# Cobbler Web界面操作（一）

上传镜像

1、上传镜像[](https://s3.51cto.com/wyfs02/M00/6F/53/wKioL1WZP0bBpPj7AAHbqpeCmyk224.jpg)

1)、点击上图标记的1处，在右边显示出导入镜像的菜单。

2）、在标记的2处填入你要上传镜像的名字（命名规则按上面方式）

3）、在标记的3处，填入镜像文件挂在的目录。（操作如下）

创建iso镜像目录   mkdir /iso

通过filezilla工具把原始镜像上传到系统服务器的/iso目录下

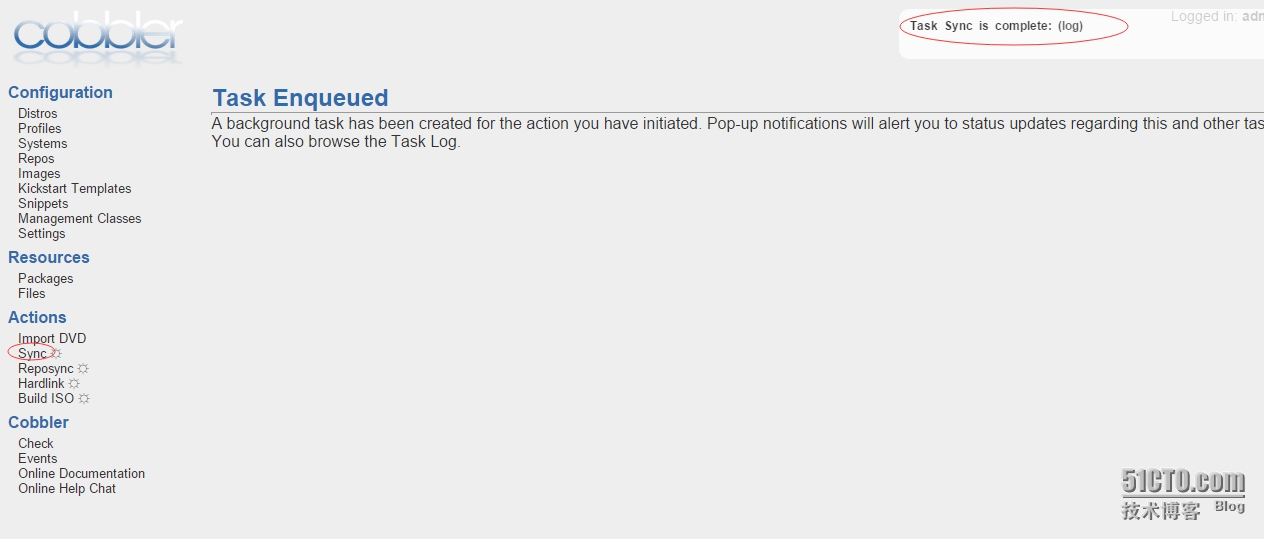
创建镜像文件的挂在目录  mkdir -pv /mnt/cdrom/CentOS-7-x86\_64

把原始镜像挂在到该目录 mount -o loop /iso/CentOS-7-x86\_64-DVD-1708.iso /mnt/cdrom/CentOS-7-x86\_64

查看挂在情况

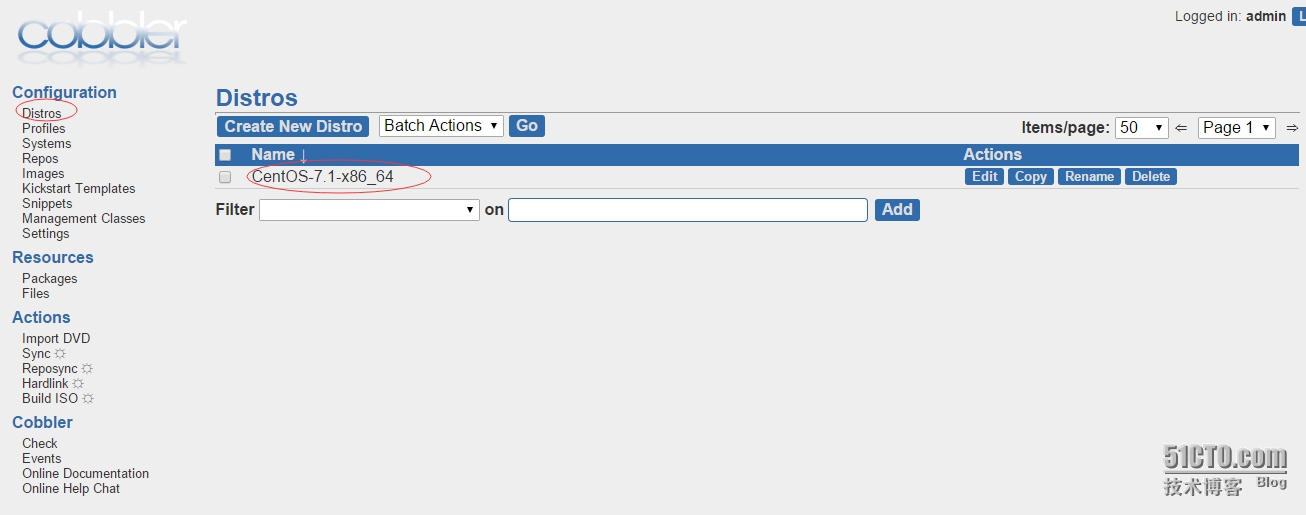
[root@localhost CentOS-7-x86\_64]# df -h

2、第一步点击run后，点击同步

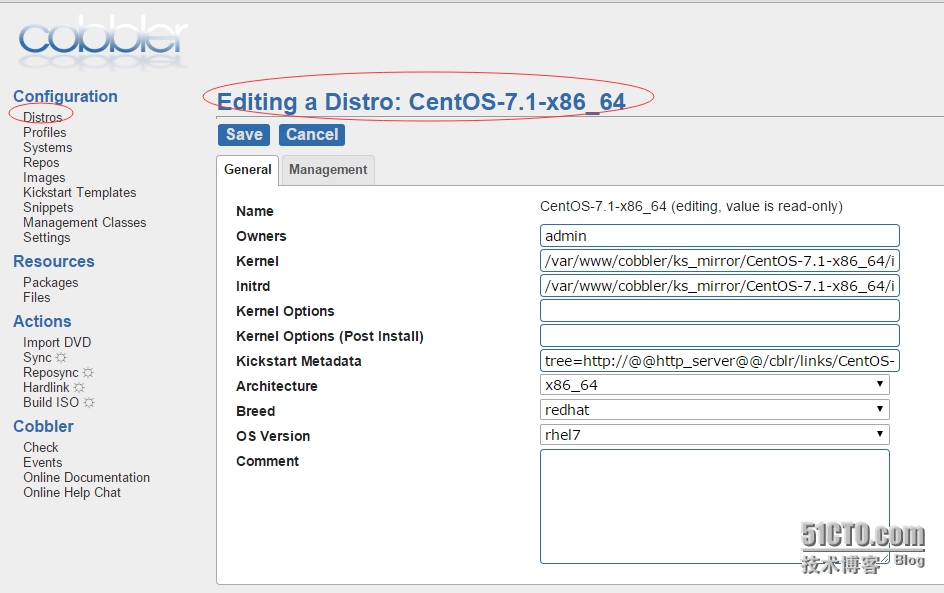
[](https://s3.51cto.com/wyfs02/M02/6F/53/wKioL1WZQfmSp6RGAAIKpFgzqeI343.jpg)

上传镜像大约需要10分钟。

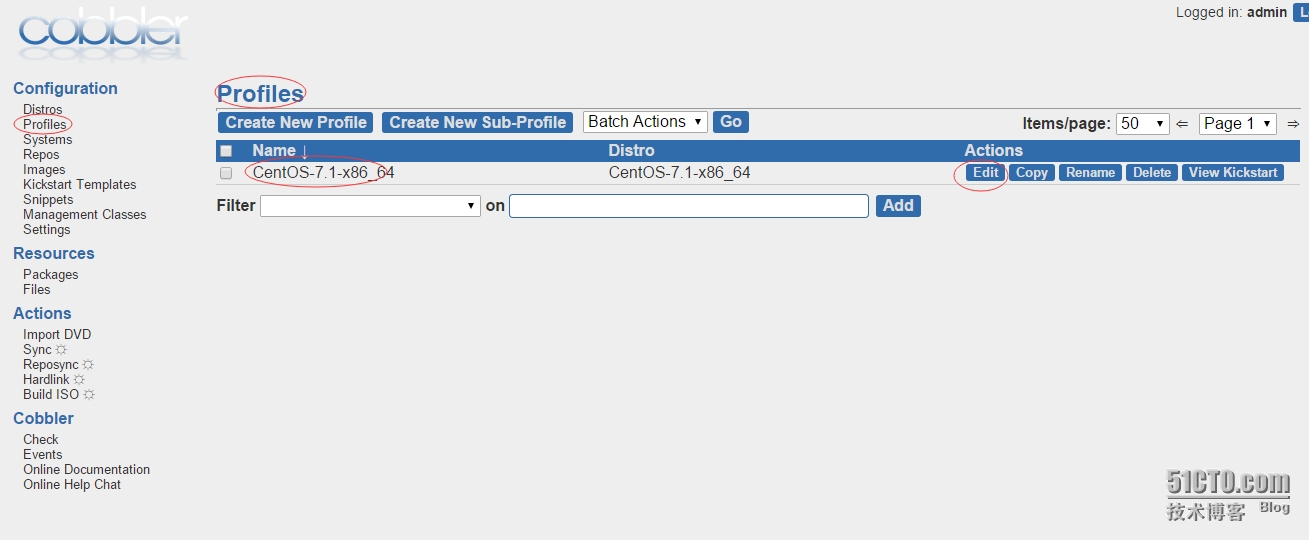
3、查看刚才上传镜像后自动生成的发型版本

[](https://s3.51cto.com/wyfs02/M02/6F/55/wKiom1WZQbOw9f9UAAIm3QwXAdU377.jpg)

