linux防火墙实现技术比较 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/354/2021_2022_linux_E9_98_B2_E7_81_c103_354008.htm 阐述linux下的防火墙的不同实现之间的区别，以ipchains, iptables, checkpoint FW1为例。 一 基本概念 2.0 在进入正题之前，我将花少许篇幅阐述一些基本概念。尽管防火墙的术语这些年基本上没有太大的变化，但是如果你以前只看过90年代初的一些文献的话，有些概念仍然会让你混淆。此处只列出一些最实用的，它们不是准确的定义，我只是尽可能的让它们便于理解而已。 2.1 包过滤： 防火墙的一类。80年代便有论文来描述这种系统。传统的包过滤功能在路由器上常可看到，而专门的防火墙系统一般在此之上加了功能的扩展，如状态检测等。它通过检查单个包的地址，协议，端口等信息来决定是否允许此数据包通过。 2.2 代理： 防火墙的一类。工作在应用层，特点是两次连接（browser与proxy之间,proxy与web server之间）。如果对原理尚有疑惑，建议用sniffer抓一下包。代理不在此文的讨论范围之内。 2.3 状态检测： 又称动态包过滤，是在传统包过滤上的功能扩展，最早由checkpoint提出。传统的包过滤在遇到利用动态端口的协议时会发生困难，如ftp。你事先无法知道哪些端口需要打开，而如果采用原始的静态包过滤，又希望用到的此服务的话，就需要实现将所有可能用到的端口打开，而这往往是个非常大的范围，会给安全带来不必要的隐患。而状态检测通过检查应用程序信息（如ftp的PORT和PASS命令），来判断此端口是否允许需要临时打开，而当传输结束时，端口又马上恢复为关闭状态。 2.4 DMZ非军事化区： 为

了配置管理方便，内部网中需要向外提供服务的服务器往往放在一个单独的网段，这个网段便是非军事化区。防火墙一般配备三块网卡，在配置时一般分别分别连接内部网，internet和DMZ。 2.5 由于防火墙地理位置的优越（往往处于网络的关键出口上），防火墙一般附加了NAT，地址伪装和VPN等功能，这些不在本文的讨论范围。 二 检测点 3.0 综述 包过滤需要检查IP包，因此它工作在网络层，截获IP包，并与用户定义的规则做比较。 3.1 ipchains 摘自【3】 总体来说，分为输入检测，输出检测和转发检测。但具体到代码的时候，输出检测实际分散到了几处（不同的上层协议走IP层的不同的流程）： UDP/RAW/ICMP报文:ip_build_xmit TCP报文:ip_queue_xmit 转发的包：ip_forward 其它：ip_build_and_send_pkt 正如ipchains项目的负责人Rusty Russell所说，在开始ipchians不久，便发现选择的检测点位置错了，最终只能暂时将错就错。一个明显的问题是转发的包在此结构中必须经过三条链的匹配。地址伪装功能与防火墙模块牵扯过于紧密，如果不详细了解其原理的话，配置规则很容易出错。 此部分详细的分析可参见我早期的一份文章【9】。 2.4内核中的防火墙系统不是2.2的简单增强，而是一次完全的重写，在结构上发生了非常大的变化。相比2.2的内核，2.4的检测点变为了五个。 在每个检测点上登记了需要处理的函数（通过nf_register_hook（）保存在全局变量nf_hooks中），当到达此检测点的时候，实现登记的函数按照一定的优先级来执行。严格的从概念上将，netfilter便是这么一个框架，你可以在适当的位置上登记一些你需要的处理函数，正式代码中已经登记了许多处理函数（在代码中

搜nf_register_hook的调用），如在NF_IP_FORWARD点上登记了装发的包过滤功能。你也可以登记自己的处理函数，具体例子可参看【8】与【10】。 3.3 FW1 FW1是chekpoint推出的用于2.2内核的防火墙。由于其发布的模组文件带了大量的调试信息，可以从反汇编的代码中窥探到到许多实现细节。FW1通过dev_add_pack的办法加载输入过滤函数，如果对这个函数不熟悉，请参看【14】。但是此处有个问题：
在net_bh()中，传往网络层的skbuff是克隆的,即
skb2=skb_clone(skb, GFP_ATOMIC). if(skb2)
pt_prev->func(skb2, skb->dev, pt_prev). 这样的话如果你想丢弃此包的话，光将其free掉是不够的，因为它只是其中的一份拷贝而已。 FW1是怎么解决这个问题的呢？见下面的代码（从汇编代码翻译成的C程序）： packet_type *fw_type_list=NULL.
static struct packet_type fw_ip_packet_type = {
__constant_htons(ETH_P_IP), NULL, /* All devices */ fw_filterin, NULL, NULL, /* next */ }. fwinstallin(int isinstall) { packet_type *temp. /*安装*/ if(isinstall==0){
dev_add_pack(amp.fw_ip_packet_type). for(temp = fw_ip_packet_type. temp. temp=temp->next)
dev_add_pack(temp). }} 不难看出，FW1把ip_packet_type歇载掉了，然后自己在自己的处理函数(fw_filterin)中调ip_recv。输出的挂载和lkm的手法一样，更改dev->hard_start_xmit。dev结构在2.2版本的发展过程中变了一次，为了兼容FW1对这点也做了处理（通过检查版本号来取偏移）。 还有一款linux下的防火墙产品WebGuard
（http://www.gennet.com.tw/b5/csub_webguard.html）采用的手

法与FW1其非常类似。有兴趣的人可以自行研究一下。