链滴

iptables 官方手册整理

作者：An

原文链接：https://ld246.com/article/1363584956910

来源网站：链滴

<p>目录：</p>
<div>
1. 简介
</div>
<div>
2. 首先，什么是包过滤?
</div>
<div>
3. 快速入门指南
</div>
<div>
4. 数据包过滤流程
</div>
<div>
5. 具体如何使用Iptables命令实现过滤功能
</div>
<div>
6. 地址转换（NAT)
</div>
<div>
7. 排除建议
</div>
<div>

</div>
<div>

</div>
<div>
1. 简介
</div>
<div>

</div>
<div>
读者们，大家好：
</div>
<div>

</div>
<div>
  在这里我们假设你已经有一定的IP地址、网络地址、子网掩码、路由、DNS基础知识。如果有建议你先阅读一下官网的networking-concepts-HOWTO文档。官网地址为：www.netfilter.org
</div>
<div>
  当前我们的网络是不安全的。但问题是我们做网络限制，同时还需要提供快速、便利的网络讯环境，而不是处于邪恶的目的。网络限制是一把双刃刀，但本文章并非解决这个问题。
</div>
<div>
  所以刀在手如何使用完全取决于你。我相信你会把这些工具用到合理的地方，并能很好地使这些工具。而不是用于其他目的。
</div>
<div>

---

2. 首先，什么是包过滤？

  包过滤是当一个数据包通过时，使用软件去查看包头信息并决定对该包的处理方式。你可以弃该数据包、接受该数据包亦或是其他更复杂的处理方式。

  在Linux中包过滤已经集成到内核中了，甚至还可以做一些数据包欺骗，但基本原则还是查数据包头并决定处理方式。

2.1 为什么我们需要包过滤？

  可控性、安全性、可监控性

  可控性：当你在局域网中使用Linux连接另一个网络时（如：互联网），你可以允许或拒绝定类型的数据。例如：数据包都会包含目标地址，这样你就可以防止数据包进入某个特定的网络。再个例子，我使用浏览器访问某个网站，在该网站上全是广告，此时浏览器会浪费我的时间去下载这些告信息。这时我可以告知包过滤工具不允许该网站的数据包通过以解决这个问题。

  安全性：当你的Linux主机是复杂的互联网与有序的局域网之间的唯一主机时，你可以通过据包限制让该Linux主机成为局域网与互联网之间的安全大门！比如：你可以会想允许所有的数据包入互联网，但你会对从外网进来的死亡之ping感到忧虑。再如，你可能不希望有人可以telnet连接你Linux主机，即使对方有密码也不可以。简单而言就是通过包过滤工具拒绝外网部分数据包进入本地。

  可监控性：当有些不正常的数据流量出现时，包过滤工具可以及时通知你是非常不错的注意

2.2 Linux系统如何过滤数据包

  Linux内核从1.1开始就已经有包过滤功能。第一代产品是1994年诞生于BSD系统的ipfw。Linux2.0增强了该功能；用户可以使用ipfwadm控制内核过滤规则。Linux2.2是用户工具变更为ipchains最后在Linux2.4中用户工具被重写，新的工具为iptables。

  你需要一个支持netfilter架构的Linux内核：netfilter是在内核中的过滤架构，而且该架构可使用插件动态加载。

  Linux防火墙主要包括两个部分：一部分为netfilter是内核过滤的基础架构，一部分为iptable是用户工具，用来编辑具体的过滤规则提供给内核netfilter。

  IPTABLES：该工具可以添加删除具体的过滤规则至内核包过滤表。这也意味着无论你如何置防火墙规则，一旦机器重启所有的规则将丢失。

  永久保存规则：你设置的防火墙规则被保存在内核中，但重启会丢失。你可以使用iptables-save和iptables-restore脚本实现永久保存与恢复。

  备注：Linux防火墙所有的规则被保存在表中，默认Iptables防火墙有4个表：filter表（实现滤功能），nat表（实现地址转换功能）， mangle表（修改数据包的TOS、TTL等信息），raw表（现数据包跟踪功能）

