

NSD SERVICES DAY06

1. [案例1：rsync基本用法](#)
2. [案例2：rsync+SSH同步](#)
3. [案例3：使用inotifywait工具](#)
4. [案例4：配置Web镜像同步](#)
5. [案例5：配置Cobbler装机平台](#)

1 案例1：rsync基本用法

1.1 问题

本例要求掌握远程同步的基本操作，使用rsync命令完成下列任务：

1. 将目录 /boot 同步到目录 /todir 下
2. 将目录 /boot 下的文档同步到目录 /todir 下
3. 在目录 /boot 下新增文件 a.txt，删除 /todir 下的子目录 grub2，再次同步使 /todir 与 /boot 一致
4. 验证 -a、-n、-v、--delete 选项的含义

1.2 方案

本地同步操作：

- rsync [选项...] 本地目录1 本地目录2
- rsync [选项...] 本地目录1/ 本地目录2

rsync同步工具的常用选项：

- -n：测试同步过程，不做实际修改
- --delete：删除目标文件夹内多余的文档
- -a：归档模式，相当于-rlptgoD
- -v：显示详细操作信息
- -z：传输过程中启用压缩/解压

1.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：rsync同步基本操作

1) 将目录 /boot 同步到目录 /todir 下

```
01. [root@svr7 ~]# ls -l /todir //同步前
02. ls: 无法访问/todir: 没有那个文件或目录
03. [root@svr7 ~]# rsync -a /boot /todir //将目录1作为目录2的子目录
04. [root@svr7 ~]# ls -l /todir //检查同步结果
05. 总用量 4
06. dr-xr-xr-x. 4 root root 4096 11月 30 18:50 boot
```

[Top](#)

2) 将目录 /boot 下的文档同步到目录 /todir 下

```

01. [root@svr7 ~]# rm -rf /todir //清理掉目录2
02. [root@svr7 ~]# rsync -a /boot/ /todir //将目录1下的文档同步到目录2下
03. [root@svr7 ~]# ls -l /todir //检查同步结果
04. 总用量 126708
05. -rw-r--r--. 1 root root 126426 10月 30 2015 config-3.10.0-327.el7.x86_64
06. drwxr-xr-x. 2 root root 4096 11月 30 18:50 extlinux
07. drwx-----. 6 root root 104 12月 9 09:58 grub2
08. ...

```

3) 同步效果测试

在目录/boot下新增文件a.txt，删除/todir下的子目录 grub2：

```

01. [root@svr7 ~]# touch /boot/a.txt
02. [root@svr7 ~]# rm -rf /todir/grub2/

```

现在目录/boot和/todir目录下的内容已经不一致了：

```

01. [root@svr7 ~]# ls -ld /boot/a.txt /todir/a.txt
02. ls: 无法访问/todir/a.txt: 没有那个文件或目录
03. -rw-r--r--. 1 root root 0 1月 11 21:09 /boot/a.txt
04. [root@svr7 ~]# ls -ld /boot/grub2 /todir/grub2
05. ls: 无法访问/todir/grub2: 没有那个文件或目录
06. drwx-----. 6 root root 104 12月 9 09:58 /boot/grub2

```

再次同步使/todir与/boot一致：

```

01. [root@svr7 ~]# rsync -a /boot/ /todir/

```

确认同步结果：

```

01. [root@svr7 ~]# ls -ld /boot/a.txt /todir/a.txt
02. -rw-r--r--. 1 root root 0 1月 11 21:09 /boot/a.txt
03. -rw-r--r--. 1 root root 0 1月 11 21:09 /todir/a.txt
04. [root@svr7 ~]# ls -ld /boot/grub2 /todir/grub2
05. drwx-----. 6 root root 104 12月 9 09:58 /boot/grub2

```

[Top](#)

06. drwx-----x. 6 root root 104 12月 9 09:58 /todir/grub2

步骤二：验证 -a、-v、-n、--delete 选项的含义

1) 验证-a选项

当目录1包含文件夹时，若缺少-a或-r选项则文件夹会被忽略：

```
01. [root@svr7 ~]# rsync /home /testa
02. skipping directory home
03. [root@svr7 ~]# ls -ld /testa
04. ls: 无法访问/testa: 没有那个文件或目录
```

添加-a后才会执行同步：

```
01. [root@svr7 ~]# rsync -a /home/ /testa
02. [root@svr7 ~]# ls -ld /testa
03. drwxr-xr-x. 4 root root 31 1月 6 17:33 /testa
```

