

NSD CLUSTER DAY01

1. [案例1：配置iSCSI服务](#)
2. [案例2：部署Multipath多路径环境](#)
3. [案例3：配置并访问NFS共享](#)
4. [案例4：编写udev规则](#)

1 案例1：配置iSCSI服务

1.1 问题

本案例要求先搭建好一台iSCSI服务器，并将整个磁盘共享给客户端：

- 服务器上要额外配置一块硬盘
- 服务端安装target，并将新加的硬盘配置为iSCSI 的共享磁盘
- 在客户端上安装initiator，挂在服务器iSCSI，要求实现开机自动挂载

1.2 方案

使用2台RHEL7虚拟机，其中一台作为iSCSI服务器（192.168.2.5）、另外一台作为测试用的客户机（192.168.2.100），如图-1所示，主机网络地址配置如表-1所示。

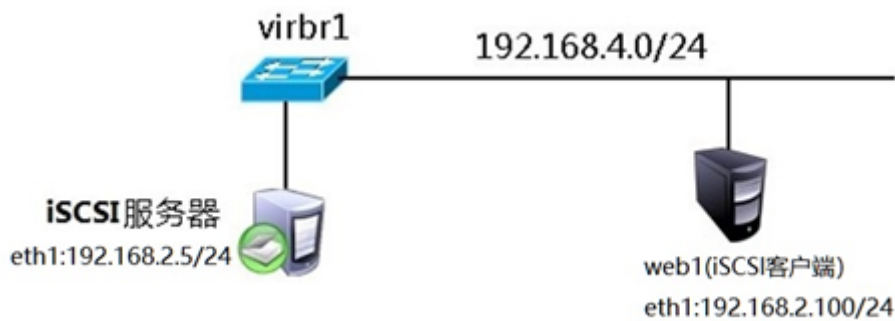


图-1

表-1 主机网络参数配置列表

主机名	网络参数配置
proxy(作为 iSCSI 服务器)	eth1:192168.2.5/24
web1(做为 iSCSI 客户端)	eth1:192.168.2.100/24

在RHEL7系统中，默认通过targetcli软件包提供iSCSI服务，因此需要在服务端安装targetcli包并配置对应的服务，iSCSI服务主要配置选项如表-1所示。

表 - 2 iSCSI配置选项列表

选项	值
后端存储	/dev/vdb1
iSCSI target 名称	iqn.2018-01.cn.tedu:server1
ACL 授权客户端访问	iqn.2018-01.cn.tedu:client1

[Top](#)

客户端挂载iSCSI服务器：

- 客户端需要安装iscsi-initiator-utils软件包
- 客户端使用命令挂载后需要分区、格式化并进行挂载测试

1.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：安装iSCSI服务器软件

1) 使用yum安装targetcli软件包

```
01. [root@proxy ~]# yum -y install targetcli
02. ...
03. [root@proxy ~]# yum info targetcli
04. ...
```

步骤二：通过命令行配置iSCSI服务

1) 真实主机准备底层存储磁盘

真实主机使用virt-manager工具为proxy虚拟机添加磁盘，如图-2所示。

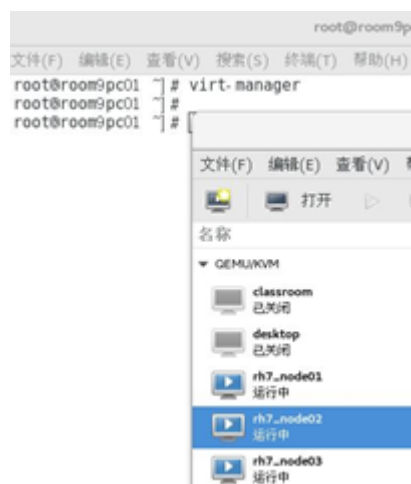


图-2

双击打开虚拟机后添加磁盘设备，如图-3和图-4所示。

[Top](#)

