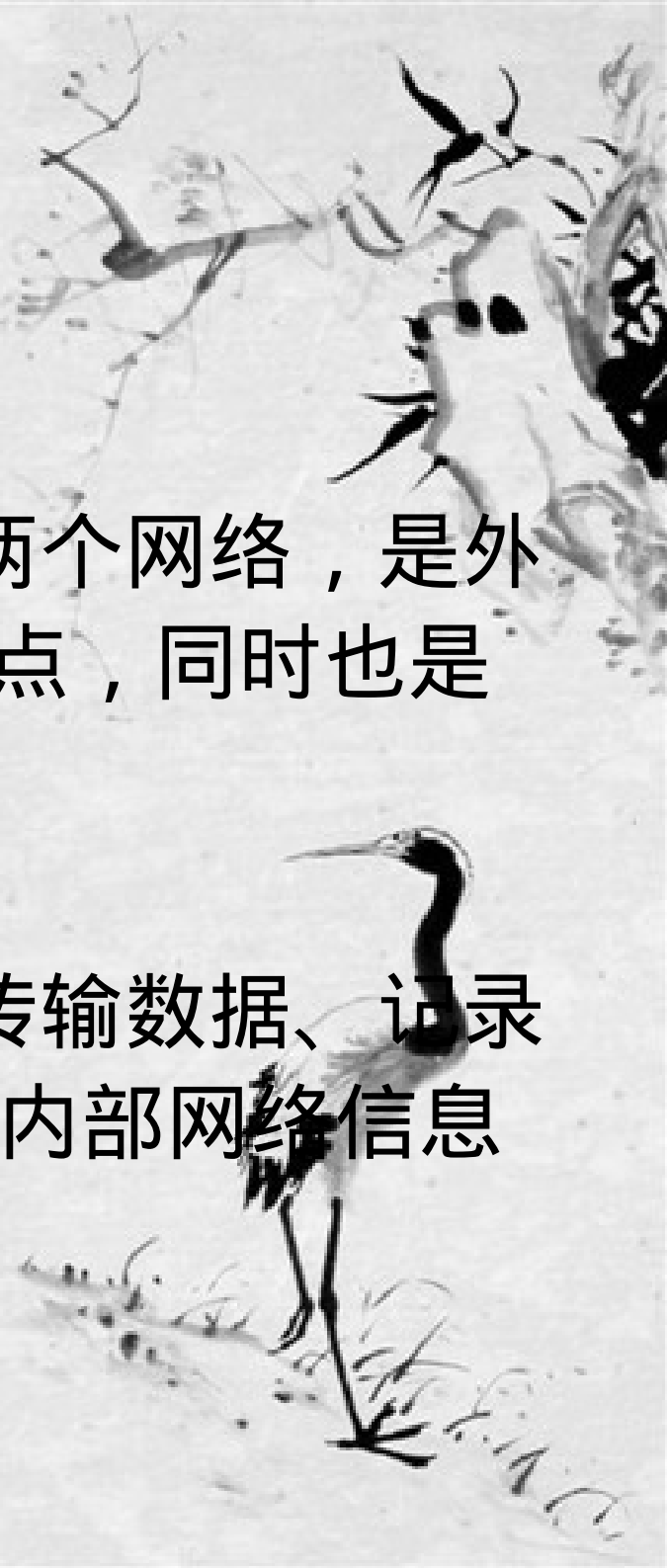
GNU/Linux Firewalld

Firewalld

防火墙 (Firewall):

计算机网络中的防火墙通常连接两个网络，是外部 Internet 和内部网络之间的交汇点，同时也是一道屏障

主要功能：实施安全策略、过滤传输数据、记录 Internet 活动、 IP 地址转换、保护内部网络信息



Firewalld

防火墙类型：

1. 包过滤防火墙

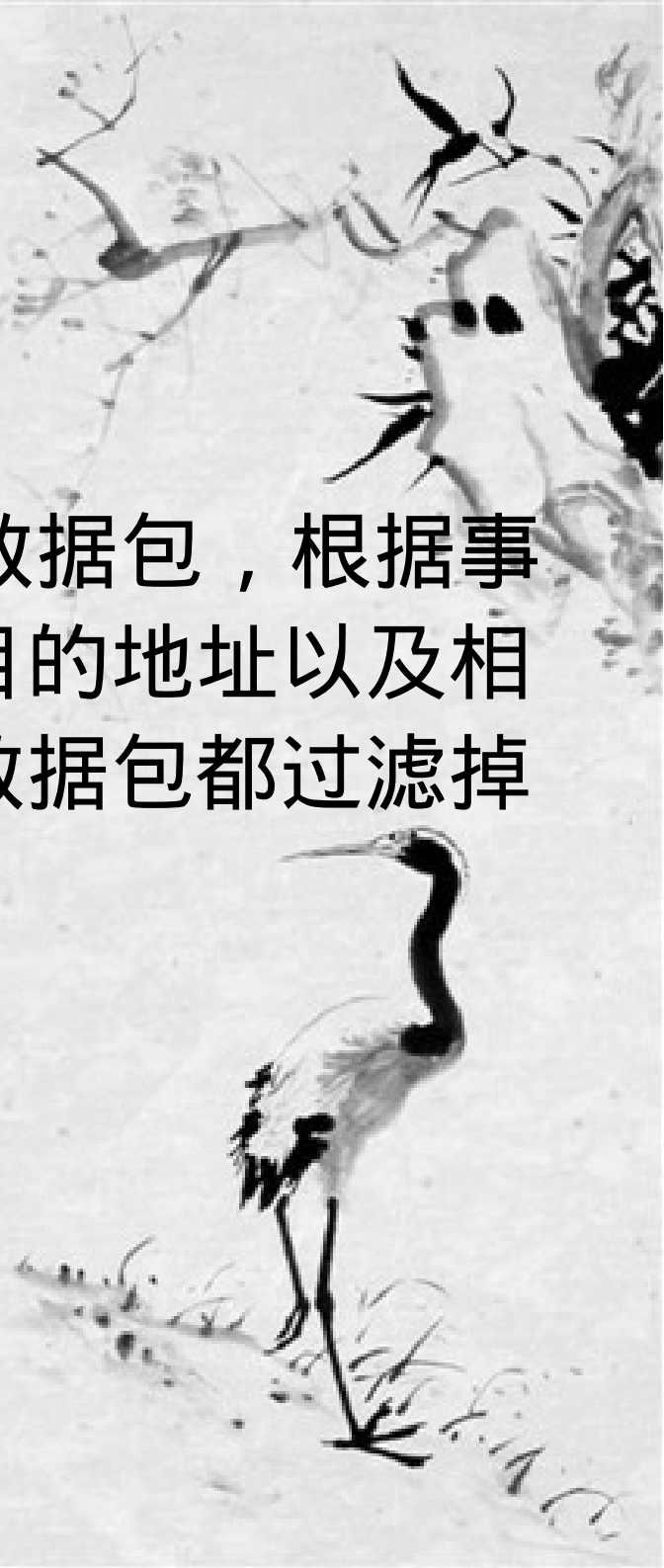
2. 代理防火墙

3. 状态检测防火墙

Firewalld

包过滤防火墙：

包过滤防火墙对通过它的每一个数据包，根据事先制订好的规则，对它的源地址、目的地址以及相应的端口进行判断，把不合规则的数据包都过滤掉。



Firewalld

包过滤防火墙：

优点：

处理来速度快，对用户透明

缺点：

不能实现用户级认证

日志记录不完善

无法防御 IP 欺骗

对特殊协议不适用

Firewalld

代理 (Proxy) 防火墙：

代理也称为应用网关防火墙，核心是代理技术，即代替客户处理对服务器连接请求。

优点：

安全性高

过滤规则灵活

详细的日志记录

Firewalld

代理 (Proxy) 防火墙：

缺点：

- 处理速度慢
- 对用户不透明
- 代理服务类型有限制

Firewalld

状态检测防火墙：

又称为动态包过滤防火墙。其主要特点在于

1) 在防火墙的核心部分建立状态连接表，并将进出网络的数据当成一个个的会话，利用状态表跟踪每个会话状态 。

2) 防火墙首先根据规则表来判断是否允许这个数据包通过，如果不符合规则就立即丢弃；如果符合规则，再将当前的数据包和状态信息，与前一时刻的数据包和状态信息进行比较，然后根据比较结果决定数据包是否能够通过防火墙 。

Firewalld

防火墙结构：

- 屏蔽路由器
- 双穴主机
- 屏蔽主机防火墙
- 屏蔽子网防火墙
- 其它结构

Firewalld

屏蔽路由器（包过滤）

优点：

处理速度快

费用低

对用户透明

Firewalld

屏蔽路由器（包过滤）

缺点：

维护困难

只能阻止较少的 IP 欺骗

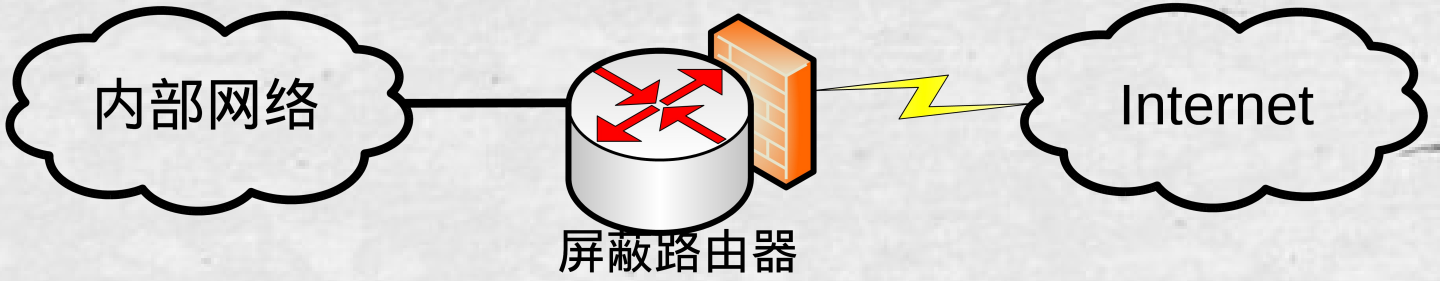
无用户认证

日志功能有限，

当规则过多时，处理速度及吞吐速度急速下降，无法对信息全面控制

Firewalld

屏蔽路由器（包过滤）

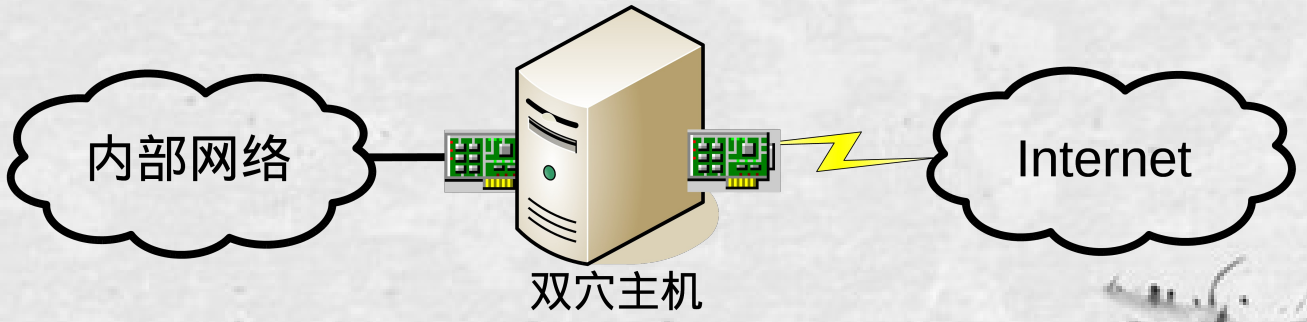


Firewalld

双穴主机（双宿网关）

优点：安全性比屏蔽路由器高

缺点：入侵者一旦得到双穴主机的访问权，内部网络就会被入侵，因此需具有强大的身份认证系统，才可以阻挡来自外部的不可信网络的非法入侵。

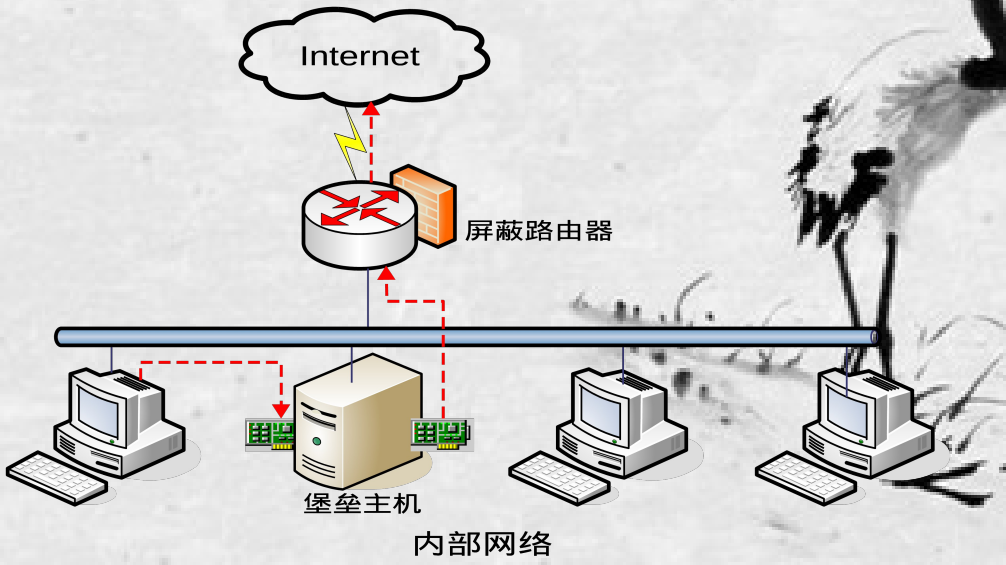


Firewalld

屏蔽主机防火墙

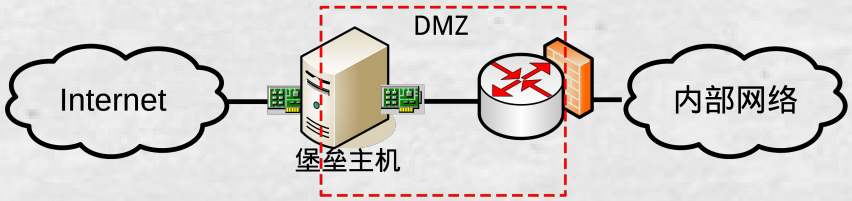
优点：实现了网络层安全（包过滤）和应用层安全（代理），因此安全等级比屏蔽路由器高