类似的情况，当目录1中的数据出现权限、归属、修改时间等变化时，若文件内容不变默认不会同步，若希望目录2也同步这些变化，也需要-a选项。

2) 验证-v选项

创建测试目录及文档：

```
01. [root@svr7 ~]# mkdir /fdir
02. [root@svr7 ~]# touch /fdir/1.txt
```

添加-v选项时，可以看到操作细节信息，比如第一次同步时：

```
01. [root@svr7 ~]# rsync -av /fdir/ /tdir
02. sending incremental file list
03. created directory /tdir
04. ./
05. 1.txt //传输文档列表
06.
07. sent 82 bytes received 34 bytes 232.00 bytes/sec
08. total size is 0 speedup is 0.00
```

[Top](#)

在目录/fdir/添加文件2.txt，再次跟踪同步信息：

```
01. [root@svr7 ~]# touch /fdir/2.txt
02. sending incremental file list
03. ./
04. 2.txt //传输文档列表
05.
06. sent 100 bytes received 34 bytes 268.00 bytes/sec
07. total size is 0 speedup is 0.00
```

确认目录1和目录2的内容已经一致：

```
01. [root@svr7 ~]# ls /fdir/ /tdir/
02. /fdir/:
03. 1.txt 2.txt
04.
05. /tdir/:
06. 1.txt 2.txt
```

再次跟踪同步信息，已经无需传输文件：

```
01. [root@svr7 ~]# rsync -av /fdir/ /tdir/
02. sending incremental file list
03.
04. sent 58 bytes received 12 bytes 140.00 bytes/sec
05. total size is 0 speedup is 0.00
```

3) 验证-n选项

将-n、-v选项合用，可以模拟同步过程，显示需要做哪些操作（但并不真的同步）。

在目录/fdir下新建文件3.txt，测试同步操作：

```
01. [root@svr7 ~]# touch /fdir/3.txt
02. [root@svr7 ~]# rsync -avn /fdir/ /tdir/
03. sending incremental file list
04. ./
05. 3.txt //提示同步时会传输哪些文件
06.
```

[Top](#)

07. sent 78 bytes received 18 bytes 192.00 bytes/sec
08. total size is 0 speedup is 0.00 (DRY RUN)
09. [root@svr7 ~]# ls -l /tdir/3.txt //但实际并未真的同步
10. ls: 无法访问/tdir/3.txt: 没有那个文件或目录

去掉-n选项才会真正同步：

01. [root@svr7 ~]# rsync -av /fdir/ /tdir/
02. sending incremental file list
03. ./
04. 3.txt
- 05.
06. sent 114 bytes received 34 bytes 296.00 bytes/sec
07. total size is 0 speedup is 0.00
08. [root@svr7 ~]# ls -l /tdir/3.txt
09. -rw-r--r--. 1 root root 0 1月 11 21:46 /tdir/3.txt

4) 验证--delete选项

rsync同步操作默认只是将目录1的数据同步到目录2，但如果目录2存在多余的文件却并不会去除，除非添加—delete选项。

在目录/fdir、/tdir已经完成同步后，删除/tdir/2.txt文件，再次同步：

01. [root@svr7 ~]# rm -rf /fdir/2.txt
02. [root@svr7 ~]# rsync -a /fdir/ /tdir/

检查发现目标文件夹/tdir下的2.txt文件还在：

01. [root@svr7 ~]# ls /fdir/ /tdir/
02. /fdir/:
03. 1.txt 3.txt
- 04.
05. /tdir/:
06. 1.txt 2.txt 3.txt

这种情况下添加--delete选项再次执行同步，两个目录的内容就一致了：

[Top](#)

```

01. [root@svr7 ~]# rsync -a --delete /fdir/ /tdir/
02. [root@svr7 ~]# ls /fdir/ /tdir/
03. /fdir/:
04. 1.txt 3.txt
05.
06. /tdir/:
07. 1.txt 3.txt

```

2 案例2：rsync+SSH同步

2.1 问题

本例要求掌握rsync与远程SSH资源的同步操作，使用rsync命令访问远程主机svr7，完成下列任务：

1. 查看远程主机的 / 目录下有哪些子目录
2. 从远程主机下载 /etc/passwd 文件到当前目录
3. 将远程主机的 /boot/ 目录同步为本地的 /fromssh
4. 将本机的 /etc 目录同步到远程主机的 /opt/下