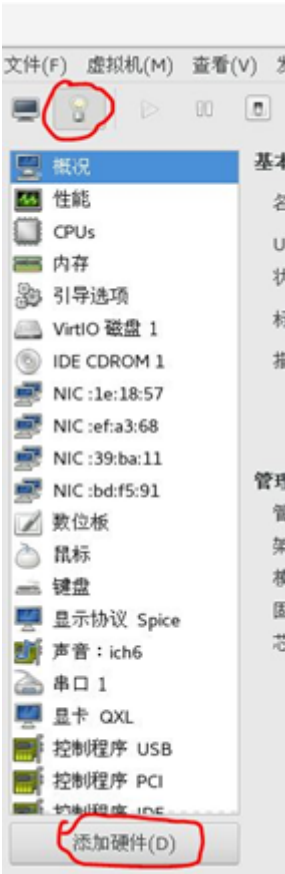


图-3

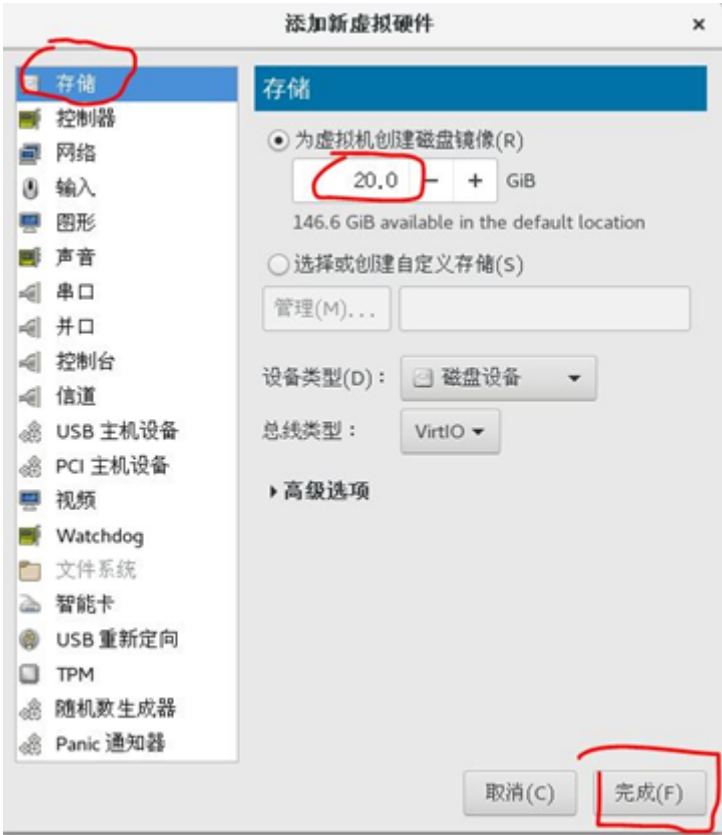


图-4

登陆到192.168.2.5主机，为新添加的磁盘准备分区：

[Top](#)

```
01 [root@proxy ~] # parted /dev/vdb mklabel gpt
```

```
02. [ root@proxy ~] # parted /dev /vdb mkpart primary 1 100%
```

2)使用targetcli定义后端存储

设置需要将哪个设备共享给其他主机，这里将/dev/vdb1设置为后端共享磁盘。

```
01. [ root@proxy ~] # targetcli
02. /> ls
03. /> backstores/block create store /dev /vdb1
04. 备注：store为任意名称
```

3) 创建iqn对象

给iSCSI共享设置一个共享名称，客户端访问时需要使用该共享名称。

```
01. /> /iscsi create iqn.2018-01.cn.tedu:server1
```

4) 授权客户机访问

类似于一个密码，设置ACL访问控制，拥有iqn.2018-01.cn.tedu:client1这个字符串的客户端才有权限访问服务器。

```
01. /> iscsi/iqn.2018-01.cn.tedu:server1/tpg1/acls create iqn.2018-01.cn.tedu:client1
```

5) 绑定存储

将iqn共享名称 (iqn.2018-01.cn.tedu:server1) 与后端实际的存储设备 (vdb) 绑定。

```
01. /> iscsi/iqn.2018-01.cn.tedu:server1/tpg1/luns create /backstores/block/store
```

#注意：block后面的store必须与前面步骤2定义后端存储create创建的名称一致。

6) 存储绑定服务监听的地址，并保存配置

```
01. /> iscsi/iqn.2018-01.cn.tedu:server1/tpg1/portals/ create 0.0.0.0
02. /> saveconfig
03. /> exit
```

[Top](#)

步骤三：服务管理

1) 启动服务

01. [root@proxy ~] # systemctl { start| restart| stop| status} target
02. [root@proxy ~] # systemctl enable target

2) 查看端口信息

01. [root@proxy ~] # ss - utlnp | grep :3260

3) 关闭防火墙与SELinux

01. [root@proxy ~] # systemctl stop firewalld
02. [root@proxy ~] # setenforce 0

步骤四：客户端访问(web1作为客户端的角色)

1) 客户端安装软件并启动服务

01. [root@web1 ~] # yum -y install iscsi-initiator-utils

2) 设置本机的iqn名称

01. [root@web1 ~] # vim /etc/iscsi/initiatorname.iscsi
02. InitiatorName=iqn.2018-01.cn.tedu:client1
03. 注意：必须跟服务器上配置的ACL一致！