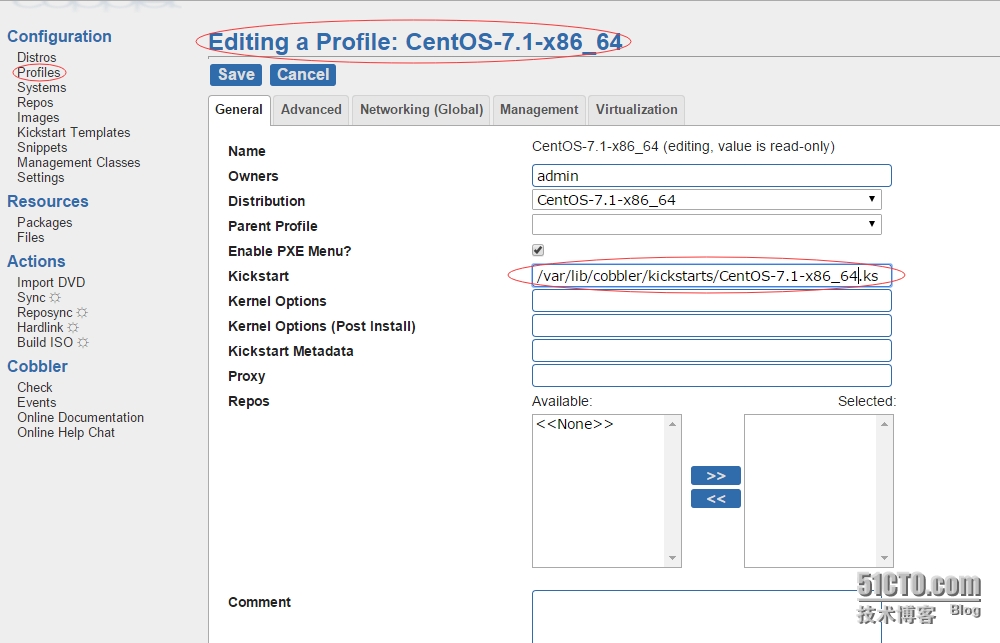
4、点击编辑查看发型版本的内容

[](https://s3.51cto.com/wyfs02/M01/6F/56/wKiom1WZQkLCPNSiAANHJZBwOL4640.jpg)

5、查看刚才上传镜像后自动生成的profile

[](https://s3.51cto.com/wyfs02/M00/6F/53/wKioL1WZRMDSErg_AAJox-CXm9w254.jpg)

6、查看并修改profile中的ks的名字（自己定义的ks文件）

[](https://s3.51cto.com/wyfs02/M01/6F/56/wKiom1WZQzaxAc6JAANIRme0wwg133.jpg)

# Cobbler Web界面操作（二）

创建repo源（以openstack kilo版本为例）

一、创建kilo版本的本地源

1、创建一个repo源的一级存储目录

mkdir /yum.repo

2、创建openstack kilo版本的目录

mkdir /yum.repo/os-kilo/

3、把openstack kilo版本安装用到的所有的rpm包，cp到/yum.repo/os-kilo/目录。

4、创建repo源

createrepo /yum.repo/os-kilo/

5、把该一级目录/yum.repo发布出去可以http访问（防止端口冲突用8080端口）

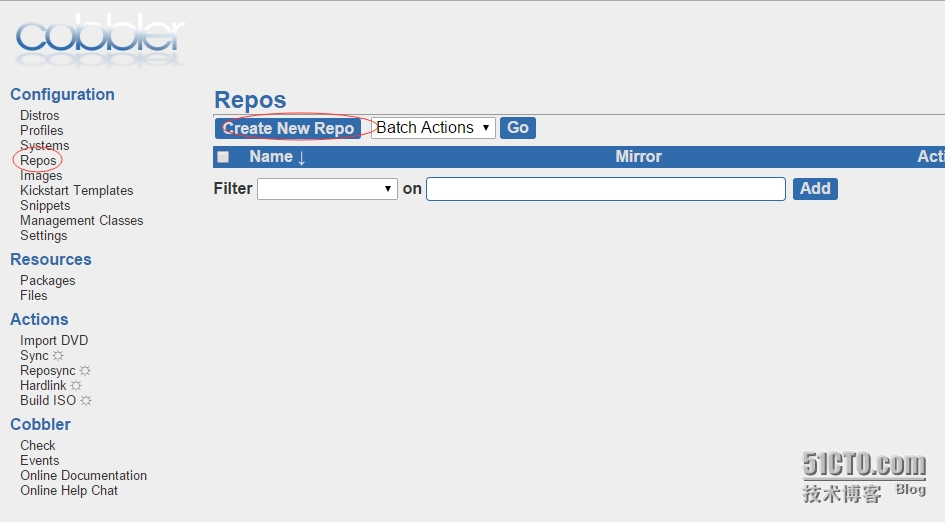
cd /yum.repo/;nohup python -m SimpleHTTPServer 8080 &

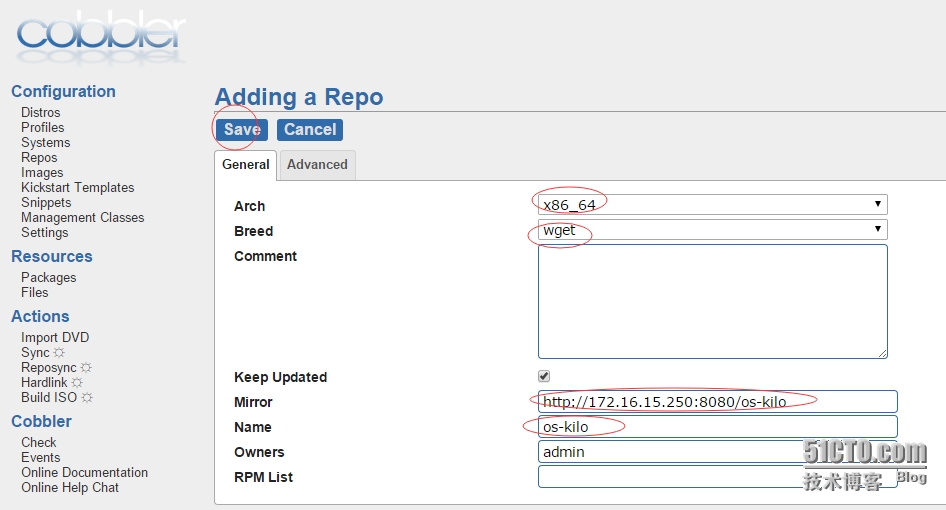
6、访问kilo版本源

[http://172.16.15.250:8080/os-kilo/](http://172.16.15.250:8080/os-kilo/" \t "https://blog.51cto.com/zhanguo1110/_blank)