2.2 方案

列出 SSH 服务端资源

- rsync user@host:远程目录/

rsync+SSH远程同步操作：

- rsync [...] user@host:远程目录 本地目录
- rsync [...] 本地目录 user@host:远程目录

2.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：列出远程主机的SSH资源

查看远程主机svr7的/目录下有哪些子目录：

```

01. [root@pc207 ~]# rsync root@192.168.4.7:/
02. root@192.168.4.7's password: //验证对方的密码
03. dr-xr-xr-x  4096 2016/12/15 10:39:34 .
04. lrwxrwxrwx    7 2016/12/07 09:21:50 bin
05. lrwxrwxrwx    7 2016/12/07 09:21:50 lib
06. lrwxrwxrwx    9 2016/12/07 09:21:50 lib64
07. lrwxrwxrwx    8 2016/12/07 09:21:50 sbin
08. dr-xr-xr-x  4096 2016/12/07 11:25:29 boot
09. drwxr-xr-x    6 2016/12/07 09:21:14 data

```

[Top](#)

- | | | | | |
|-----|------------|------|---------------------|-----|
| 10. | drwxr-xr-x | 3200 | 2016/12/15 10:46:15 | dev |
| 11. | drwxr-xr-x | 8192 | 2016/12/20 17:01:02 | etc |

步骤二：rsync+SSH同步操作

1) 从远程主机svr7下载/etc/passwd文件到当前目录

- ```

01. [root@pc207 ~]# rsync root@192.168.4.7:/etc/passwd ./
02. root@192.168.4.7's password: //验证对方的密码
03. [root@pc207 ~]# cat passwd //检查同步结果
04. root:x:0:0:root:/root:/bin/bash
05. bin:x:1:1:bin:/bin:/sbin/nologin
06. daemon:x:2:2:daemon:/sbin:/sbin/nologin
07. adm:x:3:4:adm:/var/adm:/sbin/nologin
08. lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
09.

```

2) 将远程主机svr7的/boot/目录同步为本地的/fromssh

- ```

01. [root@pc207 ~]# rsync -a root@192.168.4.7:/boot/ /fromssh
02. root@192.168.4.7's password:           //验证对方的密码
03. [root@pc207 ~]# ls /fromssh/           //检查同步结果
04. config-3.10.0-327.el7.x86_64
05. extlinux
06. grub2
07. initramfs-0-rescue-a19921505cc7e19d20dfcd5cea7d8aa2.img
08. initramfs-3.10.0-327.el7.x86_64.img
09. initramfs-3.10.0-327.el7.x86_64kdump.img
10. .. ..

```

3) 将本机的/etc目录同步到远程主机svr7的/opt/下

确认目录大小：

- ```

01. [root@pc207 ~]# du -sh /etc
02. 35M /etc

```

[Top](#)

上行同步到远程主机svr7上：

01. [root@pc207 ~]# rsync -a /etc root@192.168.4.7:/opt/
02. root@192.168.4.7's password:

在远程主机上检查同步结果：

01. [root@svr7 ~]# du -sh /opt/etc
02. 35M /opt/etc

## 3 案例3：使用inotifywait工具

### 3.1 问题

本例要求安装inotify-tools工具，并针对文件夹 /opt 启用 inotifywait 监控，完成下列任务：

1. 当此目录下出现新建、修改、更改权限、删除文件等事件时能给出提示
2. 验证上述监控事件的效果

### 3.2 方案

inotifywait监控操作：

- inotifywait [选项] 目标文件夹

inotifywait常用命令选项：

- -m，持续监控（捕获一个事件后不退出）
- -r，递归监控、包括子目录及文件
- -q，减少屏幕输出信息
- -e，指定监视的 modify、move、create、delete、attrib 等事件类别

### 3.3 步骤

实现此案例需要按照如下步骤进行。

#### 步骤一：安装inotify-tools软件包

##### 1) 解包

01. [root@svr7 ~]# tar xf inotify-tools-3.13.tar.gz -C /usr/src/

##### 2) 配置

01. [root@svr7 ~]# cd /usr/src/inotify-tools-3.13/
02. [root@svr7 inotify-tools-3.13]# ./configure

[Top](#)

##### 3) 编译



```
01. [root@svr7 inotify-tools-3.13]# make
```

#### 4) 安装

```
01. [root@svr7 inotify-tools-3.13]# make
```

#### 5) 检查安装结果 (inotifywait程序可用)

```
01. [root@svr7 ~]# inotifywait --help
02. inotifywait 3.13
03. Wait for a particular event on a file or set of files.
04. Usage: inotifywait [options] file1 [file2] [file3] [...]
05. Options:
06. -h|--help Show this help text.
07.
```

