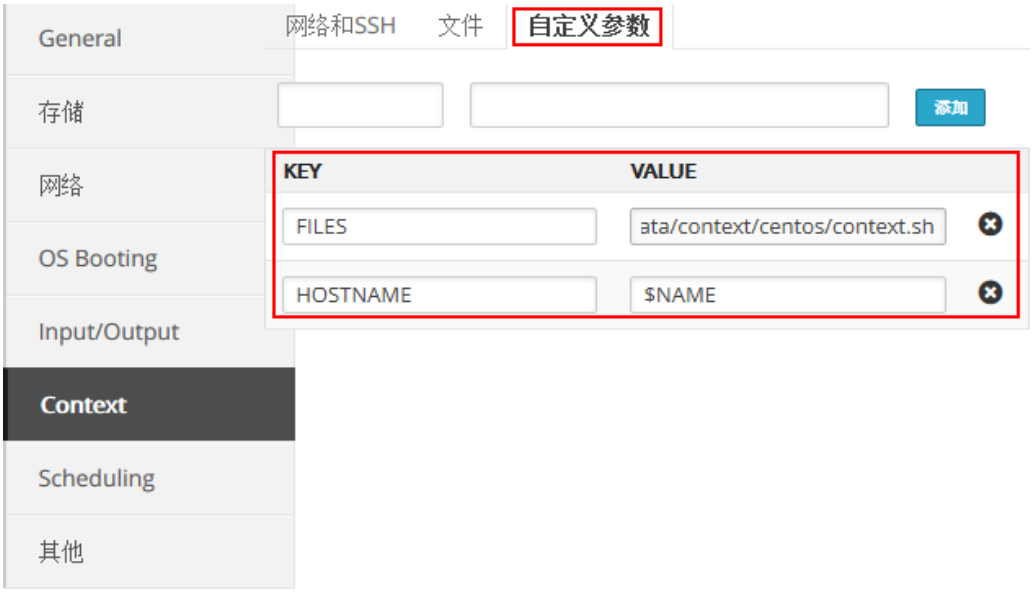

图 1.32 添加 context 自定义参数

上图中$NAME 表示创建虚拟机命名，通过 context 脚本可知，以虚拟机的命名命名系统主机名。FIFES 指定 context 脚本位置，注意脚本存放位置 oneadmin 用户必须有相关权限，建议存放在 oneadmin 用户目录下。

