高速 IDS 系统中读写缓冲区的互斥机制[1]

王晨　　龚俭　　廖闻剑
（东南大学计算机科学与工程系，江苏省计算机网络技术重点实验室，南京，210096）

A Buffer Access Algorithm in High-Speed IDS

Wang Chen，Gong Jian，Liao Wen Jian
（Dept. of Computer Science and Engineering，Southeast University，210096 Nanjing）

【摘要】在网络入侵检测系统中，通常需要在内存中开辟缓冲区对网络报文进行采集和分析，但是传统的读写缓冲区的互斥机制在高速网络环境中的效率都不甚理想，无法满足对高速 IDS 系统的性能需求。本文提出的一种基于并发锁机制的双缓冲区互斥机制可以改善传统锁机制的资源利用率，很好地解决了报文到达流和报文处理能力之间的性能瓶颈问题，显著地提高了 IDS 系统的性能。

【关键字】互斥机制，入侵检测系统（IDS），锁机制，缓冲区管理

【Abstract】 An access algorithm of buffer is usually used for collecting and analyzing network packets in network intrusion detection systems using misuse detection methods. However traditional access algorithm of buffer have indicated unsatisfactory efficiencies over high-speed networks, which can not reach the performance requirements in high-speed IDS. This paper brings forward an access algorithm based on lock algorithm, with which the performance requirements in analyzing packets in a high-speed environment have been met rather successfully and the bottle-neck between the reaching flow and the ability of analyzing packets have been solved.

【Keywords】access control algorithm，intrusion detection system（IDS），lock algorithm，buffer management。

一、引言

在面向高速主干网络的 IDS 系统中，通常情况下，主干网络流量可达到 1－2.5G（bps），经过预处理之后待检测报文的到达流强度最高可以达到 200Mpps，而在一般的通用处理平台上系统分析报文的处理能力只能达到 30Mpps 左右，可见报文到达的频率往往要大于报文处理的频率，系统需要使用缓冲区来缓解两者之间的瓶颈。因此无论对于滥用检测还是异常检测，读写缓冲区的互斥机制都是一个非常关键的问题，它将直接影响到整个入侵检测系统的效率和效果。

目前最常用的缓冲区互斥机制为并发锁机制：确保一次只能有一个进程访问特定的共享资源即所谓的临界段。对临界段的访问是互斥的，一旦有一个进程进入临界段，则其他的进程就不能再进入了，直到这个进程退出这个临界段为止。由于在等待锁被释放的过程中进程没有做任何有用的工作，因此该机制在处理器的利用上是低效的。这种锁叫做自旋锁（spin lock），锁机制又被称作忙等待（busy waiting）机制。

传统加锁策略的基本思想是【1】：对具体的数据进行对象操作前，都需要首先申请一把和这个数据对象一一对应的锁，如果锁资源申请成功，则可以继续对数据对象进行操作；否

[1]本文受国家自然科学基金项目 90104031 资助

则等待其他事务释放锁资源。在这种机制里，获得了锁资源的同时也表示获得了对应数据对象的操作权，其他的事务如果要再对这个数据对象进行操作，则必须要符合操作的相容性；同理，如果释放了一个锁资源，就表示事务不再占用数据对象，其它的事务就可以取而用之了。在实现上，它是一种脱离“数据操作层”的加锁策略，它在系统中另外开辟一个与数据对象一一对应的数据结构（锁表）作为公共资源，因此对数据对象的操作将会涉及两个层次上的操作，即锁表数据结构和内存数据对象的交叉访问。

在高速主干网络环境下，报文到达流的强度有时会过大，使得报文分析的速度远远跟不上报文到达的速度。如果采用传统的锁机制来同步读写缓冲区的操作，在单循环队列的条件下，频繁的加解锁将造成系统性能的严重下降，而且加解锁期间资源会长时间处于空闲状态，这样势必造成报文的严重丢失。本文所讨论的双缓冲区互斥机制就是基于并发锁机制的一种变体，通过改善锁机制来大大提高处理器的利用率。

除了传统的并发锁机制外，还有两种常用的互斥机制：由 Dijkstra（1968）【2】发明的信号量（Semaphore）机制和条件变量机制。由于信号量机制中，每个要访问临界资源的进程都必须自备同步操作，这就使大量的同步操作分散在各个进程中，不仅给系统的管理带来麻烦，而且会严重影响进程间通信的效率，不适合本文所论及的面向高速主干网络的 IDS 系统环境。本文所论及的系统不涉及全局条件的判断，所以条件变量的互斥机制也不适用。

本文在第二节中详细描述基于并发锁机制的双缓冲区互斥机制，在第三节分析了算法复杂性并且对比了与传统并发锁机制在丢报率上的差别，在第四节给出对本文所论及的互斥机制所做的实验及实验结果。