二、创建cobbler repo源

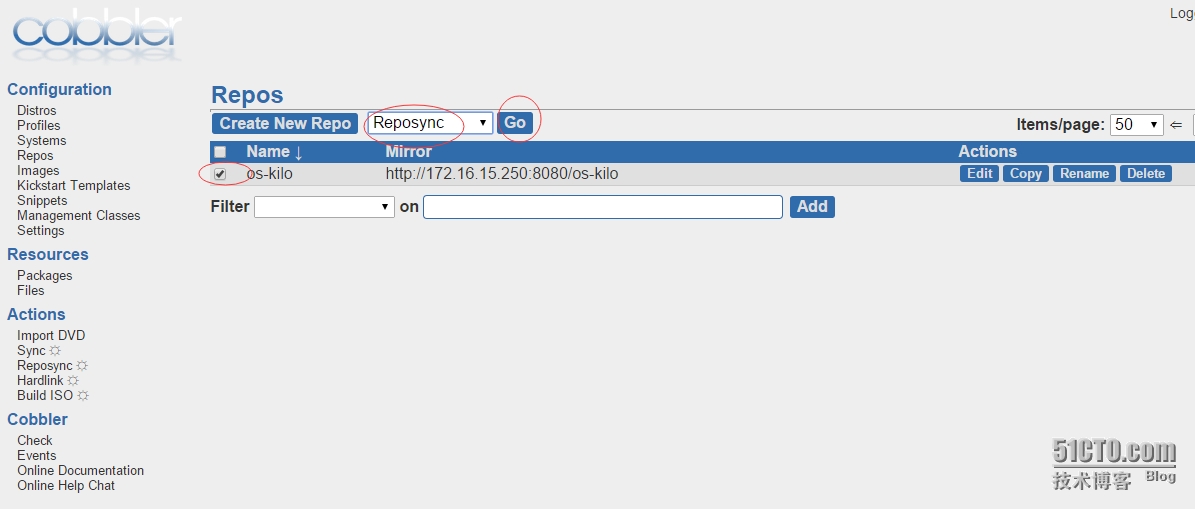
1、添加repo源

[](https://s3.51cto.com/wyfs02/M02/6F/60/wKiom1WaScShd5V0AAGst8hspMU252.jpg)

[](https://s3.51cto.com/wyfs02/M02/6F/5D/wKioL1WaS5HT6vkhAAJJ_vq2TZo277.jpg)

上图 Mirror地址就是kilo本地源所在的访问地址

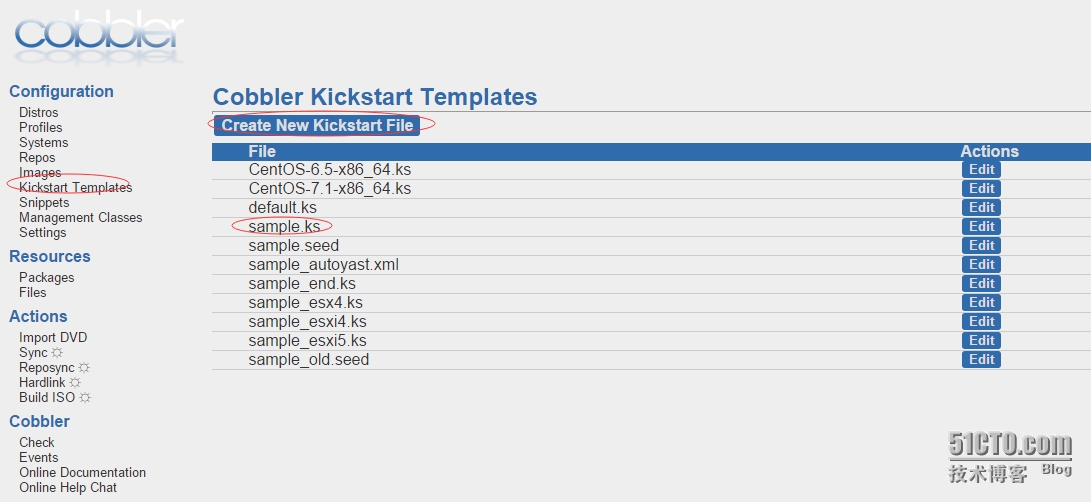
2、cobbler源与本地源同步

[](https://s3.51cto.com/wyfs02/M02/6F/5D/wKioL1WaS8XTNftqAAIdcFCL-Ng234.jpg)3、去cobbler服务器查看cobbler repo是否同步成功

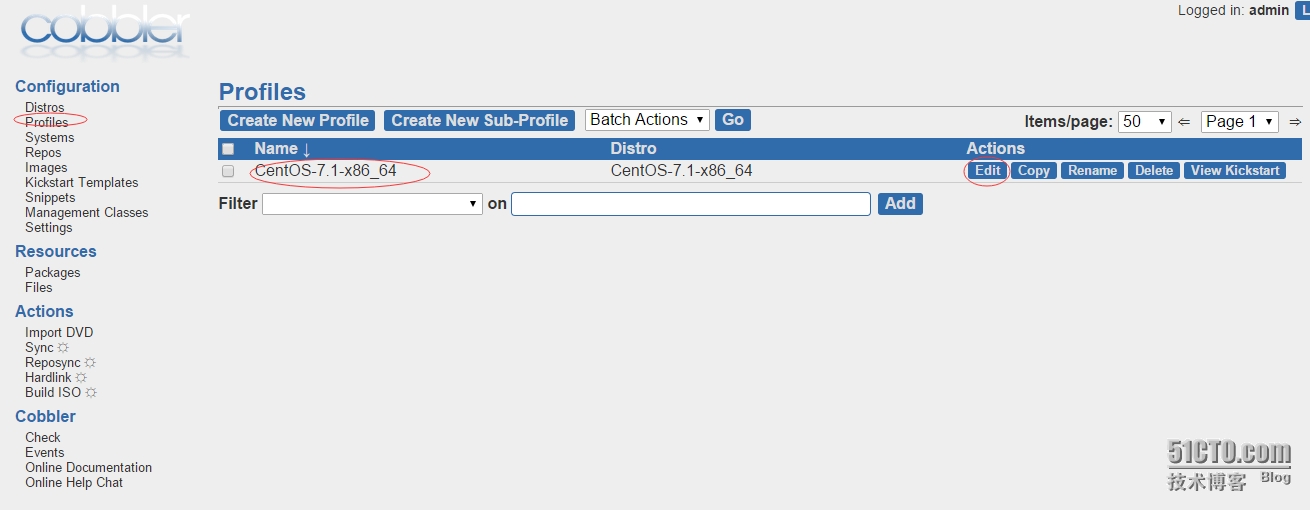
ls /var/www/cobbler/repo\_mirror/os-kilo/

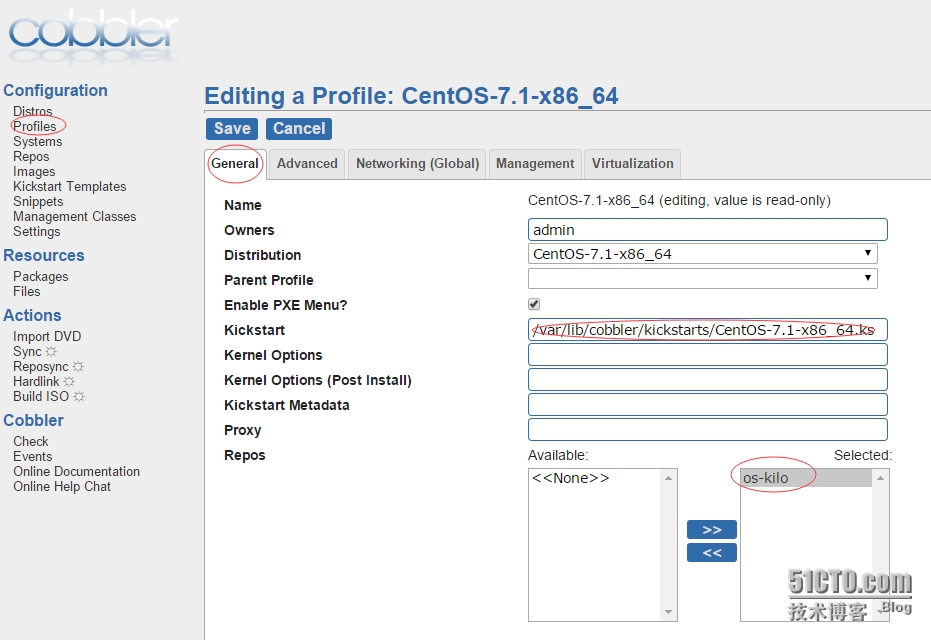
# Cobbler Web界面操作（三）

1、添加自定义的ks文件

[](https://s3.51cto.com/wyfs02/M02/6F/5E/wKioL1WaZOeT4JAzAALAyWHzbEM112.jpg)添加自己定义的ks文件时可以参照sample.ks的配置模板编写。

2、编辑profile

[](https://s3.51cto.com/wyfs02/M02/6F/61/wKiom1WaZDmz4VIRAAJLcBnHxV0822.jpg)