8.1.1.10 其他

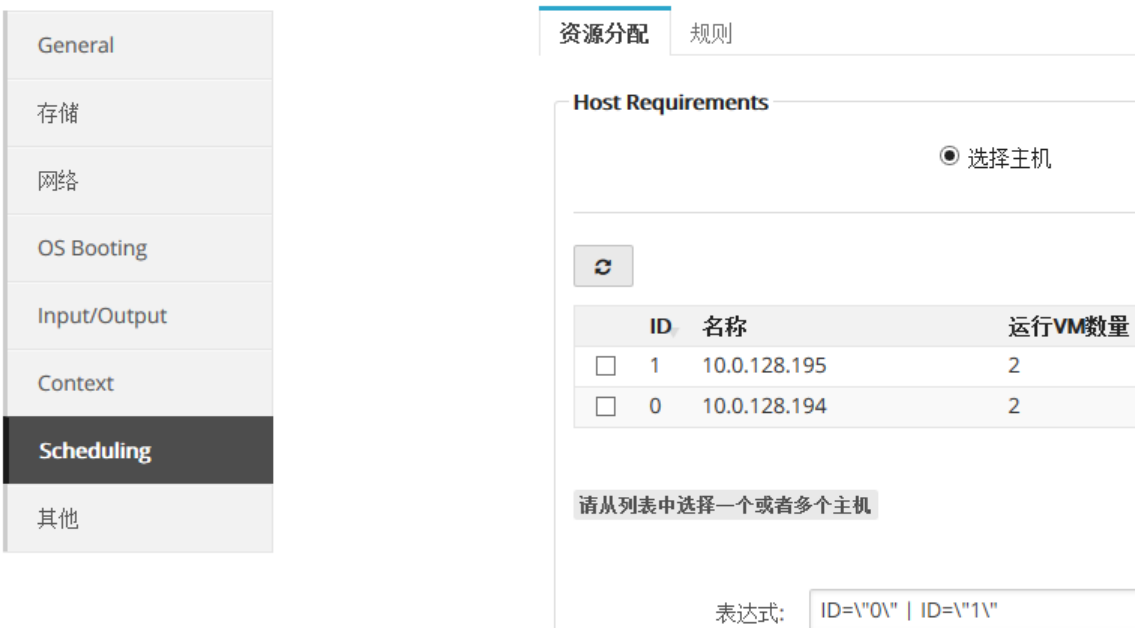

图 1.33 指定虚拟机安装主机

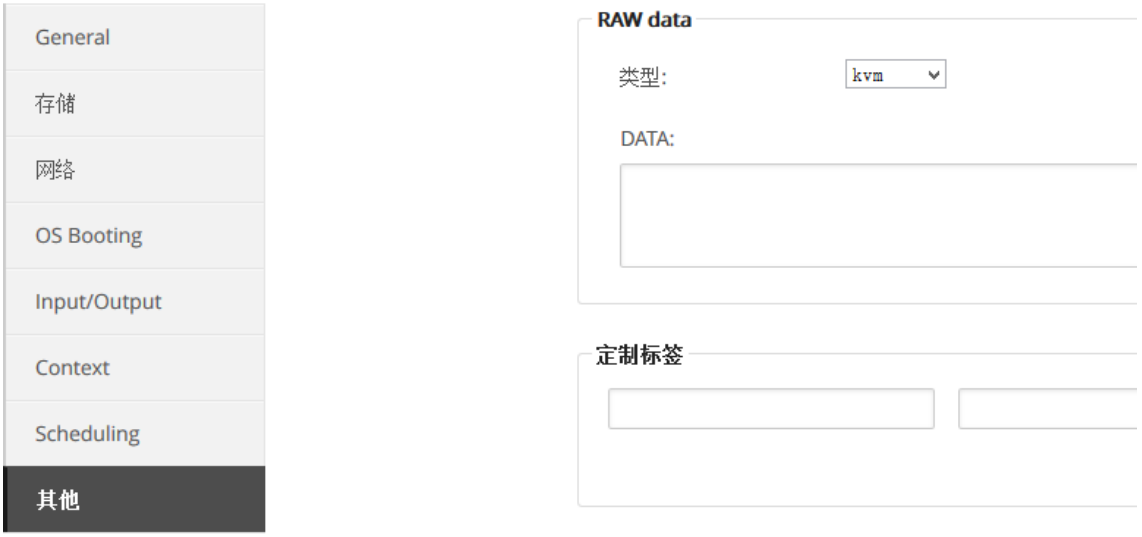

图 1.34 指定 RAW data 类型

完成之后点击创建即可。

8.1.2 命令行添加模板

8.1.2.1onetemplate 命令

onetemplate 使用方法和之前介绍的命令类似，在创建虚拟机模板之前，首先要创建模板文件，其实 web 界面创建完一个模板之后就会生成相应的内容。

选中之前创建的模板，点击右上角的更新配置

图 1.35 更新模板

选择高级模式，即可查看具体的模板内容

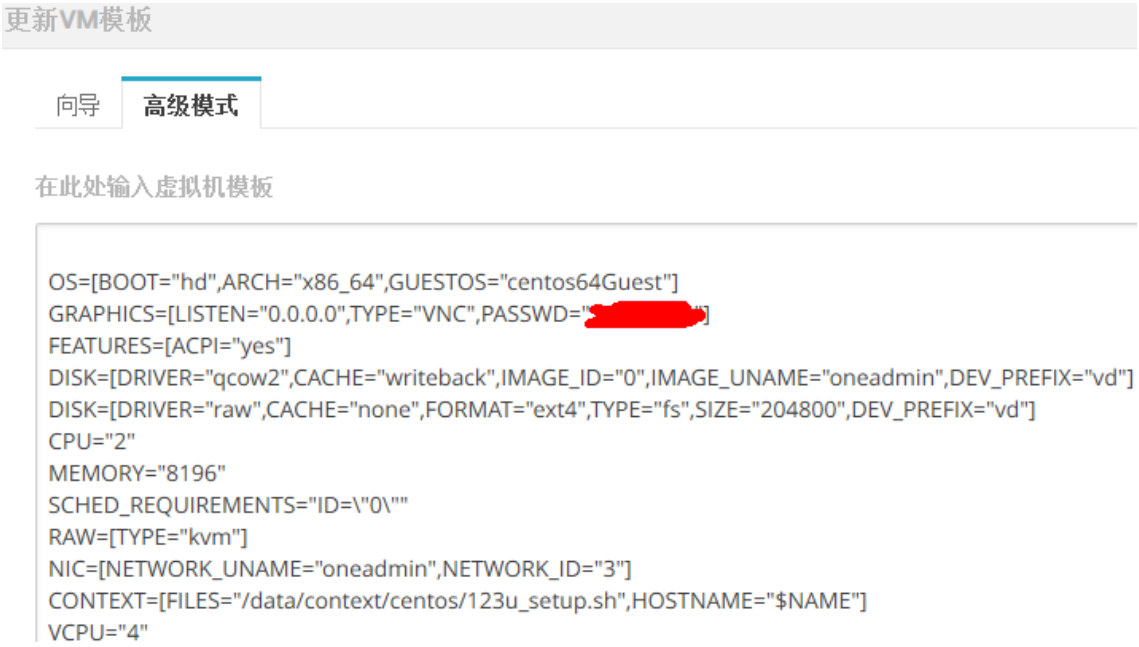

图 1.36 查看虚拟机模板

或者使用 onetemplate 命令获取以上模板信息。

查看模板详细信息 onetemplate show

```
$ onetemplate show 0
TEMPLATE 0 INFORMATION
ID             : 0
NAME           : centos5.8-tpl
USER           : oneadmin
GROUP          : oneadmin
REGISTER TIME  : 12/10 16:32:46

PERMISSIONS
OWNER          : um-
GROUP          : ---
OTHER          : ---

TEMPLATE CONTENTS                    # 之后的内容即为模板内容
CONTEXT=[
  FILES="/data/context/centos/context.sh",
  HOSTNAME="$NAME" ]
CPU="2"
DISK=[
  CACHE="writeback",
  DEV_PREFIX="vd",
  DRIVER="qcow2",
  IMAGE_ID="0",
  IMAGE_UNAME="oneadmin" ]
DISK=[
  CACHE="none",
  DEV_PREFIX="vd",
  DRIVER="raw",
  FORMAT="ext4",
  SIZE="204800",
  TYPE="fs" ]
FEATURES=[
  ACPI="yes" ]
GRAPHICS=[
  LISTEN="0.0.0.0",
  PASSWD="123123",
  TYPE="VNC" ]
MEMORY="8196"
NIC=[
  NETWORK_ID="3",
```

```
  NETWORK_UNAME="oneadmin" ]
OS=[
  ARCH="x86_64",
  BOOT="hd",
  GUESTOS="centos64Guest" ]
RAW=[
  TYPE="kvm" ]
SCHED_REQUIREMENTS="ID=\"0\""
VCPU="4"
```

直接拷贝以上 TEMPLATE CONTENTS 下内容修改相应选项即可

```
$ cat centos6-tpl.tpl
CONTEXT=[
  FILES="/data/context/centos/context.sh",
