第29卷 第8期 计 算 机 工 程 2003年5月
Vol.29 № 8 Computer Engineering May 2003
· 安全技术 · 文章编号：1000—3428(2003)08—0112—02 文献标识码：A 中图分类号： TP393.08

Linux内核防火墙Netfilter实现与应用研究

姚晓宇，赵 晨
(东南大学计算机科学与工程系，教育部计算机网络与信息集成重点实验室，南京 210096)

摘 要：介绍了Linux内核防火墙的发展，对2.4.x内核中的Netfilter框架的流程和IPv4协议栈中Netfilter的实现进行了分析，通过一个内核防火墙模块实例介绍了基于Netfilter框架下的内核防火墙设计方法，对Netfilter框架下的防火墙高级功能扩展进行了展望。

关键词：防火墙；Linux；Netfilter；内核模块

Research on Implementation and Application of Linux Kernel Firewall Netfilter

YAO Xiaoyu，ZHAO Chen
(The Key Lab of Computer Network and Information Integration, Ministry of Education,
Department of Computer Science and Engineering, Southeast University,Nanjing 210096)

【Abstract】This paper firstly introduces the history of Linux kernel firewall. With the analysis of Netfilter framework and its implementation in IPv4 protocol stack, a simple kernel filter module is presented to illustrate the kernel firewall design technique under Netfilter framework. Finally, new directions in firewall design under Netfilter framework are briefly discussed.

【Key words】Firewall；Linux；Netfilter；Kernel module

随着互联网的飞速发展，网络在给用户提供大量信息的同时，也给了恶意的攻击者提供了非法入侵个人和商用系统的媒介，因此保证服务器和个人桌面系统的安全成为困扰系统管理员的一个难题。

Linux操作系统以其开发源码、高性能和高可靠性等诸多优势在商业服务器和个人桌面系统中得到日益广泛的应用。为了解决Linux系统安全问题，Linux内核的开发团队中逐步形成了专门的内核防火墙开发小组，并随着Linux内核版本升级不断推陈出新。最早的Linux下的IP防火墙出现在1.1系列内核中，它是由Alan Cox从FreeBSD系统的IPFW防火墙移植到Linux中来的；2.0系列内核中的IPFWadm由Jos Vos等人进行了改进和增强；到了2.2.x内核发布时，IPchains和单独开发的NAT等模块已经可以比较完整地实现内核级的IP防火墙的功能，但是由于没有充分考虑到扩展性和维护性问题，因此后续开发比较困难。Rusty Russell领导的Linux内核IP防火墙项目小组在2.3.x以后的开发过程中，总结了以往的开发经验，逐步形成了抽象、通用化的可扩展防火墙核心框架——Netfilter[1]。

1 Netfilter框架在IPv4中的实现流程

图1[3]是Netfilter的实现流程，灰色框中标记的是Netfilter框架在Linux内核协议栈中的位置,它通过5个可以扩展的钩子函数(hook)，实现内核防火墙的基本框架；箭头标明了IP包在包含Netfilter框架的IP层的流向，IP包从最左端进入系统,进行IP校验和版本检查后经过第一个挂载点NF_IP_PRE_ROUTING注册的钩子函数进行处理；经过路由选择，决定该数据包需要转发还是发给本机；若该数据包是发给本机的，则经过NF_IP_LOCAL_IN注册的钩子函数处理以后然后传递给上层协议；若需要转发，则转至NF_IP_FORWARD注册的钩子函数进行处理；所有需要发送到网络的数据包，无论是本机发出的还是转发的，都要经过最后一个钩子函数NF_IP_POST_ROUTING处理以后，才能发送到网络上。本地网络层以上产生的数据包通过NF_IP_LOCAL_OUT注册的钩子函数处理后，才可以进行路由选择，然后由NF_IP_POST_ROUTING处的钩子函数处理后发送到网络上。