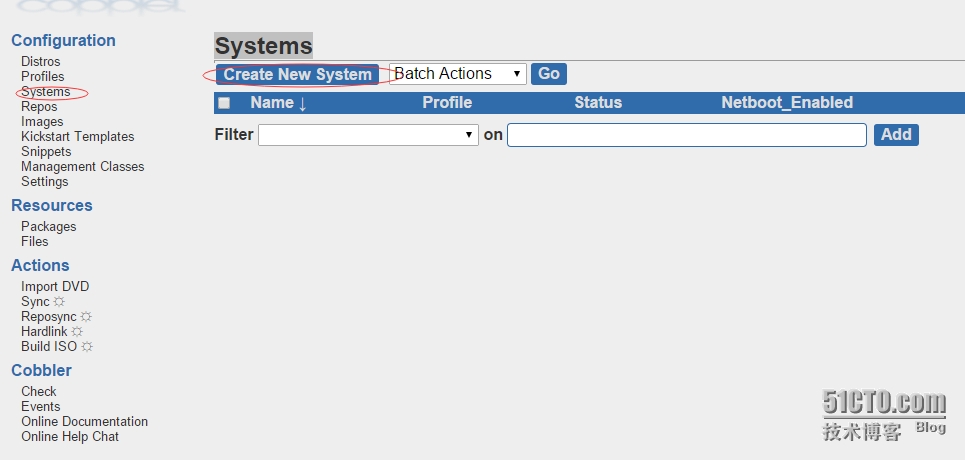
[](https://s3.51cto.com/wyfs02/M00/6F/5E/wKioL1WaZgaRyHD1AAMzRB9V3Rs159.jpg)

在General标签里：关联自定义的ks文件的路径，选择要使用的cobbler repo；

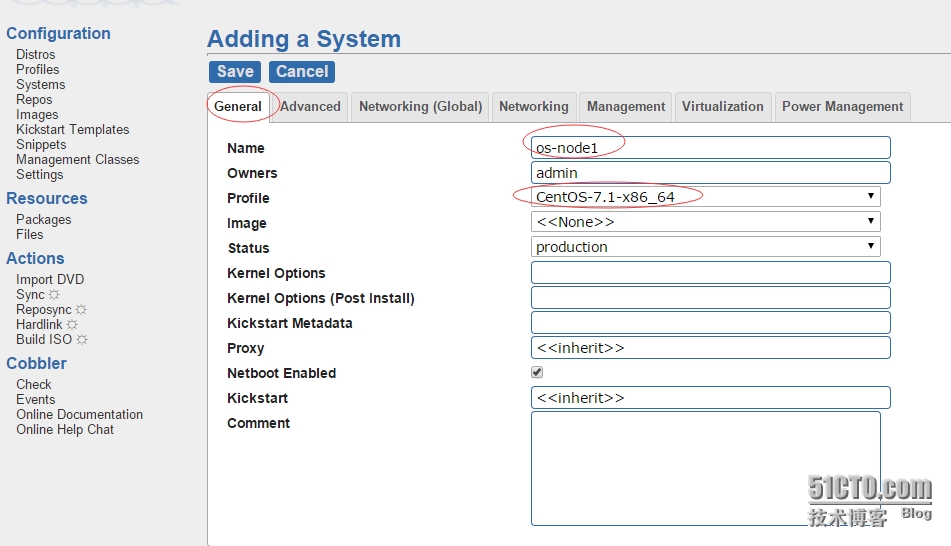
# Cobbler Web界面操作（四）

添加Systems

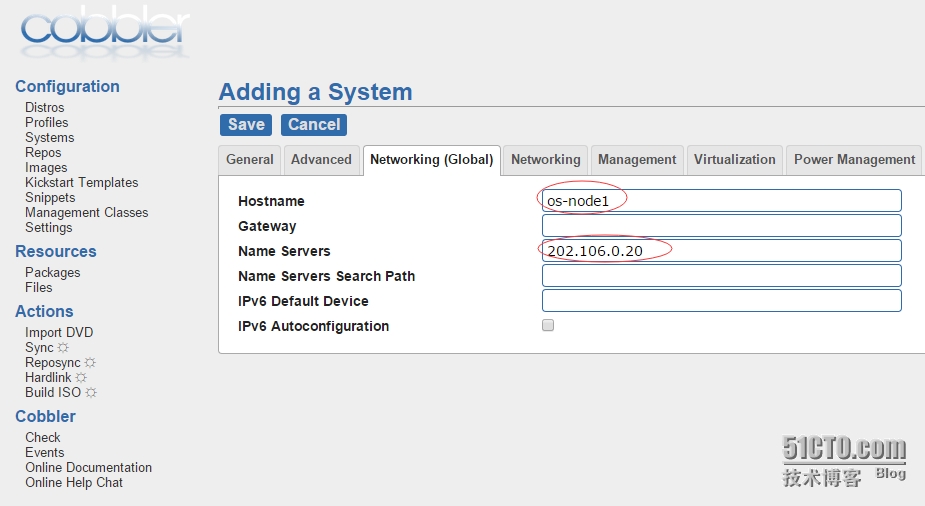
1、开始添加system系统

[](https://s3.51cto.com/wyfs02/M02/6F/5F/wKioL1Waao-BXQ9qAAGXb1ftp-Y388.jpg)

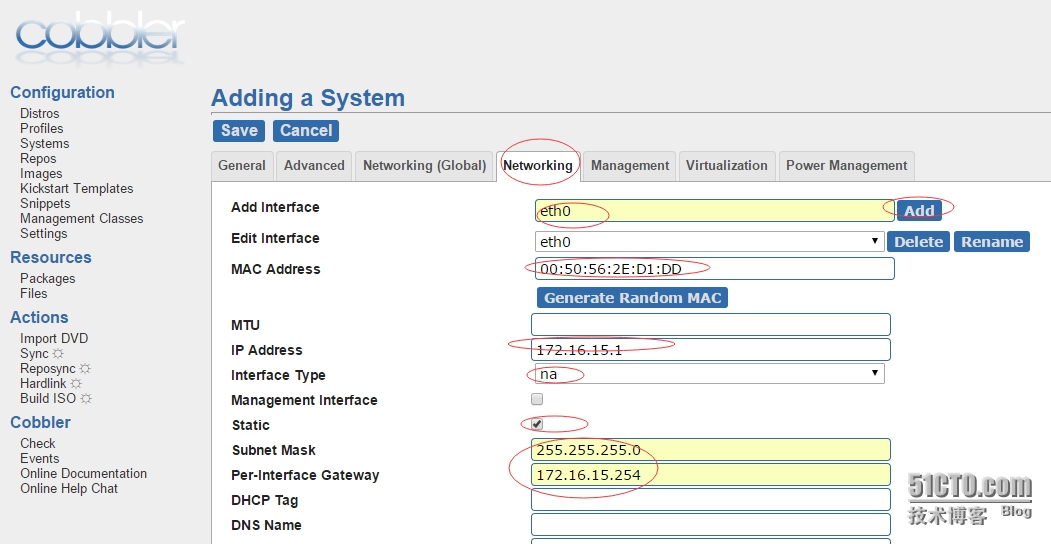
2、编辑General标签，添加系统的名字，选择对应的profile

[](https://s3.51cto.com/wyfs02/M02/6F/61/wKiom1WaaMPyfKWYAALDFoFHp-c371.jpg)

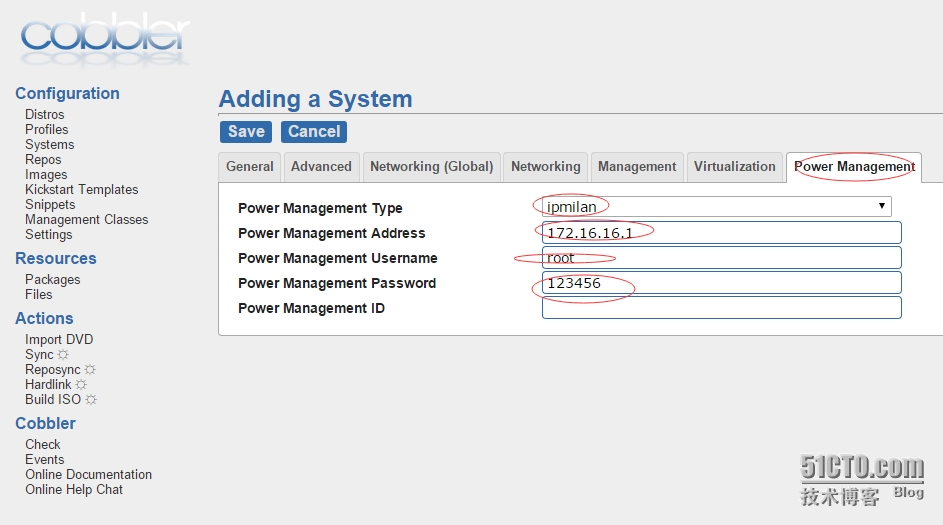
3、编辑Networking(Global)标签，添加安装主机的hostname,添加DNS服务器的地址，（网关不在全局标签内添加）

[](https://s3.51cto.com/wyfs02/M00/6F/5F/wKioL1WaapCQokPGAAJcAwjb8Tw218.jpg)

