在 Linux 下用 iptables 构建防火墙

张惠卿　严　峰　沈金龙

（南京邮电学院　南京 210003）

摘　要　简要介绍了 Linux中防火墙的发展概况，着重讲述了基于 Linux2.4 内核版本的 netfilter 结构框架及 iptalbe的使用方法，并给出相应的例子。

关键词　防火墙　iptablesnetfilter

iptables(netfilter)应用程序，被认为是 Linux 中实现包过滤功能的第四代应用程序。第一代是 Linux内核 1.1 版本所使用的 AlanCox 从 BSD UNIX 中移植过来的 ipfw。在 2.0 版的内核中，JosVos 和其他一些程序员对 ipfw进行了扩展，并且添加了 ipfwadm用户工具。在 2.2 版内核中，Russell和 MichaelNeuling做了一些非常重要的改进，在该内核中，Russell添加了帮助用户控制过滤规则的 ipchains工具。现在，Russell又完成了名为 netfilter 的内核框架。

netfilt的目的是为用户提供一个专门用于包过滤的底层结构，并且用户和开发人员可以将其建在 Linux内核中。而 iptable是一个内建在 netfilter 框架中的模块，它可以让用户访问内核过滤规则和命令。事实上 iptable和 ipchain非常相似。

1　netfilt结构框架

netfilt是 Linux24 内核中实现包过滤 /包处理 /NAT 等功能的框架。当前的 netfilt框架在 IPv4、IPv6及 DECnet 等网络协议栈中被实现，它比以前任何一版 Linux内核的防火墙子系统都要完善强大，提供了一个抽象、通用化的框架，包含以下 3 部分：

（1）为每种网络协议（IPv4、IPv6、DECnet 等）定义一套钩子函数（IPv4中定义了 5 个钩子函数）,这些钩子函数在数据包流过协议栈的几个关键点时被调用。在这几个点中，协议栈将数据包及钩子函数标号作为参数来调用 netfilt框架。

（2）内核的任何模块都可以对每种协议的一个或多个钩子进行注册，实现挂接，这样当某个数据包被传递给 netfilt框架时，内核能检测出是否有任何模块对该协议和钩子函数进行了注册。若注册了，则调用该模块注册时使用回调函数，这样这些模块就有可能检查、修改、丢弃该数据包或指示 netfilt将该数据包传入用户空间的队列。

（3）传递给用户空间的数据包队列是异步地进行处理。一个用户进程能检查、修改数据包，甚至可以重新将该数据包通过离开内核的同一个钩子函数中注入到内核中。

有了 netfilt结构框架，内核网络代码中就不再到处是混乱的修改数据包的代码了。基于 2.2 内核版本的 ipchain没有提供传递数据包到用户空间的框架，则任何需要对数据包进行处理的代码都必须运行在内核空间里，而内核编程非常复杂，并且只能用 C 语言实现，这样容易出现错误并对内核稳定性造成威胁。netfilt结构框架能很好地避免这种情况。图 1 为数据包流经 neftit系统示意图。

