CentOS7下Firewall防火墙配置用法详解(推荐)

转载 2016-12-13 作者：111cn.net 我要评论

> centos 7中防火墙是一个非常的强大的功能了，这篇文章主要介绍了CentOS7下Firewall防火墙配置用法详解(推荐)，小编觉得挺不错的，现在分享给大家，也给大家做个参考。

centos 7中防火墙是一个非常的强大的功能了，但对于centos 7中在防火墙中进行了升级了，下面我们一起来详细的看看关于centos 7中防火墙使用方法。

FirewallD 提供了支持网络/防火墙区域(zone)定义网络链接以及接口安全等级的动态防火墙管理工具。它支持 IPv4, IPv6 防火墙设置以及以太网桥接，并且拥有运行时配置和永久配置选项。它也支持允许服务或者应用程序直接添加防火墙规则的接口。 以前的 system-config-firewall/lokkit 防火墙模型是静态的，每次修改都要求防火墙完全重启。这个过程包括内核 netfilter 防火墙模块的卸载和新配置所需模块的装载等。而模块的卸载将会破坏状态防火墙和确立的连接。

相反，firewall daemon 动态管理防火墙，不需要重启整个防火墙便可应用更改。因而也就没有必要重载所有内核防火墙模块了。不过，要使用 firewall daemon 就要求防火墙的所有变更都要通过该守护进程来实现，以确保守护进程中的状态和内核里的防火墙是一致的。另外，firewall daemon 无法解析由 ip*tables 和 ebtables 命令行工具添加的防火墙规则。

守护进程通过 D-BUS 提供当前激活的防火墙设置信息，也通过 D-BUS 接受使用 PolicyKit 认证方式做的更改。

"守护进程"

应用程序、守护进程和用户可以通过 D-BUS 请求启用一个防火墙特性。特性可以是预定义的防火墙功能，如：服务、端口和协议的组合、端口/数据报转发、伪装、ICMP 拦截或自定义规则等。该功能可以启用确定的一段时间也可以再次停用。

通过所谓的直接接口，其他的服务(例如 libvirt)能够通过 iptables 变元(arguments)和参数(parameters)增加自己的规则。

amanda 、ftp 、samba 和 tftp 服务的 netfilter 防火墙助手也被"守护进程"解决了,只要它们还作为预定义服务的一部分。附加助手的装载不作为当前接口的一部分。由于一些助手只有在由模块控制的所有连接都关闭后才可装载。

因而，跟踪连接信息很重要，需要列入考虑范围。

静态防火墙(system-config-firewall/lokkit)

使用 system-config-firewall 和 lokkit 的静态防火墙模型实际上仍然可用并将继续提供，但却不能与"守护进程"同时使用。用户或者管理员可以决定使用哪一种方案。

在软件安装，初次启动或者是首次联网时，将会出现一个选择器。通过它你可以选择要使用的防火墙方案。其他的解决方案将保持完整，可以通过更换模式启用。

firewall daemon 独立于 system-config-firewall，但二者不能同时使用。

使用iptables和ip6tables的静态防火墙规则

如果你想使用自己的 iptables 和 ip6tables 静态防火墙规则, 那么请安装 iptables-services 并且禁用 firewalld ， 启用 iptables 和ip6tables:

```
yum install iptables-services
systemctl mask firewalld.service
systemctl enable iptables.service
systemctl enable ip6tables.service
```

静态防火墙规则配置文件是 /etc/sysconfig/iptables 以及 /etc/sysconfig/ip6tables .

注： iptables 与 iptables-services 软件包不提供与服务配套使用的防火墙规则. 这些服务是用来保障兼容性以及供想使用自己防火墙规则的人使用的. 你可以安装并使用 system-config-firewall 来创建上述服务需要的规则. 为了能使用 system-config-firewall, 你必须停止 firewalld.

为服务创建规则并停用 firewalld 后，就可以启用 iptables 与 ip6tables 服务了:

```
systemctl stop firewalld.service
systemctl start iptables.service
systemctl start ip6tables.service
```

什么是区域?