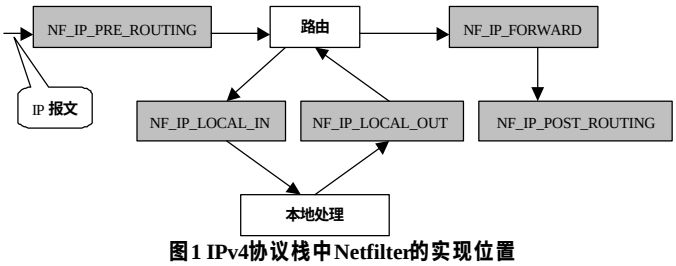

图1 IPv4协议栈中Netfilter的实现位置

具体各钩子函数挂载点、IP报文通过的时机以及通常在该点实现的防火墙功能如表1。

表1 5个钩子函数挂载点及其可以实现的防火墙功能

钩子函数挂载点	通过的时机	实现的功能
NF_IP_PRE_ROUTEING	刚刚进入网络层的IP数据包	源地址转换(SNAT)
NF_IP_LOCALIN	经过路由查找，发往本机的IP数据包	输入包过滤
NF_IP_FORWARD	经过路由查找，需要转发的IP数据包	转发包过滤
NF_IP_POST_ROUTING	所有需要从网络设备发出的IP数据包	输出包过滤
NF_IP_LOCAL_OUT	本机进程发出的IP数据包	目的地址转换(DNAT)

2 Netfilter框架在IPv4中的实现

上节描述了Netfilter框架在IPv4中的基本流程，本节以

基金项目：江苏省自然科学基金项目(BK2001205)
作者简介：姚晓宇(1978－)，男，硕士生，主研方向为高性能网络和网络安全；赵 晨，硕士生
收稿日期：2002-04-20 修回日期：2002-06-05

2.4.16内核中Netfilter在IPv4协议栈的实现中的关键数据结构和函数调用的实例分析，介绍Netfilter在IPv4中的实现。图2列举了IPv4网络层中的几个关键函数，也是Netfilter框架实现钩子函数挂载的关键。

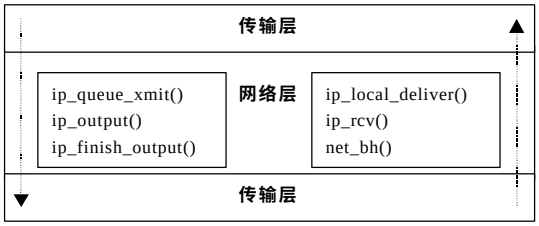

图2 IPv4网络层处理流程

Netfilter实现代码中都是通过NF_HOOK宏嵌入在网络协议栈的代码之中的，其定义在include/Linux/Netfilter.h中，具体如下：

```
#ifdef CONFIG_NETFILTER
……//其它Netfilter相关函数声明，如后面要用到的nf_register_ //hook()
#define NF_HOOK(pf, hook, skb, indev, outdev, okfn) \
(list_empty(&nf_hooks[(pf)][(hook)]) ? (okfn)(skb) : nf_hook_slow((pf), (hook), (skb), (indev), (outdev), (okfn)))
#else /* !CONFIG_NETFILTER */
#define NF_HOOK(pf, hook, skb, indev, outdev, okfn) (okfn)(skb)
#endif /*CONFIG_NETFILTER*/
```

如果编译内核时没有配置Netfilter选项，就相当于调用NF_HOOK宏的最后一个参数，执行okfn函数指针指向的函数；否则转nf_hook_slow()函数，由它执行防火墙开发者通过nf_register_hook()在该挂载点注册的防火墙处理功能，NF_HOOK宏的参数如下：

(1)PF：协议族名,除了PF_INET外，还可以是PF_INET6，PF_DECNet等；

(2)hook：HOOK点，对于IPv4，可为表1第1列的5个值之一；

(3)skb：Linux网络协议栈中数据包处理的基本单位；

(4)indev、outdev：接收和发送数据包的网络设备，以struct net_device结构表示(以上5个参数同时将被传到用nf_register_hook登记的处理函数中)；

(5)okfn: 函数指针，当所有的该HOOK点的所有登记函数调用完后，转而执行此回调函数。

在Linux2.4.16内核的IPv4协议栈实现中，IP_PRE_ROUTING挂载点在ip_rcv函数中完成IP头部校验和的计算、版本以及长度检查后进入；IP_LOCAL_IN挂载点在ip_local_deliver函数中完成IP分片的重组，准备向高层协议发送前进入；IP_FORWARD的挂载点在ip_fowrard函数中处理完应当发送ICMP报文各种出错情形后进入；IP_POSTROUTING挂载点在ip_finish_output函数中设置，所有从本地网络接口出去的报文都要经过这个检测点；IP_LOCALOUT的挂载点由于本地发送的的报文可能来自于多种不同的高层协议，所以设置IP_LOCALOUT钩子函数的函数调用点也就比较多，如ip_queue_xmit和ip_build_and_send_pkt函数中。

