

NSD ENGINEER DAY03

1. [案例1：启用SELinux保护](#)
2. [案例2：自定义用户环境](#)
3. [案例3：配置IPv6地址](#)
4. [案例4：配置聚合连接](#)
5. [案例5：配置firewalld防火墙](#)

1 案例1：启用SELinux保护

1.1 问题

本例要求为虚拟机 server0、desktop0 配置SELinux：

1. 确保 SELinux 处于强制启用模式
2. 在每次重新开机后，此设置必须仍然有效

1.2 方案

SELinux，Security-Enhanced Linux：是由美国NSA国家安全局提供的一套基于内核的增强的强制安全保护机制，针对用户、进程、文档标记安全属性并实现保护性限制。

SELinux安全体系直接集成在Linux内核中，包括三种运行模式：

- disabled：彻底禁用，内核在启动时不加载SELinux安全体系
- enforcing：强制启用，内核加载SELinux安全体系，并强制执行保护策略
- permissive：宽松模式，内核加载SELinux安全体系，只记录不执行

执行getenforce可以查看当前所处的模式。

在disabled模式与enforcing、permissive模式之间切换时，需要重新启动Linux系统；而在enforcing模式与permissive模式之间切换时，并不需要重启，可以直接执行setenforce 1|0操作。

1.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：调整当前的SELinux运行模式

1) 查看当前模式

```
01. [root@server0 ~]# getenforce
02. Permissive //表示当前为宽松模式
```

若上述操作显示的结果为Disabled，表示SELinux机制已被禁用，只能通过步骤修改固定配置后再重启；若显示的结果为Enforcing，表示已经处于强制启用模式。

[Top](#)

2) 切换为enforcing强制启用模式

如果在操作1)中显示的结果为Permissive，则执行以下操作切换为强制启用：

```
01. [ root@server0 ~] # setenforce 1           //强制启用
02. [ root@server0 ~] # getenforce             //确认切换结果
03.      Enforcing
```

如果在操作1) 中显示的结果为Disabled，则无法使用setenforcing命令：

```
01. [ root@desktop0 ~] # getenforce
02.      Disabled
03. [ root@desktop0 ~] # setenforce 1
04.      setenforce: SELinux is disabled
```

步骤二：为SELinux运行模式建立固定配置

1) 修改配置文件/etc/selinux/config

```
01. [ root@server0 ~] # vim /etc/selinux/config
02.      SELINUX=enforcing
03.      ... ..
```

2) 重启验证结果

```
01. [ root@server0 ~] # reboot
02.      ... ..
03. [ root@server0 ~] # getenforce
04.      Enforcing
```

2 案例2：自定义用户环境

2.1 问题

本例要求为系统 server0 和 desktop0 创建自定义命令，相关说明如下：

1. 自定义命令的名称为 qstat
2. 此自定义命令将执行以下操作：/bin/ps -Ao pid,tt,user,fname,rsz
3. 此自定义命令对系统中的所有用户都有效

2.2 方案

[Top](#)

命令别名：为一个复杂的命令行建立一个更加简短的命令字，方便重复使用。

基本管理操作：

- 定义别名：alias 别名='复杂的命令行'
- 查看别名：alias、alias 别名
- 取消别名：unalias 别名、unalias -a

用户登录初始化文件：

- 全局配置：/etc/bashrc、
- 用户自定义配置：~/.bashrc

2.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：为主机server0添加别名qstat

1) 为所有用户添加初始化命令

```
01. [root@server0 ~]# vim /etc/bashrc
02. .. ..
03. alias qstat='/bin/ps -Ao pid,tt,user,fname,rsz'
```

2) 验证别名qstat是否生效

```
01. [root@server0 ~]# exit //退出
02. logout
03. Connection to server0 closed.
04. [kiosk@foundation0 ~]$ ssh -X root@server0 //重登录
05. Last login: Sat Nov 26 15:30:15 2016 from 172.25.0.250
06. [root@server0 ~]# alias qstat //可查到别名
07. alias qstat='/bin/ps -Ao pid,tt,user,fname,rsz'
08. [root@server0 ~]# qstat //且此别名正常可用
09. PID TT USER COMMAND RSZ
10. 1? root systemd 6548
11. 2? root kthreadd 0
12. 3? root ksoftirq 0
```

步骤二：为主机desktop0添加别名qstat

操作与步骤一相同。

3 案例3：配置IPv6地址

3.1 问题

[Top](#)