网络区域定义了网络连接的可信等级。这是一个一对多的关系，这意味着一次连接可以仅仅是一个区域的一部分，而一个区域可以用于很多连接。

预定义的服务

服务是端口和/或协议入口的组合。备选内容包括 netfilter 助手模块以及 IPv4、IPv6地址。

端口和协议

定义了 tcp 或 udp 端口，端口可以是一个端口或者端口范围。

ICMP阻塞

可以选择 Internet 控制报文协议的报文。这些报文可以是信息请求亦可是对信息请求或错误条件创建的响应。

伪装

私有网络地址可以被映射到公开的IP地址。这是一次正规的地址转换。

端口转发

端口可以映射到另一个端口以及/或者其他主机。

哪个区域可用?

由firewalld 提供的区域按照从不信任到信任的顺序排序。

丢弃

任何流入网络的包都被丢弃，不作出任何响应。只允许流出的网络连接。

阻塞

任何进入的网络连接都被拒绝，并返回 IPv4 的 icmp-host-prohibited 报文或者 IPv6 的 icmp6-adm-prohibited 报文。只允许由该系统初始化的网络连接。

公开

用以可以公开的部分。你认为网络中其他的计算机不可信并且可能伤害你的计算机。只允许选中的连接接入。（You do not trust the other computers on networks to not harm your computer. Only selected incoming connections are accepted.）

外部

用在路由器等启用伪装的外部网络。你认为网络中其他的计算机不可信并且可能伤害你的计算机。只允许选中的连接接入。

隔离区（dmz)

用以允许隔离区（dmz）中的电脑有限地被外界网络访问。只接受被选中的连接。

工作

用在工作网络。你信任网络中的大多数计算机不会影响你的计算机。只接受被选中的连接。

家庭

用在家庭网络。你信任网络中的大多数计算机不会影响你的计算机。只接受被选中的连接。

内部

用在内部网络。你信任网络中的大多数计算机不会影响你的计算机。只接受被选中的连接。

受信任的

允许所有网络连接。

我应该选用哪个区域?

例如，公共的 WIFI 连接应该主要为不受信任的，家庭的有线网络应该是相当可信任的。根据与你使用的网络最符合的区域进行选择。

如何配置或者增加区域?

你可以使用任何一种 firewalld 配置工具来配置或者增加区域，以及修改配置。工具有例如 firewall-config 这样的图形界面工具， firewall-cmd 这样的命令行工具，以及D-BUS接口。或者你也可以在配置文件目录中创建或者拷贝区域文件。 @PREFIX@/lib/firewalld/zones 被用于默认和备用配置，/etc/firewalld/zones 被用于用户创建和自定义配置文件。

如何为网络连接设置或者修改区域

区域设置以 ZONE= 选项 存储在网络连接的ifcfg文件中。如果这个选项缺失或者为空，firewalld 将使用配置的默认区域。

如果这个连接受到 NetworkManager 控制，你也可以使用 nm-connection-editor 来修改区域。

由NetworkManager控制的网络连接

防火墙不能够通过 NetworkManager 显示的名称来配置网络连接，只能配置网络接口。因此在网络连接之前 NetworkManager 将配置文件所述连接对应的网络接口告诉 firewalld 。如果在配置文件中没有配置区域，接口将配置到 firewalld 的默认区域。如果网络连接使用了不止一个接口，所有的接口都会应用到 fiwewalld。接口名称的改变也将由 NetworkManager 控制并应用到firewalld。

为了简化，自此，网络连接将被用作与区域的关系。

如果一个接口断开了，NetworkManager也将告诉firewalld从区域中删除该接口。

当firewalld由systemd或者init脚本启动或者重启后，firewalld将通知NetworkManager把网络连接增加到区域。

由脚本控制的网络

对于由网络脚本控制的连接有一条限制：没有守护进程通知 firewalld 将连接增加到区域。这项工作仅在 ifcfg-post 脚本进行。因此，此后对网络连接的重命名将不能被应用到firewalld。同样，在连接活动时重启 firewalld 将导致与其失去关联。现在有意修复此情况。最简单的是将全部未配置连接加入默认区域。