### 步骤二：测试inotifywait监控

#### 1) 开启监控任务，置入后台

```
01. [root@svr7 ~]# inotifywait -mrq -e create,modify,move,attrib,delete /opt &
02. [1] 55564
```

#### 2) 测试/opt/目录下的新建、修改、改名、更改权限、删除文件等事件的响应消息 观察新建文件时的监控信息：

```
01. [root@svr7 ~]# touch /opt/a.txt
02. /opt/ CREATE a.txt
03. /opt/ ATTRIB a.txt
```

#### 观察修改文件内容时的监控信息：

```
01. [root@svr7 ~]# echo Hello > /opt/a.txt
02. [root@svr7 ~]# /opt/ MODIFY a.txt
03. /opt/ MODIFY a.txt
```

[Top](#)

观察将文件改名时的监控信息：

01. [root@svr7 ~]# mv /opt/a.txt /opt/b.txt
02. /opt/ MOVED\_FROM a.txt
03. /opt/ MOVED\_TO b.txt

观察修改文件权限时的监控信息：

01. [root@svr7 ~]# chmod 600 /opt/b.txt
02. /opt/ ATTRIB b.txt

观察删除文件时的监控信息：

01. [root@svr7 ~]# rm -rf /opt/b.txt
02. /opt/ DELETE b.txt

### 3) 停止监控任务

01. [root@svr7 ~]# kill -9 %1
02. [1]+ 已杀死 inotifywait -mr -e create,modify,move,attrib,delete /opt

## 4 案例4：配置Web镜像同步

### 4.1 问题

本例要求为两台Web服务器svr7、pc207的网页文档目录配置镜像同步，主要基于inotifywait监控技术实现实时触发操作，需要完成下列任务：

1. 以 svr7 为发起方，原始目录为 /var/www/html/
2. 以 pc207 为同步目标，基于SSH免密验证
3. 编写 inotify+rsync 同步脚本，验证实时同步效果

### 4.2 方案

inotifywait与rsync的结合，主要思路：

[Top](#)

01. while inotifywait监控操作
02. do

03. 需要执行的rsync同步操作
04. done

### 4.3 步骤

实现此案例需要按照如下步骤进行。

#### 步骤一：为主机svr7、pc207部署同步目录

双方的目录均为/var/www/html/，如果安装了httpd，此目录会自动出现。

##### 1) 确认svr7的目录内容

01. [root@svr7 ~]# yum -y install httpd
02. .. ..
03. [root@svr7 ~]# ls /var/www/html/ //向目录下提供一些测试文件
04. libreoffice

##### 2) 确认pc207的目录内容

01. [root@pc207 ~]# yum -y install httpd
02. .. ..
03. [root@pc207 ~]# ls /var/www/html //初始目录无数据
04. [root@pc207 ~]#

#### 步骤二：为svr7配置到pc207的SSH密钥对验证，实现免密码交互

##### 1) 检查当前用户是否已经有可用的SSH密钥对文件

01. [root@svr7 ~]# ls ~/.ssh/id\_\*
02. /root/.ssh/id\_rsa /root/.ssh/id\_rsa.pub

如果找不到id\_rsa、id\_rsa.pub密钥对文件，则需要执行下列操作创建：

01. [root@svr7 ~]# ssh-keygen
02. Generating public/private rsa key pair.
03. Enter file in which to save the key (/root/.ssh/id\_rsa): //按回车，确认存放位置
04. Enter passphrase (empty for no passphrase): //按回车，确认不要密码
05. Enter same passphrase again: //再次按回车，确认
06. Your identification has been saved in /root/.ssh/id\_rsa.

[Top](#)

```

07. Your public key has been saved in /root/.ssh/id_rsa.pub.
08. The key fingerprint is:
09. 00:a7:cb:2d:9d:b8:8a:df:f5:ff:5b:ed:bd:04:10:fe root@svr7
10. The key's randomart image is:
11. +--[RSA 2048]----+
12. | . . . |
13. | + . . |
14. | . . o |
15. | . = o o |
16. | = + S E |
17. | o .. |
18. | |
19. | . o|
20. |..oo. .+|
21. +-----+

```

## 2) 将当前用户的SSH公钥部署到远程主机

```

01. [root@svr7 ~]# ssh-copy-id root@192.168.4.207
02. The authenticity of host '192.168.4.207 (192.168.4.207)' can't be established.
03. ECDSA key fingerprint is d3:16:2c:9a:9d:91:28:c8:74:9c:af:2d:04:82:c9:66.
04. Are you sure you want to continue connecting (yes/no)? yes //首次连yes确认
05. root@192.168.4.207's password: //验证对方的密码
06.
07. Number of key(s) added: 1
08.
09. Now try logging into the machine, with: "ssh 'root@192.168.4.207'"
10. and check to make sure that only the key(s) you wanted were added.