缺点：堡垒主机可能被绕过，堡垒主机与其它内部主机间没有任何保护网络安全的东西存在，一旦被攻破，内网就将暴露

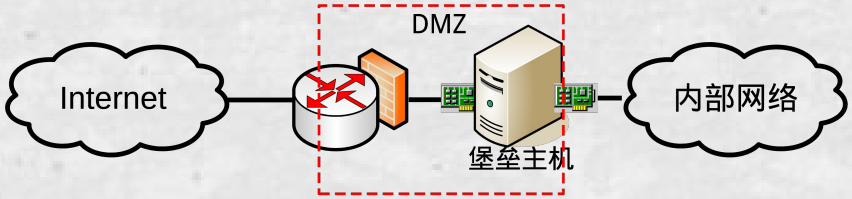


Firewalld

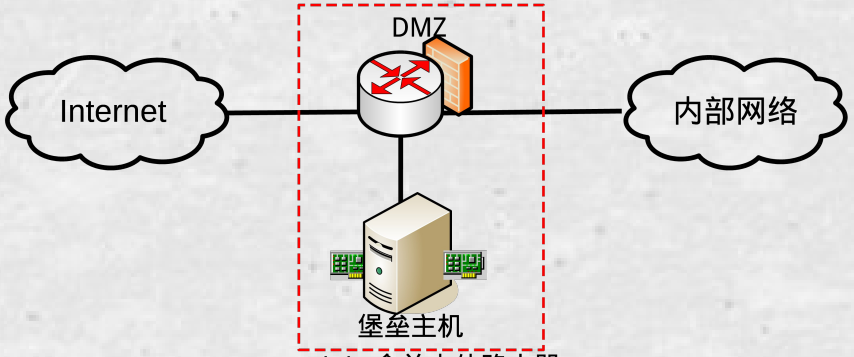
其他结构



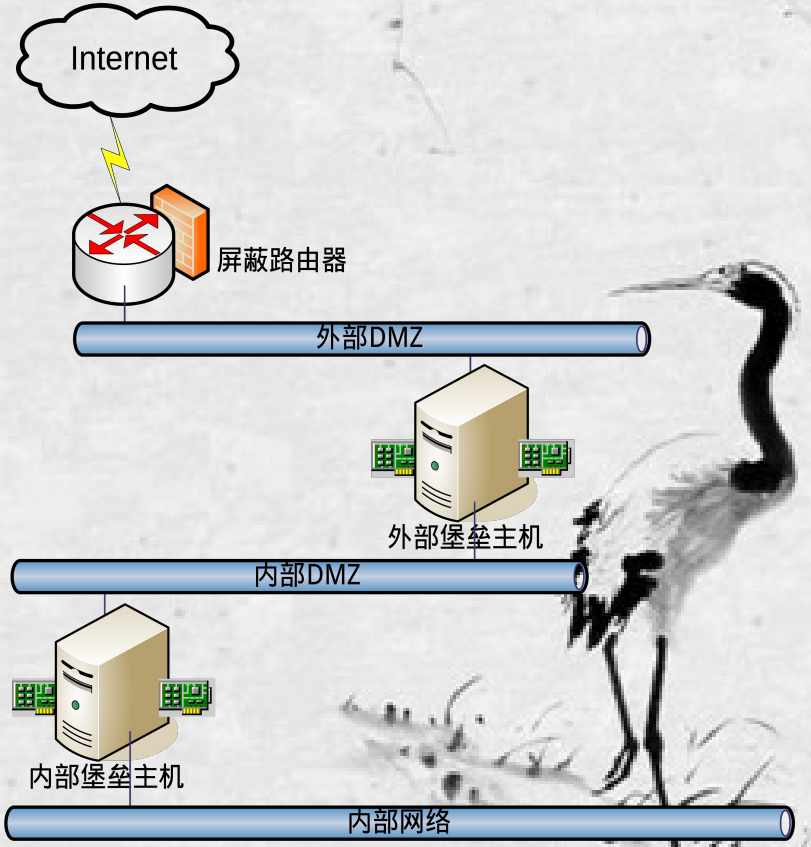
(a) 合并外部路由器与堡垒主机



(b) 一个堡垒主机和一个DMZ结构



(c) 合并内外路由器



Firewalld

Linux 防火墙

Linux 防火墙采用包过滤技术，总体看防火墙发展经历了四个发展阶段

静态防火墙：

ipfwadm(kernel 2.0)

ipchains(kernel 2.2)

iptables(kernel 2.4 later)

动态防火墙

firewalld(RHEL7 新纳入的防火墙)

Firewalld

Linux 防火墙

静态防火墙：

需要自行书写规则及相关信息，每次改变后都需要重新启动防火墙

动态防火墙

在更改防火墙规则后，不需要重新加载相关服务和模块，达到改变及应用的效果

Firewalld

Linux 防火墙

firewalld 看起来是一个非常新的防火墙，实际上仍然采用了 netfilter/iptables 的底层架构。虽然在应用上变化非常大，但底层实质意义上没有太多的改变。

Firewalld

Linux 防火墙

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

Firewalld

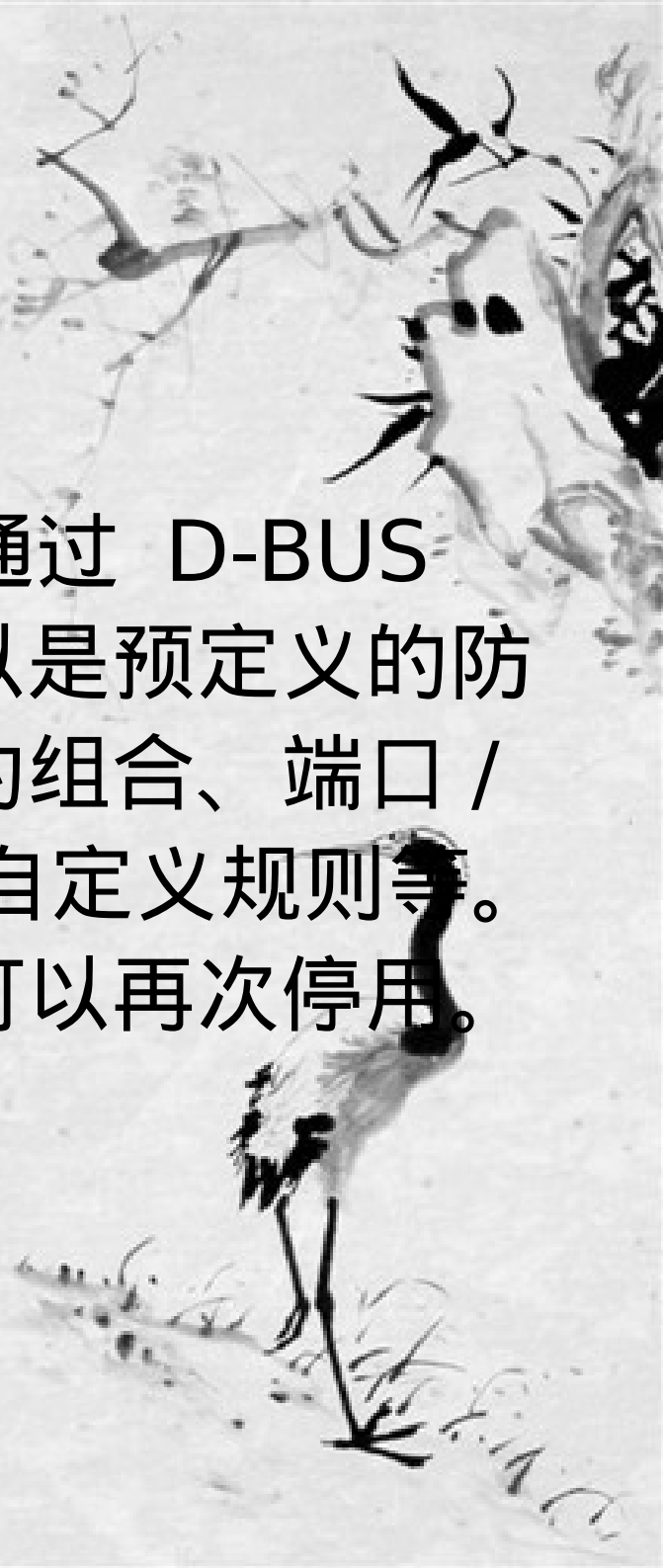
Linux 防火墙

firewall daemon 动态管理防火墙，通过 D-BUS 提供当前激活防火墙的设置信息，也通过 D-BUS 接收使用 PolicyKit 认证方式的更改。因而不需要重启整个防火墙便可应用更改。。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 iptables 和 ebtables 命令行工具添加的防火墙规则。

Firewalld

firewall daemon(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。



Firewalld

firewall daemon(守护进程)

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/tftp 等服务 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。

Firewalld

Firewalld

虽然 firewalld 使用 netfilter 子系统，但仍然跟 iptables、 ebtables 出现冲突。因此如果使用 firewalld 就必须关闭 iptables/ip6tables/ebtables

#systemctl mask iptables

#systemctl mask ip6iptables

#systemctl mask ebtables

Firewalld

Firewalld

firewalld 的工具包

#yum install firewalld system-config-firewall-base firewall-config

启用 firewalld

#systemctl enable firewalld

#systemctl start firewalld

Firewalld

Firewalld

firewalld 配置文件记录

#cd /etc/firewalld/

#ls

Firewalld

Firewalld 基本概念

区域 (zone)

firewalld 定义了 zone 概念，在默认情况下已经拥有 9 种区域，用以实现不同的功能：

Firewalld

Firewalld 基本概念

区域 (zone)

依据从不信任到信任的顺序进行排列：

drop: 任务进入网络的包都被丢弃，并且不做任何回应 . 至允许出包的链接。

Block: 任何进入网络的包都被丢弃，并返回对方 IPv4 的 icmp-host-prohibited(禁止) 或 IPv6 的 icmap6-adm-prohibited 报文 . 在此区域中至允许由系统初始化的网络链接

Firewalld

Firewalld 基本概念

区域 (zone)

public(默认): 认为当前网络中的网络信任度不高，仅允许选中的服务通过。

external: 用在路由器等启用伪装 (NAT) 的外部网络 . 当认为网络不可信任时，可指定服务通过。

Firewalld

Firewalld 基本概念

区域 (zone)

dmz: 允许 DMZ 中指定的设备的指定服务，有限的被外部网络访问。

work: 工作网络，对网络信任度高，通过指定的服务提供相关交互

Firewalld

Firewalld 基本概念

区域 (zone)

home: 家庭网络，对网络信任度高，通过指定的服务提供相关交互

internal: 内部网络，对网络信任度高，通过指定的服务提供相关交互

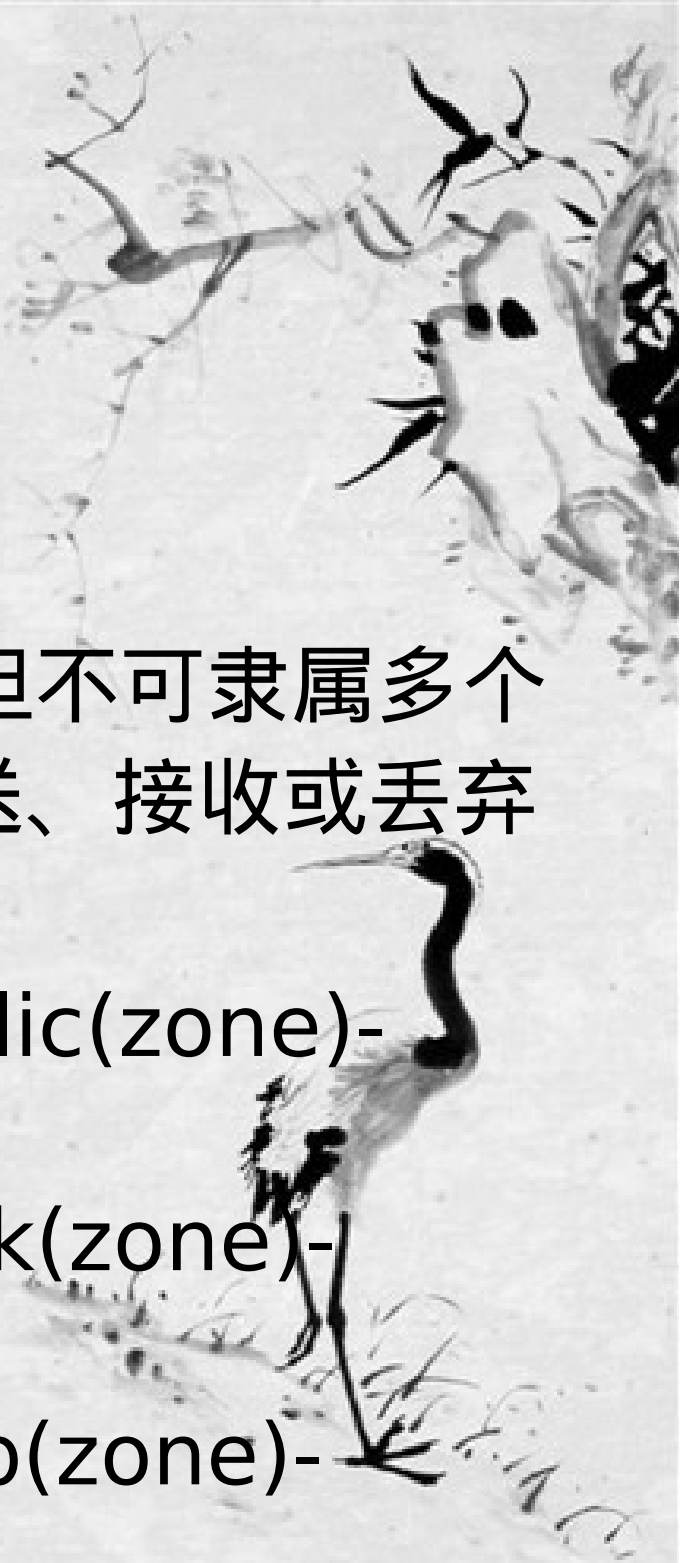
Firewalld

Firewalld 基本概念

区域 (zone)