区域定义了本区域中防火墙的特性：

使用firewalld

你可以通过图形界面工具 firewall-config 或者命令行客户端 firewall-cmd 启用或者关闭防火墙特性。

使用firewall-cmd

命令行工具 firewall-cmd 支持全部防火墙特性。对于状态和查询模式，命令只返回状态，没有其他输出。

一般应用

获取 firewalld 状态

```
firewall-cmd --state
```

此举返回 firewalld 的状态，没有任何输出。可以使用以下方式获得状态输出：

```
firewall-cmd --state && echo "Running" || echo "Not running"
```

在 Fedora 19 中, 状态输出比此前直观:

```
# rpm -qf $( which firewall-cmd )
firewalld-0.3.3-2.fc19.noarch # firewall-cmd --state
not running
```

在不改变状态的条件下重新加载防火墙：

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

如果你使用–complete-reload，状态信息将会丢失。这个选项应当仅用于处理防火墙问题时，例如，状态信息和防火墙规则都正常，但是不能建立任何连接的情况。

获取支持的区域列表

```
firewall-cmd --get-zones
```

这条命令输出用空格分隔的列表。

获取所有支持的服务

```
firewall-cmd --get-services
```

这条命令输出用空格分隔的列表。

获取所有支持的ICMP类型

```
firewall-cmd --get-icmptypes
```

这条命令输出用空格分隔的列表。

列出全部启用的区域的特性

```
firewall-cmd --list-all-zones
```

输出格式是：

```
<zone>
  interfaces: <interface1> ..
  services: <service1> ..
  ports: <port1> ..
  forward-ports: <forward port1> ..
  icmp-blocks: <icmp type1> ....
```

输出区域 <zone> 全部启用的特性。如果生略区域，将显示默认区域的信息。

```
firewall-cmd [--zone=<zone>] --list-all
```

获取默认区域的网络设置

```
firewall-cmd --get-default-zone
```

设置默认区域

```
firewall-cmd --set-default-zone=<zone>
```

流入默认区域中配置的接口的新访问请求将被置入新的默认区域。当前活动的连接将不受影响。

获取活动的区域

```
firewall-cmd --get-active-zones
```

这条命令将用以下格式输出每个区域所含接口：

<zone1>: <interface1> <interface2> ..<zone2>: <interface3> ..

根据接口获取区域

```
firewall-cmd --get-zone-of-interface=<interface>
```

这条命令将输出接口所属的区域名称。

将接口增加到区域

```
firewall-cmd [--zone=<zone>] --add-interface=<interface>
```

如果接口不属于区域，接口将被增加到区域。如果区域被省略了，将使用默认区域。接口在重新加载后将重新应用。

修改接口所属区域

```
firewall-cmd [--zone=<zone>] --change-interface=<interface>
```

这个选项与 –add-interface 选项相似，但是当接口已经存在于另一个区域的时候，该接口将被添加到新的区域。

从区域中删除一个接口

```
firewall-cmd [--zone=<zone>] --remove-interface=<interface>
```

查询区域中是否包含某接口

```
firewall-cmd [--zone=<zone>] --query-interface=<interface>
```

返回接口是否存在于该区域。没有输出。

列举区域中启用的服务

```
firewall-cmd [ --zone=<zone> ] --list-services
```

启用应急模式阻断所有网络连接，以防出现紧急状况

```
firewall-cmd --panic-on
```

禁用应急模式

```
firewall-cmd --panic-off
```

应急模式在 0.3.0 版本中发生了变化

在 0.3.0 之前的 FirewallD版本中, panic 选项是 –enable-panic 与 –disable-panic.

查询应急模式

```
firewall-cmd --query-panic
```

此命令返回应急模式的状态，没有输出。可以使用以下方式获得状态输出：

```
firewall-cmd --query-panic && echo "On" || echo "Off"
```

处理运行时区域

运行时模式下对区域进行的修改不是永久有效的。重新加载或者重启后修改将失效。

启用区域中的一种服务

```
firewall-cmd [--zone=<zone>] --add-service=<service> [--timeout=<second
```

此举启用区域中的一种服务。如果未指定区域，将使用默认区域。如果设定了超时时间，服务将只启用特定秒数。如果服务已经活跃，将不会有任何警告信息。

例: 使区域中的ipp-client服务生效60秒:

```
firewall-cmd --zone=home --add-service=ipp-client --timeout=60
```