3) 发现远程target存储

提示：参考man iscsiadm！

01. [root@web1 ~] # iscsiadm -- mode discovery db -- type sendtargets -- portal 192.168.2.5 --
02. [root@web1 ~] # iscsiadm -- mode node -- targetname iqn.2018-01.cn.tedu:server1 -- port

3) 客户端挂载iSCSI共享

```

01. [root@web1 ~] # lsblk
02. NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
03. sda        8:0  0  20G  0 disk
04. #多了一个sda设备
05. sr0        11:0  1 1024M  0 rom
06. vda        252:0  0  20G  0 disk
07. [root@web1 ~] # systemctl restart iscsi

```

4) 分区、格式化、挂载

```

01. [root@web1 ~] # parted /dev/sda mklabel gpt
02. [root@web1 ~] # parted /dev/sda mkpart primary 1 800
03. [root@web1 ~] # mkfs.xfs /dev/sda1
04. [root@web1 ~] # mount /dev/sda1 /mnt
05. [root@web1 ~] # umount /mnt

```

步骤四：附加课外实验：多台FTP或者http主机使用共享存储。

这里以FTP为例，web1和web2主机都安装vsftpd软件，使用统一的后端共享存储设备。

1) web1操作(延续前面步骤三的实验)：

```

01. [root@web1 ~] # mkdir /var/ftp/
02. [root@web1 ~] # mount /dev/sda1 /var/ftp/
03. [root@web1 ~] # yum -y install vsftpd
04. [root@web1 ~] # sed -i 's/^#anon/anon/' /etc/vsftpd/vsftpd.conf

```

备注：修改vsftpd配置文件，开启匿名上传功能。将下面2行默认的注释行打开。

#anon_upload_enable=YES

#anon_mkdir_write_enable=YES

```

01. [root@web1 ~] # chmod 777 /var/ftp/pub
02. [root@web1 ~] # systemctl start vsftpd
03. [root@web1 ~] # systemctl enable vsftpd

```

2) 真实主机访问web1的FTP共享，并任意上传一个文件到FTP服务器。

[Top](#)

打开真实主机浏览网络，如图-5所示。



图-5

输入需要连接的服务器协议与IP地址，如图-6所示。

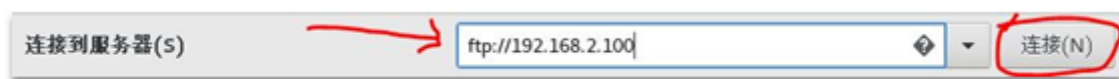


图-6

连接到服务器后，进入pub共享目录，将真实主机的任意文件拖拽到FTP的共享目录下（pub目录）。注意：仅pub目录有读写权限。



图-7

3) 当web1宕机后，web2主机可以继续使用iscsi提供FTP共享服务。
Web1关闭vsftpd服务，卸载iscsi挂载。

- ```
01. [root@web1 ~]# systemctl stop vsftpd
02. [root@web1 ~]# umount /var/ftp
```

添加iSCSI共享（web2操作）。

[Top](#)

- ```
01. [root@web2 ~]# vim /etc/iscsi/initiatorname.iscsi
02. InitiatorName=iqn.2018-01.cn.tedu:client1
```

03. 注意：必须跟服务器上配置的ACL一致！
04. [root@web2 ~] # iscsiadm -- mode discovery db -- type sendtargets -- portal 192.168.2.5 --
05. [root@web2 ~] # iscsiadm -- mode node -- targetname iqn.2018-01.cn.tedu:server1 -- port

安装部署vsftpd软件(web2操作)。

01. [root@web2 ~] # yum -y install vsftpd
02. [root@web2 ~] # sed -i 's/^#anon/anon/' /etc/vsftpd/vsftpd.conf
03. [root@web2 ~] # chmod 777 /var/ftp/pub/
04. [root@web2 ~] # systemctl start vsftpd
05. [root@web2 ~] # systemctl enable vsftpd

4) 真实主机访问web2的FTP共享，查看共享里现有的数据，并任意上传一个新文件到FTP服务器。操作步骤参考图-5至图-7所示，注意修改对应的IP地址。

思考？

写一个检测ftp的脚本，如果发现web1宕机后，web2自动mount挂载iscsi共享，自动启动vsftpd服务。

2 案例2：部署Multipath多路径环境

2.1 问题

通过Multipath，实现以下目标：

- 在共享存储服务器上配置iSCSI，为应用服务器共享存储空间
- 应用服务器上配置iSCSI，发现远程共享存储
- 应用服务器上配置Multipath，将相同的共享存储映射为同一个名称

2.2 方案

配置2台虚拟机，每台虚拟机均为两块网卡：

- eth0和eth1都可用于iSCSI存储通讯
- 具体配置如表-3所示

表-3 各节点IP地址配置

节点及网卡	地址及应用
proxy (iSCSI 服务器)	eth0: 192.168.4.5/24 eth1: 192.168.4.5/24
web1(iSCSI 客户端)	eth0: 192.168.4.100/24 eth3: 201.1.2.100/24