trusted: 允许所有网络链接，即使关闭所有服务，服务仍然可以响应。

Firewalld



Firewalld 基本概念

网卡与区域

每个网卡都将隶属与一个区域（但不可隶属多个区域）。网卡根据区域的特点来传送、接收或丢弃数据包。如

data->enp0s3(eth0)->public(zone)->kernel

data->enp0s8(eth0)->work(zone)->kernel

data->enp0s9(eth0)->drop(zone)->kernel

Firewalld

Firewalld 状态：

1. 运行状态：所有的规则均在内存中，一旦服务重新启动或设备启动，所定义的规则将消失，需要重新配定

2. 永久状态：规则写在相关的 zone 文件或服务文件中，当服务或设备启动时重新加载这些配置即可直接完成相关动作

Firewalld

查看 firewalld 帮助

```
#firewall-cmd --help
```

查看 firewalld 状态

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

Firewalld

firewalld 的预先存在的服务是用 xml 格式进行编写的，这些服务的 xml 存在与

```
#cd /usr/lib/firewalld/services/
#ls
```

可根据某个服务作为模板，来自定以额外的服务，而后用 firewall-cmd --reload 命令重新读取，这样额外服务的 xml 文件将直接被载入。

Firewalld

获取支持的区域列表

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

获取所支持的服务

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

列出所有启用的区域信息

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

Firewalld

启用应急模式阻断所有网络链接

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

禁用应急模式

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

查询应急模式

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

Firewalld

查看当前 firewalld 的默认区域

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

改变 firewalld 默认区域

```
#fireall-cmd --set-default-zone=drop
#firewall-cmd --get-default-zone
```

Firewalld

开启指定服务

1) 测试环境准备

#systemctl start http

#echo “hello” > /var/www/html/index.html

客户端浏览器测试

2) 开启 http

#firewall-cmd --zone=public --add-service=http

查询指定服务是否在 firewalld 开启

Firewalld

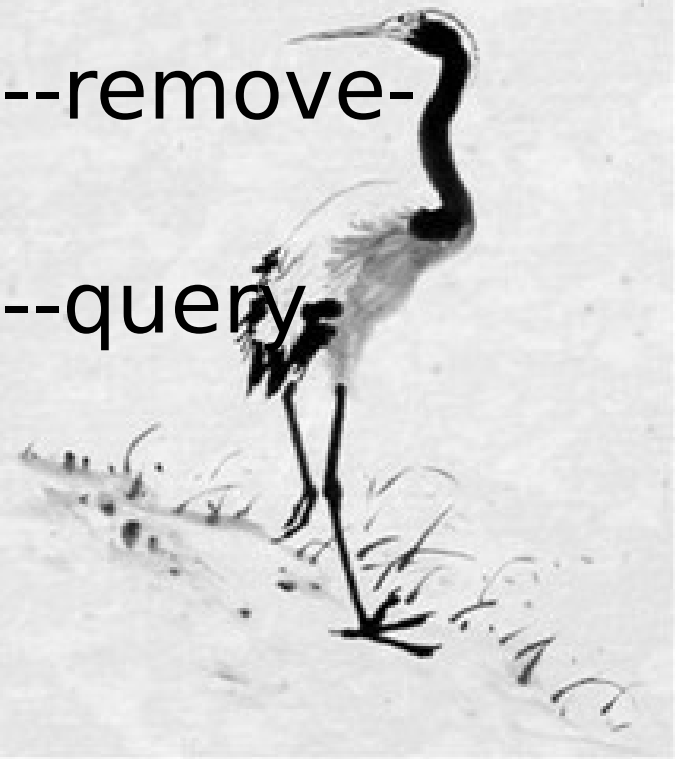
开启指定服务

3) 客户端浏览器测试

4) 关闭 http

#firewall-cmd --zone=public --remove-service=http

#firewall-cmd --zone=public --query-service=http



5) 客户端浏览器测试

Firewalld

开启指定服务

指定某个服务启动特定时间（单位：秒）

#firewall-cmd --zone=public --add-service=ipp-client --timeout=60

Firewalld

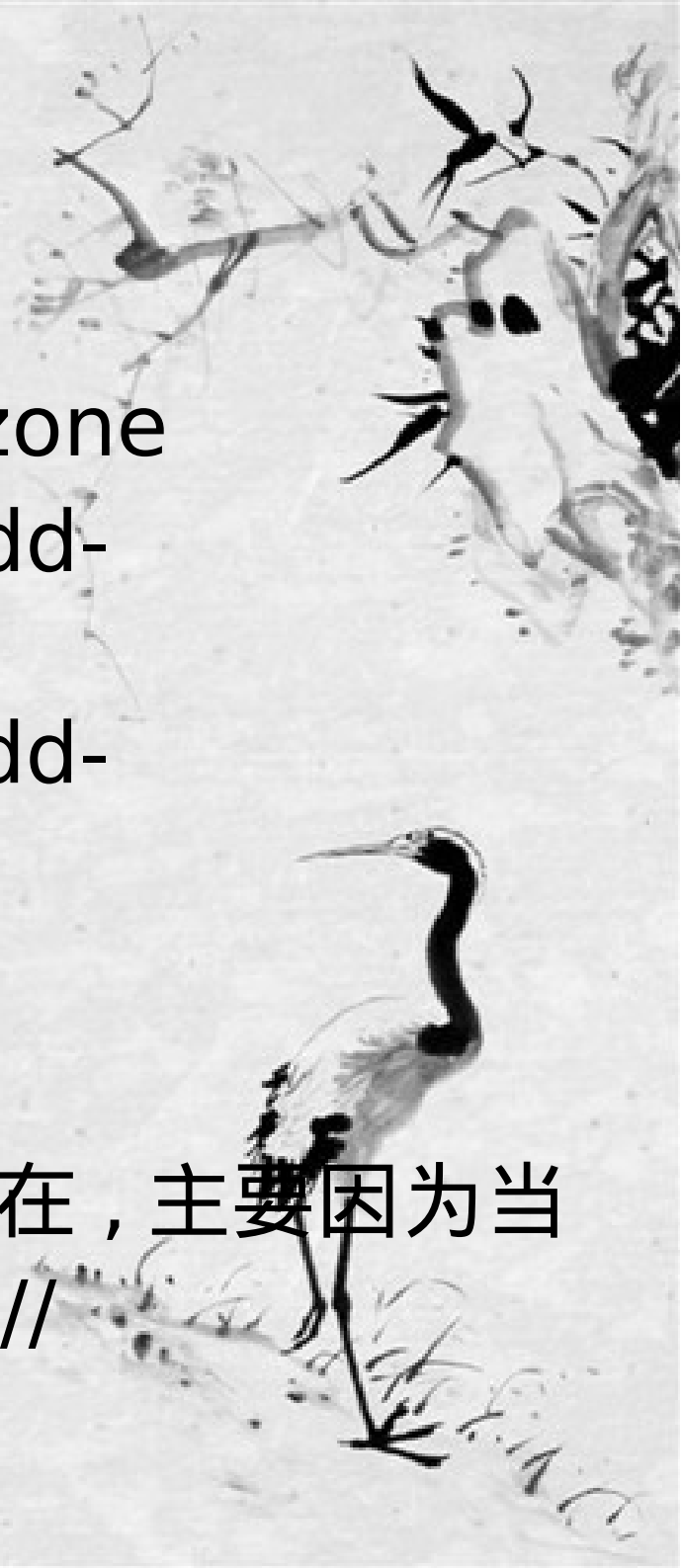
将指定服务 / 端口永久性加入默认 zone

#firewall-cmd --permanent --add-service=vnc-server

#firewall-cmd --permanent --add-port=6080/tcp

#firewall-cmd --list-all

//* 此时所加载的服务及端口并不存在，主要因为当使用 --permanent 所造成的问题 *//



Firewalld

参数 :--permanent

即将修改的信息写入相关的配置的的 zone 文件 (xml) 或服务文件 (xml) 中。

永久选项不直接影响运行时的状态。如果打算使用修改的配置，需要重载或者重启服务才可。为了使用运行时和永久设置，需要分别设置两者。 选项 --permanent 需要是永久设置的参数之一。

当指定为永久性时，必须重新加载才能看到其配置，默认情况下当时候 --permanent 时，运行状态是无法查看所更改的内容

Firewalld

查看运行时的默认区域信息

```
#firewall-cmd --list-all
```

查看默认区域永久性信息

```
#firewall-cmd --list-all --permanent
```

Firewalld

在不改变状态的条件下，可重新加载 firewalld，当 reload 成功后，永久性配置将被加载，此时当可直接查看信息

```
#firewall-cmd --reload
#firewall-cmd --list-all
```

可以使用 --complete-reload, 但状态信息将会丢失。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。