每个表中有多个数据链，而我们的具体规则被分门别类的链中。以下是每个表中的默认链：

filter表：INPUT链（入站数据过滤），FORWARD链（转发数据过滤），OUTPUT链（出站数据过）

nat表：PREROUTING链，POSTROUTING链，OUTPUT链

mangle表：PREROUTING链，POSTROUTING链，INPUT链，OUTPUT链，FORWARD链

raw表：OUTPUT链，PREROUTING链

---

3. 快速入门指南

  很多朋友使用单线PPP（拨号）连接互联网，并且不希望任何人访问你的网络，防火墙可以如下设置。

  首先加载过滤功能的模块：

#insmod ip_conntrack

#insmod ip_conntrack_ftp

以上两天命令也可以使用下面两天命令替换

#modprobe ip_conntrack

#modprobe ip_conntrack_ftp

  其次添加具体规则：

iptables -N block 新建规则链

iptables -A block -m state --state ESTABLISHED,RELATED -j ACCEPT 允许出站数据包的回应息

iptables -A block -m state --state NEW -i ! ppp0 -j ACCEPT 允许出站数据（自己可以访问外）

iptables -A block -j DROP 其余数据包全部丢弃

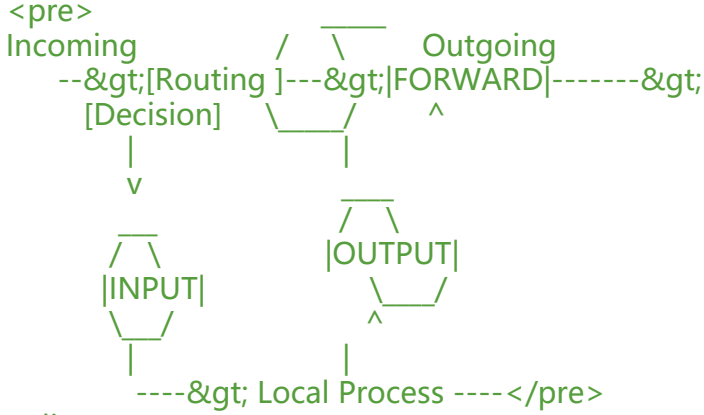
iptables -A INPUT -j block 入站数据时读取block链中的过滤规则

iptables -A FORWARD -j block 转发数据时读取block链中的过滤规则

---

4. 数据包过滤流程

  内核的过滤表默认有三个链（链中具体存放过滤规则）:INPUT,OUTPUT,FORWARD。

<pre>


</pre>

  三个圈中表示上面的三个链，当数据包到达上图中的某个圈中时，链即刻根据规则判断数据的处理方式，如果链中的某个规则说丢弃该包，该数据包将立即被丢弃；如果链中规则说接受，则数包继续向后传输。

  每个链中包含有具体的规则，每个规则具体明确说明：什么数据包头信息做怎样的处理，如你的数据包没有匹配第一个规则，则继续对比下一条规则。最后如果没有规则与你的数据包匹配，内将读取链的默认规则。出于安全的考虑，默认规则一般会被设置为丢弃（DROP）。

  1.数据包过来时内核先查看目标地址：这一步被称为路由。

  2.如果该数据是发往本地的，则继续向下传递至INPUT链，当INPUT链允许该数据包，数据进入本机等待程序接受数据。

  3.否则，如果内核未开启数据转发功能，被转发的数据包将被直接丢弃，如果内核开启了数转发功能，该数据包将传递给FORWARD链以转发数据，数据进入目标网络接口（网卡接口）；此时果 FORWARD链允许数据包通过，该数据包继续向后传递。

  4.最后在本机的一个程序发送网络数据包时，数据包会立刻进入OUPUT链，根据具体规则定允许或拒绝发送出去。

----

5. 具体如何使用Iptables命令实现过滤功能

iptables有非常详尽的手册文档(man iptalbes)，以下是iptables可以实现的几种不同的操作我们从filter过滤表开始。

  1. 创建新的自定义链 -N

  2. 删除自定义链 -X

  3. 改变默认策略 -P

  4. 显示链规则 -L

  5. 清空链中的规则 -F

  6. 将包过滤统计信息清零 -Z

 如果在链中维护具体规则：

   1. 追加新的规则 -A

   2. 插入新的规则 -I

   3. 替换旧的规则 -R

   4. 删除旧的规则 -D

 5.1 操作单条规则

   这是最基本的包过滤操作。通常你需要使用-A或-D命令选项，有时你还会使用到-I与-R命令项。

   每条规则需要指定匹配条件以及匹配后的处理方式（ACCEPT允许，DROP丢弃，REJECT拒，LOG日志等），如：你可能希望丢弃素有本地回环(127.0.0.1)的ICMP数据包，这样我们的匹配条是：ICMP协议并且源地址是127.0.0.1 匹配后做DROP处理。

   127.0.0.1是本地回环接口，即使你没有物理网卡，该接口一样存在。你可以使用ping命令生这类数据包。

 # ping -c 1 127.0.0.1

 PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.