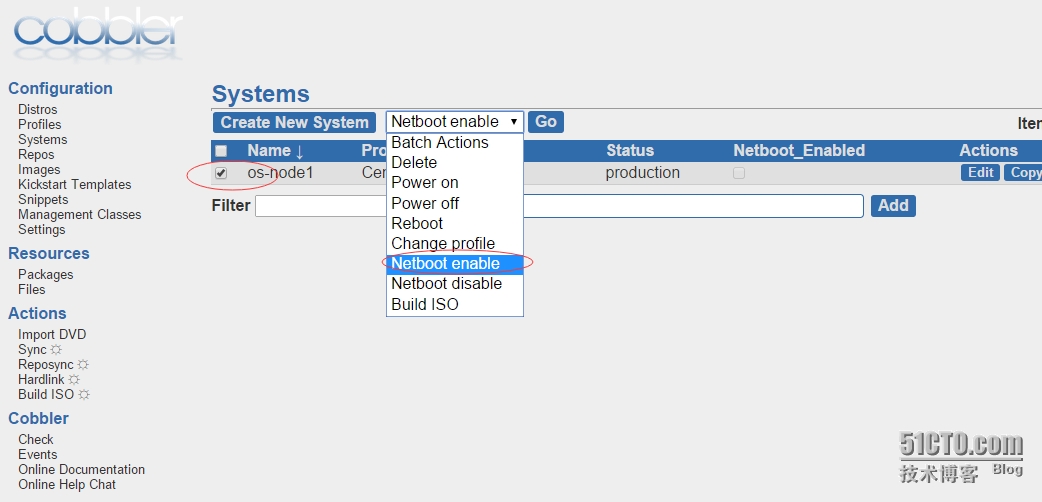
4、编辑Networking标签，添加接口eth0(此接口命名在centos7里面有变化，不过系统接口生成后会自动抓取系统识别到的名字)，添加接口的mac地址、添加IP、选定接口类型、配置子网掩码及接口网关。

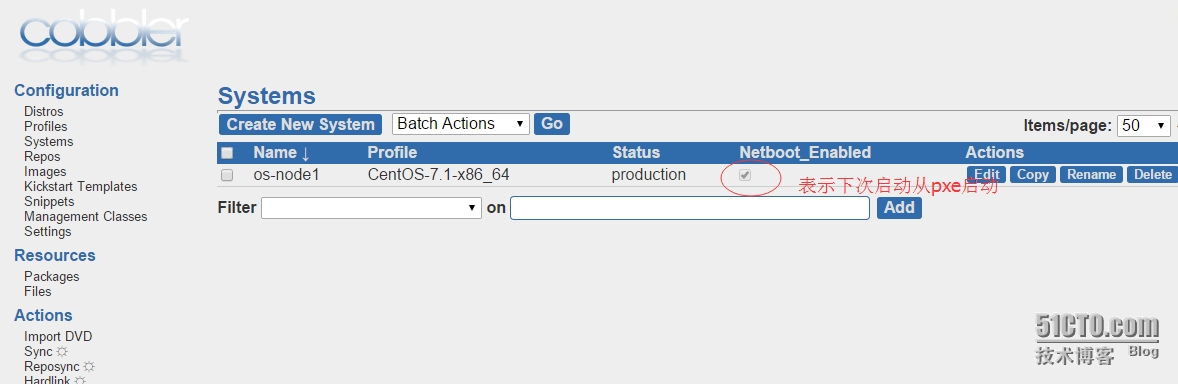
[](https://s3.51cto.com/wyfs02/M00/6F/61/wKiom1WaaMPSEZ6bAANZKIc91w0196.jpg)

5、编辑power management标签，这个可以关联ipmi、ilo等带外管理接口，通过输入ip、用户名及密码可以通过cobbler对服务器进行重启等操作，方便装系统。

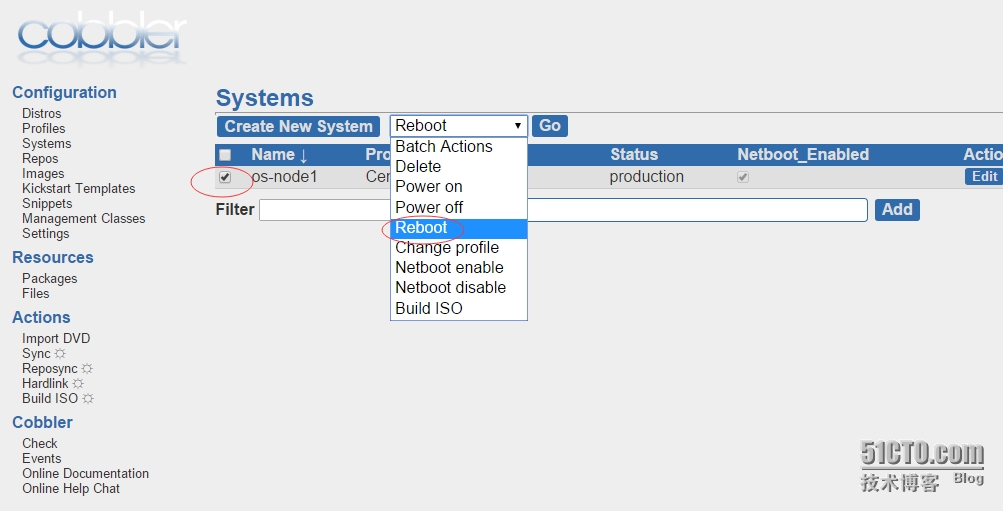
[](https://s3.51cto.com/wyfs02/M01/6F/5F/wKioL1WaapDAOOKKAAJ0BtqbPO0096.jpg)

6、重装系统

[](https://s3.51cto.com/wyfs02/M01/6F/62/wKiom1WabRmTe4dRAAJWiZsZ-SE441.jpg)

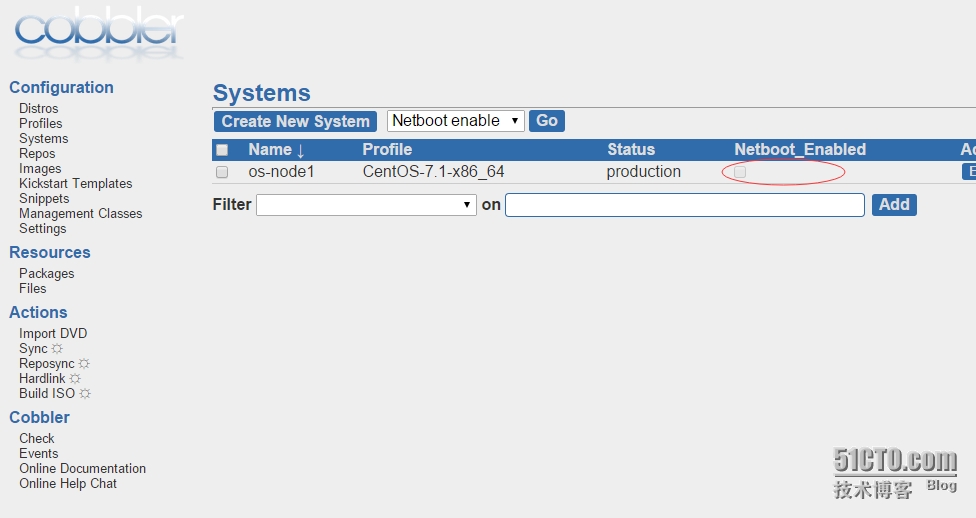
[](https://s3.51cto.com/wyfs02/M02/6F/61/wKiom1Waa-6ClyUdAAIHBWFUbIE455.jpg)

上图表示：如果服务器设置的是PXE启动，此次再重装系统就会按照服务器的启动方式从PXE启动重新安装系统

[](https://s3.51cto.com/wyfs02/M00/6F/5F/wKioL1WabbrgdPkKAAJC7hEIpPY548.jpg)

如果此系统服务器有impi或ilo等带外管理接口，可以通过cobbler对服务器进行重启来重装系统。

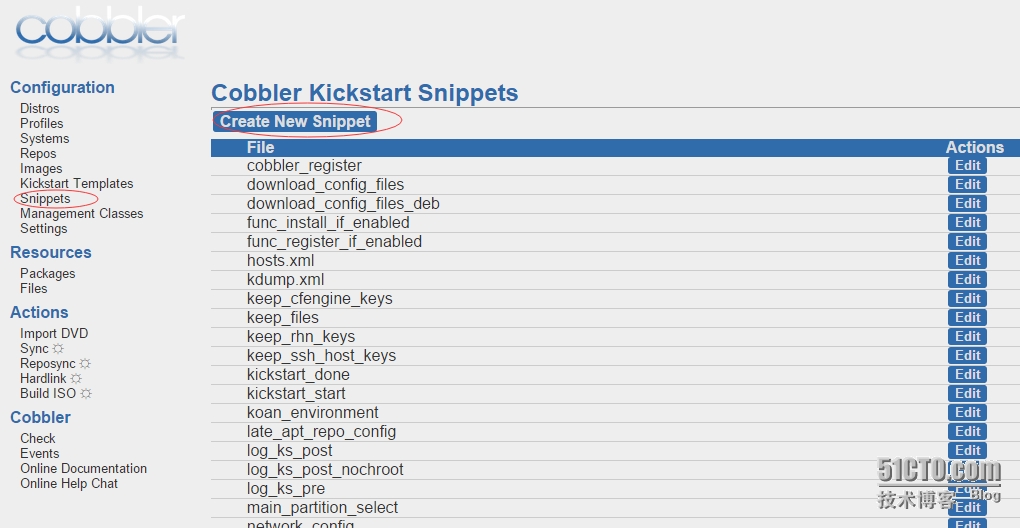
7、系统重装完后（netboot enabled 标签处的对勾消失，表示重装成功，下次服务器再启动就会跳过pxe启动）