Firewalld

开启指定端口

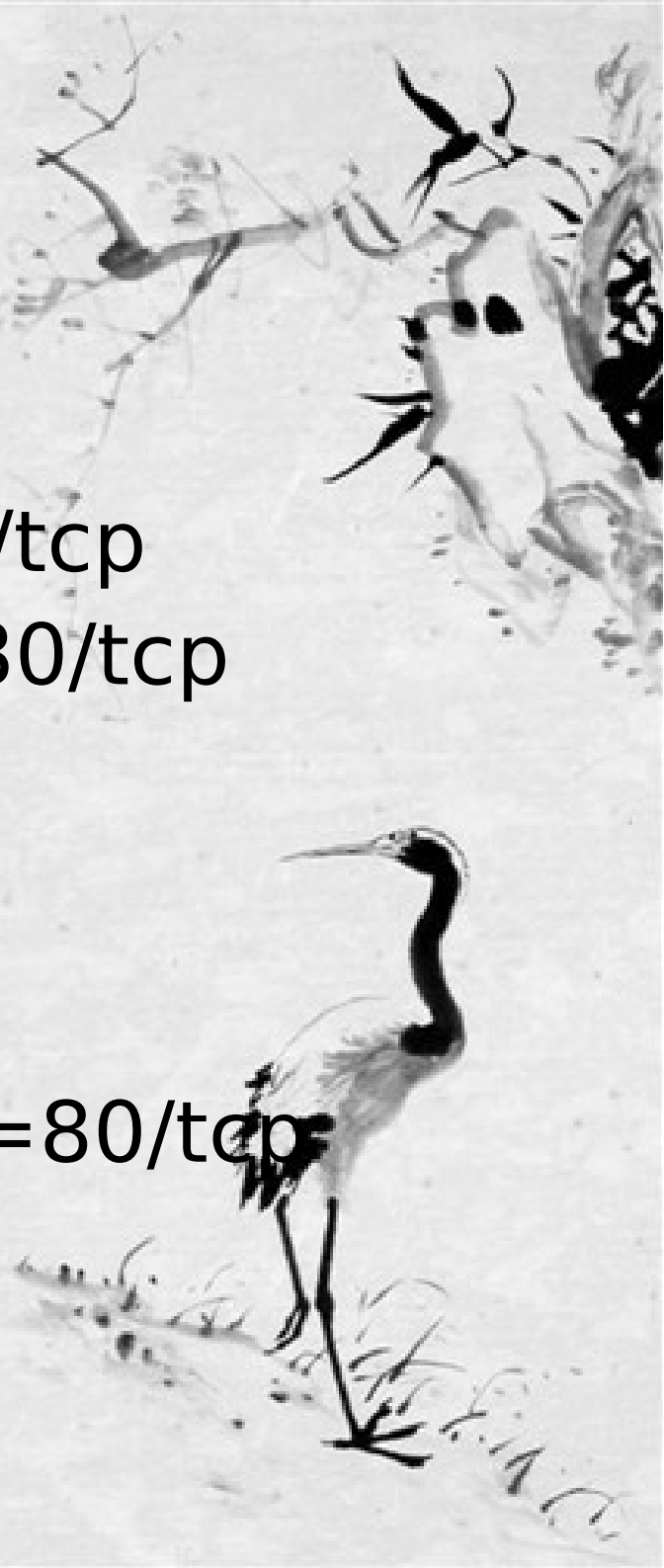
#firewall-cmd --add-port=80/tcp

#firewall-cmd --query-port=80/tcp

默认不指定区域即为默认区域

关闭指定端口

#firewall-cmd --remove-port=80/tcp



Firewalld

查看哪些网卡属于哪个 zone

```
#firewall-cmd --get-zone-of-interface=enp0s3
```

将指定网卡加入至默认 zone

```
#firewall-cmd --add-interface=enp0s8
```

将指定网卡移除默认 zone

```
#firewall-cmd --remove-interface=enp0s8
```

Firewalld

查看指定区域开启的相关服务、端口、接口

```
#firewall-cmd --zone=public --list-all
```

查看指定区域开启的相关服务

```
#firewall-cmd --zone=public --list-service
```

查看指定区域开启的相关 port

```
#firewall-cmd --zone=public --list-port
```

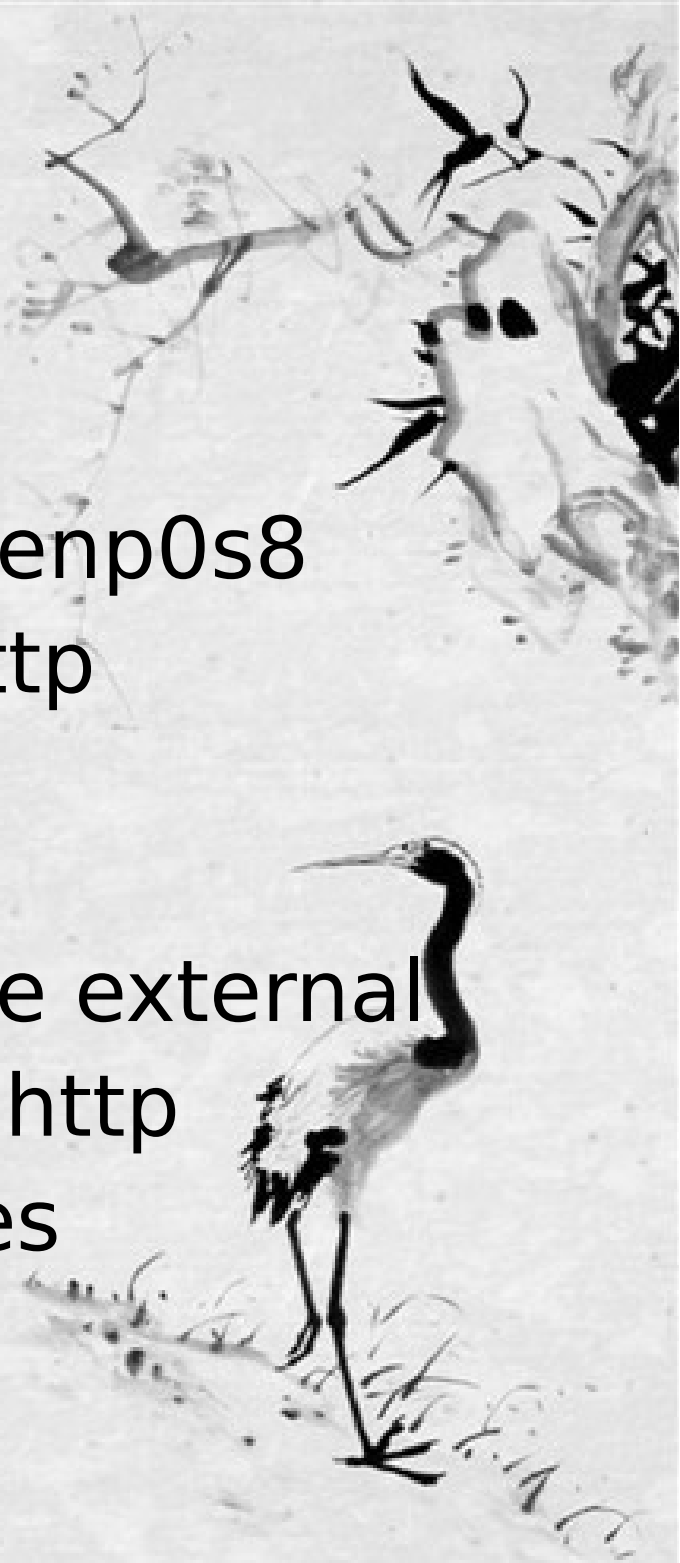
Firewalld

区域设置问题：

1. 当默认区域为 trusted 时，相关服务（ 如 http) 默认不允许通信，也可以通信。只要原因是 trusted 为最高级别信任区域

2. 当网卡在不同的区域中，其区域内允许指定服务 (如 http) 通信，而默认区域不允许，也可以完成服务的通信。此处必须将通信的网卡切换到默认区域才可以实现

Firewalld



示例：

```
#firewall-cmd --add-interface=enp0s8
#firewall-cmd --add-service=http
#firewall-cmd --get-zone-of-interface=enp0s8
#firewall-cmd --set-default-zone external
#firewall-cmd --query-service=http
#firewall-cmd --get-active-zones
external
    interfaces: enp0s3 enp0s8
```

Firewalld

示例：

测试客户端能否访问 firewalld 上的 httpd?

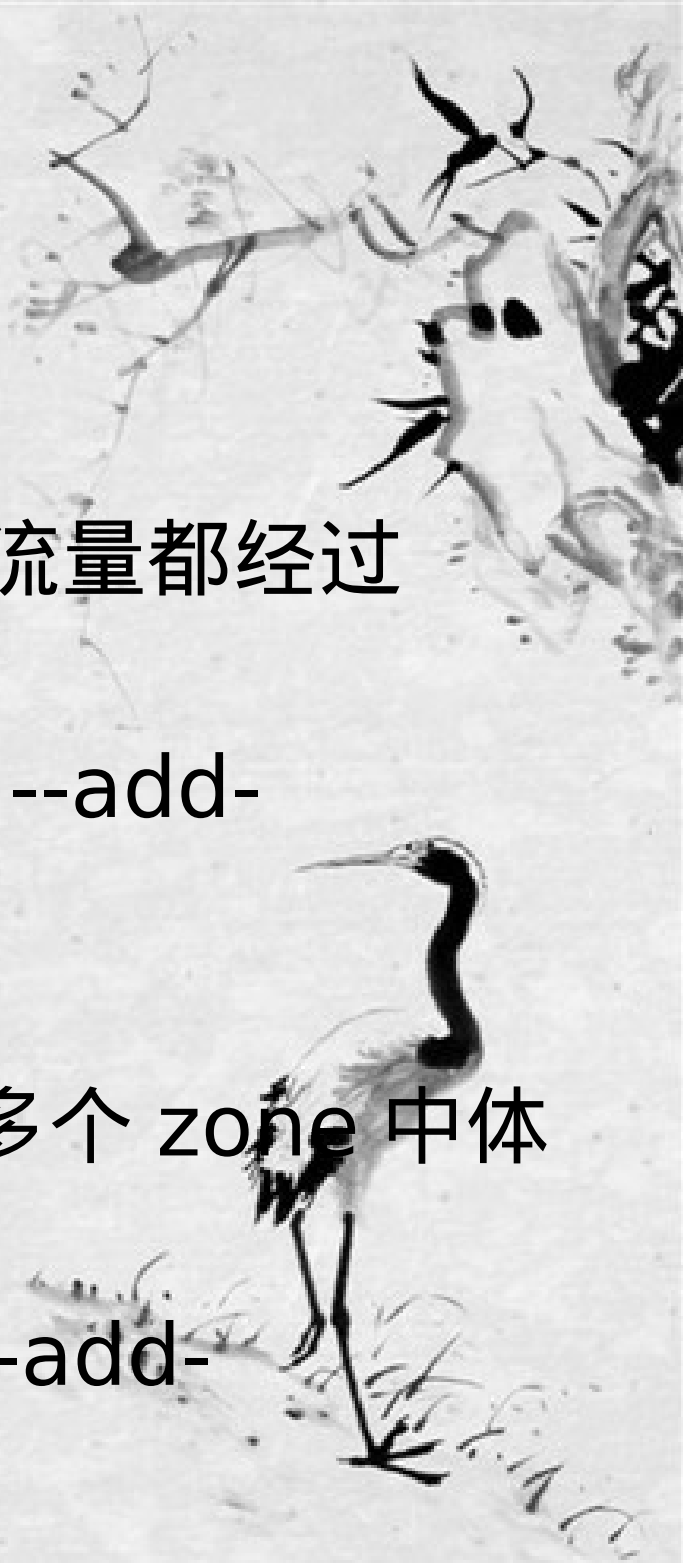
切换网卡至 public(http 服务已开启）

#firewall-cmd --zone=public --change-interface=enp0s3

#firewall-cmd --zone=public --change-interface=enp0s8

测试客户端能否访问 firewalld 上的 httpd?

Firewalld



指定源

指定源地址 192.168.3.0/24 的流量都经过 public 区域的过滤

#firewall-cmd --zone=public --add-source=192.168.3.0/24

需要注意的是一个源不能同时在多个 zone 中体现

#fireawll-cmd --zone=work --add-source=192.168.3.0/24

Firewalld

ICMP 控制：

默认情况下 ,Firewalld 允许所有的 ICMP 类型的协议通过 . 如果需要控制 ICMP, 可先查看 ICMP 类型，确定控制哪类协议。

#fireall-cmd --get-icmptypes

关闭 icmp-request, 禁止客户端 ping

#firewall-cmd --add-icmp-block=echo-request

Firewalld

开启 icmp-request, 允许客户端 ping

#firewall-cmd --remove-icmp-block=echo-request

Firewalld

开启端口转发，如表所示：

主机名	开启服务	网卡	所链接的上 / 下一台主机名称	所链接的上 / 下一台主机网卡
t1	httpd	enp0s3	firewall	enp0s3
firewall	firewalld	enp0s3/enp0s8	t2	enp0s3
t2	---	enp0s3	firewall	enp0s8

#firewall-cmd --add-masquerade

#firewall-cmd --add-forward-port=port=80:proto=tcp:toaddr=192.168.3.2

测试 t2 访问 t1 的 httpd

Firewalld

开启端口转发

#firewall-cmd --add-masquerade

#firewall-cmd --add-forward-port=port=8080:proto=tcp:toport=80

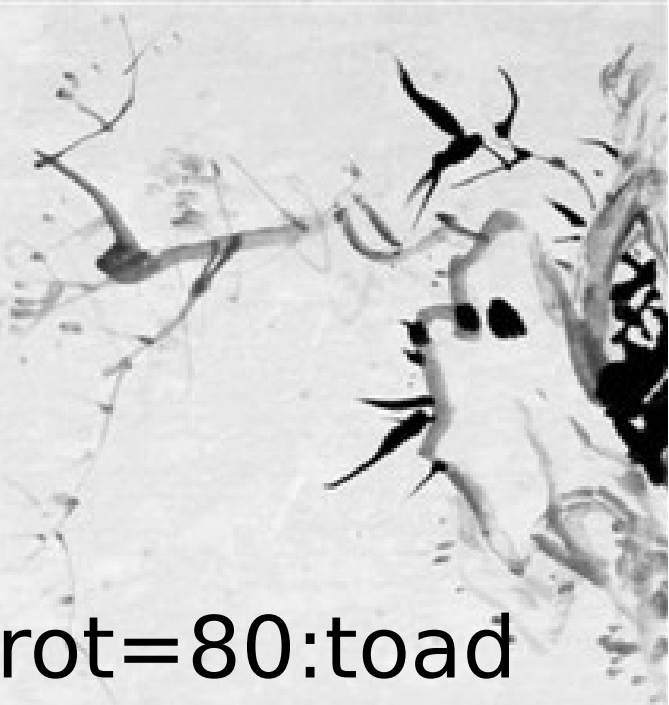
#links firewall_server_ip

Firewalld

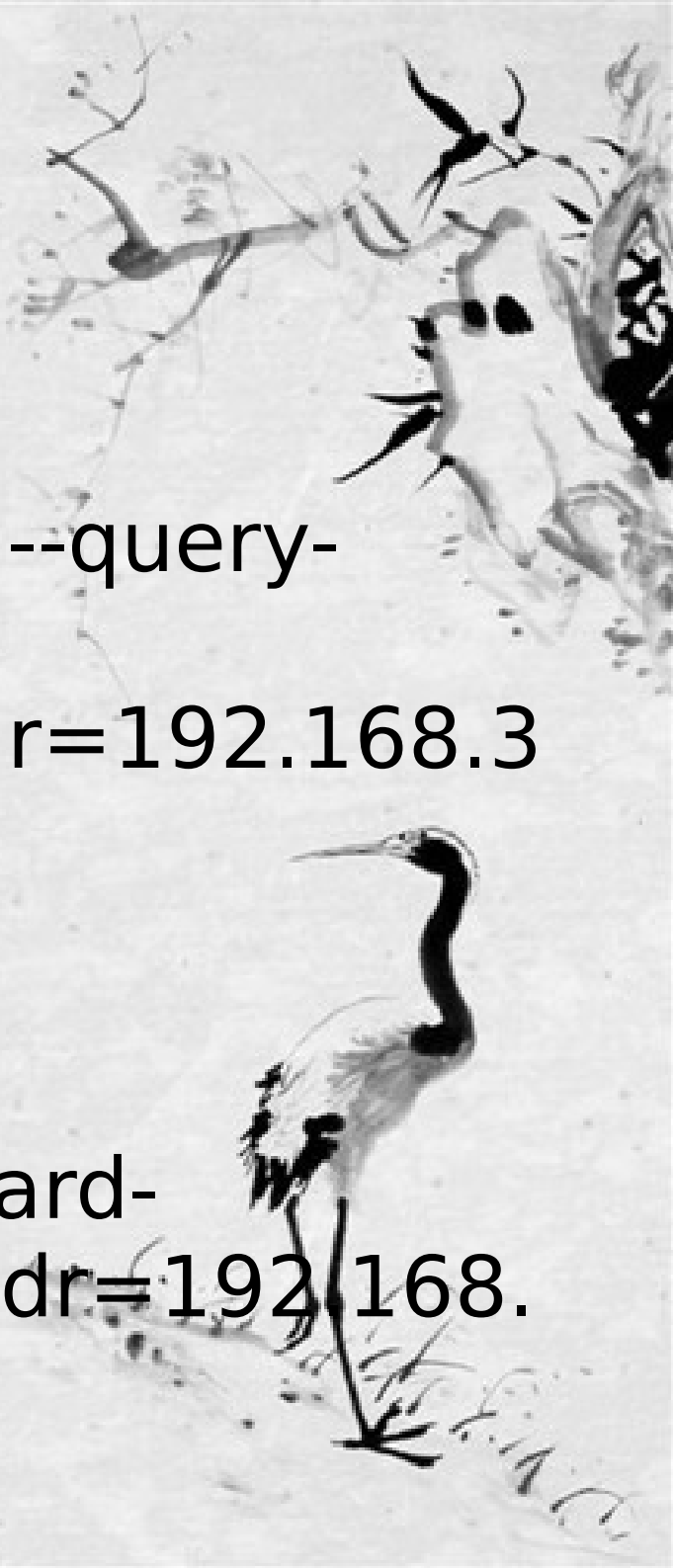
开启端口转发

#firewall-cmd --add-forward-port=port=8080:proto=tcp:toprot=80:toaddress=192.168.1.123