例: 启用默认区域中的http服务:

```
firewall-cmd --add-service=http
```

禁用区域中的某种服务

```
firewall-cmd [--zone=<zone>] --remove-service=<service>
```

此举禁用区域中的某种服务。如果未指定区域，将使用默认区域。

例: 禁止home区域中的http服务:

```
firewall-cmd --zone=home --remove-service=http
```

区域种的服务将被禁用。如果服务没有启用，将不会有任何警告信息。

查询区域中是否启用了特定服务

```
firewall-cmd [--zone=<zone>] --query-service=<service>
```

如果服务启用，将返回1,否则返回0。没有输出信息。

启用区域端口和协议组合

```
firewall-cmd [--zone=<zone>] --add-port=<port>[-<port>]/<protocol> [--t
```

此举将启用端口和协议的组合。端口可以是一个单独的端口 <port> 或者是一个端口范围 <port>-<port> 。协议可以是 tcp 或 udp。

禁用端口和协议组合

```
firewall-cmd [--zone=<zone>] --remove-port=<port>[-<port>]/<protocol>
```

查询区域中是否启用了端口和协议组合

```
firewall-cmd [--zone=<zone>] --query-port=<port>[-<port>]/<protocol>
```

如果启用，此命令将有返回值。没有输出信息。

启用区域中的IP伪装功能

```
firewall-cmd [--zone=<zone>] --add-masquerade
```

此举启用区域的伪装功能。私有网络的地址将被隐藏并映射到一个公有IP。这是地址转换的一种形式，常用于路由。由于内核的限制，伪装功能仅可用于IPv4。

禁用区域中的IP伪装

```
firewall-cmd [--zone=<zone>] --remove-masquerade
```

查询区域的伪装状态

```
firewall-cmd [--zone=<zone>] --query-masquerade
```

如果启用，此命令将有返回值。没有输出信息。

启用区域的ICMP阻塞功能

```
firewall-cmd [--zone=<zone>] --add-icmp-block=<icmptype>
```

此举将启用选中的Internet控制报文协议（ICMP）报文进行阻塞。ICMP报文可以是请求信息或者创建的应答报文，以及错误应答。

禁止区域的ICMP阻塞功能

```
firewall-cmd [--zone=<zone>] --remove-icmp-block=<icmptype>
```

查询区域的ICMP阻塞功能

```
firewall-cmd [--zone=<zone>] --query-icmp-block=<icmptype>
```

如果启用，此命令将有返回值。没有输出信息。

例: 阻塞区域的响应应答报文:

```
firewall-cmd --zone=public --add-icmp-block= echo -reply
```

在区域中启用端口转发或映射

```
firewall-cmd [--zone=<zone>] --add-forward-port=port=<port>[-<port>]:pr
```

端口可以映射到另一台主机的同一端口，也可以是同一主机或另一主机的不同端口。端口号可以是一个单独的端口 <port> 或者是端口范围 <port>-<port> 。协议可以为 tcp 或udp 。目标端口可以是端口号 <port> 或者是端口范围 <port>-<port> 。目标地址可以是 IPv4 地址。受内核限制，端口转发功能仅可用于IPv4。

禁止区域的端口转发或者端口映射

```
firewall-cmd [--zone=<zone>] --remove-forward-port=port=<port>[-<port>]
```

查询区域的端口转发或者端口映射

```
firewall-cmd [--zone=<zone>] --query-forward-port=port=<port>[-<port>]:
```

如果启用，此命令将有返回值。没有输出信息。

例: 将区域home的ssh转发到127.0.0.2

```
firewall-cmd --zone=home --add-forward-port=port=22:proto=tcp:toaddr=12
```

处理永久区域

永久选项不直接影响运行时的状态。这些选项仅在重载或者重启服务时可用。为了使用运行时和永久设置，需要分别设置两者。 选项 –permanent 需要是永久设置的第一个参数。