IPv4共有 5 个钩子函数，分别为：

[1]NF_IP_PRE_ROUTING　，

[2]NF_IP_LOCAL_IN　，

[3]NF_IP_FORWARD　，

[4]NF_IP_POST_ROUTING　，

[5]NF_IP_LOCAL_OUT　。

数据包从左边进入系统，进行 IP 校验以后，经过第一个钩子函数 NF_IP_PRE_ROUTING　进行处理；然后就进入路由代码，路由代码决定该数据包

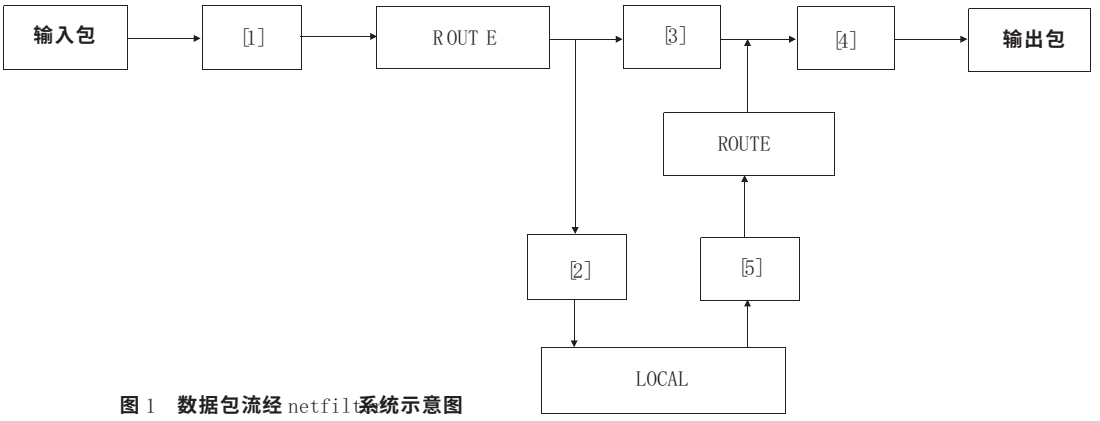

图 1　数据包流经 netfilter 系统示意图

是需要转发还是发给本机的；若该数据包是发给本机的，则该数据经过钩子函数 NF_IP_LOCAL_IN 处理以后再传递给上层协议；若该数据包应该被转发则它被 NF_IP_FORWARD 处理；经过转发的数据包经过最后一个钩子函数 NF_IP_POST_ROUTING 处理以后，再传输到网络上。

本地产生的数据经过钩子函数 NF_IP_LOCAL_OUT 处理以后，进行路由选择处理，然后经过 NF_IP_POST_ROUTING 处理，再发送到网络上。

内核模块可以对一个或多个这样的钩子函数进行注册挂接，并且在数据包经过时，调用这些钩子函数，从而模块可以修改数据包，并向 netfilter 返回如下值：

NF_ACCEPT　继续正常传输数据包；

NF_DROP　丢弃该数据包，不再传输；

NF_STOLEN　模块接管该数据包，不要继续传输该数据包；

NF_QUEUE　对该数据包进行排队（通常用于将数据包传给用户空间进程处理）；

NF_REPEAT　再次调用该钩子函数。

2　iptables 数据包选择系统

在 Linux24 内核中，采用了一个基于 netfilter 框架的数据包选择系统，用于实现数据包过滤（Filter 表），网络地址转换（NAT 表）及数据包处理（Mangle 表）功能。内核模块可以注册一个新的规则表（Table），并要求数据包流经指定的规则表。它有两种实现：iptables（IPv4）及 ip6tables（IPv6），两者都基于相同的库和基本上相同的代码。在本文中，主要对 iptables 进行讨论。iptables 是 ipchains 的后继工具，但却具有更强的可扩展性。iptables 通过插入/删除/修改规则链中的规则，可以对所有的 IP 表进行处理，包括 Filter，NAT 及 Mangle3 个表格，及以后扩展的表模块。它支持插件以匹配新的参数和目标动作，因此对 netfilter 的任何扩展都非常简单，仅仅需要编写一个完成实际目标动作处理的模块和 iptables 插件（动态连接库）来添加所需要的一切。Linux2.4 内核提供的这 3 种数据包处理功能都基于 netfilter 的钩子函数和 IP 表。它们是独立的模块，却完美地集成到由 netfilter 提供的框架中。

2.1　包过滤

filter 表格不会对数据包进行修改，而只对数据包进行过滤。iptables 优于 ipchains 的一个方面就是它更为小巧和快速。iptables 通过钩子函数 NF_IP_LOCAL_IN,NF_IP_FORWARD 及 NF_IP_LOCAL_OUT 接入 netfilter 框架，因此对于任何一个数据包只有一个地方对其进行过滤，这相对 ipchains 来说是一个巨大的改进，因为在 ipchains 中任一个被转发的数据包都要遍历 3 条链。