```

## 3) 验证免密码登录效果

```

01. [root@svr7 ~]# ssh root@192.168.4.207
02. Last login: Fri Jan 13 09:52:08 2017 from 192.168.4.110
03. [root@pc207 ~]# //确认已免密码连入远程主机
04. [root@pc207 ~]# exit //退出SSH登录环境
05. 登出
06. Connection to 192.168.4.207 closed.
07. [root@svr7 ~]# //已反对原客户机

```

[Top](#)

## 步骤三：编写镜像同步脚本并测试效果

### 1) 编写脚本文件/root/isync.sh

```
01. [root@svr7 ~]# vim /root/isync.sh
02. #!/bin/bash
03. FROM_DIR="/var/www/html/"
04. RSYNC_CMD="rsync -az --delete $FROM_DIR root@192.168.4.207:/var/www/html"
05. while inotifywait -rq -e modify,move,create,delete,attrib $FROM_DIR
06. do
07. $RSYNC_CMD
08. done &
09. [root@svr7 ~]# chmod +x /root/isync.sh
```

### 2) 运行脚本

```
01. [root@svr7 ~]# /root/isync.sh
02. [root@svr7 ~]# pgrep -l inotify //确认任务在运行
03. 56494 inotifywait
```

### 3) 测试同步效果

在svr7上向/var/www/html/目录下添加一个测试网页（触发同步）：

```
01. [root@svr7 ~]# touch /var/www/html/a.txt
02. [root@svr7 ~]# ls /var/www/html/
03. a.txt libreoffice
```

在pc207上检查/var/www/html/目录，内容应该已经与svr7上的同名目录一致：

```
01. [root@pc207 ~]# ls /var/www/html
02. a.txt libreoffice
```

### 4) 结束测试后，在svr7上停止监控任务

[Top](#)

```
01. [root@svr7 ~]# pkill -9 inotify
02. [root@svr7 ~]# pgrep -l inotify //确认已没有监控任务
```

03. [root@svr7 ~]#

## 5 案例5：配置Cobbler装机平台

### 5.1 问题

本例要求为新建虚拟机硬盘位20G，网络类型为“private1”操作系统为CentOS 7：

1. 虚拟机名设置为Cobbler
2. IP地址设置为：192.168.4.123/24
3. 关闭SELinux安全机制
4. 设置防火墙默认区域为trusted
5. 利用Cobbler部署CentOS 7与 RedHat 7双系统装机平台

### 5.2 方案

cobbler #cobbler程序包  
cobbler-web #cobbler的web服务包  
pykickstart #cobbler检查kickstart语法错误  
httpd #Apache web服务  
dhcp #Dhcp服务  
tftp-server #tftp服务

### 5.3 步骤

实现此案例需要按照如下步骤进行。

#### 步骤一：在虚拟机中解压cobbler.zip包

此cobbler.zip包，起初应该在真机上，可以通过scp命令传递到虚拟机。

1) 将真机cobbler.zip包传递到虚拟机中/root目录下，注意真机cobbler.zip绝对路径

```
01. [root@room9pc01 桌面]# scp -r /root/桌面/Cobbler/ root@192.168.4.123:/root/
02. The authenticity of host '192.168.4.123 (192.168.4.123)' can't be established.
03. ECDSA key fingerprint is SHA256:TFNqSD+oJMsA88kLwSdLSKZhSigkQIIAfrXLdKoUaJA.
04. ECDSA key fingerprint is MD5:f9:b8:7d:8d:ca:4e:20:0d:10:c4:72:a5:9f:42:28:8e.
05. Are you sure you want to continue connecting (yes/no)? yes
06. Warning: Permanently added '192.168.4.123' (ECDSA) to the list of known hosts.
07. root@192.168.4.123's password:
08. cobbler_web.png 100% 78KB 6.7MB/s 00:00
09. cobbler.zip 100% 9781KB 61.1MB/s 00:00
10. cobbler_boot.tar.gz 100% 416KB 45.7MB/s 00:00
11. [root@room9pc01 桌面]#
```

[Top](#)