获取永久选项所支持的服务

```
firewall-cmd --permanent --get-services
```

获取永久选项所支持的ICMP类型列表

```
firewall-cmd --permanent --get-icmptypes
```

获取支持的永久区域

```
firewall-cmd --permanent --get-zones
```

启用区域中的服务

```
firewall-cmd --permanent [--zone=<zone>] --add-service=<service>
```

此举将永久启用区域中的服务。如果未指定区域，将使用默认区域。

禁用区域中的一种服务

```
firewall-cmd --permanent [--zone=<zone>] --remove-service=<service>
```

查询区域中的服务是否启用

```
firewall-cmd --permanent [--zone=<zone>] --query-service=<service>
```

如果服务启用，此命令将有返回值。此命令没有输出信息。

例: 永久启用 home 区域中的 ipp-client 服务

```
firewall-cmd --permanent --zone=home --add-service=ipp-client
```

永久启用区域中的一个端口-协议组合

```
firewall-cmd --permanent [--zone=<zone>] --add-port=<port>[-<port>]/<pr
```

永久禁用区域中的一个端口-协议组合

```
firewall-cmd --permanent [--zone=<zone>] --remove-port=<port>[-<port>]/
```

查询区域中的端口-协议组合是否永久启用

```
firewall-cmd --permanent [--zone=<zone>] --query-port=<port>[-<port>]/<
```

如果服务启用，此命令将有返回值。此命令没有输出信息。

例: 永久启用 home 区域中的 https (tcp 443) 端口

```
firewall-cmd --permanent --zone=home --add-port=443/tcp
```

永久启用区域中的伪装

```
firewall-cmd --permanent [--zone=<zone>] --add-masquerade
```

此举启用区域的伪装功能。私有网络的地址将被隐藏并映射到一个公有IP。这是地址转换的一种形式，常用于路由。由于内核的限制，伪装功能仅可用于IPv4。

永久禁用区域中的伪装

```
firewall-cmd --permanent [--zone=<zone>] --remove-masquerade
```

查询区域中的伪装的永久状态

```
firewall-cmd --permanent [--zone=<zone>] --query-masquerade
```

如果服务启用，此命令将有返回值。此命令没有输出信息。

永久启用区域中的ICMP阻塞

```
firewall-cmd --permanent [--zone=<zone>] --add-icmp-block=<icmptype>
```

此举将启用选中的 Internet 控制报文协议 （ICMP） 报文进行阻塞。 ICMP 报文可以是请求信息或者创建的应答报文或错误应答报文。

永久禁用区域中的ICMP阻塞

```
firewall-cmd --permanent [--zone=<zone>] --remove-icmp-block=<icmptype>
```

查询区域中的ICMP永久状态

```
firewall-cmd --permanent [--zone=<zone>] --query-icmp-block=<icmptype>
```

如果服务启用，此命令将有返回值。此命令没有输出信息。

例: 阻塞公共区域中的响应应答报文:

```
firewall-cmd --permanent --zone=public --add-icmp-block=echo -reply
```

在区域中永久启用端口转发或映射

```
firewall-cmd --permanent [--zone=<zone>] --add-forward-port=port=<port>
```

端口可以映射到另一台主机的同一端口，也可以是同一主机或另一主机的不同端口。端口号可以是一个单独的端口 <port> 或者是端口范围 <port>-<port> 。协议可以为 tcp 或udp 。目标端口可以是端口号 <port> 或者是端口范围 <port>-<port> 。目标地址可以是 IPv4 地址。受内核限制，端口转发功能仅可用于IPv4。

永久禁止区域的端口转发或者端口映射

```
firewall-cmd --permanent [--zone=<zone>] --remove-forward-port=port=<po
```

查询区域的端口转发或者端口映射状态

```
firewall-cmd --permanent [--zone=<zone>] --query-forward-port=port=<por
```

如果服务启用，此命令将有返回值。此命令没有输出信息。

例: 将 home 区域的 ssh 服务转发到 127.0.0.2

```
firewall-cmd --permanent --zone=home --add-forward-port=port=22:proto=t
```

直接选项

直接选项主要用于使服务和应用程序能够增加规则。 规则不会被保存，在重新加载或者重启之后必须再次提交。传递的参数 <args> 与 iptables, ip6tables 以及 ebtables 一致。