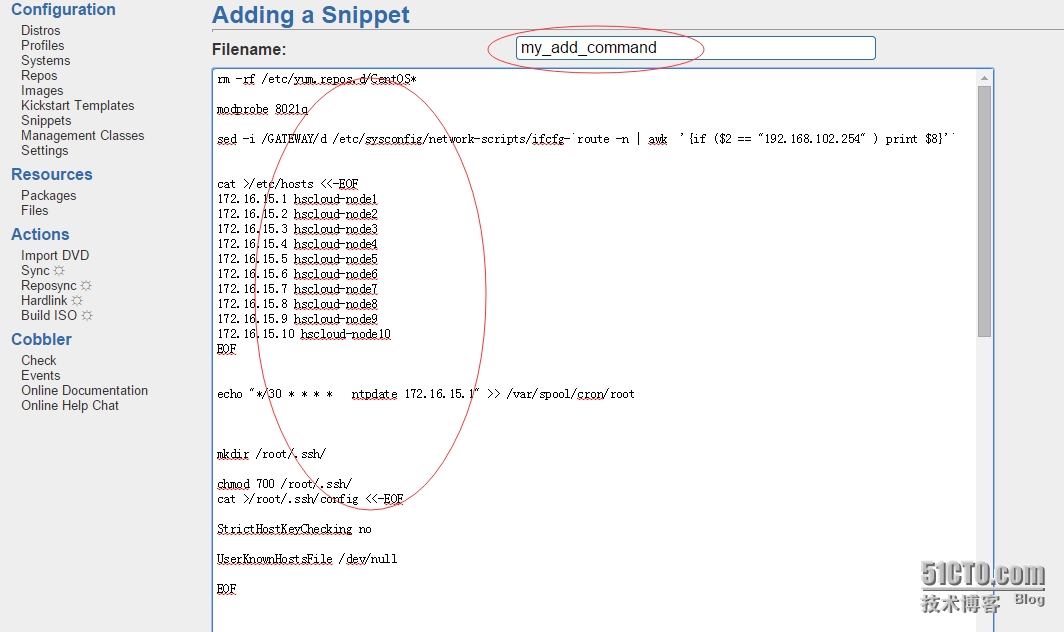
[](https://s3.51cto.com/wyfs02/M02/6F/62/wKiom1WabdLTXqonAAHt8OdldKc956.jpg)

# Cobbler Web界面操作（五）

编辑snippet

1、添加自动意的snippet文件

[](https://s3.51cto.com/wyfs02/M01/6F/5F/wKioL1WacnWhqEulAAN6DnFI6zU473.jpg)

[](https://s3.51cto.com/wyfs02/M01/6F/62/wKiom1WacKmji2lMAANaVoZv6Eg251.jpg)

2、把下面的三个自定义的snippet文件添加上

1）、my\_add\_command

rm -rf /etc/yum.repos.d/CentOS\*

modprobe 8021q

sed -i /GATEWAY/d /etc/sysconfig/network-scripts/ifcfg-`route -n | awk  '{if ($2 == "192.168.102.254" ) print $8}'`

cat >/etc/hosts <<-EOF

172.16.15.1 hscloud-node1

172.16.15.2 hscloud-node2

172.16.15.3 hscloud-node3

172.16.15.4 hscloud-node4

172.16.15.5 hscloud-node5

172.16.15.6 hscloud-node6

172.16.15.7 hscloud-node7

172.16.15.8 hscloud-node8

172.16.15.9 hscloud-node9

172.16.15.10 hscloud-node10

EOF

echo "\*/30 \* \* \* \*   ntpdate 172.16.15.1" >> /var/spool/cron/root

mkdir /root/.ssh/

chmod 700 /root/.ssh/

cat >/root/.ssh/config <<-EOF

StrictHostKeyChecking no

UserKnownHostsFile /dev/null

EOF

cat >/root/.ssh/id\_rsa <<-EOF

-----BEGIN RSA PRIVATE KEY-----

MIIEoQIBAAKCAQEA159xWLESQKxo7xm2iN5Gs//YXvOX6H4pcZJnCgc53vT4I4Di

h3SdnS7g33WezkzNjasuzuiflWB7XkPnCHWMP2LpshY7MAjMtgfv3t4Lo7W4zlhh

IgrynJbzPCFMgLQhCAzOzgOPgV/szPA0ydka82Mo9eScUnzLOuVFYEjCKe2g/TVL

mmfIgMtE4bI3G6df8BNH+9g/L5id/fWT+FUMjVX0gMWgd3Q9xi984WJaQKQu+y60

tt1s38P/5xA1+ly7jN8MgJeHTRt//Xa61g7n5qWd1z29HJObmT2KbvuWyZXxYJLd

KyWcfIRxPY7hursIBHtEmvGnaau/gD4XLe7QlwIBIwKCAQAYpIH7gfN1GwStnInj

waGuK942vMDnXuAqPJ4Pxk/BtZizmbN9MeYgl6Sr0u2T64U0wxtK2MHH7cT01I9u

rlkr0MpAPQ4UHkNIAOg2ukp5HBUe5YdxmtnmAp90lhdfKoduomDHFlmLIOfc6ECM

GM/hTSlArGmjBvKnpSyACFC0VXZJ4ZM1t/hQ4Ozsu23YFifuTC6hm3nvM+hwMSfV

Wu9kllgU2PeBOvUtmORcs9p4Drfs+N9AaOA72iRad7t6kyeXU0o9TUPzwcwCICtO

h+ez9aGmB2DZOB2FxHL3gpitVa065qGvC830u8rt68vA8VTwqhN7WjqZCBOLka++

0kmLAoGBAO8c5J74jmzKv79oUgcmYf6jfyA2cwKrbZMaXgs8ouufkKZRLlzqxF5f

ZXd1YzM/xiVFB5EmEUGC1V1MTr0gewfH3g52yIHJVO2QHw5OJ1QKrNLHSF1AT5u3

0VUUgn5AZstFy/OS66qTzNFZZwUaBBwcO188+U1XUBt7dFSmUnfXAoGBAObZ2uSL

z57yW2xh+C7Sg4PneY+/VNGXJNkN6Www1nI9oct5LM3h6/4X0DdA2W9mXUBcpYId

NoFROeSDKdBpWhlY6DRxavLu5tlavjGYwmh1U4SjNNBvmP2O9UdkwmCcgXpXZOLc

kqfLTHBCWKXId0Sub7rmotlXVPj/jwDB7FVBAoGBAIHN2zG6IW5CLZPK6rNsm5jc

aZUk31k4fU/bHRwK+VtWnvszYk+ypR1Jur0xGJgp7zjNs6aRAg2ey5kMKr5wt9EG

GXWQ8ID4RAvvJs1AXn4UbHJsLpkFp48MA+xbl0vZ0WcIoeqgNspQO/yePz1IoyUz

5bdbnUc+BupY9gIRJXRBAoGAO1yXX1cmwnjVrifSGqspIesQoU505XACKS92iYjm

r6lyv0sLhWX62vAuOhf9ZcnddvNAgIsrRdMO4v0oAmQ7vWABMg6JNyd9MJOtP/QU

u8Zl7ugjhg4RZcWsyTcqqyDuGCUZ8TFnijuXUBEPexZKjgD4KL72rOqSMWZJV/da

B0sCgYAwQu2sW2ohR5ZCtI5WZTmSIi5KcBtGCHM8UIMQhl0TgQni83LfkBaUcHAr

Wi6lVvEkFDKpjkS1GICA1JT7vQWDKJY211Ut5snkV6wGKb5KkVczzRjmvCv77rH9

b80ZLdnIvTS8clQIQE/0DVwmEQ5LNf9vdIMKr6gZ2CEMySyzsA==

-----END RSA PRIVATE KEY-----

EOF

cat >>/root/.ssh/authorized\_keys <<-EOF

ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEA159xWLESQKxo7xm2iN5Gs//YXvOX6H4pcZJnCgc53vT4I4Dih3SdnS7g33WezkzNjasuzuiflWB7XkPnCHWMP2LpshY7MAjMtgfv3t4Lo7W4zlhhIgrynJbzPCFMgLQhCAzOzgOPgV/szPA0ydka82Mo9eScUnzLOuVFYEjCKe2g/TVLmmfIgMtE4bI3G6df8BNH+9g/L5id/fWT+FUMjVX0gMWgd3Q9xi984WJaQKQu+y60tt1s38P/5xA1+ly7jN8MgJeHTRt//Xa61g7n5qWd1z29HJObmT2KbvuWyZXxYJLdKyWcfIRxPY7hursIBHtEmvGnaau/gD4XLe7Qlw== root@bgw-os-node152