2) 确认虚拟机中的目录内容

- ```
01. [root@cob ~]# ls /root/Cobbler/
02. cobbler_boot.tar.gz cobbler_web.png cobbler.zip
03. [root@cob ~]#
```

步骤二：搭建Yum仓库，安装Cobbler

1) 将cobbler.zip包，解压缩到根目录下

- ```
01. [root@cob ~]# unzip /root/Cobbler/cobbler.zip -d /
02.
03. [root@cob ~]# ls /cobbler/
04.
```

2) 搭建Yum仓库，利用yum安装所有rpm软件包

- ```
01. [root@cob ~]# mount /dev/cdrom /mnt/ #首先通过图形将CentOS光盘放入光驱设备
02. mount: /dev/sr0 写保护，将以只读方式挂载
03. [root@cob ~]# ls /mnt/
04. [root@cob ~]# rm -rf /etc/yum.repos.d/*
05. [root@cob ~]# vim /etc/yum.repos.d/dvd.repo
06. [CentOS7]
07. name=CentOS 7.4
08. baseurl=file:///mnt
09. enabled=1
10. gpgcheck=0
11. [root@cob ~]# yum repolist
12. ....
13. 源标识          源名称          状态
14. CentOS7          CentOS 7.4          3,894
15. repolist: 3,894
16. [root@cob ~]# yum -y install /cobbler/*.rpm
```

步骤三：设置防火墙与SELinux

1) 设置防火墙默认区域为trusted

- ```
01. [root@cob ~]# firewall-cmd --set-default-zone=trusted
```

[Top](#)

## 2) 修改SELinux状态

01. [root@cob ~]# setenforce 0
02. [root@cob ~]# getenforce
03. Permissive
04. [root@cob ~]# vim /etc/selinux/config
05. ....
06. SELINUX=permissive
07. ....

## 步骤四：配置cobbler

### 1) 修改配置文件/etc/cobbler/settings

01. [root@cob ~]# vim /etc/cobbler/settings
02. ....
03. next\_server: 192.168.4.123 #设置下一个服务器还为本机
04. server: 192.168.4.123 #设置本机为cobbler服务器
05. manage\_dhcp: 1 #设置cobbler管理dhcp服务
06. pxe\_just\_once: 1 #防止客户端重复安装操作系统,增加默认从本机硬盘启动菜

### 2) 配置cobbler的dhcp分配网段及IP地址范围

01. [root@cob ~]# vim /etc/cobbler/dhcp.template
02. ....
03. ....
04. ....
05. :%s /192.168.1/192.168.4/g #将所有192.168.1替换为192.168.4

### 3) 绝对路径解压cobbler\_boot.tar.gz

01. [root@cob ~]# tar -tf /root/Cobbler/cobbler\_boot.tar.gz #众多的引导文件
02. ....
03. [root@cob ~]# tar -xPf /root/Cobbler/cobbler\_boot.tar.gz
04. [root@cob ~]# ls /var/lib/cobbler/loaders/ #默认cobbler存放引导文件路径

## 步骤四：启动相关服务



### 1) 启动cobblerd主服务

01. `[root@cob ~]# systemctl restart cobblerd`
02. `[root@cob ~]# systemctl enable cobblerd`
03. Created symlink from `/etc/systemd/system/multi-user.target.wants/cobblerd.service` to `/usr/lib/systemd/system/cobblerd.service`

### 2) 启动httpd主服务，主要提供Web页面、装机光盘内容，ks应答文件等

01. `[root@cob ~]# systemctl restart httpd`
02. `[root@cob ~]# systemctl enable httpd`
03. Created symlink from `/etc/systemd/system/multi-user.target.wants/httpd.service` to `/usr/lib/systemd/system/httpd.service`

### 3) 启动tftp主服务，主要提供pxelinux.0、菜单文件等

01. `[root@cob ~]# systemctl restart tftp`
02. `[root@cob ~]# systemctl enable tftp`
03. Created symlink from `/etc/systemd/system/sockets.target.wants/tftp.socket` to `/usr/lib/systemd/system/tftp.socket`

### 4) 启动rsyncd主服务，主要提供cobbler各个组件之间同步数据

01. `[root@cob ~]# systemctl restart rsyncd`
02. `[root@cob ~]# systemctl enable rsyncd`
03. Created symlink from `/etc/systemd/system/multi-user.target.wants/rsyncd.service` to `/usr/lib/systemd/system/rsyncd.service`