  HOSTNAME="$NAME" ]
NAME="contos6-tpl"          //添加模板名，需要加上
CPU="2"
DISK=[
  CACHE="writeback",
  DEV_PREFIX="vd",
  DRIVER="qcow2",
  IMAGE_ID="0",
  IMAGE_UNAME="oneadmin" ]
DISK=[
  CACHE="none",
  DEV_PREFIX="vd",
  DRIVER="raw",
  FORMAT="ext4",
  SIZE="204800",
  TYPE="fs" ]
FEATURES=[
  ACPI="yes" ]
GRAPHICS=[
  LISTEN="0.0.0.0",
  PASSWD="123123",
  TYPE="VNC" ]
MEMORY="8196"
NIC=[
  NETWORK_ID="3",
  NETWORK_UNAME="oneadmin" ]
OS=[
  ARCH="x86_64",
```

```
  BOOT="hd",
  GUESTOS="centos64Guest" ]
RAW=[
  TYPE="kvm" ]
SCHED_REQUIREMENTS="ID=\"0\""
VCPU="4"
```

创建模板 **onetemplate create**

```
$ onetemplate create centos6-tpl.tpl
ID: 1
```

模板列表 **onetemplate list**

```
$ onetemplate list
  ID USER            GROUP           NAME                          REGTIME
   0 oneadmin        oneadmin        centos5.8-tpl           12/10 16:32:46
   1 oneadmin        oneadmin        contos6-tpl             12/13 15:18:06
```

删除指定模板 **onetemplate delete**

```
$ onetemplate delete 1
$ onetemplate list
  ID USER            GROUP           NAME                          REGTIME
   0 oneadmin        oneadmin        centos5.8-tpl           12/10 16:32:46
```

九. 创建虚拟机

完成以上的内容之后就可以创建虚拟机了，虚拟机这边我也以 web 和命令行方式叙述

9.1 web 界面创建虚拟机

图 1.37 创建虚拟机

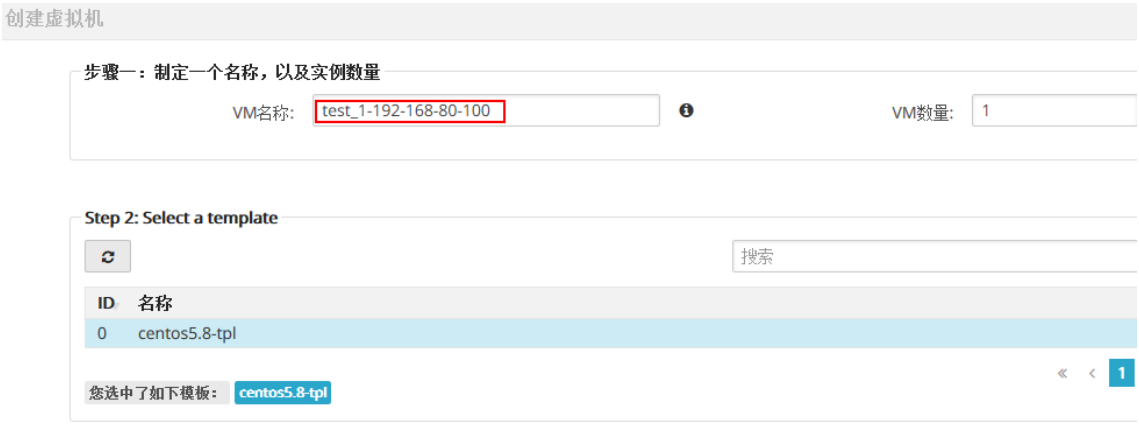

图 1.38 创建虚拟机选项

虚拟机名称按照之前 context.sh 脚本定义的来，`IP=`echo $HOSTNAME | awk -F'-' '{print "192.168."$4"."$5}'`` 因为 IP 配置也是按照虚拟机名来配置的，所以需要定义正确格式的虚拟机名才可以配置好 IP。