二、高速 IDS 系统的缓冲区互斥机制

本文建议采用双缓存队列来缓冲报文。读写进程独立的操作一块缓冲队列，考虑到低效率的读进程将造成重要数据的丢失，同时也为了保证写进程所在队列中的新到的数据能够得到及时的处理，因此人为地把写进程的优先级设置成高于读进程，当写进程写满当前队列时就强行切换读写进程所在的队列，这时读进程所在队列中剩下的未处理的老数据将丢失。这种互斥机制与传统的加锁机制不同之处在于：

(1) 不同的事务可以同时对不同的数据对象进行操作，而不需要像传统的锁机制一样同一时间只能有一个事务工作；
(2) 没有频繁的加解锁操作；
(3) 设置事务的优先级，高优先级事务可以掠夺低优先级事务所操作的数据对象而不需要低优先级事务释放相应的锁资源。

这种双缓冲队列的互斥访问控制机制使用的锁表数据结构包括：

a. **读指针当前队列标识（read_buffer）**：用来标志读指针所在队列的标号。
b. **写指针当前队列标识（write_buffer）**：用来标志写指针所在队列的标号。
c. **读指针（reader）**：用来标志读进程正在处理的数据块编号。
d. **写指针（writer）**：用来标志写进程正在处理的数据块编号。
e. **队列繁忙向量（busy）**：是一个包含 2 个元素的数组，每个元素相应的代表特定队列的使用状态，其初始值是 0 表示队列处于空闲状态。
f. **队列切换向量（change）**：是一个包含 2 个元素的数组，每个元素相应的代表特定队列的切换状态，起初始值是 0 表示队列处于未切换状态，一旦变为 1 代表另一个队列已经写满，需要切换读写指针所在的队列。

互斥中涉及读写缓冲区两个独立的进程，因此需要分别对读写算法进行讨论，算法的具体流程如图 1 所示。设 Blocknum 为单个缓冲队列能够存储的最大数据块个数。

写入缓冲区算法：

⑴ 如果 writer < Blocknum，则转向步骤 5；否则转向步骤 2，因为读指针所在的队列缓冲区已经写满了。

⑵ 将 change[read_buffer]置位，表示写指针已经写满队列，需要强行切换队列，同时将 change[write_buffer]复位，等待下一次的队列切换。

⑶ 如果 busy[read_buffer] != 0，则转向步骤 4，表示读进程目前没有处理数据，可以切换队列；否则继续留在步骤 3 判断 busy[read_buffer]，这样可以保证目前读进程所处理的数据不会收到影响，避免脏数据的出现。