 64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=21.9 ms

--- 127.0.0.1 ping statistics ---

1 packets transmitted, 1 received, 0% packet loss, time 0ms

rtt min/avg/max/mdev = 21.966/21.966/21.966/0.000 ms

iptables -A INPUT -s 127.0.0.1 -p icmp -j DROP 添加一条规则

ping -c 1 127.0.0.1

PING 127.0.0.1 (127.0.0.1): 56 data bytes

--- 127.0.0.1 ping statistics ---

1 packets transmitted, 0 packets received, 100% packet loss

  你可以看到第一次ping是成功的(-c 1 说明仅ping一次)，然后我们追加了一条规则到NPUT链，该规则指定从127.0.0.1发送的ICMP协议的数据包将被丢弃。第二次再执行ping命令所有数据100%丢失。

  我们有两种方式可以删除规则，首先我们知道INPUT链中只有一条规则，我们可以使用编号除：

#iptables -D INPUT 1 删除INPUT链中的第一条规则

  第二种方法类似与-A选项，使用-D替换-A。当你的规则比较复杂并搞不清编号时可以使用种方式：

#iptables -D INPUT -s 127.0.0.1 -p icmp -j DROP

5.2 特定过滤规则

  上面我们已经看到可以使用-p指定协议，-s指定源地址，但还有很多可以用来过滤的条件匹符。下面我们分别介绍：

  源地址与目标地址

  源地址(-s,--source或--src)，目标地址(-d,--destination或--dst)有四种使用方式：最常用是使用名称，比如"localhost"或者"http://www.kernel.org"。第二种方法使用IP地址如"127.0.0.1"。第三四种方法可以匹配IP地址区域，如"199.95.207.0/4"或"199.95.207.0/255.255.255.0"。它们都可以匹配199.95.207.0到199.95.20地址可以使用0/0匹配所有地址。

iptables -A INPUT -s 0/0 -j DROP 拒绝所有源地址访问本机

  取反匹配

  很多标签"-s","-d"等都可以在后面添加"!"以表示否匹配，如"-s ! localhost"将匹配所有非本地源地址。

   协议匹配

   匹配协议可以使用-p标签,协议的指定可以使用数字编号（如果你知道协议的编号)或使用名称如"TCP","UDP","ICMP"等)。协议名称前可以添加"!&uot;如"-p ! TCP"匹配所有非TCP协议数据包。

   接口匹配

   "-i(--in-interface)"和"-o(--out-interface)"匹配指定的接口。接是真实的物理网卡接口，-i(数据包从哪个网卡进来的),-o(数据从哪个网卡出去的)。你可以使用ifconfi命令查看哪些接口是开启的。

   注意：在INPUT链不可以使用-o选项，因为入站的数据不走出站接口。所以在INPUT链中的-规则将无法匹配任何数据。

         同理，在OUTPUT链中不可以使用-i选项。

   数据段匹配

   很多数据包因为太大无法一次完成数据的传输。此时数据包将被分割为数据片段再发送出去接收端接受完数据后将把这些数据片段重新组合成完整的数据包。

   但问题在于当数据被分割后，只有前面的初始数据片段包含全部的数据头部信息（IP,TCP,UP,ICMP等），后续的数据片段仅包含数据包头部信息的一部分信息。这时去再检查后续数据片段的部信息是不可能的。

