Linux防火墙扩展技术与入侵检测实现 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/260/2021_2022_Linux_E9_98_B2_E7_81_c103_260895.htm 摘 要:该文介绍基于Linux netfilter/iptables架构实现机制和扩展技术,在此基础上提出扩展匹配选项实现防火墙的入侵检测功能，扩充后的防火墙可以像Snort一样具有入侵检测功能，从而扩展了防火墙的安全控制功能，并且可将Snort规则转化为防火墙规则实现防火墙规则集的扩充。 关键词：防火墙 入侵检测 扩展 按照防火墙对内外来往的数据的处理方法，防火墙可以分为包过滤防火墙和应用层防火墙，包过滤防火墙工作在网络层，它只是检测包的协议头对数据包进行裁决，它运行速度快但无法对高层的协议内容进行检查，应用层防火墙则可以对高层数据进行转发和过滤并强制身份验证，但对不同的服务需要提供代理应用程序并且建立了网络瓶颈；并且将包过滤技术和多种应用技术融合到一起，构成复合型防火墙是目前国内防火墙产品的一个特点，也是防火墙今后发展的主流技术。鉴于在防火墙中整合数据包检测功能是一种良好的解决方法，它可以弥补现有防火墙的缺点并且具有像入侵检测系统一样的检测功能，本文将介绍基于Linux netfilter/iptables架构实现机制和扩展技术,在此基础上提出了扩展匹配选项实现防火墙的入侵检测功能，扩充后的防火墙可以像Snort一样具有入侵检测功能，并且可将Snort规则转化为防火墙规则实现规则集的扩充。 1 Linux防火墙的扩展netfilter/iptables的技术 Linux中防火墙Netfilter/Iptables系统主要包括两个基本组件 :定义在内核空间中的通用框架Net filter和数据包选择系统(Packet Selection).

其中后者又由两部分构成:在Net filter框架上定义的数据结构“ IP表 ”(IP Tables)和在用户空间实现的应用程序iptables.具体防火墙工作流程见[1][2]。 由于Net filter架构的加入，可以通过简单的内核模块化来实现新功能的扩展，在现有的Netfilter/Iptables中可以通过两种方式对现有的防火墙进行扩充，一种是扩展Net filter通过编写相关内核模块调用nf_register_hook()直接在相关的钩子上注册从而获得新特性，一种是扩展IP表通过编写相关的匹配标准和目标来实现新特性；扩展IP表方式是对现有表的匹配规则的扩充与具体表无关。扩展IP表需要编写内核和用户两方的代码，内核模块提供了实际的数据包匹配规则代码，用户方代码提供了IPTABLE 新的命令行选项的共享库。 2 Linux防火墙入侵检测扩展匹配设计 目前入侵检测系统普遍采用精确的模式匹配算法，如Snort采用基于规则的方式对数据包进行规则匹配来检测多种不同的入侵行为和探测活动，这种方式简单而有效，因此可以借鉴这种思想在防火墙的匹配选项中加入匹配选项来检测数据包中的内容，由于扩展IP表具有很好地灵活性，为此可以选用这种方式扩充匹配标准来实现入侵检测模块。这种方式需要编写内核和用户空间代码，Netfilter/Iptables的标准化提供了两方使用的重要数据结构，在实现这两部分代码时主要是填充相应的数据结构内容然后将它们注册从而扩展功能。 2.1 内核模块数据结构 新的MATCH功能可作为一个独立的模块，为了能使新模块能被别的模块使用，可以使用iptable提供的ipt-register-match()将该模块进行注册，新的MATCH模块的核心是ipt-match结构，它将作为ipt-register-match()的参数注册到MATCH链表中备用从而增

加新的规则匹配选项。 Struct ipt_match{struct list_head list.一般设定为{NULL,NULL}，由核心使用const char name[].MTACH功能的名称，该名称必须与模块名相匹配int (*match)().一个指向MTACH功能函数的指针，返回非0表示匹配int (*check entry)().一个指向检查规则规范的指针，如果返回0，规则不会加入iptablesvoid (*destroy)(). 当一个使用该MATCH的入口被删除时，该函数调用以释放所占资源struct module me 是否是模块的定义，是模块设置为THIS_MODULE 否则NULL}在该数据结构中重要的是match ，check entry 函数，MATCH函数将实现接收从底层传来的数据包，检查数据包实现匹配功能，如果数据包与所定义的规则相同那么返回TRUE，如果不成功返回FALSE并且可以设置参数表示数据包可以被立即被丢弃。Check entry函数指向一个检查规则规范的指针，如果返回0表明这条规则不能从用户空间接受。 2.2用户空间数据结构 在内核中加入相关的内核模块选项后，为了在用户空间使用iptables软件提供相关的规则必须为该软件提供相关的命令行选项，为了使各个扩展模块使用一个版本的iptables软件而不必编写相关扩展的特定软件版本，采用共享库可以解决该问题，共享库应该具有-init()功能，它的功能和内核模块功能相似，在装载时被自动调用，该功能根据添加的新MATCH和新TARGET不同分别调用register-match()或register-target()，共享库可以提供初始化数据结构和提供相关选项的功能。编写共享库中使用的重要数据结构是iptables_match，它作为参数传递给register-match()注册相关的命令行匹配选项让iptables识别该新匹配。 Struct iptables_match{struct iptables_match *next.用于形成一

个MATCH列表的指针，初始化为NULLipt_chainlabel name.MATCH功能的名字，必须与库函数名相同便于主程序根据MATCH名加载相应的动态连接库const char *version.版本信息通常被设置IPTABLES_version宏size_t size.该MATCH的数据大小size_t userspacesize.由于内核可能修改某些域，在这里填写被改变数据区大小，它一般和size大小同void (*help)(void).打印帮助选项大纲void (*init)().初始化ipt-entry-match结构 int (*parse)().扫描并接收本MATCH的命令行参数，正确接受返回非0void (*final_check)().检查是否强制选项（如--ids）被描述，如果不正确退出void (*print)().查询当前表中的规则时，显示使用了当前match规则的额外信息void (*save)().PARSE的反转，被iptable-save调用再生match的命令行参数const struct option *extra_opts.NULL结尾的参数列表，提供命令行其余选项/*以下参数由iptables内部使用，用户不必填写*/Unsigned int option_offset.Struct ipt_entry_match *m.Unsigned int mflags.Unsigned int used.} 