本例要求为两个虚拟机 server0、desktop0的接口 eth0 配置下列 IPv6 地址：

1. server0 上的地址应该是 2003:ac18::305/64
2. desktop0 上的地址应该是 2003:ac18::306/64
3. 两个系统必须能与网络 2003:ac18/64 内的系统通信
4. 地址必须在重启后依旧生效
5. 两个系统必须保持当前的IPv4地址并能通信

3.2 方案

如何表示一个IP地址：

- IPv4地址（32位）—— 点 分隔 十进制，比如172.25.0.11
- IPv6地址（128位）—— 冒号 分隔 十六进制，比如fe80::5054:ff:fe00:b。前置0可以省略，多个连续的冒号分隔可简写成两个（::）。

针对IPv6目标地址的连通性测试应使用ping6命令工具。

3.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：修改主机server0的网卡eth0的配置

1）确认网卡eth0所属的网络连接名（NAME）

```
01. [root@server0 ~]# nmcli connection show
02. NAME      UUID                                  TYPE      DEVICE
03. System eth0  5fb06bd0-0bb0-7ffb-45f1-d6edd65f3e03 802-3 ethernet eth0
```

2）修改此连接的IPv6地址配置

使用方法一（命令行）：

```
01. [root@server0 ~]# nmcli connection modify "System eth0" ipv6.method manual ipv6.a
```



或者，使用方法二（图形工具），运行nm-connection-editor，在打开的图形程序界面中双击连接名称System eth0，选择“IPv6 Settings”选项卡（如图-1所示）。

[Top](#)

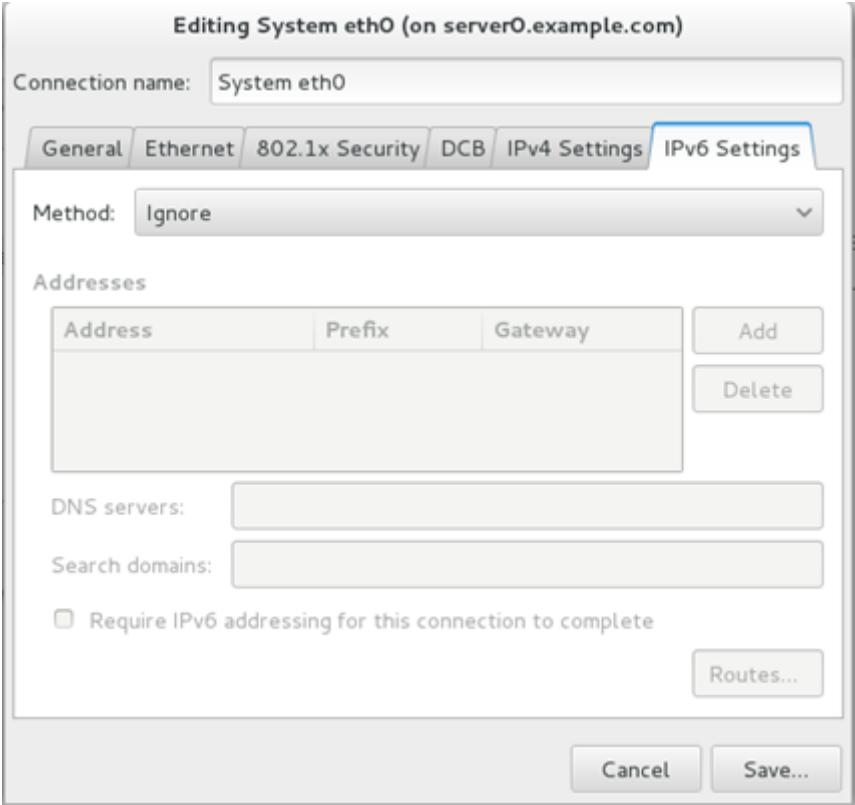


图-1

然后在“Method”处下拉选择“Manual”，再单击中间栏右侧的“Add”按钮添加指定的IPv6地址2003:ac18::305、掩码长度64，勾选底部的“Require IPv6 addressing for this connection to complete”（如图-2所示），最后单击右下角的“Save”保存，并关闭配置窗口。