   当然，如果你想匹配第二个及后面被分片的数据，可以使用"-f"选项。

 # iptables -A OUTPUT -f -d 192.168.1.1 -j DROP 丢弃发送至192.168.1.1的所有数据以及分片数据

   扩展iptables规则

   iptables有很好的可扩展性，也就是说内核架构与iptables工具都可以添加扩展功能。

   内核功能扩展一般放在内核模块子目录中:/lib/modules/2.6.32-220.el6.i686/kernel/net/nefilter(这里以CentOS6.2为例)。这些模块在你使用iptables时会自动加载。

   mac

   该模块需要使用"-m mac"选项启用，这对应过滤进站数据包的源MAC址很用帮助。

 #iptables -I INPUT -m mac --mac-source 00:60:0891:CC:B7 -j REJECT 拒绝MAC地址为00:60:08:91:CC:B7的数据包进入

   limit

   该模块需要使用"-m limit"选项启用，该功能对限制网速很有效。

 #iptables -A OUTPUT -p tcp -m limit --limit 100/s -j ACCEPT 每秒包个数100以内将允许发送

 #iptables -P OUTPUT REJECT 默认拒绝所有

   说明：以上两天可以实现，当每秒包个数大于100时，拒绝所有连接。

   应用实例

 Syn-flood protection:

  # iptables -A FORWARD -p tcp --syn -m limit --limit 1/s -j ACCEPT

 Furtive port scanner:

  # iptables -A FORWARD -p tcp --tcp-flags SYN,ACK,FIN,RST RST -m limit --limit 1/s -j ACCEPT

 Ping of death:

  # iptables -A FORWARD -p icmp --icmp-type echo-request -m limit --limit 1/s -j ACCPT

   状态匹配

   该模块需要使用"-m state"选项启用，数据包的状态包括：NEW,ESTALISHED,RELATED,INVALID

   NEW：创建连接的数据包

   ESTABLISHED：通过已经创建的连接通道传输的数据包

   RELATED：与已经创建的连接相关的数据包，如ICMP错误数据包

   INVALID：无法识别的数据包

 #iptables -A INPUT -m state --state NEW -j &nbs;DROP   拒绝进站的连接请求（外网无法访问本机）

#iptables -A INPUT -m state --state RESTABLISHED,RELAED -j ACCEPT   允许外网数据对本机的回应信息

#iptables -P OUTPUT ACCEPT 允许访问外网

---

6. 地址转换（NAT）

<pre>[ ]
|>>> [Internet] >>>> (www.google.com)
| [ ]
v
v
_____
[eth0:202.106.22.31]
[网关]
[___eth1:192.168.1.1___]
|
|
v
v
[ ] >>> (PC:192.168.1.200)
[交换机] >>> (PC:192.168.1.201)
[ ] >>> (PC:192.168.1.202)</pre>

  上图中局域网（192.168.1.0/24)通过交换机与公司网关连通在一起，网关与互联网可以实通讯。现在我们使用iptables的SNAT功能实现内网所有主机上网。

  实现步骤：

  1. 开启网关路由功能（路由器就有该功能，如果是Linux软路由可以使用：#echo "1quot; > /proc/sys/net/ipv4/ip_forward命令开启）

  2. 设置iptables的SNAT功能：

#iptables -t nat -A POSTROUTING -s 192.168.1.0/4 -o eth0 -j SNAT --to-source 202.106.22.31

  上图中如果需要实现通过在外网任意主机通过访问公司网关IP地址：202.106.22.31，即可问位于公司内部192.168.1.200上的Web服务器，我们可以使用iptables的DNAT功能实现。

  实现步骤：

  1. 开启网关路由功能

  2. 设置iptables的DNAT功能：

#iptables -t nat -A PREROUTING -i eth0 -d 202.106.22.31 --dport 80 -j DNAT --to-destination   192.168.1.200

7. 排除建议

  1.注意过滤规则的顺序，如果规则的第一条拒绝所有tcp连接，第二条允许192.168.1.1访问机的tcp连接，则第二条将无效。

#iptables -A INPUT -p tcp -j DROP

#iptables -A INPUT -p tcp -s 192.168.1.1 -j ACCEPT

  以上两条规则由于第一条规则已经丢弃所有的tcp数据包，所以不会再匹配第二条规则。

  2.设置完规则后未保存，导致重启后所有规则丢失。解决方法：可以使用iptables-save，或service iptables save实现永久保存

#iptables-save > /etc/sysconfig/iptables

#service iptables save

  以上以CentOS为例，两天命令任选其一即可永久保存。

  3.无效的规则及时删除，否则影响效率。

  4.匹配端口号时必须指定协议，否则会报错。

  5.公司有FTP服务器时，提前加载ftp模块：#modprobe ip_nat_ftp

转自：http://my.oschina.net/u/945017/blog/107791