如果模板没有指定安装主机选项 **SCHED_REQUIREMENTS**，则创建之后还需要指定部署主机

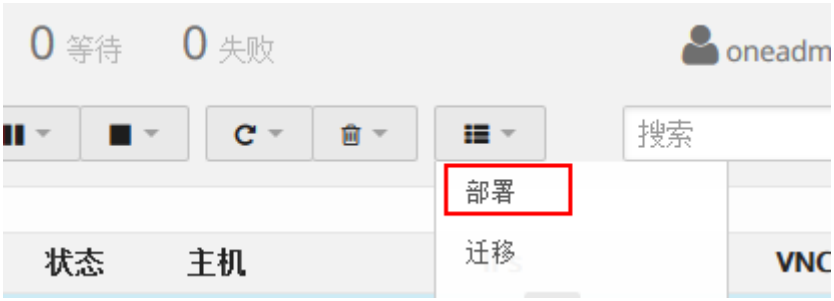

图 1.39 指定虚拟机部署主机

图 1.40 选择对应主机

完成以上操作之后，等待虚拟机创建，新创建的主机经历状态是 等待 → PROLOG → BOOT → RUNNING，创建完成之后，即可通过 web 端的 VNC 或者远程 ssh 管理虚拟主机。对于虚拟机的状态，OpenNebula 官方文档有详细的说明 vm_guide_2 。

9.2 命令行创建虚拟机

命令行创建主机使用 onevm 命令

9.2.1 onevm 命令

查看虚拟机详细信息 onevm show

```
$ onevm show 1          //1 为虚拟机 ID
… …
```

```
VIRTUAL MACHINE TEMPLATE          //之后内容即是 onevm 创建虚拟机所需文件内容
AUTOMATIC_REQUIREMENTS="!(PUBLIC_CLOUD = YES)"
CONTEXT=[
  DISK_ID="2",
  FILES="/data/context/centos/context.sh",
  HOSTNAME="test_1-192-168-80-200",
  TARGET="hda" ]
CPU="2"
FEATURES=[
  ACPI="yes" ]
GRAPHICS=[
  LISTEN="0.0.0.0",
  PASSWD="123123",
  PORT="5925",
  TYPE="VNC" ]
MEMORY="8196"
OS=[
  ARCH="x86_64",
  BOOT="hd",
  GUESTOS="centos64Guest" ]
TEMPLATE_ID="0"
VCPU="4"
VMID="25"
```

拷贝以上内容修改成实际选项到文件 test1.vm，然后使用 onevm 创建。

创建虚拟机 onevm create

```
$ onevm create test1.vm
```

查看虚拟机列表 onevm list

```
$ onevm list
```

删除指定虚拟机 onevm delete

```
$ onevm delete test1.vm
```

十. Windows Server 镜像制作

10.1 创建虚拟机镜像文件

```
# qemu-img create -f qcow2 win_2008.qcow2 50G
```

10.2 安装虚拟机

网卡推荐使用 e1000，磁盘类型使用 virtio。使用 virtio 需要安装 virtio 设备驱动，否则默认是不能识别 virtio 设备的。

下载 fedora 项目组最新的 virtio 驱动 iso：

- virtio 官网地址目前的提供 virtio 最新驱动 iso 为 virtio-win-0.1-65.iso ，下载之后使用如下方式安装

```
# kvm -m 2048 -cdrom cn_windows_server_2008_r2.iso \
-drive file=/data/virtio-win-0.1-65.iso,media=cdrom -drive file=win_2008.qcow2,if=virtio \
-net nic,model=e1000 -net tap,ifname=tap0,script=no  -boot d -nographic -vnc :0
```

