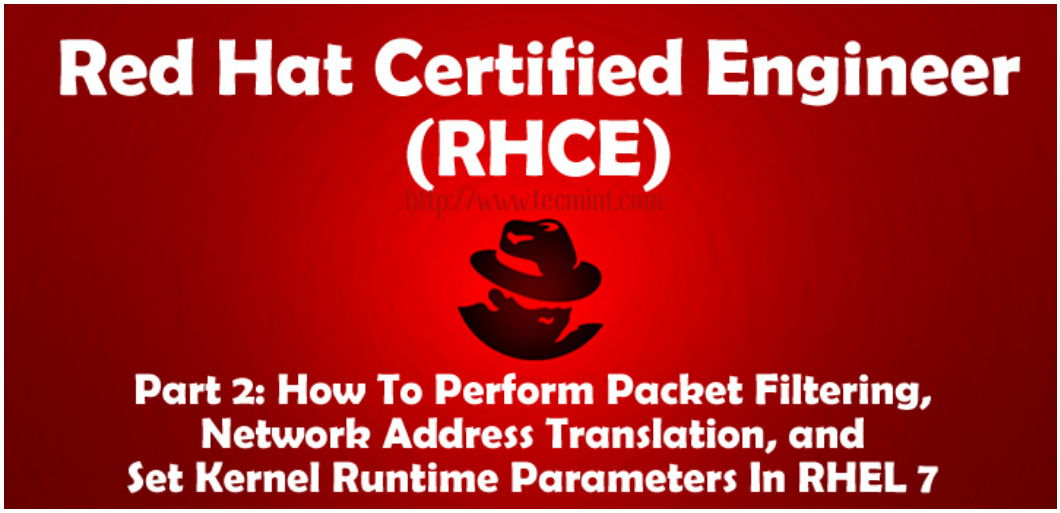
RHCE 系列（二）：如何进行包过滤、网络地址转换和设置内核运行时参数

2015-10-23 10:24 评论: 3 收藏: 5

参考原文：http://www.tecmint.com/perfor... 作者： Gabriel Cánepa
编译文章：LCTT https://linux.cn/article-6458-1.html 译者： ictlyh

正如第一部分（“设置静态网络路由 <https://linux.cn/article-6451-1.html> ”）提到的，在这篇文章（RHCE 系列第二部分），我们首先介绍红帽企业版 Linux 7（RHEL）中包过滤和网络地址转换（NAT）的原理，然后再介绍在某些条件发生变化或者需要变动时设置运行时内核参数以改变运行时内核行为。



RHCE 第二部分：网络包过滤

RHEL 7 中的网络包过滤

当我们讨论数据包过滤的时候，我们指防火墙读取每个试图通过它的数据包的包头所进行的处理。然后，根据系统管理员之前定义的规则，通过采取所要求的动作过滤数据包。

正如你可能知道的，从 RHEL 7 开始，管理防火墙的默认服务是 firewalld <http://www.tecmint.com/firewalld-rules-for-centos-7/> 。类似 iptables，它和 Linux 内核的 netfilter 模块交互以便检查和操作网络数据包。但不像 iptables，Firewalld 的更新可以立即生效，而不用中断活跃的连接 - 你甚至不需要重启服务。

Firewalld 的另一个优势是它允许我们定义基于预配置服务名称的规则（之后会详细介绍）。

在第一部分，我们用了下面的场景：

静态路由网络示意图

然而，你应该记得，由于还没有介绍包过滤，为了简化例子，我们停用了2号路由器的防火墙。现在让我们来看看如何使接收的数据包发送到目的地的特定服务或端口。

首先，让我们添加一条永久规则允许从 enp0s3 (192.168.0.19) 到 enp0s8 (10.0.0.18) 的入站流量：

```
# firewall-cmd --permanent --direct --add-rule ipv4 filter FORWARD 0 -i enp0s3 -o enp0s8 -j ACCEPT
```

上面的命令会把规则保存到 /etc/firewalld/direct.xml 中：

```
# cat /etc/firewalld/direct.xml
```

```
[root@router2 ~]# cat /etc/firewalld/direct.xml
<?xml version="1.0" encoding="utf-8"?>
<direct>
  <rule priority="0" table="filter" ipv="ipv4" chain="FORWARD">-i enp0s3 -o enp0s8 -j ACCEPT</rule>
</direct>
[root@router2 ~]#
```