[Top](#)

多路径示意图，如图-8所示。

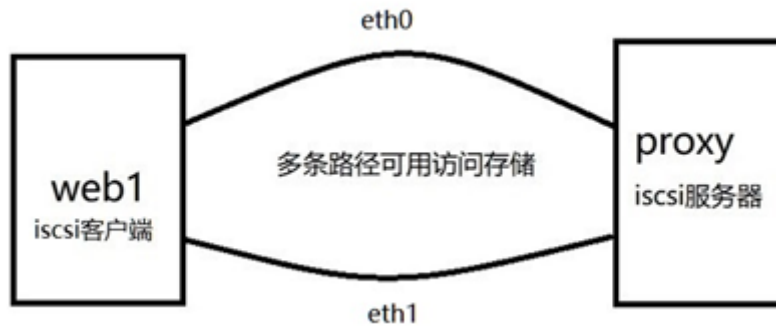


图-8

2.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：存储服务器上添加额外的磁盘（如果已经完成案例1，此步骤可以忽略）

使用KVM软件新建（或修改）虚拟机，为虚拟机额外添加一块硬盘。

步骤二：存储服务器上安装并配置共享存储（如果已经完成案例1，此步骤可用忽略）

1) 定义后端存储

```

01. [root@proxy ~]# targetcli
02. /> ls
03. /> backstores/block create store /dev/vdb1

```

2) 创建iqn对象

```

01. /> /iscsi create iqn.2018-01.cn.tedu:server1

```

3) 授权客户机访问

```

01. /> iscsi/iqn.2018-01.cn.tedu:server1/tpg1/acls create iqn.2018-01.cn.tedu:client1

```

4) 绑定存储

```

01. /> iscsi/iqn.2018-01.cn.tedu:server1/tpg1/luns create /backstores/block/store

```

5) 绑定存储绑定监听地址，并保存配置

[Top](#)

```

01.  /> iscsi/iqn.2018-01.cn.tedu:server1/tpg1/portals/ create 0.0.0.0
02.  /> saveconfig
03.  /> exit

```

步骤三：在client服务器上安装并配置iSCSI客户端

1) 安装客户端软件（前面的案例1已经完成的情况下，可以忽略此步骤）

```

01.  [root@web1 ~] # yum list | grep iscsi
02.  iscsi-initiator-utils.x86_64 6.2.0.873-14.el6
03.  [root@web1 ~] # yum install -y iscsi-initiator-utils

```

2) 发现存储服务器的共享磁盘

因为有两条链路都可以连接到共享存储，所以需要在两条链路上都发现它。

注意：两次发现使用的IP地址不同！

```

01.  [root@web1 ~] # iscsiadm --mode discoverydb --type sendtargets --portal 192.168.2.5 --
02.  192.168.2.5:3260,1 iqn.2018-01.cn.tedu:client1
03.
04.  [root@web1 ~] # iscsiadm --mode discoverydb --type sendtargets --portal 192.168.4.5 --
05.  192.168.4.5:3260,1 iqn.2018-01.cn.tedu:client1

```

3) 登陆共享存储

只需要将iscsi服务重启就可以自动登陆（就不需要再login了）。

在login之前，只能看到本地的存储，登陆之后，将会多出两块新的硬盘。

```

01.  ... ..
02.
03.  [root@web1 ~] # service iscsi restart
04.  停止 iscsi : [确定]
05.  正在启动 iscsi : [确定]
06.  [root@web1 ~] # lsblk
07.  NAME                                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
08.  sda                                8:0  0  20G  0 disk
09.  └─sda1                            8:1  0  20G  0 part
10.  sdb                                8:0  0  20G  0 disk

```

[Top](#)

```

11.  |—sdb1          8:1 0 20G 0 part
12.  vda            252:0 0 20G 0 disk
13.  |—vda1         252:1 0 1G 0 part /boot

```

提示：登陆的是同一个服务器的同一个iSCSI，但客户端看到的是两个独立的设备，sda和sdb。其实，这两个设备是同一个设备。

4) 设置开机自启动

iscsi用于自动login远程存储，iscsid是守护进程。

```

01.  [ root@web1 ~] # systemctl enable iscsid
02.  [ root@web1 ~] # systemctl enable iscsi

```

步骤四：配置Multipath多路径

1) 安装多路径软件包

```

01.  [ root@web1 ~] # yum list | grep multipath
02.  device-mapper-multipath.x86_64 0.4.9-111.el7 Server
03.  device-mapper-multipath-libs.i686 0.4.9-111.el7 Server
04.  device-mapper-multipath-libs.x86_64 0.4.9-111.el7 Server
05.
06.  [ root@web1 ~] # yum install -y device-mapper-multipath