选项–direct需要是直接选项的第一个参数。

将命令传递给防火墙。参数 <args> 可以是 iptables, ip6tables 以及 ebtables 命令行参数。

```
firewall-cmd --direct --passthrough { ipv4 | ipv6 | eb } <args>
```

为表 <table> 增加一个新链 <chain> 。

```
firewall-cmd --direct --add-chain { ipv4 | ipv6 | eb } <table> <chain>
```

从表 <table> 中删除链 <chain> 。

```
firewall-cmd --direct --remove-chain { ipv4 | ipv6 | eb } <table> <chai
```

查询 <chain> 链是否存在与表 <table>. 如果是，返回0,否则返回1.

```
firewall-cmd --direct --query-chain { ipv4 | ipv6 | eb } <table> <chain
```

如果启用，此命令将有返回值。此命令没有输出信息。

获取用空格分隔的表 <table> 中链的列表。

```
firewall-cmd --direct --get-chains { ipv4 | ipv6 | eb } <table>
```

为表 <table> 增加一条参数为 <args> 的链 <chain> ，优先级设定为 <priority>。

```
firewall-cmd --direct --add-rule { ipv4 | ipv6 | eb } <table> <chain> <
```

从表 <table> 中删除带参数 <args> 的链 <chain>。

```
firewall-cmd --direct --remove-rule { ipv4 | ipv6 | eb } <table> <chain
```

查询带参数 <args> 的链 <chain> 是否存在表 <table> 中. 如果是，返回0,否则返回1.

```
firewall-cmd --direct --query-rule { ipv4 | ipv6 | eb } <table> <chain>
```

如果启用，此命令将有返回值。此命令没有输出信息。

获取表 <table> 中所有增加到链 <chain> 的规则，并用换行分隔。

```
firewall-cmd --direct --get-rules { ipv4 | ipv6 | eb } <table> <chain>
```

当前的firewalld特性

D-BUS接口

D-BUS 接口提供防火墙状态的信息，使防火墙的启用、停用或查询设置成为可能。

区域

网络或者防火墙区域定义了连接的可信程度。firewalld 提供了几种预定义的区域。区域配置选项和通用配置信息可以在firewall.zone(5)的手册里查到。

服务

服务可以是一系列本读端口、目的以及附加信息，也可以是服务启动时自动增加的防火墙助手模块。预定义服务的使用使启用和禁用对服务的访问变得更加简单。服务配置选项和通用文件信息在 firewalld.service(5) 手册里有描述。

ICMP类型

Internet控制报文协议 (ICMP) 被用以交换报文和互联网协议 (IP) 的错误报文。在 firewalld 中可以使用 ICMP 类型来限制报文交换。 ICMP 类型配置选项和通用文件信息可以参阅 firewalld.icmptype(5) 手册。

直接接口

直接接口主要用于服务或者应用程序增加特定的防火墙规则。这些规则并非永久有效，并且在收到 firewalld 通过 D-Bus 传递的启动、重启、重载信号后需要重新应用。

运行时配置

运行时配置并非永久有效，在重新加载时可以被恢复，而系统或者服务重启、停止时，这些选项将会丢失。

永久配置

永久配置存储在配置文件种，每次机器重启或者服务重启、重新加载时将自动恢复。

托盘小程序

托盘小程序 firewall-applet 为用户显示防火墙状态和存在的问题。它也可以用来配置用户允许修改的设置。

图形化配置工具

firewall daemon 主要的配置工具是 firewall-config 。它支持防火墙的所有特性（除了由服务/应用程序增加规则使用的直接接口）。 管理员也可以用它来改变系统或用户策略。

命令行客户端

firewall-cmd是命令行下提供大部分图形工具配置特性的工具。

对于ebtables的支持

要满足libvirt daemon的全部需求，在内核 netfilter 级上防止 ip*tables 和 ebtables 间访问问题，ebtables 支持是需要的。由于这些命令是访问相同结构的，因而不能同时使用。

/usr/lib/firewalld中的默认/备用配置

该目录包含了由 firewalld 提供的默认以及备用的 ICMP 类型、服务、区域配置。由 firewalld 软件包提供的这些文件不能被修改，即使修改也会随着 firewalld 软件包的更新被重置。 其他的 ICMP 类型、服务、区域配置可以通过软件包或者创建文件的方式提供。