客户端通过 VNC 连接，启动之后 virtio 的硬盘是不能被识别的，如下图

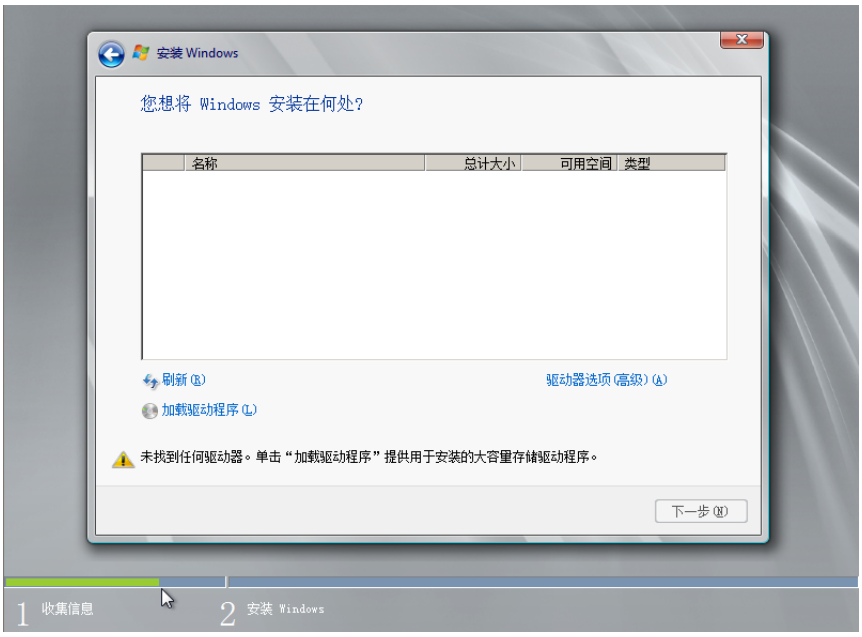

图 1.41 virtio 磁盘不被识别

此时，需要安装 virtio 驱动，点击加载驱动程序，Windows Server 2008 对应 win7 版本

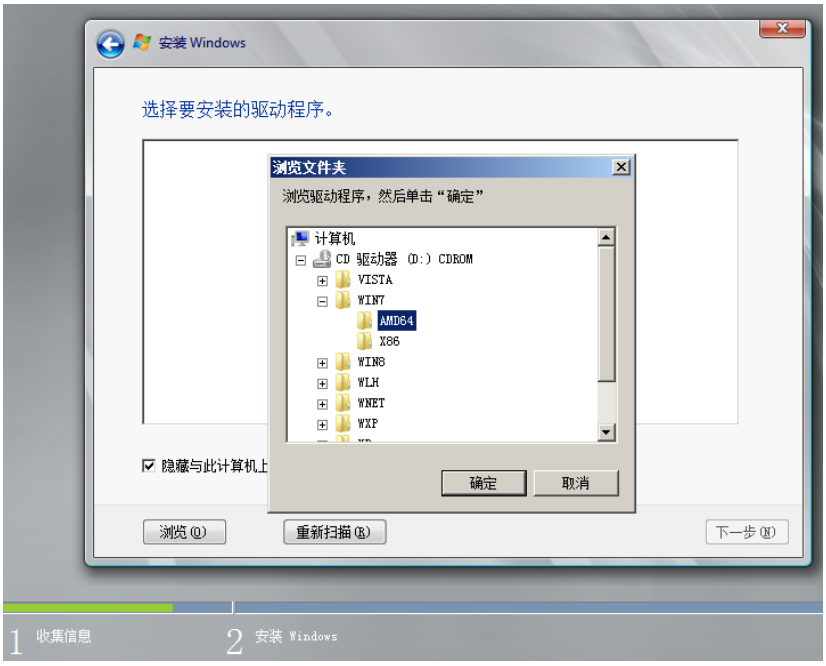

图 1.42 添加驱动所在目录

选择之后，按提示操作安装，安装完之后就会识别之前建立的磁盘

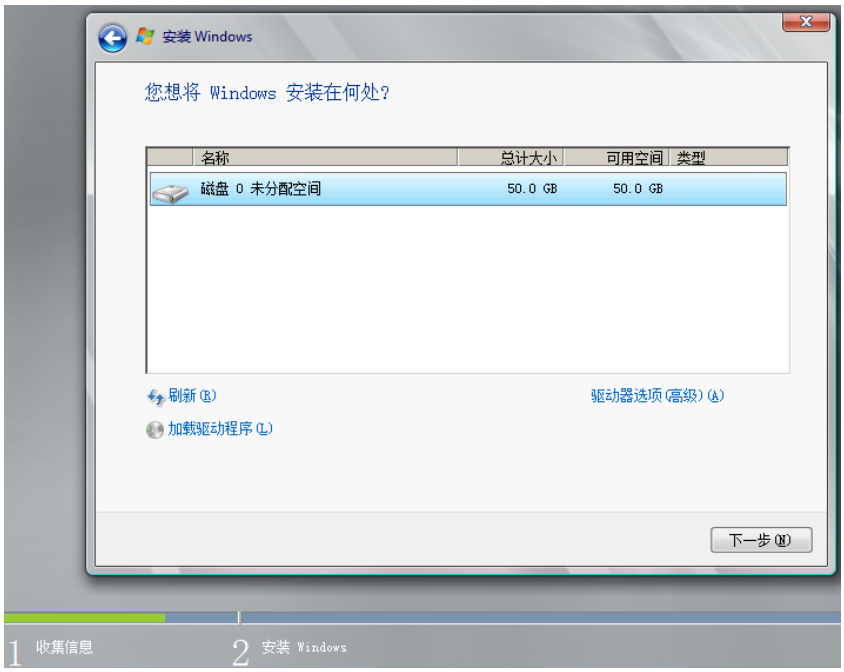

图 1.43 驱动安装识别磁盘

磁盘识别之后就可以之后的完整系统的安装，关于 Windows Server 2008 的安装步骤这里不再赘述。

10.3 Windows 相关配置

10.3.1 激活操作

安装完 Windows 之后首先需要激活操作

10.3.2 配置远程桌面

10.3.2.1 步骤 1 开启远程桌面

鼠标右击 [计算机] 选择 [属性]，之后点击远程设置，如下图，选择允许任意版本远程桌面的计算机连接，默认用户 administrator 已经有远程访问权限了。