### 步骤五：同步刷新cobbler所有配置

01. `[root@cob ~]# cobbler sync`
02. task started: 2018-03-22\_200534\_sync
03. task started (id=Sync, time=Thu Mar 22 20:05:34 2018)
04. running pre-sync triggers
05. cleaning trees
06. removing: `/var/lib/tftpboot/grub/images`
07. copying bootloaders

[Top](#)

08. copying: `/var/lib/cobbler/loaders/pxelinux.0 -> /var/lib/tftpboot/pxelinux.0`
09. copying: `/var/lib/cobbler/loaders/menu.c32 -> /var/lib/tftpboot/menu.c32`
10. copying: `/var/lib/cobbler/loaders/yaboot -> /var/lib/tftpboot/yaboot`
11. copying: `/usr/share/syslinux/memdisk -> /var/lib/tftpboot/memdisk`
12. copying: `/var/lib/cobbler/loaders/grub-x86.efi -> /var/lib/tftpboot/grub/grub-x86.efi`
13. copying: `/var/lib/cobbler/loaders/grub-x86_64.efi -> /var/lib/tftpboot/grub/grub-x86_`
14. copying distros to tftpboot
15. copying images
16. generating PXE configuration files
17. generating PXE menu structure
18. rendering DHCP files
19. generating `/etc/dhcp/dhcpd.conf`
20. rendering TFTP files
21. generating `/etc/xinetd.d/tftp`
22. cleaning link caches
23. running post-sync triggers
24. running python triggers from `/var/lib/cobbler/triggers/sync/post/*`
25. running python trigger `cobbler.modules.sync_post_restart_services`
26. running: `dhcpd -t -q`
27. received on stdout:
28. received on stderr:
29. running: `service dhcpd restart`
30. received on stdout:
31. received on stderr: Redirecting to `/bin/systemctl restart dhcpd.service`
- 32.
33. running shell triggers from `/var/lib/cobbler/triggers/sync/post/*`
34. running python triggers from `/var/lib/cobbler/triggers/change/*`
35. running python trigger `cobbler.modules.scm_track`
36. running shell triggers from `/var/lib/cobbler/triggers/change/*`
37. \*\*\* TASK COMPLETE \*\*

## 步骤六：导入系统光盘镜像数据

1) 导入CentOS系统光盘镜像，cobbler默认提供ks应答文件，但应答文件为最小化安装，命令格式：

`cobbler import --path=挂载点 --name=导入系统命名 --arch=操作系统架构`

cobbler导入的镜像放在：`/var/www/cobbler/ks_mirror`

01. `[root@cob ~]# cobbler import --path=/mnt --name=CentOS7 --arch=x86_64` [Top](#)
02. task started: 2018-03-22\_201215\_import

03. task started (id=Media import, time=Thu Mar 22 20:12:15 2018)
04. ....
05. Keeping repodata as-is :/var/www/cobbler/ks\_mirror/CentOS7-x86\_64/repodata
06. \*\*\* TASK COMPLETE \*\*\*
07. [root@cob ~]# ls /var/www/cobbler/ks\_mirror/
08. CentOS7-x86\_64 config
09. [root@cob ~]#

2) 首先卸载光驱设备挂载，通过图形将光驱设备中的光盘，换成RHEL7光盘如图-1所示，导入RHEL7系统光盘镜像，cobbler默认提供ks应答文件，但应答文件为最小化安装