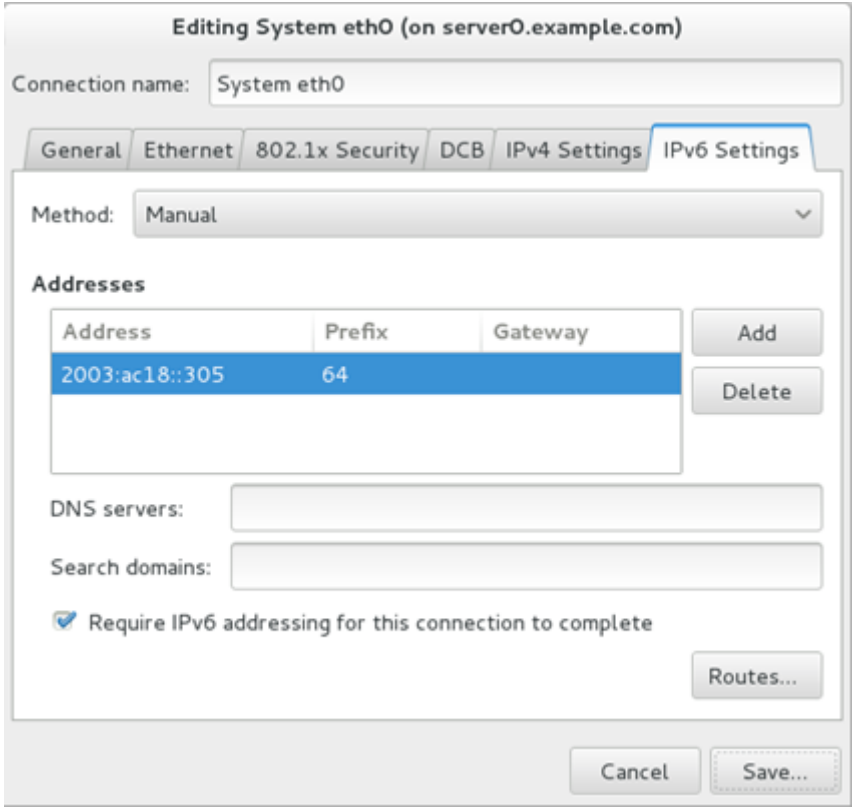


图-2

[Top](#)

3) 激活新配置

01. [root@server0 ~] # nmcli connection up "System eth0"
02. Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager

4) 确认地址已成功设置

执行ifconfig命令可以看到新增加的IPv6地址：

01. [root@server0 ~] # ifconfig eth0 | grep inet6
02. eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
03. inet 172.25.0.11 netmask 255.255.255.0 broadcast 172.25.0.255
04. inet6 2003:ac18::305 prefixlen 64 scopeid 0x0<global> //确认地址
05. inet6 fe80::5054:ff:fe00:b prefixlen 64 scopeid 0x20<link>
06. ether 52:54:00:00:00:0b txqueuelen 1000 (Ethernet)
07. RX packets 8697 bytes 5617496 (5.3 MB)
08. RX errors 0 dropped 0 overruns 0 frame 0
09. TX packets 6681 bytes 5803117 (5.5 MB)
10. TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

步骤二：修改主机desktop0的网卡eth0的配置

除了IPv6地址应使用2003:ac18::306以外，其他操作与步骤一相同。

步骤三：测试主机server0、desktop0之间的IPv6地址互连

在server0上，使用ping6命令测试desktop0的IPv6地址，可以正常连通：

01. [root@server0 ~] # ping6 2003:ac18::306
02. PING 2003:ac18::306(2003:ac18::306) 56 data bytes
03. 64 bytes from 2003:ac18::306: icmp_seq=1 ttl=64 time=0.656 ms
04. 64 bytes from 2003:ac18::306: icmp_seq=2 ttl=64 time=1.33 ms
05. 64 bytes from 2003:ac18::306: icmp_seq=3 ttl=64 time=1.29 ms
06. 64 bytes from 2003:ac18::306: icmp_seq=4 ttl=64 time=1.48 ms
07. ...

步骤四：确保配置有正确的静态主机名

避免重启后无法确定本机的主机名，容易引起混淆。

对于主机server0：

[Top](#)

01. [root@server0 ~] # hostnamectl set-hostname server0.example.com
02. [root@server0 ~] # hostnamectl

- 03. Static hostname: server0.example.com
- 04. Icon name: computer
- 05.

对于主机desktop0：

- 01. [root@desktop0 ~] # hostnamectl set-hostname desktop0.example.com
- 02. [root@desktop0 ~] # hostnamectl
- 03. Static hostname: desktop0.example.com
- 04. Icon name: computer
- 05.