EOF

chmod 600 /root/.ssh/id\_rsa

chmod 600 /root/.ssh/authorized\_keys

useradd nova -U  -d /var/lib/nova

mkdir -p /var/lib/nova/.ssh/

chmod 700 /var/lib/nova/.ssh/

cat >/var/lib/nova/.ssh/config <<-EOF

StrictHostKeyChecking no

UserKnownHostsFile /dev/null

EOF

cat > /var/lib/nova/.ssh/id\_rsa <<-EOF

-----BEGIN RSA PRIVATE KEY-----

MIIEoQIBAAKCAQEA159xWLESQKxo7xm2iN5Gs//YXvOX6H4pcZJnCgc53vT4I4Di

h3SdnS7g33WezkzNjasuzuiflWB7XkPnCHWMP2LpshY7MAjMtgfv3t4Lo7W4zlhh

IgrynJbzPCFMgLQhCAzOzgOPgV/szPA0ydka82Mo9eScUnzLOuVFYEjCKe2g/TVL

mmfIgMtE4bI3G6df8BNH+9g/L5id/fWT+FUMjVX0gMWgd3Q9xi984WJaQKQu+y60

tt1s38P/5xA1+ly7jN8MgJeHTRt//Xa61g7n5qWd1z29HJObmT2KbvuWyZXxYJLd

KyWcfIRxPY7hursIBHtEmvGnaau/gD4XLe7QlwIBIwKCAQAYpIH7gfN1GwStnInj

waGuK942vMDnXuAqPJ4Pxk/BtZizmbN9MeYgl6Sr0u2T64U0wxtK2MHH7cT01I9u

rlkr0MpAPQ4UHkNIAOg2ukp5HBUe5YdxmtnmAp90lhdfKoduomDHFlmLIOfc6ECM

GM/hTSlArGmjBvKnpSyACFC0VXZJ4ZM1t/hQ4Ozsu23YFifuTC6hm3nvM+hwMSfV

Wu9kllgU2PeBOvUtmORcs9p4Drfs+N9AaOA72iRad7t6kyeXU0o9TUPzwcwCICtO

h+ez9aGmB2DZOB2FxHL3gpitVa065qGvC830u8rt68vA8VTwqhN7WjqZCBOLka++

0kmLAoGBAO8c5J74jmzKv79oUgcmYf6jfyA2cwKrbZMaXgs8ouufkKZRLlzqxF5f

ZXd1YzM/xiVFB5EmEUGC1V1MTr0gewfH3g52yIHJVO2QHw5OJ1QKrNLHSF1AT5u3

0VUUgn5AZstFy/OS66qTzNFZZwUaBBwcO188+U1XUBt7dFSmUnfXAoGBAObZ2uSL

z57yW2xh+C7Sg4PneY+/VNGXJNkN6Www1nI9oct5LM3h6/4X0DdA2W9mXUBcpYId

NoFROeSDKdBpWhlY6DRxavLu5tlavjGYwmh1U4SjNNBvmP2O9UdkwmCcgXpXZOLc

kqfLTHBCWKXId0Sub7rmotlXVPj/jwDB7FVBAoGBAIHN2zG6IW5CLZPK6rNsm5jc

aZUk31k4fU/bHRwK+VtWnvszYk+ypR1Jur0xGJgp7zjNs6aRAg2ey5kMKr5wt9EG

GXWQ8ID4RAvvJs1AXn4UbHJsLpkFp48MA+xbl0vZ0WcIoeqgNspQO/yePz1IoyUz

5bdbnUc+BupY9gIRJXRBAoGAO1yXX1cmwnjVrifSGqspIesQoU505XACKS92iYjm

r6lyv0sLhWX62vAuOhf9ZcnddvNAgIsrRdMO4v0oAmQ7vWABMg6JNyd9MJOtP/QU

u8Zl7ugjhg4RZcWsyTcqqyDuGCUZ8TFnijuXUBEPexZKjgD4KL72rOqSMWZJV/da

B0sCgYAwQu2sW2ohR5ZCtI5WZTmSIi5KcBtGCHM8UIMQhl0TgQni83LfkBaUcHAr

Wi6lVvEkFDKpjkS1GICA1JT7vQWDKJY211Ut5snkV6wGKb5KkVczzRjmvCv77rH9

b80ZLdnIvTS8clQIQE/0DVwmEQ5LNf9vdIMKr6gZ2CEMySyzsA==

-----END RSA PRIVATE KEY-----

EOF

cat >>/var/lib/nova/.ssh/authorized\_keys <<-EOF

ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEAuWcKohvVRpFCGUAONrYYzFBWr5xZj0BZvWJ7JKf9Y7kEW7yX0A41j4XgV+xjajjye8UxI4eXaW2fgCH6fZe5WKrztlLEZzA6XdD5HnAmNSpys3311ChBYjPbPUvZLdRP2eq/Jc8BRFXQ5C7qgHxunwLP1Q4I+wf7/oaRUIhhoudtK9Dm1EFgWwO+FuTwLqPqV80+b/StVhEsjT9+10G4+I1UAQlOFMHIAWJf15H9rfMYelp979430tQnlzccKtTIkaq2jXeF8TNQosH9c3oL+r4PswL3xMTVbWAsFSmby5YEj5R+0d3Ah7yXDwFEcLhlcbh0cctncCG0HDXk+sAo6Q== nova@bgw-os-node151

EOF

chmod 600 /var/lib/nova/.ssh/id\_rsa

chmod 600 /var/lib/nova/.ssh/authorized\_keys

chmod 600 /var/lib/nova/.ssh/config

chown nova:nova /var/lib/nova -R

2）、my\_root\_ssh\_key

cd /root

mkdir --mode=700 .ssh

cat >> .ssh/authorized\_keys << "PUBLIC\_KEY"

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAAAgQCwww24e5eBdcNAFIYL/4LJ54eH5XQr1GmpuRB4m7NAcsC7PTt2BDob2P5bt+Rg1o64Aah2m1ajt3kIewEeVCKIC6Msocu4W2ERD893aVgo72Dn9+ogPpg7w/GA4GKyXg2DS4mCNPN8pLQa1wxrozyIFqyyNYDFzWE0mSJdwaWzqQ==

chmod 600 .ssh/authorized\_keys

3)、my\_pre\_install\_network\_config

#if $getVar("system\_name","") != ""

# Start pre\_install\_network\_config generated code

#raw

# generic functions to be used later for discovering NICs

mac\_exists() {

  [ -z "$1" ] && return 1

  if which ip 2>/dev/null >/dev/null; then

    ip -o link | grep -i "$1" 2>/dev/null >/dev/null

    return $?

  elif which esxcfg-nics 2>/dev/null >/dev/null; then

    esxcfg-nics -l | grep -i "$1" 2>/dev/null >/dev/null

    return $?

  else

    ifconfig -a | grep -i "$1" 2>/dev/null >/dev/null

    return $?

  fi

}

get\_ifname() {

  if which ip 2>/dev/null >/dev/null; then

    IFNAME=$(ip -o link | grep -i "$1" | sed -e 's/^[0-9]\*: //' -e 's/:.\*//')

  elif which esxcfg-nics 2>/dev/null >/dev/null; then

    IFNAME=$(esxcfg-nics -l | grep -i "$1" | cut -d " " -f 1)

  else

    IFNAME=$(ifconfig -a | grep -i "$1" | cut -d " " -f 1)

    if [ -z $IFNAME ]; then

      IFNAME=$(ifconfig -a | grep -i -B 2 "$1" | sed -n '/flags/s/:.\*$//p')

    fi

  fi

}

#end raw

    #set ikeys = $interfaces.keys()

    #import re

    #set $vlanpattern = $re.compile("[a-zA-Z0-9]+[\.][0-9]+")

    #set $routepattern = $re.compile("[0-9/.]+:[0-9.]+")

    ##

    ## Determine if we should use the MAC address to configure the interfaces first

    ## Only physical interfaces are required to have a MAC address

    #set $configbymac = True

    #for $iname in $ikeys

        #set $idata = $interfaces[$iname]