/etc/firewalld中的系统配置设置

存储在此的系统或者用户配置文件可以是系统管理员通过配置接口定制的，也可以是手动定制的。这些文件将重载默认配置文件。

为了手动修改预定义的 icmp 类型，区域或者服务，从默认配置目录将配置拷贝到相应的系统配置目录，然后根据需求进行修改。

如果你加载了有默认和备用配置的区域，在 /etc/firewalld下的对应文件将被重命名为 <file>.old 然后启用备用配置。

正在开发的特性

富语言

富语言特性提供了一种不需要了解iptables语法的通过高级语言配置复杂 IPv4 和 IPv6 防火墙规则的机制。
Fedora 19 提供了带有 D-Bus 和命令行支持的富语言特性第2个里程碑版本。第3个里程碑版本也将提供对于图形界面 firewall-config 的支持。

锁定

锁定特性为 firewalld 增加了锁定本地应用或者服务配置的简单配置方式。它是一种轻量级的应用程序策略。

Fedora 19 提供了锁定特性的第二个里程碑版本，带有 D-Bus 和命令行支持。第3个里程碑版本也将提供图形界面 firewall-config 下的支持。

永久直接规则

这项特性处于早期状态。它将能够提供保存直接规则和直接链的功能。通过规则不属于该特性。

从ip*tables和ebtables服务迁移

这项特性处于早期状态。它将尽可能提供由iptables,ip6tables 和 ebtables 服务配置转换为永久直接规则的脚本。此特性在由firewalld提供的直接链集成方面可能存在局限性。
此特性将需要大量复杂防火墙配置的迁移测试。

计划和提议功能

防火墙抽象模型

在 ip*tables 和 ebtables 防火墙规则之上添加抽象层使添加规则更简单和直观。要抽象层功能强大，但同时又不能复杂，并不是一项简单的任务。为此，不得不开发一种防火墙语言。使防火墙规则拥有固定的位置，可以查询端口的访问状态、访问策略等普通信息和一些其他可能的防火墙特性。

对于conntrack的支持

要终止禁用特性已确立的连接需要 conntrack 。不过，一些情况下终止连接可能是不好的，如：为建立有限时间内的连续性外部连接而启用的防火墙服务。

用户交互模型

这是防火墙中用户或者管理员可以启用的一种特殊模式。应用程序所有要更改防火墙的请求将定向给用户知晓，以便确认和否认。为一个连接的授权设置一个时间限制并限制其所连主机、网络或连接是可行的。配置可以保存以便将来不需通知便可应用相同行为。 该模式的另一个特性是管理和应用程序发起的请求具有相同功能的预选服务和端口的外部链接尝试。服务和端口的限制也会限制发送给用户的请求数量。

用户策略支持

管理员可以规定哪些用户可以使用用户交互模式和限制防火墙可用特性。

端口元数据信息(由 Lennart Poettering 提议)

拥有一个端口独立的元数据信息是很好的。当前对 /etc/services 的端口和协议静态分配模型不是个好的解决方案，也没有反映当前使用情况。应用程序或服务的端口是动态的，因而端口本身并不能描述使用情况。

元数据信息可以用来为防火墙制定简单的规则。下面是一些例子：

- 允许外部访问文件共享应用程序或服务
- 允许外部访问音乐共享应用程序或服务
- 允许外部访问全部共享应用程序或服务
- 允许外部访问 torrent 文件共享应用程序或服务
- 允许外部访问 http 网络服务

这里的元数据信息不只有特定应用程序，还可以是一组使用情况。例如：组"全部共享"或者组"文件共享"可以对应于全部共享或文件共享程序(如：torrent 文件共享)。这些只是例子，因而，可能并没有实际用处。

这里是在防火墙中获取元数据信息的两种可能途径：

第一种是添加到 netfilter (内核空间)。好处是每个人都可以使用它，但也有一定使用限制。还要考虑用户或系统空间的具体信息，所有这些都需要在内核层面实现。

第二种是添加到 firewall daemon 中。这些抽象的规则可以和具体信息(如：网络连接可信级、作为具体个人/主机要分享的用户描述、管理员禁止完全共享的应归则等)一起使用。