#firewall-cmd --add-forward-port=port=8000-9000:proto=tcp:toprot=1000-2000:toaddress=192.168.1.123



Firewalld



查询转发

#firewall-cmd --zone=public --query-forward-port=port=80:proto=tcp:toaddr=192.168.3.2

删除转发

#firewall-cmd --remove-forward-port=port=80:protco=tcp:toaddr=192.168.3.2

Firewalld

开启 vsftpd

t1:vsftpd 服务 (192.168.3.2)

Firewalld: 操作

#firewall-cmd --add-service=ftp

#firewall-cmd --add-forward-port=port=21:protco=tcp:toaddr=192.168.3.2

T2: 客户端 -> 测试

Firewalld

Rich 规则 (firewall 扩展规则)

如认为 firewalld 所提供的规则有限，可以使用 (direct rules) 直接规则来直接操作 iptables,ip6tables,ebtalbes 。虽然对刚学习防火墙来说有些困难，但其灵活和强大是不容忽视的。

直接规则有限与 firewalld 规则进行分析

rich 规则，在不指定的情况下将直接加入默认 zones 中

Firewalld

基本的 rich rule 语法格式

rule
  [source]
  [destination]
  service|port|protocol|icmp-block|masquerade|forward-port
  [log]
  [audit]
  [accept|reject|drop]

Firewalld

基本的 rich rule 语法格式

log: 将指定信息记录至日志

audit: 限制 ='< 速率 / 持续时间 >'

Firewalld

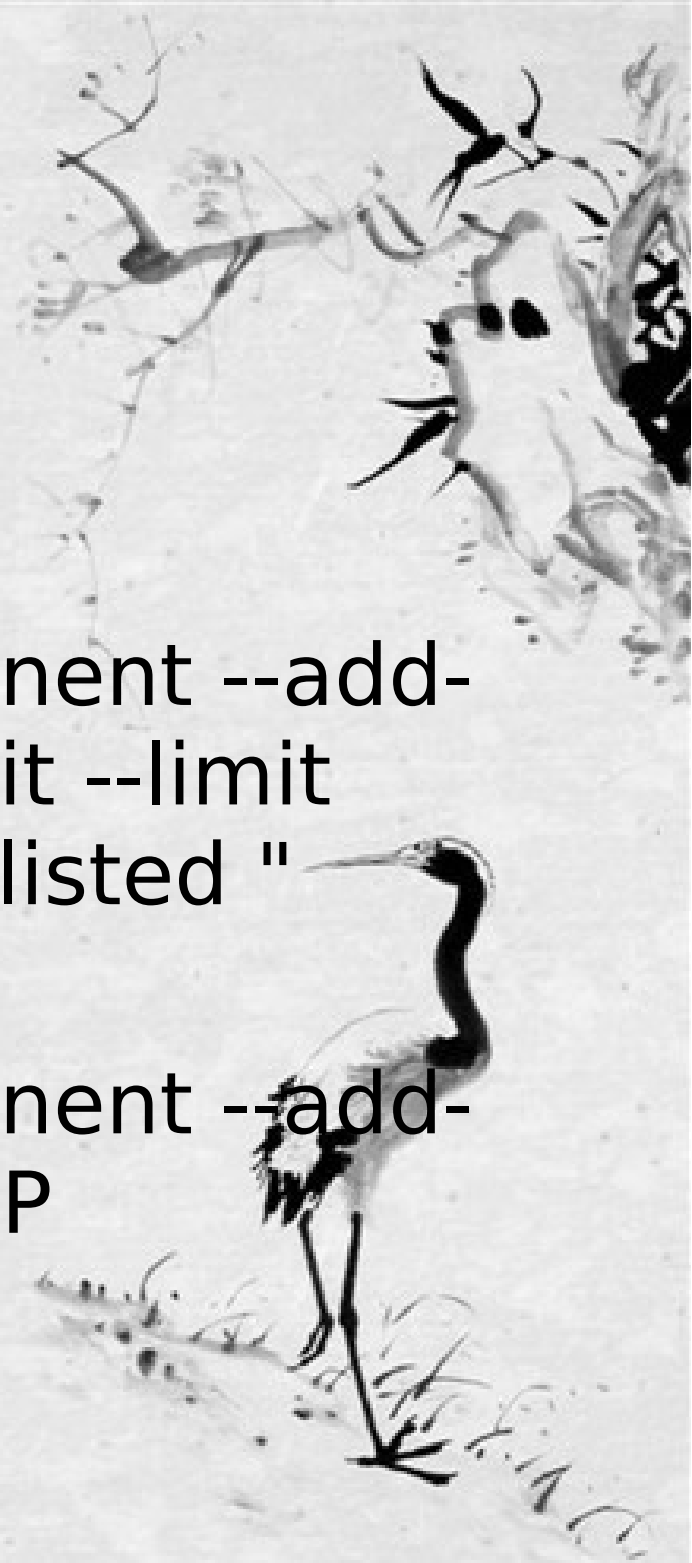
Rich 规则（丰富规则）

列示 1: 将符合规则的条目加入至黑列表

```
# firewall-cmd --direct --permanent --add-chain ipv4 raw blacklist

# firewall-cmd --direct --permanent --add-rule ipv4 raw PREROUTING 0 -s 192.168.0.0/24 -j blacklist
```

Firewalld



Rich 规则 (丰富规则)

列示 1:(审计 ->limit)

```
# firewall-cmd --direct --permanent --add-rule ipv4 raw blacklist 0 -m limit --limit 1/min -j LOG --log-prefix "blacklisted "

# firewall-cmd --direct --permanent --add-rule ipv4 raw blacklist 1 -j DROP
```

Firewalld

示例 2

1. 将 ssh 链接记录至日志中，每分钟 3 次

#firewall-cmd --add-rich-rule='rule service name=ssh log prefix=”SSHCONNECT “ limit value=3/m'

2. 找客户端链 3 此服务器

3. 查看服务

#grep SSHCONNECT /var/log/messages

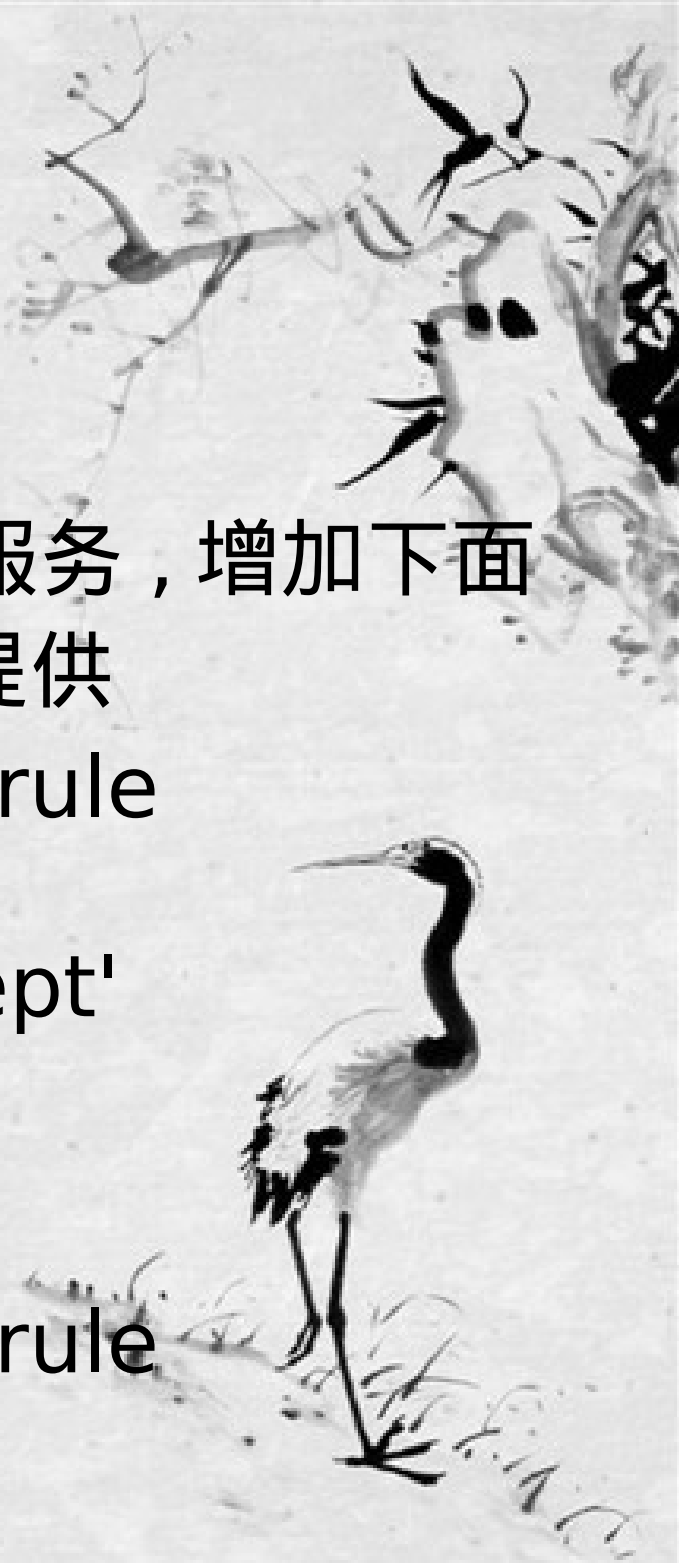
Firewalld

示例 2

1. 拒绝 2001:db8::/64 地址 5 分钟内使用 DNS 服务，每小时记录 1 次此信息

#firewall-cmd --add-rich-rule='rule family=ipv6 source address="2001:db8::/64" service name="dns" audit limit value="1/h" reject' --timeout=300

Firewalld



示例 3

1. 假设 http 在服务端不允许提供服务，增加下面内容可让指定 IP 完成 http 服务的提供

```
#firewall-cmd --add-rich-rule='rule family=ipv4 source address=192.168.1.11/32 accept'
```

2. 拒绝指定网络段

```
#firewall-cmd --add-rich-rule='rule family=ipv4 source address=192.168.0.0/24 drop'
```

Firewalld

示例 3

3. 查看 rich 规则

#firewall-cmd --list-all-rules

或

#firewall-cmd --list-all[-zones]

4. 查询指定的 rich 规则

#firewall-cmd --query-rich-rule='rule family=ipv4 source address=192.168.0.0/24 drop'

Firewalld

示例

1. 使 192.168.8.109 可以访问 192.168.8.123 的 http 服务

#firewall-cmd --add-rich-rule='rule family=ipv4 source address=192.168.8.109/32 destination address=192.168.8.123/32 port protocol=tcp port=80 accept'

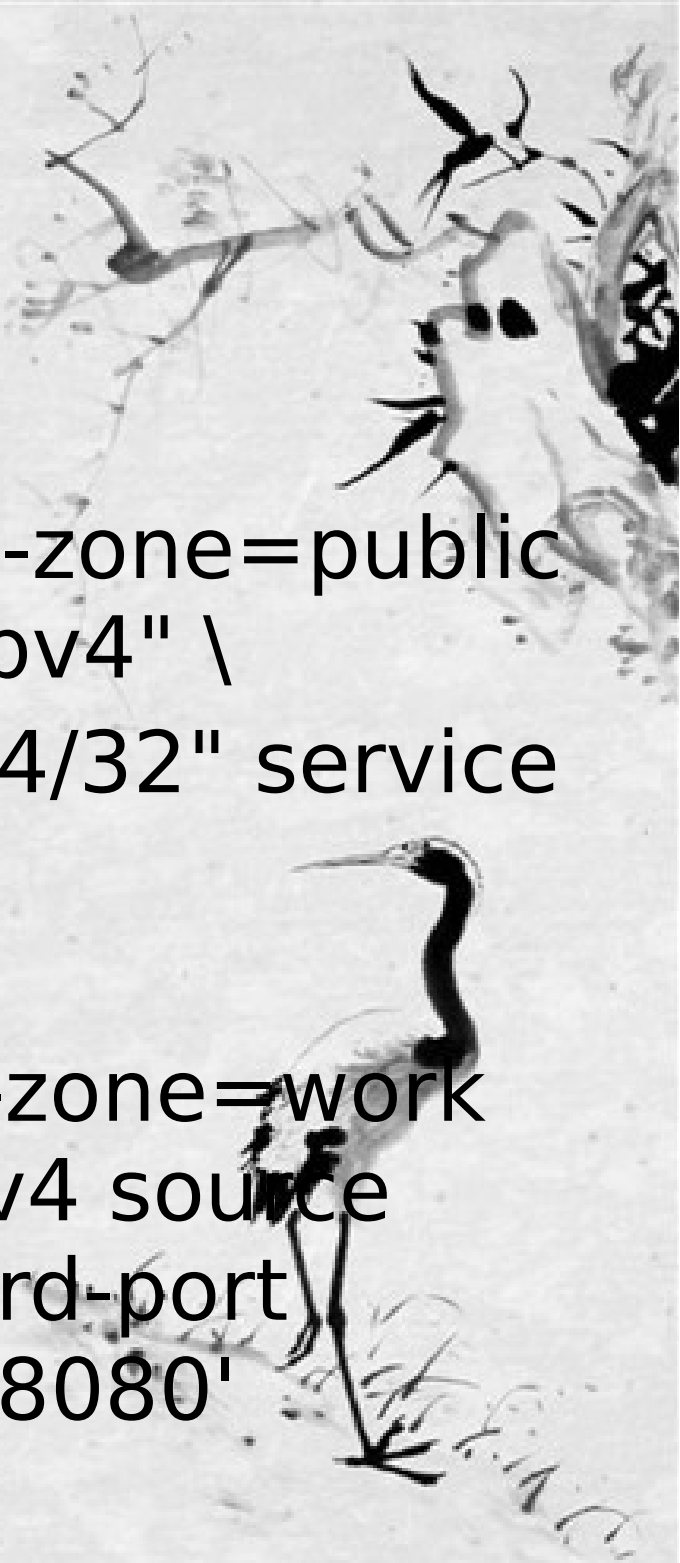
Firewalld

示例

2. 使 192.168.8.109 可以访问 192.168.8.123 的 http 服务

#firewall-cmd --add-rich-rule='rule family=ipv4 source address=192.168.8.109/32 destination address=192.168.8.123/32 service name=http accept'