```

2) 生成配置文件

```

01.  [ root@web1 ~] # cd /usr/share/doc/device-mapper-multipath-0.4.9/
02.  [ root@web1 ~] # ls multipath.conf
03.  [ root@web1 ~] # cp multipath.conf /etc/multipath.conf

```

3) 获取wwid

登陆共享存储后，系统多了两块硬盘，这两块硬盘实际上是同一个存储设备。应用服务器使用哪个都可以，但是如果使用sdb时，sdb对应的链路出现故障，它不会自动切换到sda。

为了能够实现系统自动选择使用哪条链路，需要将这两块磁盘绑定为一个名称。

通过磁盘的wwid来判定哪些磁盘是相同的。

取得一块磁盘wwid的方法如下：

```

01.  [ root@web1 ~] # /usr/lib/udev/scsi_id --whitelisted --device=/dev/sdb

```

[Top](#)

02. 360014059e8ba68638854e9093f 3ba3a0

4) 修改配置文件

首先声明自动发现多路径：

```
01. [root@web1 ~] # vim /etc/multipath.conf
02. defaults {
03.     user_friendly_names yes
04.     find_multipaths yes
05. }
```

然后在文件的最后加入多路径声明，如果哪个存储设备的wwid和第（3）步获取的wwid一样，那么，为其取一个别名，叫mpatha。

```
01. multipaths {
02.     multipath {
03.         wwid "360014059e8ba68638854e9093f 3ba3a0"
04.         alias mpatha
05.     }
06. }
```

步骤五：启用Multipath多路径，并测试

注意：如果做案例1时，已经挂载了iSCSI设备，一定要先umount卸载掉再启动多路径。

1) 启动Multipath，并设置为开机启动

```
01. [root@web1 ~] # systemctl start multipathd
02. [root@web1 ~] # systemctl enable multipathd
```

2) 检查多路径设备文件

如果多路径设置成功，那么将在/dev/mapper下面生成名为mpatha的设备文件：

```
01. [root@web1 ~] # ls /dev/mapper/
02. control mpatha mpatha1
```

[Top](#)

3) 对多路径设备文件执行分区、格式化、挂载操作

提示：如果前面已经对iscsi做过分区操作，则这里可以直接识别到mpatha1（就不需要再次分区了）。

```

01. [root@web1 ~] # fdisk -cu /dev/mapper/mpatha
02. Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
03. Building a new DOS disklabel with disk identifier 0x205c887e.
04. Changes will remain in memory only, until you decide to write them.
05. After that, of course, the previous content won't be recoverable.
06.
07. Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)
08.
09. Command (m for help): n      # 创建分区
10. Command action
11.   e   extended
12.   p   primary partition (1-4)
13. p           # 分区类型为主分区
14. Partition number (1-4): 1    # 分区编号为1
15. First sector (2048-4194303, default 2048): # 起始扇区回车
16. Using default value 2048
17. Last sector, +sectors or +size{K,M,G} (2048-4194303, default 4194303): # 回车
18. Using default value 4194303
19.
20. Command (m for help): w      # 保存并退出
21. The partition table has been altered!
22.
23. Calling ioctl() to re-read partition table.

```

新的分区名称应该是/dev/mapper/mpathap1，如果该文件不存在，则执行以下命令进行配置的重新载入：

```

01. [root@web1 ~] # ls /dev/mapper/ # 再次查看，将会看到新的分区
02. control mpatha mpatha1

```

创建目录并挂载（如果已经格式化，这里就不需要再次格式化，直接挂载即可）：

```

01. [root@web1 ~] # mkfs.xfs /dev/mapper/mpathap1
02. [root@web1 ~] # mkdir /data
03. [root@web1 ~] # mount /dev/mapper/mpathap1 /data/

```

[Top](#)

```

04. [root@web1 ~]# df -h /data/
05. Filesystem      Size  Used Avail Use% Mounted on
06. /dev/mapper/mpatha1 20G  3.0M  19G   1%  /data

```

4) 验证多路径

查看多路径，sda和sdb都是running状态。

```

01. [root@web1 ~]# multipath -rr
02. reload: mpatha (360014059e8ba68638854e9093f3ba3a0) undef LIO- ORG ,store
03. size=9.3G features='0' hwhandler='0' wp=undef
04. | - + policy='service-time 0' prio=1 status=undef
05. | ` - 2:0:0:0 sda 8:0 active ready running
06. ` - + policy='service-time 0' prio=1 status=undef
07. ` - 3:0:0:0 sdb 8:16 active ready running

