jasenwan88的专栏

个人资料

jasenwan88

访问： 41907次
积分： 579
等级： BLOG 3
排名： 千里之外

原创： 7篇 转载： 48篇
译文： 1篇 评论： 0条

Linux下Libpcap源码分析和包过滤机制（3）

分类： 网络基础知识 2012-07-19 13:43 151人阅读 评论(0) 收藏 举报

linux 算法 网络 filter flex graph

大量的网络监控程序目的不同，期望的数据包类型也不同，但绝大多数情况都都只需要所有数据包的一（小）部分。

数据包过滤机制

大量的网络监控程序目的不同，期望的数据包类型也不同，但绝大多数情况都都只需要所有数据包的一（小）部分。例如：对邮件系统进行监控可能只需要端口号为 25（smtp）和 110（pop3) 的 TCP 数据包，对 DNS 系统进行监控就只需要端口号为 53 的 UDP数据包。包过滤机制的引入就是为了解决上述问题，用户程序只需简单的设置一系列过滤条件，最终便能获得满足条件的数据包。包过滤操作可以在用户空间执行，也可以在内核空间执行，但必须注意到数据包从内核空间拷贝到用户空间的开销很大，所以如果能在内

核空间进行过滤，会极大的提高捕获的效率。内核过滤的优势在低速网络下表现不明显，但在高速网络下是非常突出的。在理论研究和实际应用中，包捕获和包过滤从语意上并没有严格的区分，关键在于认识到捕获数据包必然有过滤操作。基本上可以认为，包过滤机制在包捕获机制中占中心地位。

包过滤机制实际上是针对数据包的布尔值操作函数，如果函数最终返回true，则通过过滤，反之则被丢弃。形式上包过滤由一个或多个谓词判断的并操作（AND）和或操作（OR）构成，每一个谓词判断基本上对应了数据包的协议类型或某个特定值,例如：只需要 TCP 类型且端口为110的数据包或ARP类型的数据包。包过滤机制在具体的实现上与数据包的协议类型并无多少关系，它只是把数据包简单的看成一个字节数组，而谓词判断会根据具体的协议映射到数组特定位置的值。如判断ARP类型数据包，只需要判断数组中第 13、14 个字节（以太头中的数据包类型）是否为0X0806。从理论研究的意思上看，包过滤机制是一个数学问题，或者说是一个算法问题，其中心任务是如何使用最少的判断操作、最少的时间完成过滤处理，提高过滤效率。

BPF

libpcap 重点使用 BPF（BSD Packet Filter）包过滤机制，BPF 于 1992 年被设计出来，其设计目的主要是解决当时已存在的过滤机制效率低下的问题。BPF的工作步骤如下：当一个数据包到达网络接口时，数据链路层的驱动会把它向系统的协议栈传送。但如果 BPF 监听接口，驱动首先调用 BPF。BPF 首先进行过滤操作，然后把数据包存放在过滤器相关的缓冲区中，最后设备驱动再次获得控制。注意到BPF是先对数据包过滤再缓冲，避免了类似sun的NIT过滤机制先缓冲每个数据包直到用户读数据时再过滤所造成的效率问题。参考资料D是关于BPF设计思想最重要的文献。

BPF 的设计思想和当时的计算机硬件的发展有很大联系，相对老式的过滤方式CSPF（CMU/Stanford Packet Filter）它有两大特点。1：基于寄存器的过滤机制，而不是早期内存堆栈过滤机制，2：直接使用独立的、非共享的内存缓冲区。同时，BPF 在过滤算法是也有很大进步，它使用无环控制流图（CFG control flow graph）,而不是老式的布尔表达式树（boolean expression tree）。布尔表达式树理解上比较直观，它

文章搜索

文章分类

linux内核 (25)
编辑工具 (2)
移植 (1)
网络基础知识 (16)
linux基础 (4)
服务器类 (4)
unix高级环境编程 (2)
项目需求 (1)
luci (1)

文章存档

2014年11月 (1)
2012年07月 (29)
2012年06月 (22)
2012年04月 (1)
2012年03月 (2)
展开

阅读排行

emacs 命令小结---开关、 (8295)

天玥网络安全审计系统（ (4592)
关于IP选项 (1729)
Ubuntu linux下安装sqlite (1619)
Openssl EVP 说明三 分享 (1386)
ifconf和ifreq (1261)
Openssl ASN.1 说明二（ (1113)
POSIX semaphore: sem (1056)
Openssl ASN.1 说明一 分 (996)
libpcap的使用 (954)