3 基于Netfilter的防火墙开发

通过前面关于IPv4中Netfilter实现的分析，可以看出Netfilter本身没有提供任何过滤报文的代码，它仅仅是一个可扩展的框架。基于该框架下进行开发的程序员无须过多涉及内核协议栈的改动，只须对一个或多个这样的挂载点上挂载内核模块，从而在数据报通过挂载点时通过内核模块过滤或修改这些数据报，并向Netfilter返回预定义的返回值[2]。用户挂载内核模块要通过nf_register_hook函数实现，其函数原型为：

```
int nf_register_hook(struct nf_hook_ops *reg)
```

参数reg的类型struct nf_hook_ops定义如下：

```
struct nf_hook_ops{
        struct list_head list;
        nf_hookfn *hook; /* 用户在此注册自己的处理函数 */
        int pf;
        int hooknum;
        int priority;
};
```

其中list项总要初始化为{NULL,NULL}，在IPv4中pf为PF_INET；hooknum就是选择的挂载点，一个挂载点上可以挂多个处理函数，执行的顺序根据priority值的大小决定。

防火墙开发者的工作便是编写防火墙在不同HOOK点的处理函数nf_hookfn，然后填充到struct nf_hook_ops结构的实例中，并用nf_register_hook将其注册在指定的HOOK上。其中nf_hookfn的原型为：

```
unsigned int nf_hookfn(unsigned int hooknum, struct sk_buff **skb,const struct net_device *in,const struct net_device *out, int (*okfn)(struct sk_buff *));
```

它的5个参数将由前面介绍过的NF_HOOK宏传进去，返回值常用的是NF_ACCEPT和NF_DROP，分别通知内核继续传输数据报和丢弃数据报，其它的返回值如：NF_STOLEN、NF_QUEUE和NF_REPEAT的介绍见文献[1]。

下面以一个简单的防火墙内核过滤模块为例介绍基于Netfilter框架下的防火墙开发，例子主要完成对nmap半开扫描报文进行过滤，注册的过滤函数samplefilter(nf_hookfn)和挂载点(NF_IP_PRE_ROUTING)写入在ipfilter(nf_nf_hook_ops类型)的实例中，这样所有进入系统的分组都将在经过ip_rcv()调用时进入挂载的过滤模块samplefilter。

```
static struct nf_hook_ops samplefilter_ops
={      {NULL,NULL}     ,samplefilter,PF_INET,NF_IP_PRE_ROUTING,NF_IP_PRI_FILTER-1};
static unsigned int samplefilter(unsigned int hooknum,struct sk_buff **skb,const struct net_device *in,const struct net_device *out,int (*okfn)(struct sk_buff *))
{
struct iphdr *iph = (*skb)->nh.iph;
__u32   sip=iph->saddr;
__u32   dip=iph->daddr;
struct tcphdr *tcph;
if(iph->protocol == 6){                 //针对IPv4报文的过滤
tcph=(struct tcphdr*)((__u32 *)iph+iph->ihl);//获得TCP的头指针
if((tcph->fin)&&(tcph->syn)&&(!tcph->rst)&&(!tcph->psh)&&(!tcph->ack)&&(!tcph->urg)){
printk("SF_scan from %d.%d.%d.%d to %d.%d.%d.%d\n", NIPQUAD(sip),NIPQUAD(dip));
return NF_DROP; /*nmap 半开扫描过滤*/ } }
return NF_ACCEPT;//通过本模块过滤的报文可以做进一步处理
}
```

内核模块的初始化和退出处理分别调用nf_register_hook(&samplefilter_ops)和nf_unregister_hook(&samplefilter_ops)完成钩子函数的挂载和卸载，内核模块编写可参考文献[4]。

编译上述内核模块，将生成的filter.o作为一个内核模块插入系统，就可以构成一个基于Netfilter的具有过滤nmap半开扫描功能的防火墙。当然，一个真正的防火墙系统的功能

(下转第163页)

流、最小负荷率、负荷曲线形状系数K、总表抄表数、分表抄表数之和、站内线损量、站内线损率。

②节点配变

自身属性：ID、编号、位置、型号、容量、是否是高压直供用户。

计算属性：节点平均电流、代表日铁损、月铁损、代表日铜损、月铜损。

2)线模型

①线路各分段

自身属性：ID、编号、线内编号、手段节点号、末端节点号、长度、导线型号

计算属性：线段平均电流、线段等值电阻

②线路整体

自身属性：ID、编号、首端、末端、配变台数。

计算属性：等值电阻、代表日损耗电量、月损耗电量、总损耗、线损率、导线线损率、配变铜线线损率、配变铁损线损率。

3)关系模型

点、线模型间有不可分割的逻辑关系，线损计算必然涉及到判断节点相邻、线路端点等问题，需要建立专门的模型来表征这些复杂的关系，本系统中将某些可以融合在点、线模型属性中的数据项专门提出来设计成一组关系模型。