01. [root@cob ~]# umount /mnt/ #卸载光驱设备，将光盘换成RHEL7

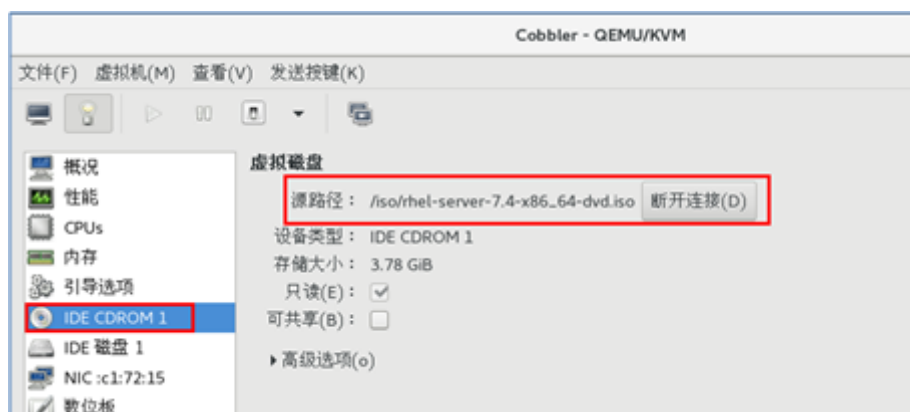


图-1

01. [root@cob ~]# mount /dev/cdrom /mnt/
02. mount: /dev/sr0 写保护，将以只读方式挂载
03. [root@cob ~]# ls /mnt/
04. addons extra\_files.json isolinux Packages RPM-GPG-KEY-redhat-release
05. EFI GPL LiveOS repodata TRANS.TBL
06. EULA images media.repo RPM-GPG-KEY-redhat-beta
07. [root@cob ~]# cobbler import --path=/mnt --name=RedHat --arch=x86\_64
08. task started: 2018-03-22\_202531\_import
09. task started (id=Media import, time=Thu Mar 22 20:25:31 2018)
10. ....
11. Keeping repodata as-is :/var/www/cobbler/ks\_mirror/RedHat-x86\_64/addons/Resilie
12. \*\*\* TASK COMPLETE \*\*\*
13. [root@cob ~]# ls /var/www/cobbler/ks\_mirror/
14. CentOS7-x86\_64 config RedHat-x86\_64
15. [root@cob ~]#

[Top](#)

## 步骤七：新建虚拟机测试

### 1) 新建一台虚拟机测试：

- 选择pxe安装方式
- 注意如果安装CentOS系统虚拟机内容必须为2G以上，安装RedHat内存1G以上，硬盘均在9G以上
- 测试虚拟机网络类型选择为：private1 如图-2与图-3所示

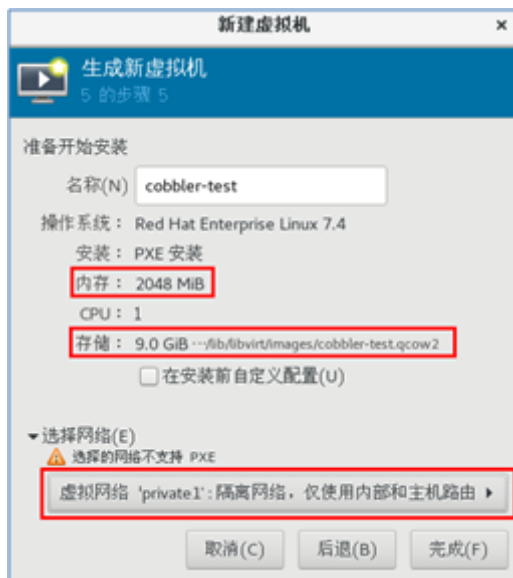


图-2

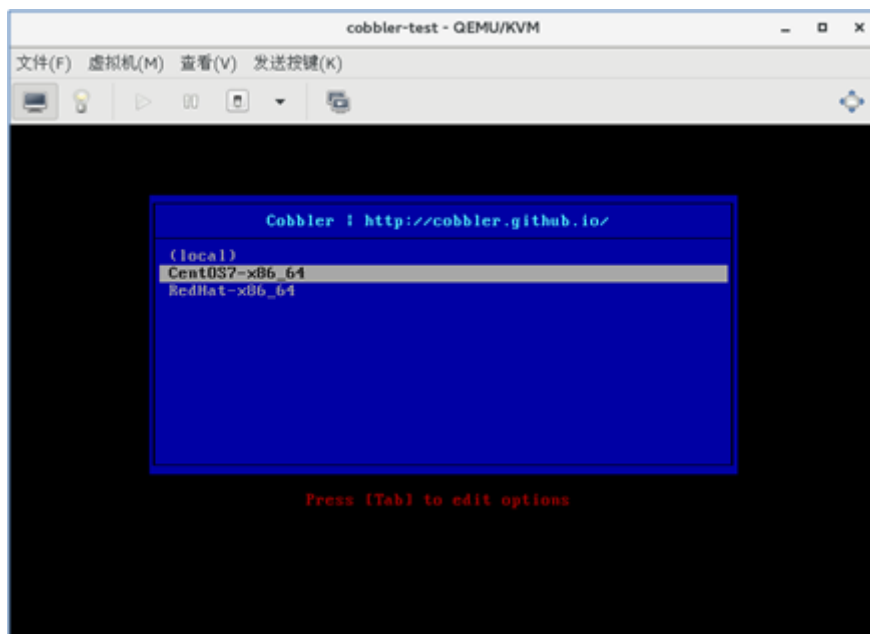


图-3

[Top](#)