2.2　NAT

NAT 表格监听 3 个 netfilter 钩子函数：NF_IP_PRE_ROUTING，NF_IP_POST_ROUTING 及 NF_IP_LOCAL_OUT。NF_IP_PRE_ROUTING 实现对需要转发的数据包的源地址进行地址转换；而 NF_IP_POST_ROUTING 则对需要转发的数据包的目的地址进行地址转换。对于本地数据包的目的地

址的地址转换则由 NF_IP_LOCAL_OUT 来实现。NAT 表格不同于 Filter 表格，因为只有新连接的第一个数据包将遍历表格，而随后的数据包将根据第一个数据包的结果进行同样的转换处理。NAT 表格用于源 NAT 、目的 NAT 、伪装（其是源 NAT 的一个特例）及透明代理（其是目的 NAT 的一个特例）。

2.3 数据包处理（Packet mangling）

Mangle 表格在 NF_IP_PRE_ROUTING 和 NF_IP_LOCAL_OUT 钩子中进行注册。使用 Mangle 表可以实现对数据包的修改，或给数据包附上一些带外数据。当前 Mangle 表支持修改 TOS 位及设置 skb 的 nfmard 字段。

3 iptables 基本用法

iptables 命令基本上包含如下 5 部分：

（1）希望工作在哪个表上（Filter、NAT 、Mangle）；

（2）希望使用该表的哪个链（INPUT 、OUT - PUT 、FORWARD ）；

（3）进行的操作（插入、添加、删除、修改）；

（4）对特定规则的目标动作；

（5）匹配数据包条件。

3.1 iptables 的基本操作

-A 在链尾添加一条规则，

-I 插入规则，

-D 删除规则，

-R 替代一条规则，

-L 列出规则。

3.2 基本目标动作（适用于所有的链）

ACCEPT 接收该数据包，

DROP 丢弃该数据包，

QUEUE 排队该数据包到用户空间，

RETURN 返回到前面调用的链，

FOOBAR 用户自定义链。

3.3 基本匹配条件（适用于所有的链）

-p 指定协议（tcp/icmp/udp），

-s 源地址（ipaddress/masklen），

-d 目的地址（ipaddress/masklen），

-i 数据包输入接口，

-o 数据包输出接口。

3.4 iptables 的基本语法

iptables -t table -Operation chain -j target match (es)

例如，希望添加一个规则，允许所有从任何地方到本地 http 端口的连接：

iptables -t filter -A INPUT -j ACCEPT -p tcp --dport http

当然，还有其他的对规则进行操作的命令，如清空链表，设置链缺省策略，添加一个用户自定义的链等。

4 利用 iptables 进行数据包过滤

我们知道，Filter 表和 3 个钩子函数进行了挂接，因此提供了 3 条链表进行数据包过滤。所有来自网络并且发给本机的数据包会遍历 INPUT 规则链；所有被转发的数据包将仅仅遍历 FORWARD 规则链；而本地发出的数据包将遍历 OUTPUT 链。