检查 Firewalld 保存的规则

然后启用规则使其立即生效：

```
# firewall-cmd --direct --add-rule ipv4 filter FORWARD 0 -i enp0s3 -o enp0s8 -j ACCEPT
```

现在你可以从 RHEL 7 中通过 telnet 到 web 服务器并再次运行 tcpdump <http://www.tecmint.com/12-tcpdump-commands-a-network-sniffer-tool/> 监视两台机器之间的 TCP 流量，这次2号路由器已经启用了防火墙。

```
# telnet 10.0.0.20 80
# tcpdump -qnnvvv -i enp0s3 host 10.0.0.20
```

如果你想只允许从 192.168.0.18 到 web 服务器（80 号端口）的连接而阻塞 192.168.0.0/24 网络中的其它来源呢？

在 web 服务器的防火墙中添加以下规则：

```
# firewall-cmd --add-rich-rule 'rule family="ipv4" source
address="192.168.0.18/24" service name="http" accept'
# firewall-cmd --add-rich-rule 'rule family="ipv4" source
address="192.168.0.18/24" service name="http" accept' --permanent
# firewall-cmd --add-rich-rule 'rule family="ipv4" source
address="192.168.0.0/24" service name="http" drop'
# firewall-cmd --add-rich-rule 'rule family="ipv4" source
address="192.168.0.0/24" service name="http" drop' --permanent
```

现在你可以从 192.168.0.18 和 192.168.0.0/24 中的其它机器发送到 web 服务器的 HTTP 请求。第一种情况连接会成功完成，但第二种情况最终会超时。

任何下面的命令可以验证这个结果：

```
# telnet 10.0.0.20 80
# wget 10.0.0.20
```

我强烈建议你看看 Fedora Project Wiki 中的 Firewalld Rich Language <https://fedoraproject.org/wiki/Features/FirewalldRichLanguage> 文档更详细地了解关于富规则的内容。

RHEL 7 中的网络地址转换（NAT）

网络地址转换（NAT）是为专用网络中的一组计算机（也可能是其中的一台）分配一个独立的公共 IP 地址的过程。这样，在内部网络中仍然可以用它们自己的私有 IP 地址来区别，但外部“看来”它们是一样的。

另外，网络地址转换使得内部网络中的计算机发送请求到外部资源（例如因特网），然后只有源系统能接收到对应的响应成为可能。

现在让我们考虑下面的场景：

网络地址转换

在2号路由器中，我们会把 enp0s3 接口移动到外部区域（external），enp0s8 到内部区域（external），伪装（masquerading）或者说 NAT 默认是启用的：

```
# firewall-cmd --list-all --zone=external
# firewall-cmd --change-interface=enp0s3 --zone=external
# firewall-cmd --change-interface=enp0s3 --zone=external --permanent
# firewall-cmd --change-interface=enp0s8 --zone=internal
# firewall-cmd --change-interface=enp0s8 --zone=internal --permanent
```

对于我们当前的设置，内部区域（internal）- 以及和它一起启用的任何东西都是默认区域：

```
# firewall-cmd --set-default-zone=internal
```