Firewalld



创建一个指定规则

```
    # firewall-cmd --permanent --zone=public
--add-rich-rule="rule family="ipv4" \
    source address="192.168.0.4/32" service
name="http" accept"
```

```
    #firewall-cmd --permanent --zone=work
--add-rich-rule='rule family=ipv4 source
address=192.168.0.0/26 forward-port
port=80 protocol=tcp to-port=8080'
```

Firewalld

创建一个指定规则

#firewall-cmd --permanent --zone=work --add-rich-rule='rule family=ipv4 source address=192.168.0.0/24 forward-port port=80 protocol=tcp to-port=8080'

#firewall-cmd --permanent --zone=work --add-rich-rule='rule family=ipv4 source address=192.168.0.5/32 forward-port port=80 protocol=tcp to-port=8080 to-ipadd=192.168.0.1'

Firewalld

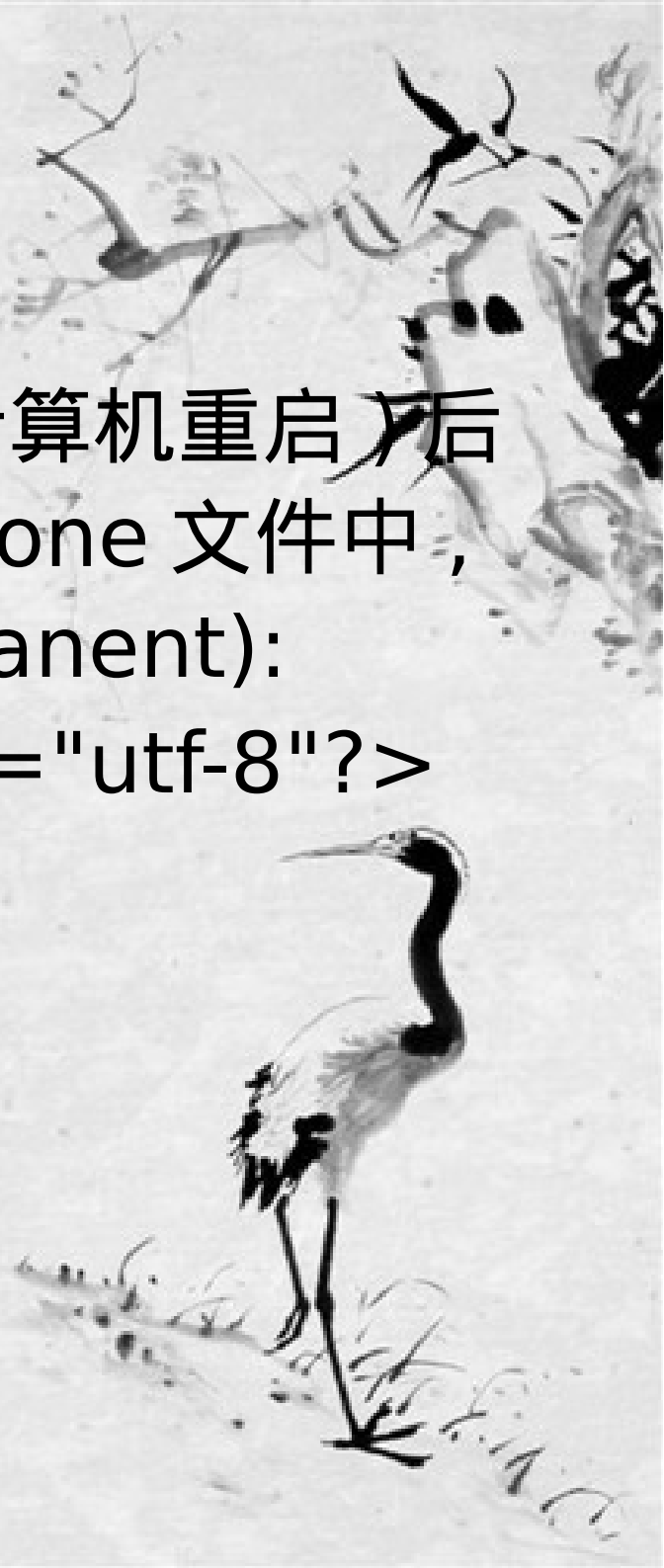
firewalld 区域文件所在

```
#cd /etc/firewalld/zones/
#vim public.xml
```

Firewalld

指定规则在每次 Firewalld 重启（计算机重启）后，将必须再次提交，可以直接写在 zone 文件中，以便于自动处理（或直接加 --permanent):

```
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>Public</short>
```



Firewalld

<description>For use in public areas. You do not trust the other computers on networks

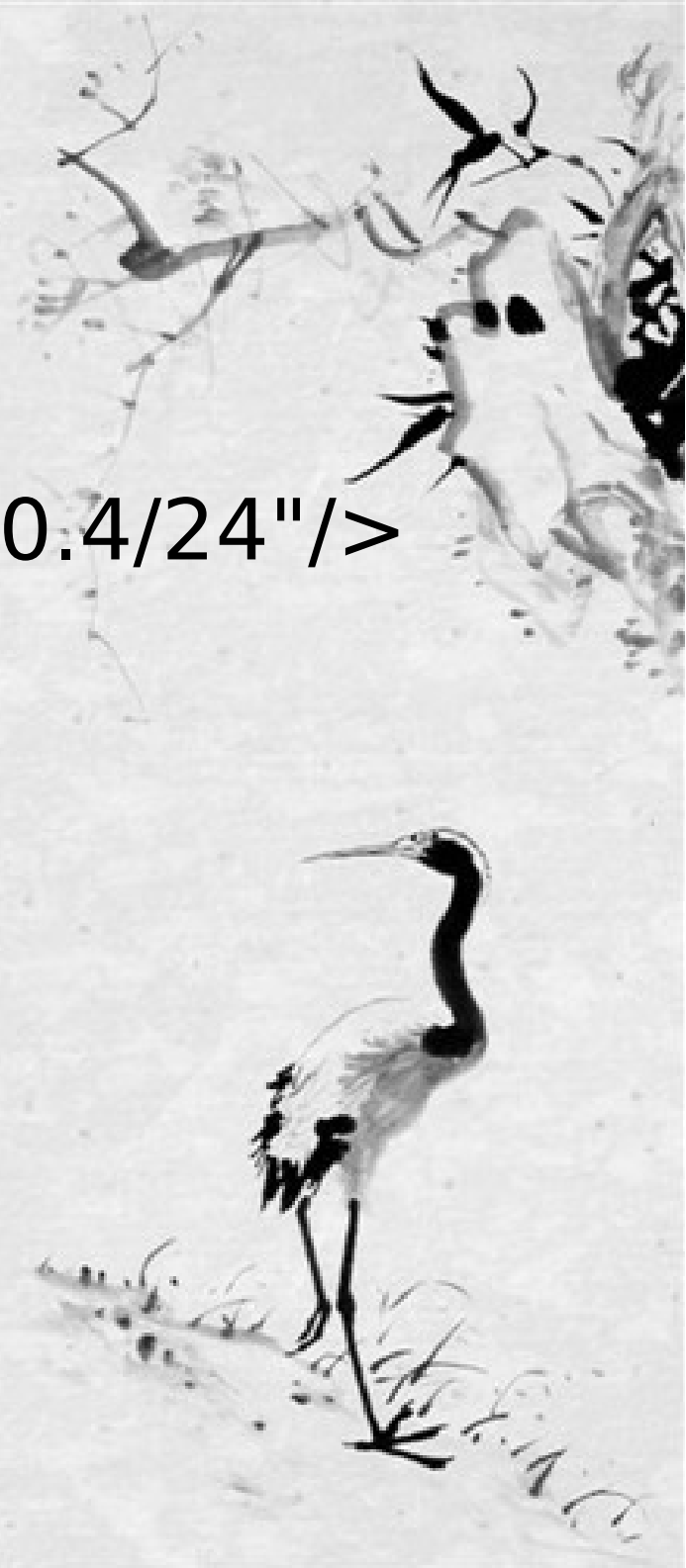
to not harm your computer. Only selected incoming connections are accepted.</description>

<service name="dhcpv6-client"/>

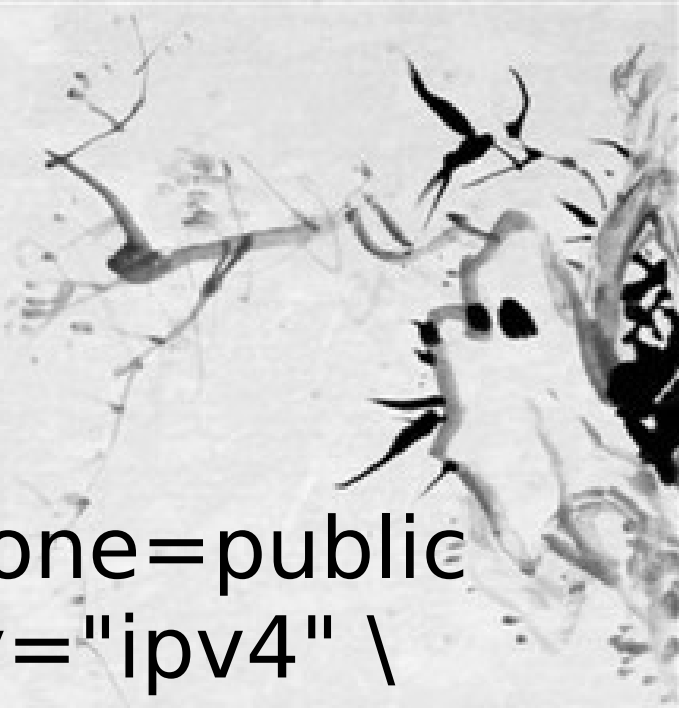
<service name="ssh"/>

Firewalld

```
<rule family="ipv4">
    <source address="192.168.0.4/24"/>
    <service name="http"/>
    <accept/>
  </rule>
</zone>
```



Firewalld



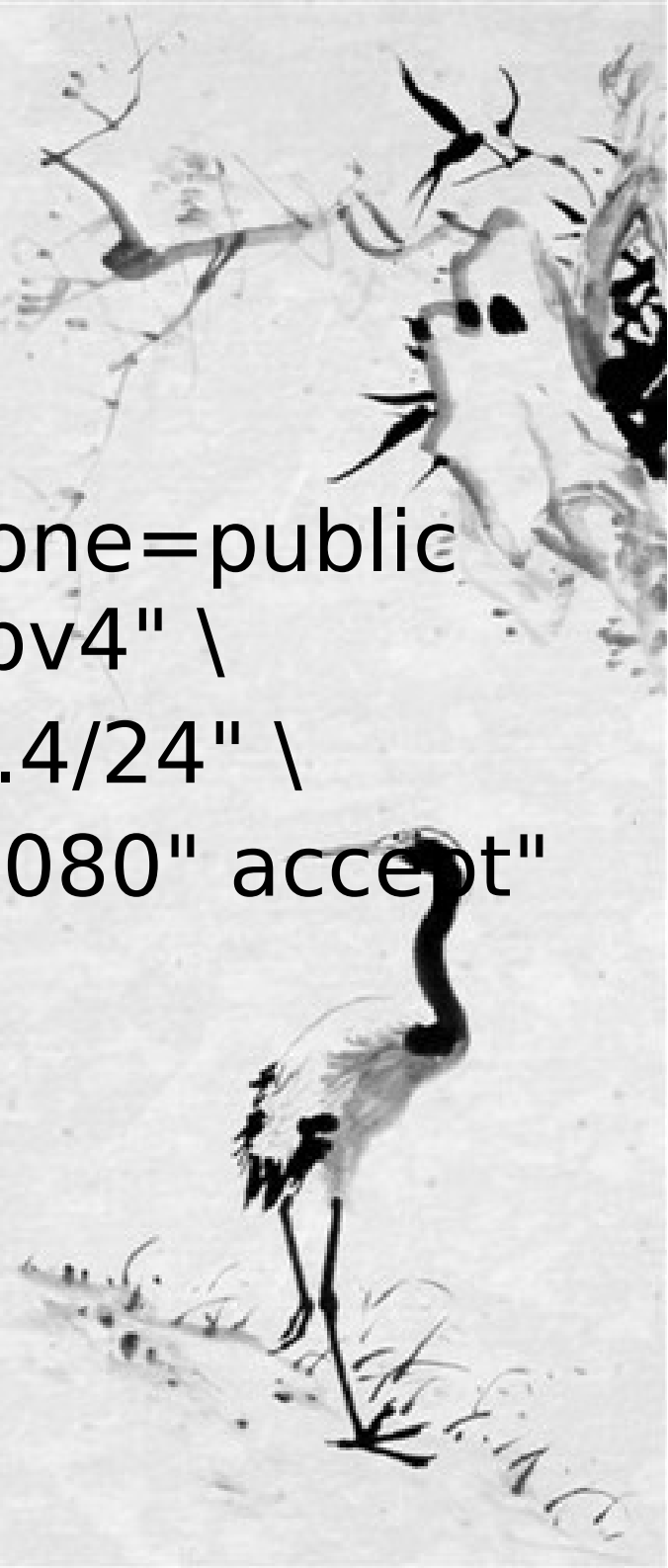
删除自定义规则

```
# firewall-cmd --permanent --zone=public
--remove-rich-rule="rule family="ipv4" \
    source address="192.168.0.4/24" service
name="http" accept"
```