```

关闭某个链路后，再次查看效果，此时会发现sdb为运行失败状态。

```

01. [root@web1 ~]# nmcli connection down eth1
02. [root@web1 ~]# multipath -rr
03. reject: mpatha (360014059e8ba68638854e9093f3ba3a0) undef LIO- ORG ,store
04. size=9.3G features='0' hwhandler='0' wp=undef
05. | - + policy='service-time 0' prio=0 status=undef
06. | ` - 2:0:0:0 sda 8:0 active undef running
07. ` - + policy='service-time 0' prio=0 status=undef
08. ` - 3:0:0:0 sdb 8:16 active faulty running

```

使用-II选项查看，仅sda为有效运行状态。

```

01. [root@web1 ~]# multipath -II
02. reject: mpatha (360014059e8ba68638854e9093f3ba3a0) undef LIO- ORG ,store
03. size=9.3G features='0' hwhandler='0' wp=undef
04. ` - + policy='service-time 0' prio=0 status=undef
05. ` - 2:0:0:0 sda 8:0 active undef running

```

3 案例3：配置并访问NFS共享

[Top](#)

3.1 问题

服务器利用NFS机制发布2个共享目录，要求如下：

- 将目录/root共享给192.168.2.100，客户机的root用户有权限写入
- 将/usr/src目录共享给192.168.2.0/24网段，只开放读取权限

从客户机访问NFS共享：

- 分别查询/挂载上述NFS共享目录
- 查看挂载点目录，并测试是否有写入权限

3.2 方案

使用2台RHEL7虚拟机，其中一台作为NFS共享服务器（192.168.2.5）、另外一台作为测试用的Linux客户机（192.168.2.100），如图-10所示。

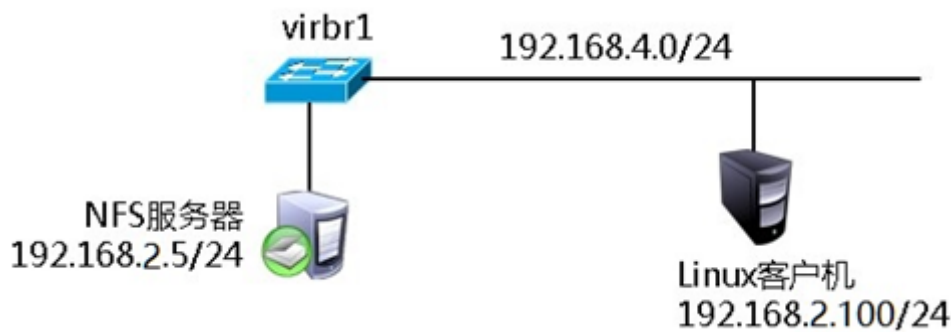


图-10

NFS共享的配置文件：/etc/exports。

配置记录格式：文件夹路径 客户地址1(控制参数...) 客户地址2(...)。

3.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：配置NFS服务器，发布指定的共享

1) 确认服务端程序、准备共享目录

软件包nfs-utils用来提供NFS共享服务及相关工具，而软件包rpcbind用来提供RPC协议的支持，这两个包在RHEL7系统中一般都是默认安装的：

```
01. [root@proxy ~]# rpm -q nfs-utils rpcbind
02. nfs-utils-1.3.0-0.48.el7.x86_64
03. rpcbind-0.2.0-42.el7.x86_64
```

根据本例的要求，需要作为NFS共享发布的有/root、/usr/src这两个目录：

```
01. [root@proxy ~]# ls -ld /root /usr/src/
02. dr-xr-x--. 35 root root 4096 1月 15 18:52 /root
03. drwxrwxr-x+ 4 root root 4096 1月 15 17:35 /usr/src/
```

[Top](#)

2) 修改/etc/exports文件，添加共享目录设置

默认情况下，来自NFS客户端的root用户会被自动降权为普通用户，若要保留其root权限，注意应添加no_root_squash控制参数(没有该参数，默认root会被自动降级为普通账户)；另外，限制只读的参数为ro、可读可写为rw，相关配置操作如下所示：

```
01. [root@proxy ~] # vim /etc/exports
02. /root          192.168.2.100(rw,no_root_squash)
03. /usr/src       192.168.2.0/24(ro)
```

3) 启动NFS共享相关服务，确认共享列表

依次启动rpcbind、nfs服务：

```
01. [root@proxy ~] # systemctl restart rpcbind ; systemctl enable rpcbind
02.
03. [root@proxy ~] # systemctl restart nfs ; systemctl enable nfs
```

使用showmount命令查看本机发布的NFS共享列表：

```
01. [root@proxy ~] # showmount -e localhost
02. Export list for localhost:
03. /usr/src 192.168.2.0/24
04. /root    192.168.2.100
```