评论排行

openwrt中luci界面中简单 (0)
Linux下Libpcap源码分析 (0)
Linux下Libpcap源码分析 (0)
Linux下Libpcap源码分析 (0)
以開源碼 dansguardian- (0)
如何定制Ubuntu 12.04 C (0)
linux如何建立IP隧道 (0)
linux下的多进程服务器框 (0)
精通top,ps命令 (0)
Authentication Proxy原理 (0)

推荐文章

的每一个叶子节点即是一个谓词判断，而非叶子节点则为 AND 操作或 OR操作。CSPF有三个主要的缺点。1：过滤操作使用的栈在内存中被模拟，维护栈指针需要使用若干的加/减等操作，而内存操作是现代计算机架构的主要瓶颈。2：布尔表达式树造成了不需要的重复计算。3：不能分析数据包的变长头部。BPF 使用的 CFG 算法实际上是一种特殊的状态机，每一节点代表了一个谓词判断，而左右边分别对应了判断失败和成功后的跳转，跳转后又是谓词判断，这样反复操作，直到到达成功或失败的终点。CFG算法的优点在于把对数据包的分析信息直接建立在图中，从而不需要重复计算。直观的看，CFG 是一种"快速的、一直向前"的算法。

过滤代码的编译

BPF 对 CFG 算法的代码实现非常复杂，它使用伪机器方式。BPF 伪机器是一个轻量级的，高效的状态机，对 BPF 过滤代码进行解释处理。BPF 过滤代码形式为"opcode jt jfk"，分别代表了操作码和寻址方式、判断正确的跳转、判断失败的跳转、操作使用的通用数据域。BPF 过滤代码从逻辑上看很类似于汇编语言，但它实际上是机器语言，注意到上述 4 个域的数据类型都是 int 和 char 型。显然，由用户来写过滤代码太过复杂，因此 libpcap 允许用户书写高层的、容易理解的过滤字符串，然后将其编译为BPF代码。

libpcap使用了4个源程序gencode.c、optimize.c、grammar.c、scanner.c完成编译操作，其中前两个实现了对过滤字符串的编译和优化，后两个主要是为编译提供从协议相关过滤条件到协议无关(的字符数组)位置信息的映射，并且它们由词汇分析器生成器 flex 和 bison 生成。参考资料 C 有对此两个工具的讲解。

```
flex -Ppcap_ -t scanner.l > $$.scanner.c; mv $$.scanner.c
scanner.c
bison -y -p pcap_ -d grammar.y
mv y.tab.c grammar.c
mv y.tab.h tokdefs.h
```

* struts2国际化
* 无需超级用户mpi多机执行
* 从视图索引说Notes数据库（下）
* Java虚拟机解析篇之---垃圾回收器
* 2015互联网校招总结—一路走来
* SSL 3.0曝出Poodle漏洞的解决方案-----开发者篇

编译过滤字符串调用了函数 pcap_compile()[getcode.c]，形式为：

```
int pcap_compile(pcap_t *p, struct bpf_program *program,
            char *buf, int optimize, bpf_u_int32 mask)
```

其中 buf 指向用户过滤字符串，编译后的 BPF 代码存在在结构 bpf_program中，标志 optimize 指示是否对 BPF 代码进行优化。

```
/* [pcap-bpf.h] */
struct bpf_program {
u_int bf_len; /* BPF 代码中谓词判断指令的数目 */
struct bpf_insn *bf_insns; /* 第一个谓词判断指令 */
};

/* 谓词判断指令结构，含意在前面已描述 [pcap-bpf.h] */
struct bpf_insn {
u_short code;
u_char  jt;
u_char  jf;
bpf_int32 k;
};
```

上一篇 Linux下Libpcap源码分析和包过滤机制（2）
下一篇 Linux下Libpcap源码分析和包过滤机制（4）

主题推荐 源码 linux 汇编语言 计算机 算法

猜你在找

linux-3.2.36内核启动4-setup_arch中的内存初始化3（arm
Linux netfilter源码分析(3)
Linux下Libpcap源码分析和包过滤机制（2）
Linux下Libpcap源码分析和包过滤机制（1）
Linux下Libpcap源码分析和包过滤机制（4）
深入分析Linux内核源码-Linux管道的实现机制
Linux下Libpcap源码分析和包过滤机制（4）
Linux下Libpcap源码分析和包过滤机制（2）
Linux下Libpcap源码分析和包过滤机制（1）
linux下libpcap抓包分析

查看评论

暂无评论

您还没有登录,请[登录]或[注册]

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

核心技术类目

全部主题 Hadoop AWS 移动游戏 Java Android iOS Swift 智能硬件 Docker OpenStack
VPN Spark ERP IE10 Eclipse CRM JavaScript 数据库 Ubuntu NFC WAP jQuery