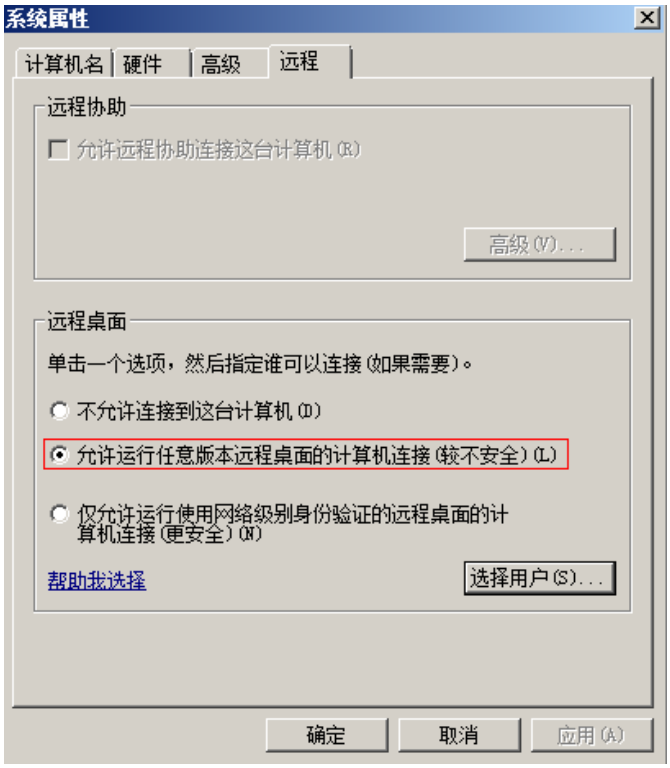

图 1.44 开启远程桌面

10.3.2.2 步骤 2 每个用户支持 2 个远程会话

默认每个远程桌面只能进行一个会话，通过简单配置最多可以使得每个用户进行 2 个会话。简单设置步骤比较简单，[控制面板] -- [管理工具] -- [远程桌面服务] -- [远程桌面会话主机配置]