        #if $idata["mac\_address"] == "" and not $vlanpattern.match($iname) and not $idata["interface\_type"].lower() in ("master","bond","bridge","bonded\_bridge\_slave")

            #set $configbymac = False

        #end if

    #end for

    #set $i = 0

    #if $configbymac

        ## Output diagnostic message

# Start of code to match cobbler system interfaces to physical interfaces by their mac addresses

    #end if

    #for $iname in $ikeys

#  Start $iname

        #set $idata         = $interfaces[$iname]

        #set $mac           = $idata["mac\_address"]

        #set $static        = $idata["static"]

        #set $ip            = $idata["ip\_address"]

        #set $netmask       = $idata["netmask"]

        #set $iface\_type    = $idata["interface\_type"]

        #set $iface\_master  = $idata["interface\_master"]

        #set $if\_gateway    = $idata["if\_gateway"]

        #set $static\_routes = $idata["static\_routes"]

        #set $devfile       = "/etc/sysconfig/network-scripts/ifcfg-" + $iname

        #if $vlanpattern.match($iname)

            ## If this is a VLAN interface, skip it, anaconda doesn't know

            ## about VLANs.

            #set $is\_vlan = "true"

        #else

            #set $is\_vlan = "false"

        #end if

        #if ($configbymac and $is\_vlan == "false" and $iface\_type.lower() not in ("slave","bond\_slave","bridge\_slave","bonded\_bridge\_slave")) or $iface\_type.lower() in ("master","bond","bridge")

            ## This is a physical interface, hand it to anaconda. Do not

            ## process slave interface here.

            #if $iface\_type.lower() in ("master","bond","bridge","bonded\_bridge\_slave")

                ## Find a slave for this interface

                #for $tiname in $ikeys

                    #set $tidata = $interfaces[$tiname]

                    #if $tidata["interface\_type"].lower() in ("slave","bond\_slave","bridge\_slave") and $tidata["interface\_master"].lower() == $iname

                        #if $tidata["mac\_address"] != '':

                            #set $mac = $tidata["mac\_address"]

#  Found a slave for this interface: $tiname ($mac)

                            #break

                        #end if

                    #else if $tidata["interface\_type"].lower() == "bonded\_bridge\_slave" and $tidata["interface\_master"].lower() == $iname

                        ## find a slave for this slave interface...

                        #for $stiname in $ikeys

                            #set $stidata = $interfaces[$stiname]

                            #if $stidata["interface\_type"].lower() in ("slave","bond\_slave","bridge\_slave") and $stidata["interface\_master"].lower() == $tiname

                                #if $stidata["mac\_address"] != '':

                                    #set $mac = $stidata["mac\_address"]

#  Found a slave for this interface: $tiname -> $stiname ($mac)

                                    #break

                                #end if

                            #end if

                        #end for

                    #end if

                #end for

            #end if

            #if $static and $ip != ""

                #if $netmask == ""

                    ## Netmask not provided, default to /24.

                    #set $netmask = "255.255.255.0"

                #end if

                #set $netinfo = "--bootproto=static --ip=%s --netmask=%s" % ($ip, $netmask)

                #if $if\_gateway != ""

            #set $netinfo = "%s --gateway=%s" % ($netinfo, $if\_gateway)

                #else

                    #set $netinfo = "%s --gateway=0.0.0.0" % ($netinfo)

                #end if

             #if $len($name\_servers) > 0

                 #set $netinfo = "%s --nameserver=%s" % ($netinfo, $name\_servers[0])

                #end if

            #else if not $static

                #set $netinfo = "--bootproto=dhcp"

            #else

                ## Skip this interface, it's set as static, but without

                ## networking info.

#  Skipping (no configuration)...

                #continue

            #end if

            #if $hostname != ""

                #set $netinfo = "%s --hostname=%s" % ($netinfo, $hostname)

            #end if

# Configuring $iname ($mac)

if mac\_exists $mac

then

  get\_ifname $mac

  echo "network --device=\$IFNAME $netinfo " >> /tmp/pre\_install\_network\_config

            #for $route in $static\_routes

                #if $routepattern.match($route)

                    #set $routebits = $route.split(":")

                    #set [$network, $router] = $route.split(":")

  ip route add $network via $router dev \$IFNAME

                #else

  # Warning: invalid route "$route"

                #end if

            #end for

fi

        #else

            #if $iface\_type.lower() in ("slave","bond\_slave","bridge\_slave","bonded\_bridge\_slave")

#  Skipping (slave-interface)

            #else

#  Skipping (not a physical interface)...

            #end if

        #end if

    #end for

# End pre\_install\_network\_config generated code

#end if

# cobbler initrd.img引导镜像中添加HP DL388 Gen8 B320i阵列卡驱动

1、利用cobbler安装HP DL388 Gen8服务器时不能成功，报错提示找不到可用的硬盘，和HP售后沟通了解到HP DL388 Gen8服务器的raid卡的型号是B320i。centos6或7系统的initrd.img中没有B320i阵列卡的驱动，如果想把Gen8装上centos或redhat的系统，需要先在官网下载B320i阵列卡的驱动，CentOS7的下载地址如下：

阵列卡驱动下载

[http://h20564.www2.hp.com/hpsc/swd/public/detail?sp4ts.oid=5293150&swItemId=MTX\_9200a10168684afbbb4efce88a&swEnvOid=4176](http://h20564.www2.hp.com/hpsc/swd/public/detail?sp4ts.oid=5293150&swItemId=MTX_9200a10168684afbbb4efce88a&swEnvOid=4176" \t "https://blog.51cto.com/zhanguo1110/_blank)

2、下载完B320i的阵列卡驱动后，需要把阵列卡打包到initrd.img引导镜像中，具体步骤如下：

1）、把iso镜像挂在到一个临时目录

mkdir /mnt/cdrom/CentOS-7-x86\_64/

mount -o loop /iso/CentOS-7-x86\_64-DVD-1503-01.iso /mnt/cdrom/CentOS-7-x86\_64/

2)、因为mount到的临时目录没有写权限所以需要把这些文件复制到另外的目录

[root@localhost ~]# mkdir /mnt/CentOS-7-x86\_64

[root@localhost ~]# cd /mnt/CentOS-7-x86\_64/

[root@localhost CentOS-7-x86\_64]# cp -r /mnt/cdrom/CentOS-7-x86\_64/\* .

3）、找到initrd.img引导文件所在的目录

/mnt/CentOS-7-x86\_64/isolinux/initrd.img

/mnt/CentOS-7-x86\_64/p\_w\_picpaths/pxeboot/initrd.img

4）、在/mnt/CentOS-7-x86\_64/isolinux/目录创建一个initrd的临时目录

mkdir initrd

3)、复制initrd.img镜像到initrd临时目录内

cp initrd.img initrd

4)、解压initrd临时目录中的initrd.img镜像（用file initrd.img 查看此文件的格式）

file initrd.img

initrd.img: xz compressed data

通过查看该文件时XZ格式，用xz 工具解压该文件

xz -dc initrd.img | cpio -id

查看解压完后的目录内容

[root@localhost initrd]# ls

bin  dev  etc  init  lib  lib64  proc  root  run  sbin  shutdown  sys  sysroot  tmp  usr  var

[root@localhost initrd]#

5）、查看下载的驱动的格式

[root@localhost ~]# file hpvsa-1.2.14-100.rhel7u1.x86\_64.dd.gz

hpvsa-1.2.14-100.rhel7u1.x86\_64.dd.gz: Linux rev 1.0 ext2 filesystem data (mounted or unclean) (extents) (64bit) (huge files)

6)、解压上面的驱动文件

mount -o loop hpvsa-1.2.14-100.rhel7u1.x86\_64.dd.gz  /bbb

7）、解压kmod-hpvsa-1.2.14-100.rhel7u1.x86\_64.rpm文件

rpm2cpio kmod-hpvsa-1.2.14-100.rhel7u1.x86\_64.rpm | cpio -div

8）、复制驱动文件到initrd.img解压后的module目录内

cp -r /root/bbb/lib/modules/3.10.0-229.el7.x86\_64/extra/ /mnt/CentOS-7-x86\_64/isolinux/initrd/usr/lib/modules/3.10.0-229.el7.x86\_64/kernel

9）、获得添加模块的模块别名

modinfo -F alias kernel/extra/hpvsa/hpvsa.ko  | sed -e 's/^/alias /' -e 's/$/ hpvsa/' >>modules.alias

10）、回到initrd目录重新压缩initrd目录生成initrd.img文件

find . | cpio -c -o | xz -9 --format=lzma > initrd.img

11）、删除原来的initrd.img引导文件把新的initrd.img文件复制到/mnt/CentOS-7-x86\_64/isolinux/和/mnt/CentOS-7-x86\_64/p\_w\_picpaths/pxeboot/目录