第二种解决方案的好处是不需要为有新的元数据组和纳入改变(可信级、用户偏好或管理员规则等等)重新编译内核。这些抽象规则的添加使得 firewall daemon 更加自由。即使是新的安全级也不需要更新内核即可轻松添加。

sysctld

现在仍有 sysctl 设置没有正确应用。一个例子是，在 rc.sysinit 正运行时，而提供设置的模块在启动时没有装载或者重新装载该模块时会发生问题。

另一个例子是 net.ipv4.ip_forward ，防火墙设置、libvirt 和用户/管理员更改都需要它。如果有两个应用程序或守护进程只在需要时开启 ip_forwarding ，之后可能其中一个在不知道的情况下关掉服务，而另一个正需要它，此时就不得不重启它。

sysctl daemon 可以通过对设置使用内部计数来解决上面的问题。此时，当之前请求者不再需要时，它就会再次回到之前的设置状态或者是直接关闭它。

防火墙规则

netfilter 防火墙总是容易受到规则顺序的影响，因为一条规则在链中没有固定的位置。在一条规则之前添加或者删除规则都会改变此规则的位置。 在静态防火墙模型中，改变防火墙就是重建一个干净和完善的防火墙设置，且受限于 system-config-firewall / lokkit 直接支持的功能。也没有整合其他应用程序创建防火墙规则，且如果自定义规则文件功能没在使用 s-c-fw / lokkit 就不知道它们。默认链通常也没有安全的方式添加或删除规则而不影响其他规则。

动态防火墙有附加的防火墙功能链。这些特殊的链按照已定义的顺序进行调用，因而向链中添加规则将不会干扰先前调用的拒绝和丢弃规则。从而利于创建更为合理完善的防火墙配置。

下面是一些由守护进程创建的规则，过滤列表中启用了在公共区域对 ssh , mdns 和 ipp-client 的支持：

```
*filter
:INPUT ACCEPT [0:0]:FORWARD ACCEPT [0:0]:OUTPUT ACCEPT [0:0]:FORWARD_Z
-A INPUT -i lo -j ACCEPT
-A INPUT -j INPUT_direct
-A INPUT -j INPUT_ZONES
-A INPUT -p icmp -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -i lo -j ACCEPT
-A FORWARD -j FORWARD_direct
-A FORWARD -j FORWARD_ZONES
-A FORWARD -p icmp -j ACCEPT
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
-A OUTPUT -j OUTPUT_direct
-A IN_ZONE_public -j IN_ZONE_public_deny
-A IN_ZONE_public -j IN_ZONE_public_allow
-A IN_ZONE_public_allow -p tcp -m tcp --dport 22 -m conntrack --ctstat
-A IN_ZONE_public_allow -d 224.0.0.251 /32 -p udp -m udp --dport 5353
```

```
-A IN_ZONE_public_allow -p udp -m udp --dport 631 -m conntrack --ctsta
```

使用 deny/allow 模型来构建一个清晰行为(最好没有冲突规则)。例如： ICMP块将进入 IN_ZONE_public_deny 链(如果为公共区域设置了的话)，并将在 IN_ZONE_public_allow 链之前处理。

该模型使得在不干扰其他块的情况下向一个具体块添加或删除规则而变得更加容易。

以上就是本文的全部内容，希望对大家的学习有所帮助，也希望大家多多支持脚本之家。

您可能感兴趣的文章:

- Linux 中firewall的使用方法总结
- CentOS 7下用firewall-cmd控制端口与端口转发详解
- centos 7中firewall防火墙的常用命令总结
- Centos 7之Firewalld相关命令详细介绍
- 详解CentOS7使用firewalld打开关闭防火墙与端口
- 详解CentOS7防火墙管理firewalld
- CentOS 7 中firewall-cmd命令详细介绍
- centos7中firewall防火墙命令详解
- Centos7(Firewall)防火墙开启常见端口命令

原文链接：http://www.centoscn.com/CentOS/Intermediate/2015/0313/4879.html