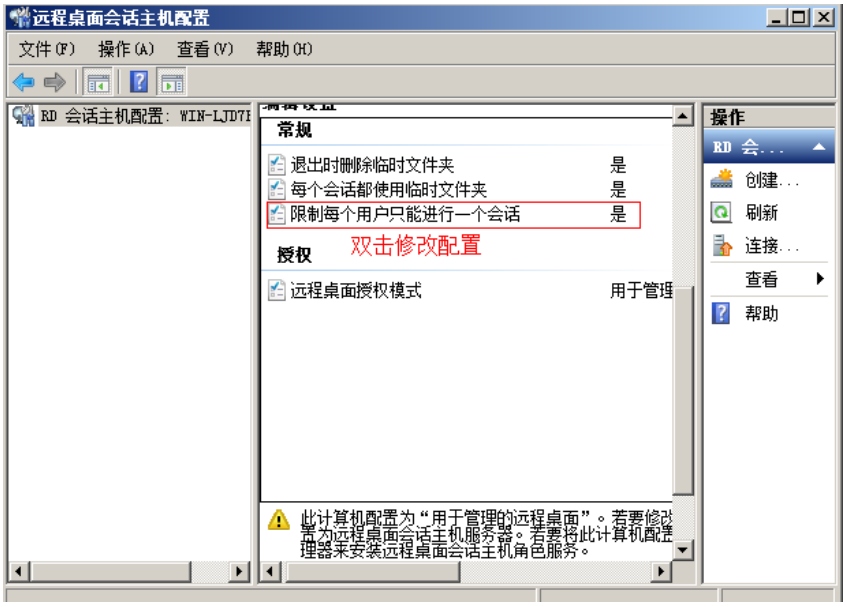

图 1.45 限制用户会话设置

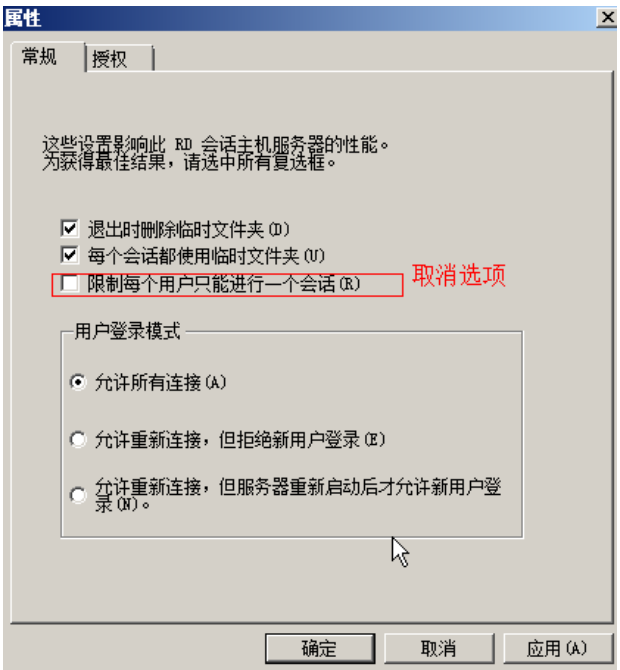

图 1.46 取消限制会话

如上操作之后，之前已经说明只能支持一个用户同时登录 2 个会话，如果需要支持更多的会话那么久需要进行步骤 3 的操作，即添加授权服务器。

10.3.2.3 步骤 3 添加授权服务器

关于授权服务器的添加，可以参考 Windows server 2008 R2 远程桌面终端连接数的破解 ，根据文档的步骤一步一步来即可。

10.3.3 msdtc 设置

因为虚拟机的批量产生相当于克隆，会造成虚拟机的 msdtc 值相同，需要将 msdtc 服务删掉，命令行执行如下操作步：

```
命令行运行删除命令：MSDTC －uninstall
```

如果需要开启，则命令行运行如下：

```
MSDTC －install
net start msdtc
```

10.3.4 其它配置

1. 关闭防火墙，[控制面板] -- [Windows 防火墙] -- [打开或关闭 Windows 防火墙] ，选择关闭。
2. 修改如下两处注册表：

✓ [HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\policies\system]
- "ShutdownWithoutLogon"=dword:00000001

✓ [HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Windows]
- "ShutdownWarningDialogTimeout"=dword:00000001

10.4 导入镜像和新增模板

导入镜像参见之前章节内容，这里贴出模板内容

内容可参考如下信息，实际操作时需要做相应的修改

```
CPU="1"
DISK=[
  BUS="virtio",
  CACHE="writeback",
```

```
  DRIVER="qcow2",
  IMAGE_ID="2",          # 镜像选择，需要根据实际情况修改
  TARGET="vda",
  TYPE="disk" ]
FEATURES=[
  ACPI="yes" ]           # 如果不添加则 OpenNebula VNC 连接黑屏
GRAPHICS=[
  LISTEN="0.0.0.0",
  PASSWD="123123",
  TYPE="vnc" ]
MEMORY="4096"
NAME="win2008_4G_8C"   # 镜像名，需根据实际情况修改
NIC=[
  MODEL="e1000",
  NETWORK_ID="2" ]     # 虚拟网络 ID，根据实际情况修改
NIC=[
  MODEL="e1000",
  NETWORK_ID="3" ]     # 同上
OS=[
  ARCH="x86_64",
  BOOT="hd" ]
RAW=[
  DATA="<clock offset='localtime'>
                <timer name='pit' tickpolicy='delay'/>
                <timer name='rtc' tickpolicy='catchup'/>
                <timer name='hpet' present='no'/>
        </clock>",
  TYPE="kvm" ]
REQUIREMENTS="HOSTNAME=\"pm\""
TEMPLATE_ID="7"
VCPU="8"
```

Windows 相关的东西暂且只介绍这些，之前涉及到一些所以也就记录下来了。

十一. 磁盘热插拔

KVM 支持以下两种磁盘类型的热插拔

- sd: SCSI (default).

- vd: virtio.

KVM 虚拟机需要开启 acpi 才支持磁盘的热插拔，使用 OpenNebula 的时候需要设置如下选项：

```
FEATURES = [ acpi="yes" ]
```