4 案例4：配置聚合连接

4.1 问题

本例要求在两个虚拟机 server0、desktop0之间配置一个链路，要求如下：

1. 此链路使用接口 eth1 和 eth2
2. 此链路在其中一个接口失效时仍然能工作
3. 此链路在 server0 上使用下面的地址 172.16.3.20/255.255.255.0
4. 此链路在 desktop0 上使用下面的地址 172.16.3.25/255.255.255.0
5. 此链路在系统重启之后依然保持正常状态

4.2 方案

聚合连接（team）：指的是网络连接的捆绑/组队，通过将多个实际网卡（team-slave）整个为逻辑上的单个连接，实现负载均衡、热备份等单块网卡难以完成的特殊功能。

聚合连接的类型：热备份activebackup、轮询负载均衡roundrobin。

定义聚合连接的类型配置时，采用JSON语法标记，主要特点如下：

- 标记一个对象 —— { 对象 }
- 每一个对象 —— 名称:值
- 每一个字符串 —— "字符串"

热备份-聚合连接（activebackup）：

```
01    { "runner": { "name": "activebackup" } }
```

负载均衡-聚合连接（roundrobin）：

```
01    { "runner": { "name": "roundrobin" } }
```

[Top](#)

4.3 步骤

除了所配置的IP地址不一样以外，在server0、desktop0主机上的其他操作相同。此处仅列出在server0上的配置过程。

实现此案例需要按照如下步骤进行。

步骤一：准备练习用网卡环境

新建的聚合连接将组合新增加的两块网卡eth1、eth2。

```

01. [root@server0 ~]# ifconfig
02. eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
03.     inet 172.25.0.11 netmask 255.255.255.0 broadcast 172.25.0.255
04.     inet6 2003:ac18::305 prefixlen 64 scopeid 0x0<global>
05.     inet6 fe80::5054:ff:fe00:b prefixlen 64 scopeid 0x20<link>
06.     ether 52:54:00:00:00:0b txqueuelen 1000 (Ethernet)
07.     RX packets 172995 bytes 23870389 (22.7 MB)
08.     RX errors 0 dropped 0 overruns 0 frame 0
09.     TX packets 54053 bytes 34274222 (32.6 MB)
10.     TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
11.
12. eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
13.     ether 52:54:00:f8:86:c1 txqueuelen 1000 (Ethernet)
14.     RX packets 104217 bytes 5437855 (5.1 MB)
15.     RX errors 0 dropped 0 overruns 0 frame 0
16.     TX packets 171 bytes 17171 (16.7 KB)
17.     TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
18.
19. eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
20.     ether 52:54:00:38:79:d9 txqueuelen 1000 (Ethernet)
21.     RX packets 104118 bytes 5428927 (5.1 MB)
22.     RX errors 0 dropped 2060 overruns 0 frame 0
23.     TX packets 0 bytes 0 (0.0 B)
24.     TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
25.     ...

```

步骤二：创建聚合连接配置

1) 新建聚合连接

[Top](#)

```

01. [root@server0 ~]# nmcli con add con-name team0 type team ifname team0 config '
02. Connection 'team0' (8e61d730-50ff-4a7b-8ca0-fcf5955f6ea7) successfully added.

```


2) 配置IPv4地址

```
01. [root@server0 ~]# nmcli con modify team0 ipv4.method manual ipv4.addresses '172.
```

3) 新建聚合成员连接

```
01. [root@server0 ~]# nmcli con add con-name team0-p1 type team-slave ifname eth1
02. Connection 'team0-p1' (a62d23a2-9a2a-4855-8fbc-60ce1fd43f0b) successfully added.
03. [root@server0 ~]# nmcli con add con-name team0-p2 type team-slave ifname eth2
04. Connection 'team0-p2' (f4d4980e-8123-4840-89ac-1af148cc2eea) successfully added.
```

步骤三：激活聚合连接

1) 激活聚合连接

```
01. [root@server0 ~]# nmcli connection up team0
02. Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager
```

2) 激活聚合成员连接

```
01. [root@server0 ~]# nmcli connection up team0-p1
02. Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager
03. [root@server0 ~]# nmcli connection up team0-p2
04. Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager
```

步骤四：确认聚合连接状态

1) 查看聚合连接地址

```
01. [root@server0 ~]# ifconfig team0
02. team0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
03.     inet 172.16.3.20 netmask 255.255.255.0 broadcast 172.16.3.255
```

[Top](#)

- 04. inet6 fe80::c80d:eff:fe08:ca57 prefixlen 64 scopeid 0x20<link>
- 05. ether ca:0d:ef:08:ca:57 txqueuelen 0 (Ethernet)
- 06. RX packets 0 bytes 0 (0.0 B)
- 07. RX errors 0 dropped 36 overruns 0 frame 0
- 08. TX packets 68 bytes 8695 (8.4 KiB)
- 09. TX errors 0 dropped 1 overruns 0 carrier 0 collisions 0