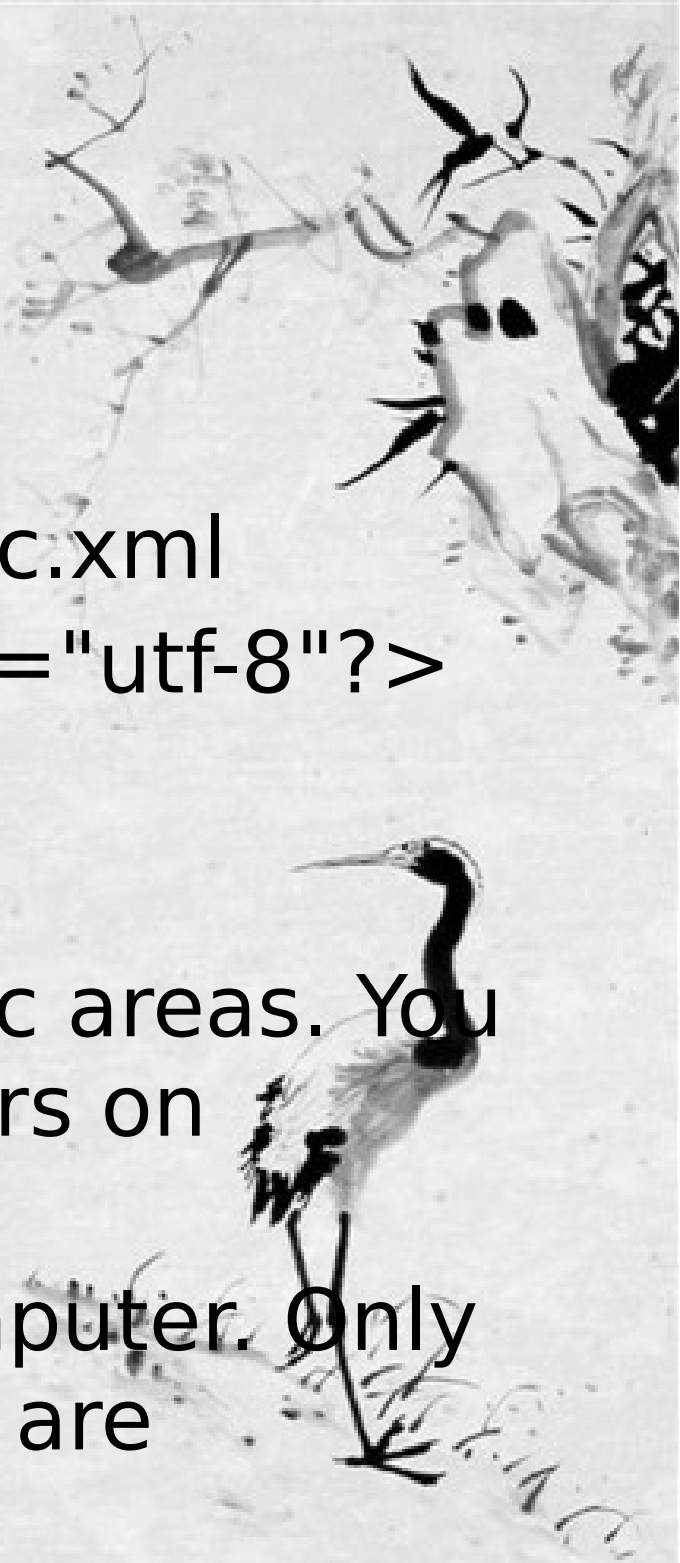
Firewalld

添加指定源至指定协议的指定端口

```
# firewall-cmd --permanent --zone=public
--add-rich-rule="rule family="ipv4" \
    source address="192.168.0.4/24" \
    port protocol="tcp" port="8080" accept"
```



Firewalld



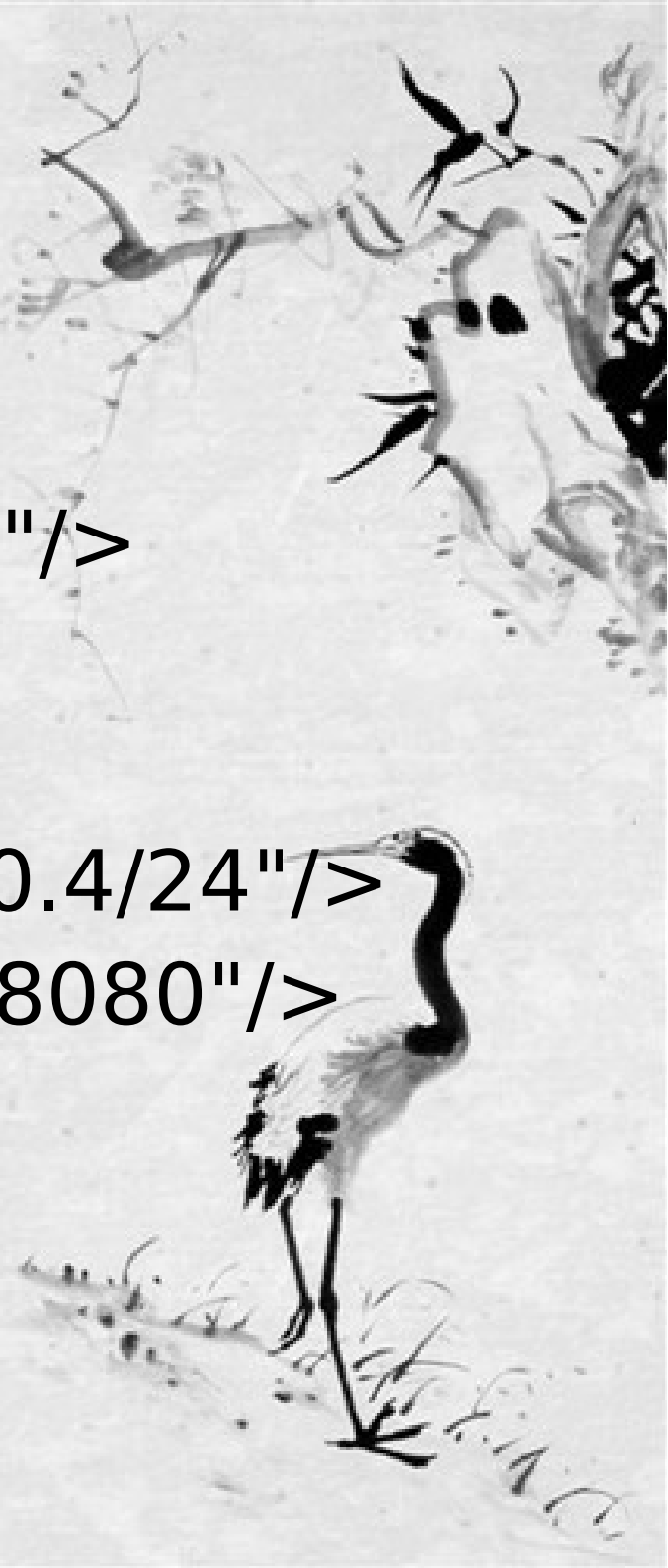
zone 文件写法

```
# cat /etc/firewalld/zones/public.xml
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>Public</short>
  <description>For use in public areas. You do not trust the other computers on networks
               to not harm your computer. Only selected incoming connections are accepted.</description>
```

Firewalld

zone 文件写法

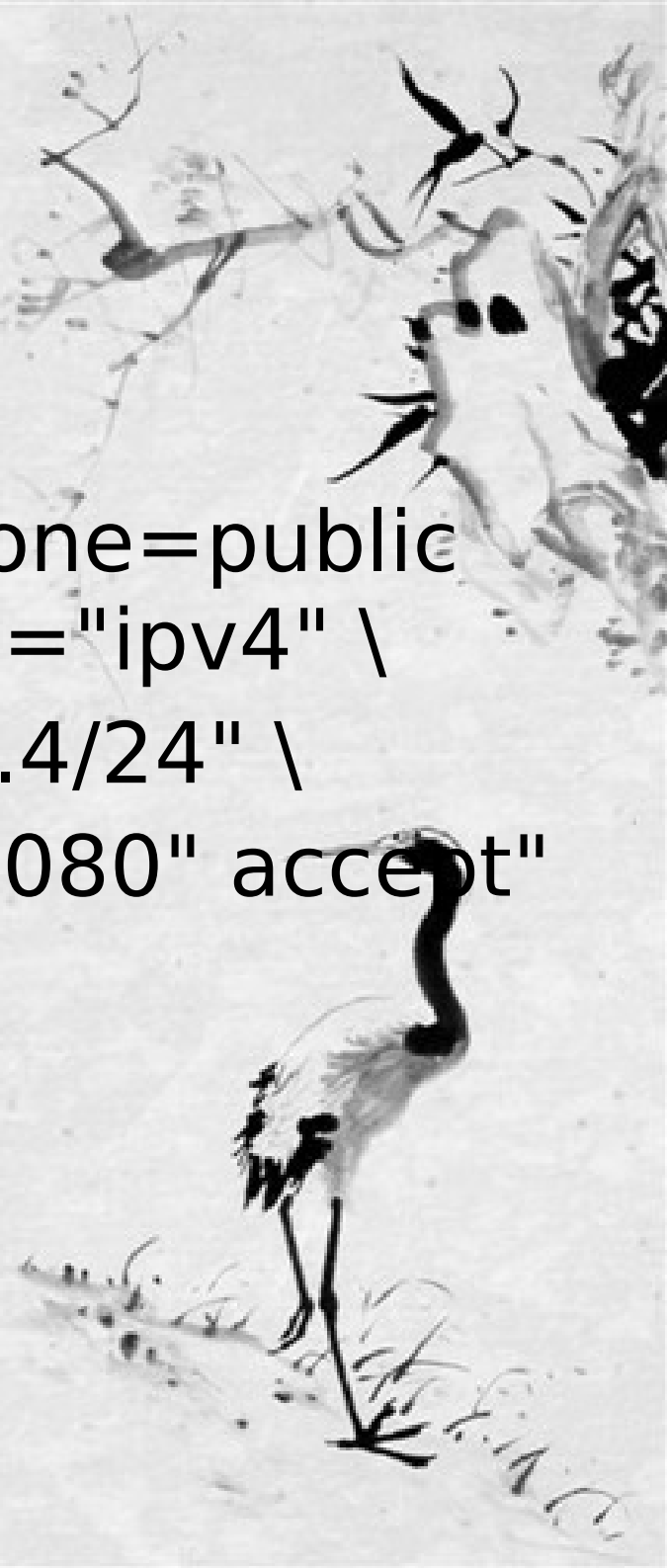
```
<service name="dhcpv6-client"/>
  <service name="ssh"/>
  <rule family="ipv4">
    <source address="192.168.0.4/24"/>
    <port protocol="tcp" port="8080"/>
    <accept/>
  </rule>
</zone>
```



Firewalld

删除指定源至指定协议的指定端口

```
# firewall-cmd --permanent --zone=public
--remove-rich-rule="rule family="ipv4" \
    source address="192.168.0.4/24" \
    port protocol="tcp" port="8080" accept"
```



Firewalld

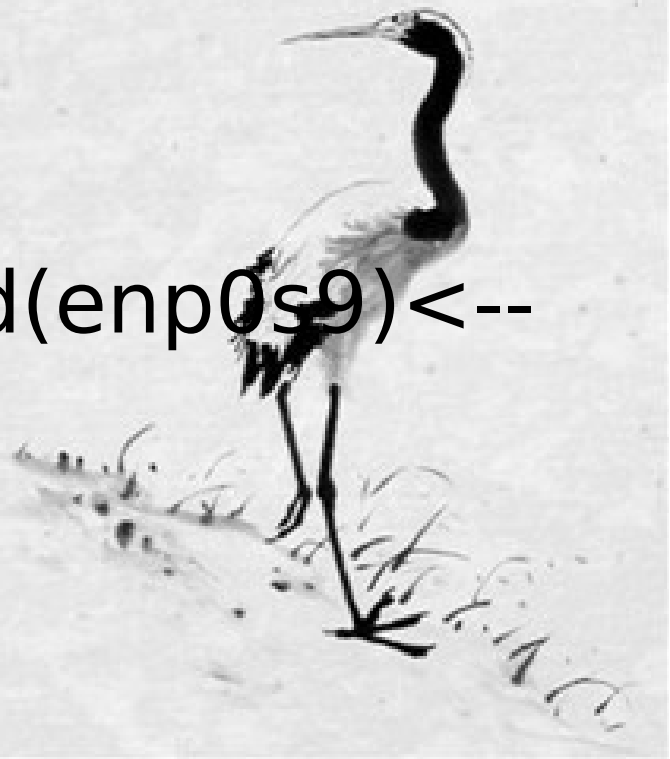
启用 masquerade(伪装)

默认情况下，区域 external 已启用 masquerade

#firewall-cmd --list-all --zone=external

如示意：

Internet<->(enp0s8)Firewalld(enp0s9)<-->t2(client)



Firewalld

实现 masquerade(伪装)

1. 当前默认 zone 为 public

   #firewall-cmd --list-all

2. 将 enp0s8 切换至 external

   #firewall-cmd --change-interface=enp0s8 --zone=external --permanent ← 永久性支持 , 即永久性切换至 external, 将此内容保存至 external.xml 文件中

Firewalld

实现 masquerade(伪装)

2. 将 enp0s8 切换至 external

...

