基于 Snort 的入侵检测系统的改进

魏葆雅[1,2]　吴顺祥[2]
(1.漳州师范学院计算机科学与工程系　2.厦门大学自动化系)

[摘　要] 在高速网络中,提高入侵检测系统检测速率和效率是目前入侵检测系统需要解决的主要问题。本文对 snort 的数据包捕获机制 libpcap 和检测引擎进行分析,根据其所存在的局限提出改进方法。

[关键词] Snort　数据包捕获　模式匹配

0. 引言

Snort 是一个以开放源代码形式发行的轻量级网络入侵检测系统,是目前使用最广泛的 NIDS。通过研究 Snort的实现方法,能够对基于网络的入侵检测系统有更深入的了解。Snort 可以运行在多种操作系统平台上(如 Unix/ Linux/ Windows 等),和很多商业产品相比,它对操作系统的依赖性比较低,并且对系统影响最小,而且用户可以根据自己的需要调整检测策略[1]。

1. Snort介绍

Snort 可分为 5 个主要的组件,每个组件对入侵检测都很关键。第一个是捕包装置。Snort依赖一个外部捕包程序库 Libpcap 来抓包。在包被以原始状态捕获后,要送给包解码器。解码器是进入 Snort 自身体系的第一步。包解码器将特殊协议元素翻译成内部数据结构。在最初的捕包和解码完成后,由预处理程序处理流量。在许多插入式预处理器程序对包进行检查或操作后将他们交给下一个组件 检测引擎。检测引擎对每个包进行检测判断入侵。最后是输出组件,它对可疑行为产生警报。图 1 是一个简单的数据流程示意图。

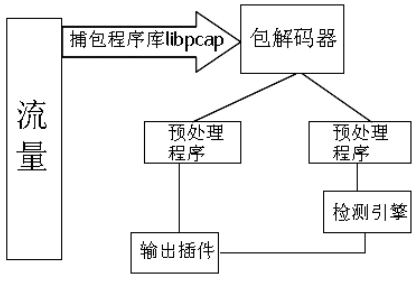


图 1

文献[2]建立了实验环境,利用标准测试数据和实际测试数据对 Snort 性能进行了测试。测试表明,Snort 系统数据包捕获大约占用了系统时间的 20%;其中大量的时间用于软中断和系统调用。攻击检测约占用了系统时间的 57 % ,其中大量的时间用于模式匹配。随着网络流量的逐渐增加,Snort 系统的丢包率不断上升,造成大量漏报,因此在高速网络环境下有必要对 Snort 的性能进行改进。

2. Snort数据捕获机制分析

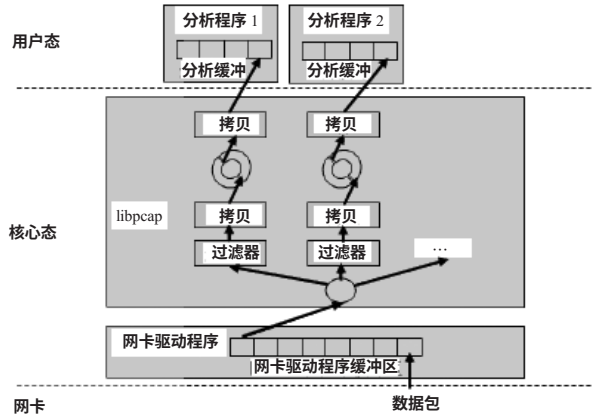
目前,Snort 还没有自己的捕包工具,直接从网卡捕包的任务由捕包程序 Libpcap承担。Libpcap 是一个与实现无关的访问操作系统所提供的分组捕获机制的分组捕获函数库,用于访问数据链路层。这个库为不同的平台提供了一致的编程接口,在安装了 Libpcap 的平台上,以 Libpcap 为接口写的程序、应用,能够自由的跨平台使用。它也是用户态的数据包截获 API 函数接口,有独立和可移植性。Libpcap系统框架如图 2 所示[3]。可以看出,Libpcap 通过对数据帧的过滤、缓冲、拷贝,将数据帧传递到各个应用程序缓冲区。在这个过程中,数据包经历了从网卡到内核空间、从内核空间到用户空间两次数据包拷贝。在 Libpcap 技术构架下开发完成的 Snort系统必然存在如下性能问题:

(1) 多次数据拷贝造成性能降低;

(2) 在 Libpcap 和应用程序之间拷贝数据帧,系统必须不断地在核心态和用户态之间不断切换,这将消耗大量的系统资源;

(3) 分析程序中包含的多种复杂算法需要实时运行,但在 Libpcap 构架中,分析程序作为普通进程,运行于操作系统用户态,受到其他进程的影响,性能极不稳定,导致大量丢包。

针对上述不足,我们采用以下措施提高数据采集性能。首先是采用零拷贝技术,零拷贝技术通过精简协议,完全旁路操作系统,这样就可以避免数据在内存中多次拷贝产生的效率问题。目前,实现零拷贝用到的最主要技术是 DMA 数据传输技术和内存区域映射技术。为了避免频繁的状态切换,可将分析程序放到系统核心态,以内核模式运行,这样数据帧不再被传递到应用程序分析,系统就不需要频繁地在核心态和用户态之间切换。对数据帧的实时分析功能全部在核心态完成,不再受操作系统进程调度的约束,性能得到提高。



3. Snort攻击检测效率改进

Snort 系统攻击检测的大部分工作量是在模式匹配上。因此提高模式匹配的效率是提高系统攻击检测速度的关键,它直接影响系统的性能。目前,对于提高模式匹配速度的研究已经很多,也提出了许多高效的匹配算法。

BM 算法是单模式字符串比较算法,扫描一次数据包,只能完成对一个模式串的匹配工作。在网络入侵检测系统中,一个包里的内容可能会部分匹配很多规则,因此在匹配每条规则时,都会重新运行匹配算法,这会造成匹配效率的降低。BM 算法在模式数量很小时速度很快,当模式数量增加到一定范围后性能明显不如多模式匹配算法。BM 算法时间复杂度为 O(mn),最优时为 O(n/m)。

多模式匹配算法 Aho- Corasic 算法并行搜索模式集合要比应用 BM 算法逐一地搜索多个模式串高效得多。其预处理时间取决于规则集中最长模式的长度。AC 算法预处理阶段的时间复杂度为 0(M)(M 为模式集的总长度),匹配阶段为 O(N)(N 为文本长度)。

Wu- Manber 算法是另一种应用广泛的多模式匹配算法。该算法采用了 Boyer- Moore 算法的框架,在预处理所有模式时使用一个 2- byte 的跳转表。此外,在进行匹配的时候,它使用散列表选择关键词集合中的一个子集与当前文本进行匹配,减少无谓的匹配运算。

这些算法都大大提高了规则匹配的速度。然而随着各种入侵攻击手法不断增加,进行模式匹配所需的特征库也越来越大,导致在采用相同硬件资源和模式匹配算法的情况,系统对数据包处理速度下降。针对这个问题,可在检测引擎中增加高速缓存模块,高速缓存存放的是最近匹配的规则。捕获的数据包经过预处理后先与高速缓存中的规则进行匹配,如果不匹配再继续与规则库中的规则进行匹配。一次攻击可以展开成百上千个包,这些包可能匹配的规则相同,采用高速缓存可以提高检测效率。我们的目标是要提高检测效率,所以高速缓存的设计要尽量减少内存消耗以及避免复杂的运算。

4. 总结

随着网络带宽的增长以及网络攻击种类的急剧增加,致使 Snort 的检测任务越来越重,从而有可能漏掉一些造成严重后果的网络攻击行为。因此需对 Snort 入侵检测系统进行改进,以消除基于 Snort 的网络入侵检测系统瓶颈,提高系统效率。

参考文献

[1] 高平利. 基于 Snort 入侵检测系统的分析与实现[J].计算机应用与软件,2006 ,23 (8) :1342138.

[2] 汪世义. 基于 Snort 的入侵检测系统研究[D].大连:大连理工大学,2005.

[3] Degioanni L, Baldi M, Risso F, et al. Profiling and optimization of software- based network- analysisapplication[EB/ OL].http:/ / www.winpcap. org/ docs/ WinPcap- SBAC03.pdf, 2003- 11- 19.