步骤二：从客户机访问NFS共享

1) 启用NFS共享支持服务

客户机访问NFS共享也需要rpcbind服务的支持，需确保此服务已开启：

```
01. [root@web1 ~] # systemctl restart rpcbind ; systemctl enable rpcbind
```

2) 查看服务器提供的NFS共享列表

```
01. [root@web1 ~] # showmount -e 192.168.2.5
02. Export list for 192.168.2.5:
03. /usr/src 192.168.2.0/24
04. /root    192.168.2.100
```

[Top](#)

3) 从客户机192.168.2.100访问两个NFS共享，并验证权限

将远程的NFS共享/root挂载到本地的/root5文件夹，并验证可读可写：

```

01. [root@web1 ~] # mkdir /root5 //建立挂载点
02. [root@web1 ~] # mount 192.168.2.5:/root /root5 //挂载NFS共享目录
03. [root@web1 ~] # df -hT /root5 //确认挂载结果
04. Filesystem      Type  Size  Used Avail Use% Mounted on
05. 192.168.2.5:/root nfs   50G   15G   33G   31% /root5
06.
07. [root@web1 ~] # cd /root5 //切换到挂载点
08. [root@web1 root5] # echo "NFS Write Test" > test.txt //测试写入文件
09. [root@web1 root5] # cat test.txt //测试查看文件
10. NFS Write Test

```

将远程的NFS共享/usr/src挂载到本地的/mnt/nfsdir，并验证只读：

```

01. [root@web1 ~] # mkdir /mnt/nfsdir //建立挂载点
02. [root@web1 ~] # mount 192.168.2.5:/usr/src /mnt/nfsdir/ //挂载NFS共享目录
03. [root@web1 ~] # df -hT /mnt/nfsdir/ //确认挂载结果
04. Filesystem      Type  Size  Used Avail Use% Mounted on
05. 192.168.2.5:/usr/src nfs   50G   15G   33G   31% /mnt/nfsdir
06.
07. [root@web1 ~] # cd /mnt/nfsdir/ //切换到挂载点
08. [root@web1 nfsdir] # ls //读取目录列表
09. debug install.log kernels test.txt
10.
11. [root@web1 nfsdir] # echo "Write Test." > pc.txt //尝试写入文件失败
12. -bash: pc.txt: 只读文件系统

```

!!!! 如果从未授权的客户机访问NFS共享，将会被拒绝。比如从NFS服务器本机尝试访问自己发布的/root共享（只允许192.168.2.100访问），结果如下所示：

```

01. [root@proxy ~] # mkdir /root5
02. [root@proxy ~] # mount 192.168.2.5:/root /root5
03. mount.nfs: access denied by server while mounting 192.168.2.5:/root

```

[Top](#)

4) 设置永久挂载

```
01. [root@web1 ~]# vim /etc/fstab
02. ...
03. 192.168.2.5:/usr/src nfs default,ro 0 0
04. 192.168.2.5:/root root5 nfs default 0 0
```

4 案例4：编写udev规则

4.1 问题

编写udev规则，实现以下目标：

1. 当插入一个U盘时，该U盘自动出现一个链接称为udisk
2. U盘上的第1个分区名称为udisk1，以此类推
3. 终端上出现提示信息“udisk plugged in”

4.2 方案

问题：加载一个USB设备后，系统可能识别为sda也可能识别为sdb，能不能固定呢？

对于Linux kernel 2.6及更新的操作系统版本会将设备的相关信息动态写入/sys文件系统中，而udev程序可以通过读取这些设备信息，并根据自己的udev规则进行设备管理器，实现如下功能：

- 处理设备命名
- 决定要创建哪些设备文件或链接
- 决定如何设置属性
- 决定触发哪些事件

udev默认规则存放在/etc/udev/rules.d目录下，通过修改此目录下的规则实现设备的命名、属性、链接文件等。

4.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：编写udev规则

1) 准备USB设备（如果使用真实机演示，下面为虚拟机添加USB设备可以忽略）

使用virt-manager为虚拟机添加USB设备，如图-5所示。注意添加设备时一定要选择正确的USB设备，图-9仅是参考案例，每个人的USB品牌与型号都有可能不一样！

[Top](#)