①变电站—线路各分段关系

在输配电网中，每个变电站对应多条出线，而每条出线段又是某条线路整体的一个分段，为了明确表示这种对应关系，并使数据库编程更方便，在系统中建立变电站—线路各分段关系模型，变电站—线路各分段关系时一对多关系，即一个变电站对应多条线路分段(见表1)。

表1 变电站—线路各分段关系表

变电站ID	出现线路段1的ID	出现线路段2的ID	……	出现线路段n的ID

②线路各分段—线路整体关系

线路整体包含多条线路各分段，且前者的等值电阻是由后者的等值电阻求各得到的，因此两者之间存在一对多的关系,即一条线路整体对应多条线路分段(见表2)。

表2 线路各分段—线路整体关系表

线路整体ID	线路分段1的ID	线路分段2的ID	……	线路分段3的ID

③配变节点—配变节点关系

在线损计算算法中，线路各分段的平均电流由配变节点的平均电流计算而来，线路分段是由首、末节点确定的，因此，通过配变节点与配变节点之间有向相邻关系，就可唯一确定每一个线路分段。一个配变节点可以与多个配变节点相邻，因此两者之间是一对多的关系(见表3)。

表3 配变节点—配变节点关系表

配变节点ID	相邻节点数	相邻节点1的ID	……	相邻节点n的ID

(3)应用开发实施

将输配电网图数字化并在ArcInfo中建立起上述3类模型后，通过ArcSDE将空间数据和属性数据引入到SQL Server数据库中，构建起GIS数据服务器。

在Visual C++开发环境下开发线损计算客户端应用。客户端应用开发核心过程是：利用ArcSDE提供的API实现对GIS数据库中空间数据或属性数据的访问，调用所需数据按照选用的算法进行线损计算，将更新结果写入到数据库中，利用ArcObjects的Customization Framework等子库开发定制符合电力专业人员工作习惯的GIS应用界面，利用Display、Output等子库将线损计算结果图形化显示或输出，并支持直观图形查询和统计。

3 结束语

本文提出的基于GIS实现电网线损分析的技术方案目前在南阳电力GIS系统开发项目中已得到应用，并且将会在实际开发中不断改进和完善。线损分析是电力GIS系统的一个重要模块，为了实现系统全局最优，具体的模型设计还要兼顾查询、统计等模块的要求，进行整体优化。对象模型设计的优劣是决定系统性能的关键，也是后期应用开发的基础，因此在系统开发中有着非常重要的地位。

参考文献

1 虞忠年陈星莺刘 昊等电力网电能损耗北京中国电力出版社，2000

2 杨永康.徐有升余德文等新编用电管理工作手册郑州:河南科学技术出版社,2001

3 陈述彭鲁学军周成虎地理信息系统导论北京科学出版社,1999

4 Box D.潘爱民译.COM本质论北京中国电力出版社,2001

5 朱世伟胡春琳王兆祥构件对象技术模型在电力GIS系统的应用. 计算机系统应用,1999,(4):3-5

6 华奇兵许文波李 琳等.COM技术及其程序设计.重庆邮电学院学报,2001,13(1):51-61

☆☆

(上接第113页)

还要包括完备的规则库和日志处理等功能，但是有了Netfilter框架，防火墙系统的开发的难度大大降低了，这也是近年来国内基于Linux平台的防火墙产品迅速发展的原因之一。

4 结束语

Linux内核防火墙经历了三代的发展，到Netfilter为止，基本上形成了一个结构合理、功能完备的防火墙框架。Netfilter将框架与用户可定制的功能分离，提高了系统的可扩展性和可维护性，也为开发内核防火墙系统提供了可以借鉴的思路和快速开发的底层支持。

目前，在Netfilter框架的基础上，通过内核模块扩展还可以实现IP的QoS、流量控制(traffic control)和负载均衡(Linux virtual server)等高级功能,这些开放源码的扩展模块为构建一个功能强大的防火墙系统提供了有力的支持。

参考文献

1 Linux Netfilter Hacking HOWTO Rusty Russell.Mailing list Netfilter @lists.samba.org

2 Writing a Module for Netfilter by Paul "Rusty" Russell Linux Magazine.http://www.Linux-mag.com/2000-06/gear_01.html

3 The Netfilter Framework in Linux 2.4.http://www.gnumonks.org/papers/Netfilter-lk2000/presentation.html

4 Linux Kernel Module Programming Guide.Ori Pomerantz,1999