下一步，让我们重载防火墙规则并保持状态信息：

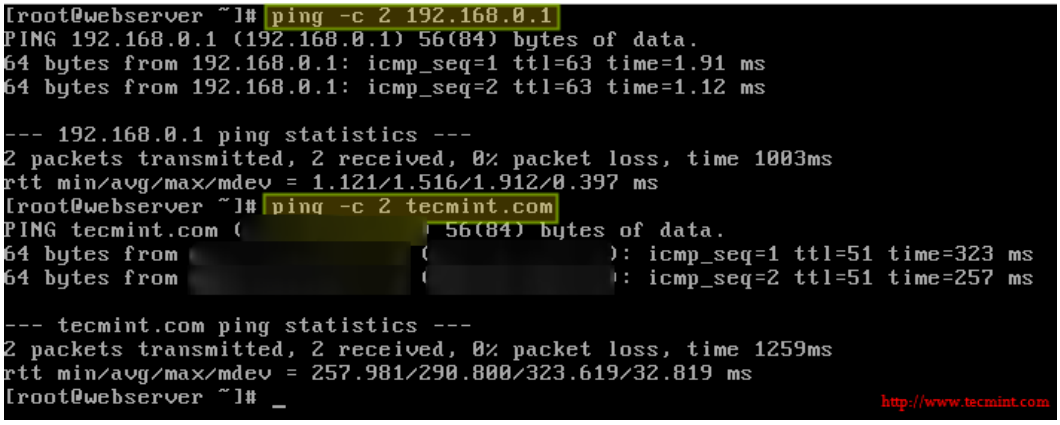
```
# firewall-cmd --reload
```

最后，在 web 服务器中添加2号路由器为默认网关：

```
# ip route add default via 10.0.0.18
```

现在你会发现在 web 服务器中你可以 ping 1号路由器和外部网站（例如 tecmint.com）：

```
# ping -c 2 192.168.0.1
# ping -c 2 tecmint.com
```



验证网络路由

在 RHEL 7 中设置内核运行时参数

在 Linux 中，允许你更改、启用以及停用内核运行时参数，RHEL 也不例外。当操作条件发生变化时， `/proc/sys` 接口（sysctl）允许你实时设置运行时参数改变系统行为，而不需太多麻烦。

为了实现这个目的，会用 shell 内建的 echo 写 `/proc/sys/<category>` 中的文件，其中 `<category>` 一般是以下目录中的一个：

- dev: 连接到机器中的特定设备的参数。
- fs: 文件系统配置（例如 quotas 和 inodes）。
- kernel: 内核配置。
- net: 网络配置。
- vm: 内核的虚拟内存的使用。

要显示所有当前可用值的列表，运行

```
# sysctl -a | less
```

在第一部分中，我们通过以下命令改变了 `net.ipv4.ip_forward` 参数的值以允许 Linux 机器作为一个路由器。

```
# echo 1 > /proc/sys/net/ipv4/ip_forward
```

另一个你可能想要设置的运行时参数是 `kernel.sysrq` ，它会启用你键盘上的 `Sysrq` 键，以使系统更好的运行一些底层功能，例如如果由于某些原因冻结了后重启系统：

```
# echo 1 > /proc/sys/kernel/sysrq
```

要显示特定参数的值，可以按照下面方式使用 `sysctl` ：

```
# sysctl <parameter.name>
```

例如，

```
# sysctl net.ipv4.ip_forward
# sysctl kernel.sysrq
```

有些参数，例如上面提到的某个，只需要一个值，而其它一些（例如 `fs.inode-state` ）要求多个值：

```
[root@rhel7 ~]# sysctl net.ipv4.ip_forward
net.ipv4.ip_forward = 0
[root@rhel7 ~]# sysctl kernel.sysrq
kernel.sysrq = 16
[root@rhel7 ~]# sysctl fs.inode-state
fs.inode-state = 17099  329     0       0       0       0       0
[root@rhel7 ~]#
```


查看内核参数

不管什么情况下，做任何更改之前你都需要阅读内核文档。

请注意系统重启后这些设置会丢失。要使这些更改永久生效，我们需要添加内容到 `/etc/sysctl.d` 目录的 .conf 文件，像下面这样：

```
# echo "net.ipv4.ip_forward = 1" > /etc/sysctl.d/10-forward.conf
```

（其中数字 10 表示相对同一个目录中其它文件的处理顺序）。

并用下面命令启用更改：

```
# sysctl -p /etc/sysctl.d/10-forward.conf
```

总结

在这篇指南中我们解释了基本的包过滤、网络地址变换和在运行的系统中设置内核运行时

参数并使重启后能持久化。我希望这些信息能对你有用，如往常一样，我们期望收到你的回复！

别犹豫，在下面的表单中和我们分享你的疑问、评论和建议吧。

---

via: http://www.tecmint.com/perform-packet-filtering-network-address-translation-and-set-kernel-runtime-parameters-in-rhel/ <http://www.tecmint.com/perform-packet-filtering-network-address-translation-and-set-kernel-runtime-parameters-in-rhel/>

作者：Gabriel Cánepa <http://www.tecmint.com/author/gacanepa/> 译者：ictlyh <https://github.com/ictlyh> 校对：wxy <https://github.com/wxy>

本文由 LCTT <https://github.com/LCTT/TranslateProject> 原创翻译，Linux中国 <https://linux.cn/> 荣誉推出