#firewall-cmd --reload

#firewall-cmd --list-all --zone=external

Firewalld

实现 masquerade(伪装)

3. 重起 firewalld

#firewalld-cmd --complete-reload

4. 将 enp0s9 切换至 internal

#firewall-cmd --change-interface=enp0s9 --zone=internal --permanent

#firewall-cmd --list-all --zone=internal

Firewalld

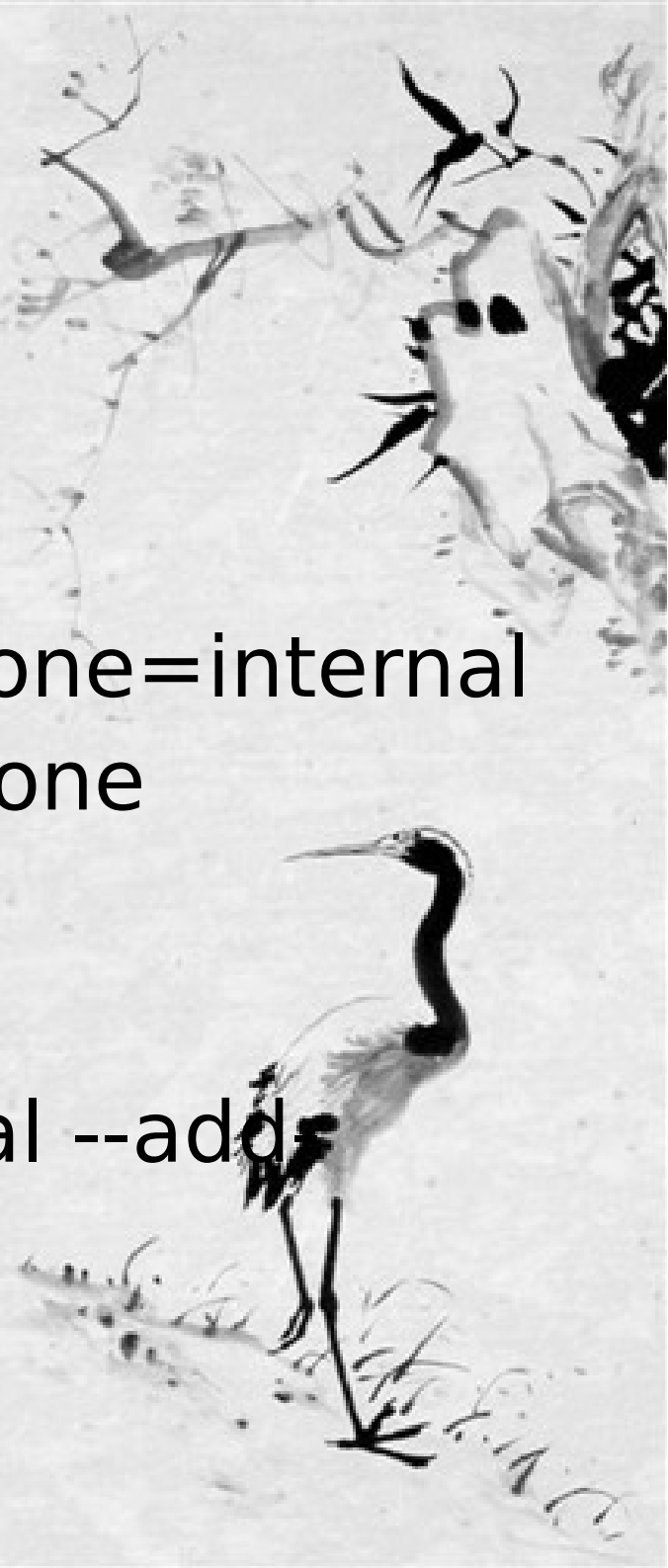
实现 masquerade(伪装)

5. 设置默认区域为 internal

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

6. 添加 NAT 所需的服务

```
#firewall-cmd --zone=internal --add-service=dns --permanent
```



Firewalld

实现 masquerade(伪装)

7. 对客户端设置 DNS 服务器地址

   #cat /etc/resolv.conf

   nameserver 202.160.0.20

   Nameserver 202.106.148.1

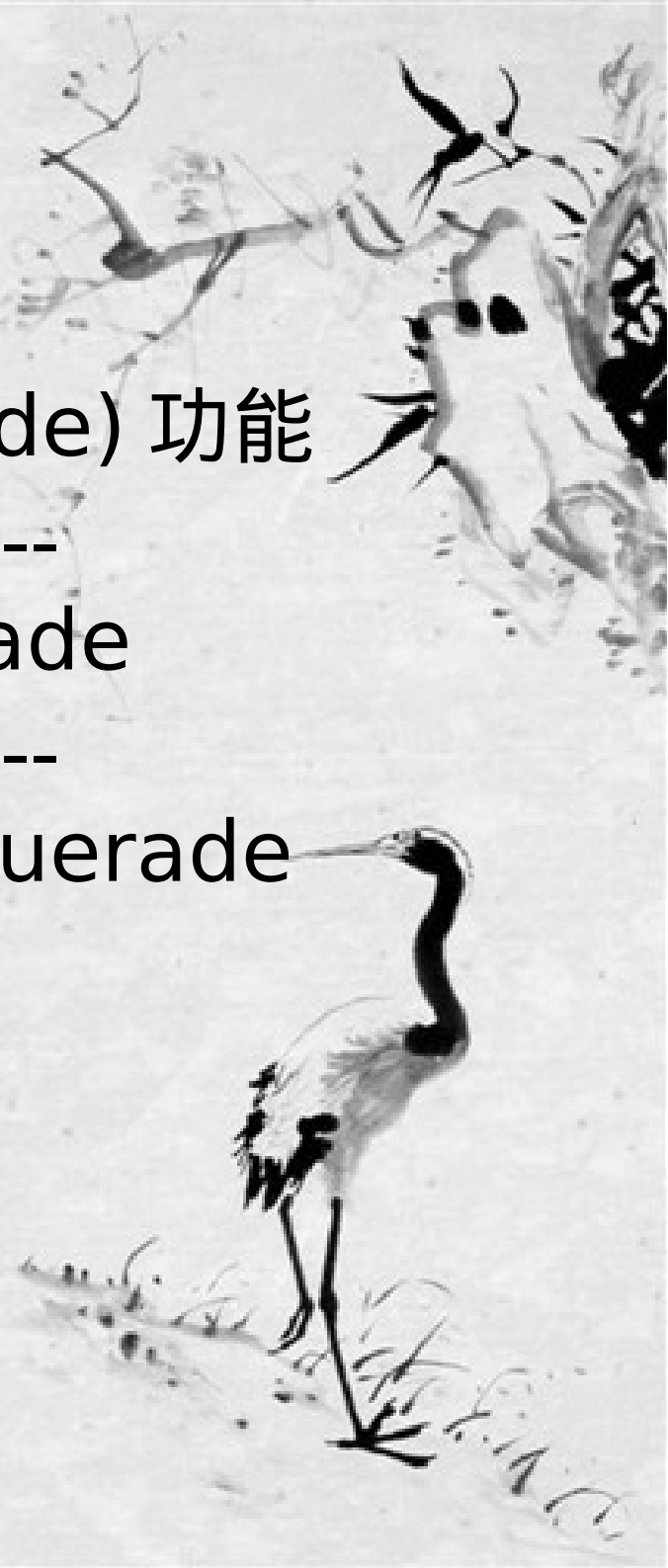
8. 测试

Firewalld

对区域增加 / 删除伪装 (masquerade) 功能

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

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



Firewalld

对区域增加 / 删除伪装 (masquerade) 功能 (富规则)

#firewall-cmd --permanent --zone=<ZONE> --add-rich-rule='rule family=ipv4 source address=192.168.0.0/24 masquerade'

#firewall-cmd --permanent --zone=<ZONE> --remove-rich-rule='rule family=ipv4 source address=192.168.0.0/24 masquerade'

Firewalld

查询某区域是否开启 masquerade

#firewall-cmd --zone=external --query-masquerad

Firewalld

firewalld 也可以使用图形工具来进行配置及完成

#firewall-conifg

或

应用程序 -> 杂项 -> 防火墙

Firewalld

创建自定义区域文件

当区域文件需要自定时，可以在 /usr/lib/firewalld/zones 和 /etc/firewalld/zones 中创建，当两个目录中都存在相同的区域， firewalld 将优先从 /etc/firewalld 中读取区域配置

用户可根据某个存在的区域当成模板进行修改

Firewalld

解读区域 public.xml

#cd /etc/firewalld/zones

#vim /etc/public.xml

Firewalld

解读区域 public.xml

<?xml version="1.0" encoding="utf-8"?>

←xml 格式标记

<zone> ←zone 开头

<short>Public</short> ← 区域名

<description> ← 区域说明

For use in public areas. You do not trust the other computers

on networks not to harm your computer. Only selected incoming

Firewalld

解读区域 public.xml

connections are accepted.

</description> ← 结束区域说明

<interface name="eno1"/> ← 区域包含的网卡

<source address="172.26.0.0/24"/> ← 源地址

<service name="ssh"/> ← 允许提供服务的服务名称

<port protocol="tcp" port="123"/> ← 允许使用的协议及端口号

Firewalld

创建自定义服务文件

当服务文件需要自定时，可以在 /usr/lib/firewalld/services 和 /etc/firewalld/services 中创建，当两个目录中都存在相同的区域， firewalld 将优先从 /etc/firewalld/services 中读取区域配置

用户可根据某个存在的区域当成模板进行修改

Firewalld

解读服务 smaba 文件

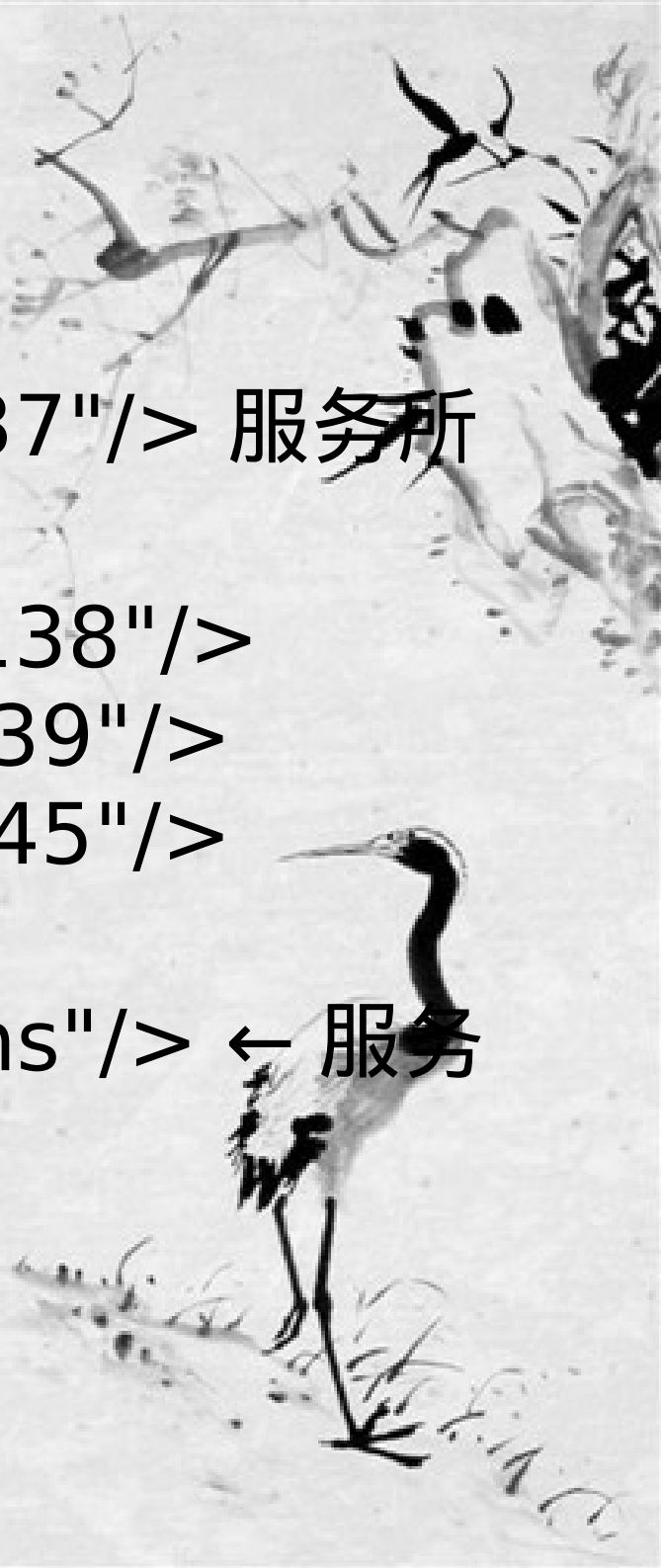
#cd /etc/firewalld/zones

#vim /etc/public.xml

Firewalld

<?xml version="1.0" encoding="utf-8"?>
←xml 格式定义
<service> ← 服务定义起始位置
  <short>Samba</short> ← 服务名称
  <description> ← 注释
    This option allows you to access and participate in Windows
    file and printer sharing networks. You need the samba package
    installed for this option to be useful.
  </description> ← 服务注释结束

Firewalld



<port protocol="udp" port="137"/> 服务所用协议及端口
  <port protocol="udp" port="138"/>
  <port protocol="tcp" port="139"/>
  <port protocol="tcp" port="445"/>
  <module name="nf_conntrack_netbios_ns"/> ← 服务所需要的 kernel 模块
</service> ← 服务定义结束位置