11.1 virtio 磁盘热插拔

虚拟机加载 acpiphp 驱动

```
# modprobe acpiphp     # 加载驱动
```

选择需要添加磁盘的虚拟机-存储-挂载新硬盘 ，配置完成之后选择挂载使用

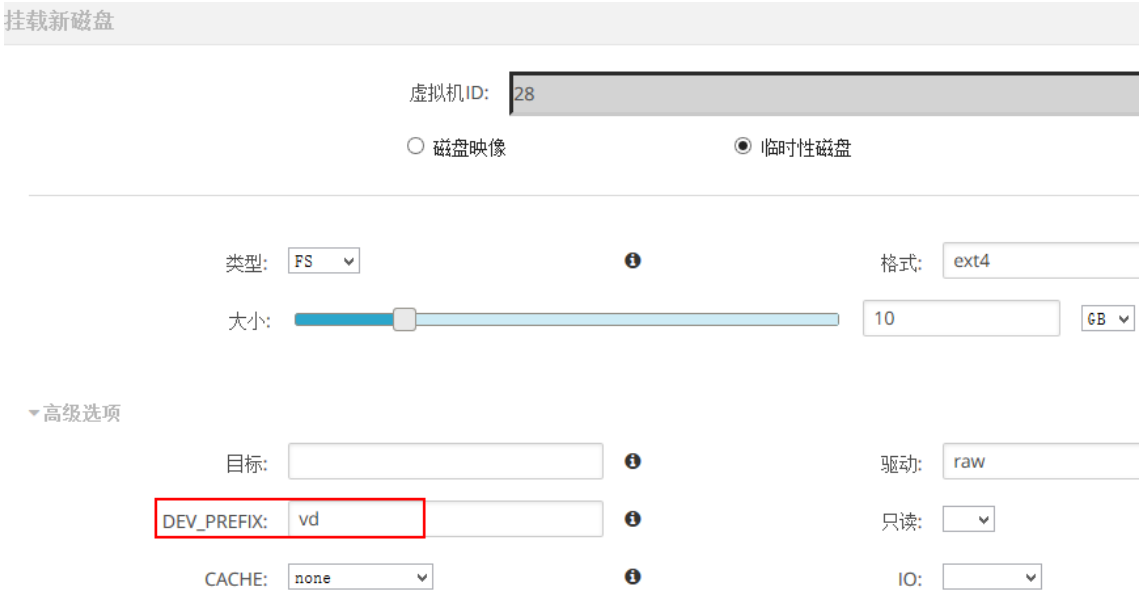

图 1.47 挂载 virtio 类型硬盘选项

Device Prefix 选择 sd 表示 scsi 磁盘，如果是 vd 则是 vitio 类型磁盘。选择挂载之后，刷新页面就会看到新建的磁盘。

11.2 scsi 磁盘热插拔

虚拟机加载 acpiphp 驱动

```
# modprobe acpiphp     # 加载驱动
```

选择需要添加磁盘的虚拟机-存储-挂载新硬盘 ，配置完成之后选择挂载使用

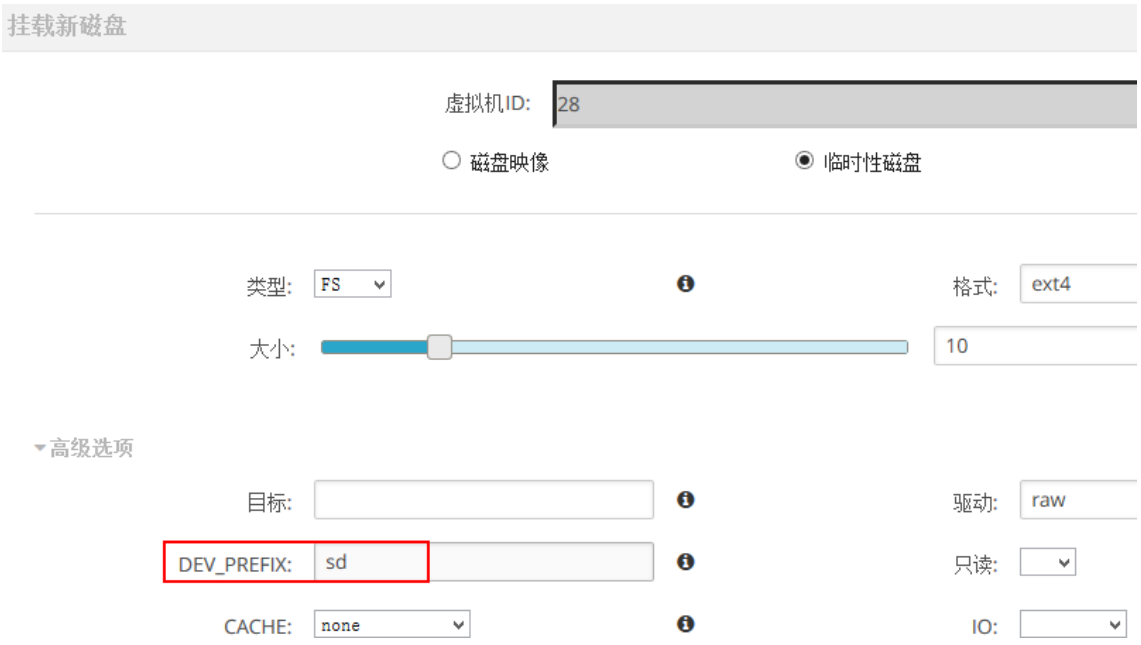

图 1.48 挂载 SCSI 类型硬盘选项

如果未识别 scsi 磁盘，执行如下命令，使得 KVM 虚拟机识别

```
# echo '- - -' > /sys/class/scsi_host/hostX/scan # X 一般为 0
```

十二. 总结

本文档特别感谢老大 **san** 的悉心指导，另外文档的完成离不开网络的帮助，OpenNebula 官网的文档目前已经非常完善了，其中大部分的内容都是按照最新官方文档来做的，KVM 虚拟化的部分参考了**《KVM 虚拟化：实战与原理解析》**一书，部分图片采用网上移动刘卫军**《OpenNebula 架构及应用案例分析》**一文。之所以共享出来，是希望可以帮助到需要的朋友，本文取之于网络，如此也就归于网络。