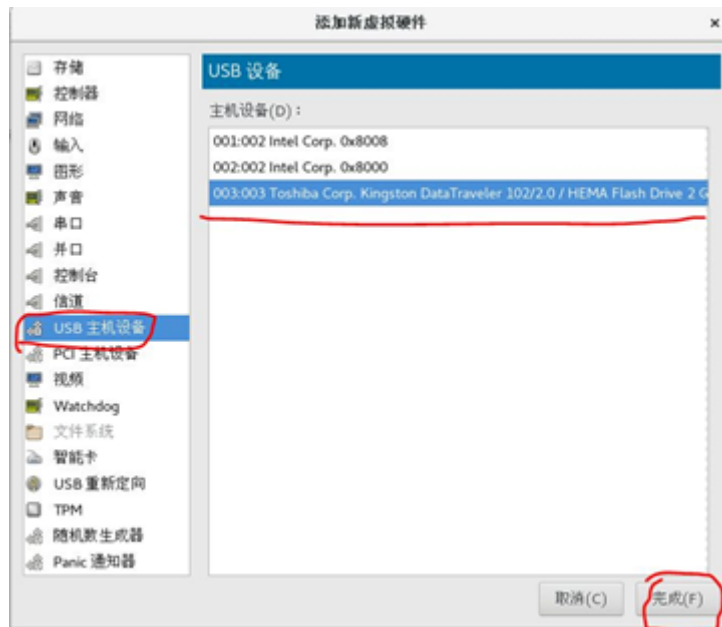


图-9

1) 查看设备属性

加载USB设备的同时实时查看设备的相关属性，可以使用monitor指令。

```
01. [ root@proxy ~] # udevadm monitor --property
```

如果设备已经加载则无法使用monitor查看相关属性。可以使用下面的命令查看设备属性。

```
01. [ root@proxy ~] # udevadm info --query=path --name=/dev/sda
```

```
02. [ root@proxy ~] # udevadm info --query=property --path=/block/sda
```

单独查看某个磁盘分区的属性信息。

```
01. [ root@proxy ~] # udevadm info --query=property --path=/block/sdada1
```

2) 编写udev规则文件 (实现插拔USB设备时有屏幕提示信息)

注意：修改规则文件不能照抄，这里的变量都是需要根据实际情况而修改的！！

每个设备的属性都有所不同！！一定要根据前面查询的info信息填写。

```
01. [ root@proxy ~] # vim /etc/udev/rules.d/70_usb.rules
```

```
02. SUBSYSTEMS=="usb", ENV{ID_VENDOR}=="TOSHIBA", ENV{serial}=="60A44CB4665EEE4133E
```

[Top](#)

在virt-manager中删除、添加USB设备，测试自己的udev规则是否成功。

排错方法：通过查看/var/log/messages日志文件排错。

3) 继续修改规则文件（实现给分区命名）

```
01. [ root@proxy ~] # udevadm info --query=property --path=/block/sdb/sdb1
02. [ root@proxy ~] # /etc/udev/rules.d/70-usb.rules
03. ACTION="add", ENV{ID_VENDOR}="TOSHIBA", ENV{DEVTYPE}="partition", ENV{ID_SERIAL
```

在virt-manager中删除、添加USB设备，测试自己的udev规则是否成功。

4) 继续修改规则文件（修改设备所有者和权限）

```
01. [ root@proxy ~] # /etc/udev/rules.d/70-usb.rules
02. ACTION="add", ENV{ID_VENDOR}="TOSHIBA", ENV{DEVTYPE}="partition", ENV{ID_SERIAL
```

在virt-manager中删除、添加USB设备，测试自己的udev规则是否成功。

5) 继续修改规则文件（插拔U盘等于启停服务）

注意：启动服务的程序systemctl，必须使用绝对路径。

```
01. [ root@proxy ~] # /etc/udev/rules.d/70-usb.rules
02. ACTION="add", ENV{ID_VENDOR}="TOSHIBA", ENV{ID_SERIAL_SHORT}="60A44CB4665E1
03. ACTION="remove", ENV{ID_VENDOR}="TOSHIBA", ENV{ID_SERIAL_SHORT}="60A44CB46
```

在virt-manager中删除、添加USB设备，测试自己的udev规则是否成功。

总结知识点：

udev规则文件，常见指令操作符如表-4所示。

表 - 4 udev常见指令操作符

[Top](#)

选项	值
==	表示匹配
!=	表示不匹配
=	指定赋予的值
+=	添加新值
:=	指定值，且不允许被替换
NAME="udisk"	定义设备名称
SYMLINK+="data1"	定义设备的别名
OWNER="student"	定义设备的所有者
GROUP="student"	定义设备的所属组
MODE="0600"	定义设备的权限
ACTION=="add"	判断设备的操作动作（添加或删除设备等）
KERNEL=="sd[a-z]1"	判断设备的内核名称
RUN+=程序	为设备添加程序

udev常用替代变量：

- %k：内核所识别出来的设备名，如sdb1
- %n：设备的内核编号，如sda3中的3
- %p：设备路径，如/sys/block/sdb/sdb1

[Top](#)