4.1 数据包匹配手段

TCP 匹配扩展能匹配源端口、目的端口及 tcp 标记的任意组合，tcp 选项等。

UPD 匹配扩展能匹配源端口和目的端口。

ICMP 匹配扩展能匹配 ICMP 类型。

MAC 匹配扩展能匹配接收到的数据的 mac 地址。

MARK 匹配扩展能匹配 nfmark。

OWNE 匹配扩展（仅仅应用于本地产生的数据包）来匹配用户 ID、组 ID、进程 ID 及会话 ID。

LIMIT 匹配扩展用来匹配特定时间段内的数据包限制。这个匹配扩展对于限制 DoS 攻击数据流非常有用。

TOS 匹配扩展用来匹配 IP 头的 TOS 字段的值。

STATE 匹配扩展用来匹配特定状态下的数据包（由连接跟踪子系统来决定状态），可能的状态包括：

（1）INVALID （不匹配于任何连接），

（2）ESTABLISHED （属于某个已经建立的链接的数据包），

（3）NEW （建立连接的数据包），

（4）RELATED （和某个已经建立的连接有一定相关的数据包，例如一个 ICMP 错误消息或 ftp

数据连接)。

4.2 iptables的数据包过滤目标动作扩展

LOG 将匹配的数据包传递给 syslog() 进行记录；

ULOG 将匹配的数据用用户空间的 log进程进行记录；

REJECT 不仅仅丢弃数据包，同时返回给发送者一个可配置的错误信息；

MIRROR 互换源和目的地址以后重新传输该数据包。

5 利用 iptables进行 NAT

Linux以前的内核仅仅支持有限的 NAT 功能，被称为伪装。netfilt则支持任何一种 NAT 。一般来讲 NAT 可以分为源 NAT 和目的 NAT 。 源 NAT 在数据包经过 NF_IP_POST_ROUTING 时修改数据包的源地址。伪装是一个特殊的 SNAT 。而目的 NAT 在数据包经过 NF_IP_LOCAL_OUT 或 NF_IP_PRE_ROUTING 时修改数据包目的地址。端口转发和透明代理都是 DNAT 。

1. SNAT

变换数据包的源地址。

iptables-t nat -A POSTROUTING -j SNAT --to-source1.2.3.4

2. MASQUERADE

用于具有动态 IP 地址的拨号连接的 SNAT ，但是如果连接断开，所有的连接跟踪信息将被丢弃，而去使用重新连接以后的 IP 地址进行 IP 伪装。

iptables-t nat-A POSTROUTING -jMASQUER -ADE -o ppp0

3. DNAT

转换数据包的目的地址，这是在 PRE_ ROUT -ING 钩子链中处理的，也就是在数据包刚刚进入时。因此 Linux随后的处理得到的都是新的目的地址。

iptables-t nat -A PREROUTING -j DNAT --to-destination1.2.3.4:8080 -p tcp--dport80 -ieth1

4. REDIRECT

重定向数据包的目的为本地，和 DNAT 将目的地址修改为接到数据包的接口地址情况完全一样。

iptables-t nat-A PREROUTING -j REDIRECT --to-port3128 -ieth1-p tcp--dport80

6 利用 iptables进行数据包处理 (Packet mangling)

Mangle表提供了修改数据包各个字段的值的方法。

1. MARK

设置 nfmark字段的值。我们可以修改 nfmark 字段的值。nfmark仅仅是一个用户定义的数据包的标记（可以是无符号长整数范围内的任何值)。该标记值用于基于策略的路由，通知 ipqmpd （运行在用户空间的队列分捡器守护进程）将该数据包排队给哪个进程等信息。

例 ： iptables-t mangle -A PREROUTING -j MARK --set-mark0x0a-p tcp

2. TOS

设置数据包的 IP 头的 TOS 字段值。若希望适用基于 TOS 的数据包调度及路由，这个功能是非常有用的。

例：iptables-t mangle-A PREROUTING -jTOS --set-tos0x10 -p tcp--dportssh

张惠卿，1978 年生，女，广东惠州人。南京邮电学院计算机科学与技术系计算机应用专业硕士研究生，2000 年毕业于南京邮电学院计算机科学与技术系。目前研究方向为计算机通信及网间互连技术。

严 峰，1975 年生，男，江苏南京人。南京邮电学院通信工程系硕士研究生，1997 年毕业于南京航空航天大学电子工程系。目前研究方向为通信协议的一致性测试。

沈金龙，1942 年生，男，上海市人。南京邮电学院计算机科学与技术系教授。目前研究方向为计算机通信及网间互连技术。