2) 查看聚合连接运行状态

- 01. [root@server0 ~] # teamdctl team0 state
- 02. setup:
- 03. runner: activebackup //运行模式/类型
- 04. ports:
- 05. eth1 //成员网卡1
- 06. link watches:
- 07. link summary: up
- 08. instance[link_watch_0]:
- 09. name: ethtool
- 10. link: up
- 11. eth2 //成员网卡2
- 12. link watches:
- 13. link summary: up
- 14. instance[link_watch_0]:
- 15. name: ethtool
- 16. link: up
- 17. runner:
- 18. active port: eth1 //当前活动的成员网卡

5 案例5 : 配置firewalld防火墙

5.1 问题

本例要求为两个虚拟机 server0、desktop0配置防火墙策略：

1. 允许从172.25.0.0/24网段的客户机访问 server0、desktop0 的任何服务
2. 禁止从my133t.org域 (172.34.0.0/24网段) 的客户机访问 server0、desktop0 的任何服务
3. 在172.25.0.0/24网络中的系统，访问 server0 的本地端口5423将被转发到80
4. 上述设置必须永久有效

5.2 方案

[Top](#)

RHEL7的防火墙体系根据所在的网络场所区分，提供了预设的安全区域：

- public：仅允许访问本机的sshd等少数几个服务
- trusted：允许任何访问
- block：阻塞任何来访请求
- drop：丢弃任何来访的数据包
-

新增防火墙规则的位置包括：

- 运行时（runtime）：仅当前有效，重载防火墙后失效
- 永久（permanent）：静态配置，需要重载防火墙才能生效

本地端口转发（端口1 --> 端口2）：

- 从客户机访问防火墙主机的 端口1 时，与访问防火墙的 端口 2 时等效
- 真正的网络应用服务其实在 端口2 提供监听

5.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：采取“默认全允许，仅拒绝个别”的防护策略

1) 启用防火墙服务

```
01. [root@server0 ~]# systemctl restart firewalld
02. [root@server0 ~]# systemctl enable firewalld
```

2) 将默认区域设置为trusted

```
01. [root@server0 ~]# firewall-cmd --get-default-zone           //修改前
02. public
03. [root@server0 ~]# firewall-cmd --set-default-zone=trusted  //修改操作
04. success
05. [root@server0 ~]# firewall-cmd --get-default-zone           //修改后
06. trusted
```

步骤二：封锁指定的IP网段

1) 添加永久配置“阻塞来自网段172.34.0.0/24的任何访问”

```
01. [root@server0 ~]# firewall-cmd --permanent --zone=block --add-source=172.34.0.0/24
02. success
```

[Top](#)

2) 重载防火墙

```
01. [ root@server0 ~] # firewall-cmd --reload
02. success
```

3) 检查运行时规则

```
01. [ root@server0 ~] # firewall-cmd --list-all --zone=block
02. block
03. interfaces:
04. sources: 172.34.0.0/24
05. services:
06. ports:
07. masquerade: no
08. forward-ports:
09. icmp-blocks:
10. rich rules:
```

步骤三：实现5423-->80端口转发

1) 针对80端口部署测试应用

快速搭建一个测试网站：

```
01. [ root@server0 ~] # yum -y install httpd //装包
02. ...
03. [ root@server0 ~] # vim /var/www/html/index.html //部署测试网页
04. test site.
05. [ root@server0 ~] # systemctl restart httpd //起服务
```

从客户端访问，确认测试网页：

```
01. [ root@desktop0 ~] # yum -y install elinks
02. ...
03. [ root@desktop0 ~] # elinks -dump http://server0.example.com/
04. test site.
```

2) 配置5423-->80端口转发策略

[Top](#)

```

01. [ root@server0 ~] # firewall-cmd --permanent --zone=trusted --add-forward-port=por
02. success
03. [ root@server0 ~] # firewall-cmd --reload //重载服务
04. Success
05. [ root@server0 ~] # firewall-cmd --list-all //确认运行时规则
06. trusted ( default, active)
07. interfaces: eth1 eth2 eth0 team0
08. sources:
09. services:
10. ports:
11. masquerade: no
12. forward-ports: port=5423:proto=tcp:toport=80:toaddr=
13. icmp-blocks:
14. rich rules:

```

3) 验证端口转发策略

从desktop0上访问server0的5423端口，与访问server0的80端口效果一样：

```

01. [ root@desktop0 ~] # elinks -dump http://server0.example.com:5423/
02. test site.
03. [ root@desktop0 ~] # elinks -dump http://server0.example.com/
04. test site.

```

[Top](#)