⑷ write_buffer 与 read_buffer 值互换，切换读写进程所在的队列，转向步骤 5。

⑸ 将数据写入写指针当前所在的队列的数据块中，写指针前移。

读取缓冲区算法：

⑴ 如果 change[read_buffer] = 1，则转向步骤 2，表示写指针所在队列已经写满，需要强行切换读写队列；否则转向步骤 3。

⑵ 读进程处理完读指针所指数据块的数据，并将 busy[read_buffer]复位，表示读进程的当前数据已经处理完，可以切换读写队列了，转向步骤 3。

⑶ 如果 reader < Blocknum，则转向步骤 4，表示读指针尚有数据待处理；否则表示没有数据可以处理，转向步骤 1。

⑷ 从读指针所在队列的数据块中读取数据，读指针前移。

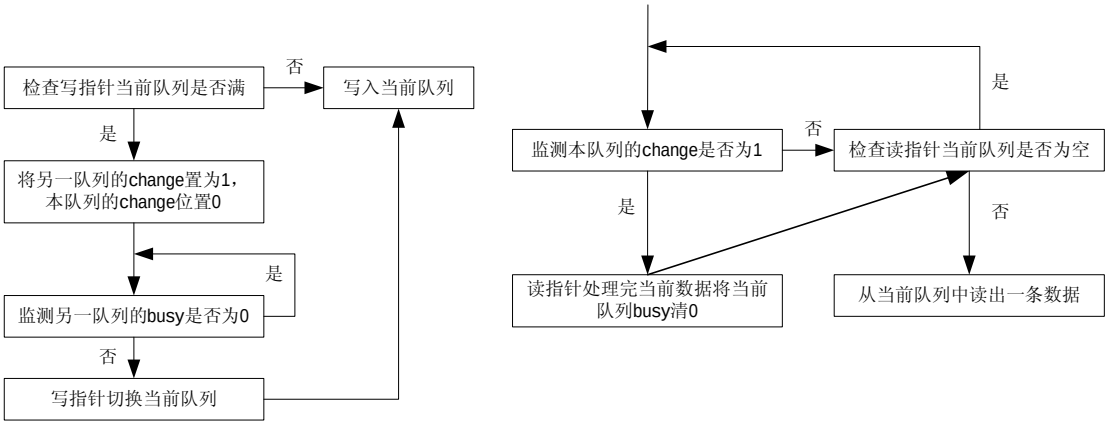

图1 读写交互机制算法

三、算法的复杂性分析

现在来分析处理 n 个报文的所用的时间复杂度。设缓冲队列的长度为 n，第 i 个报文的包长为 l_i，模式串的长度固定为 l_p，设一次缓冲区拷贝耗时为 t_0，由上一节的算法描述，可以知道判断写队列满与判断读队列 busy 信号将耗费 2 个原子动作，将数据拷贝进缓冲区与拷出缓冲区将耗时 $2t_0$，判断读队列的 change 信号与判断读队列空将耗费 2 个原子动作，处理报文的动作可以简单地归为模式串匹配，传统的 KMP 算法的时间复杂度为 $O(l_i + l_p)$，

因此处理 n 个报文的所用的时间复杂度为：$\sum_{i=1}^{n}(4 + 2t_0 + O(l_i + l_p))$。

下面将改进后的互斥机制同传统的并发锁机制在丢报率进行对比。设报文到达流的强度为 $M(pps)$，传统的并发锁机制中单位时间的丢报数为 $M \times O(l_i + l_p)$；改进后的互斥机制中单位时间的丢报数为：$\dfrac{n - \dfrac{n/M}{O(l_i + l_p)}}{2 \times n/M} = \dfrac{M}{2} - \dfrac{1}{2 \times O(l_i + l_p)}$，（写满一个缓缓冲区的时间为 n/M，除以单个报文的处理时间 $O(l_i + l_p)$ 即为处理的报文个数，缓冲队列的长度与处理报文的个数之差即为丢失的报文个数）。可见 M 的值越大，改进机制相对于传统的并发锁机制丢包率就越低，其原因为：

设改进机制的丢报率为 $a(M)$，传统并发锁机制的丢报率为 $b(M)$，为了证明简单，设 $O(l_i + l_p)$ 为常数 K。

M ＝ 1 时，　$b(M) - a(M) = K - 1/2 - 1/2K = \dfrac{K-1}{2K}$

M ＝ n 时，　$b(M) - a(M) = n \times K - n/2 - 1/2K = \dfrac{K \times n - 1}{2K}$

当 $K \geq 1$ 时，改进机制的丢报率低于传统并发锁机制的丢报率，而且 M 的值越大，两者之间的差值就越大。

由于本文所论及的互斥机制，只有写满整个缓冲队列才会切换队列，所以读进程所获取的数据会产生一定的延时，设报文到达流的强度为 $M(pps)$，单个报文所产生的延时为报文写满整个队列的时间（即 n/M ）加上拷贝缓冲区的时间 $n \times t_0$,那么报文的平均延时为 $n/M + n \times t_0$，可见在队列长度和数据拷贝时间一定的情况下，M 越大，延时越小，所以在面向高速主干网络的 IDS 中，这种延时是可以忽略的。

四、 算法的实测分析

在滥用检测系统中，一般将检测分为若干步骤来完成，每一步完成特定的报文过滤和分析过程。本文所设计并实现的 IDS 系统所采用的体系结构分为报文采集（Packet Sensor）、报文分析（Packet Analyzer）、协同管理（Cooperation Manager）、后台数据库等模块【3】。本文构建了简单的测试环境以监测上述读写缓冲区互斥机制的性能。由于只需对双缓冲区互斥机制进行测试，所以实验中屏蔽了 IDS 系统后端的数据库模块和协同模块，而前端的报文采集模块则由一个报文生成器取代，只保留了报文分析模块用来测试读写缓冲区的互斥机制。分析模块的体系结构如图 2，数据接收服务器接收来自前端报文采集模块的报文，写入读写缓冲区；安全检测分析模块从缓冲区读出报文进行分析，将被检测出报文作为安全事件发往后端的协同管理模块，同时记录会话日志。

由于报文分析器主要用于报文的模式串匹配，系统性能与待匹配的模式串的个数有着密切的关系，测试中，规定模式串个数为 180（即对于每一个报文，进入报文分析器之后都要

进行 180 次的模式匹配)；同时，测试的报文也根据实际情况构建了不同长度的报文，而且包括了非攻击报文流量（攻击流量占总流量的 1/3)。

测试中，采用一台 Pentium Ⅲ 1.13G x 2 、512 M 内存的机器作为流量发生器，采用 Pentium Ⅲ 1.26G x2、512M 内存的机器作为报文分析器，2 台机器均采用 RedHat 6.2(kernel-2.2.14)内核版本，Intel PRO/1000 MF 千兆以太网卡进行收发流量。

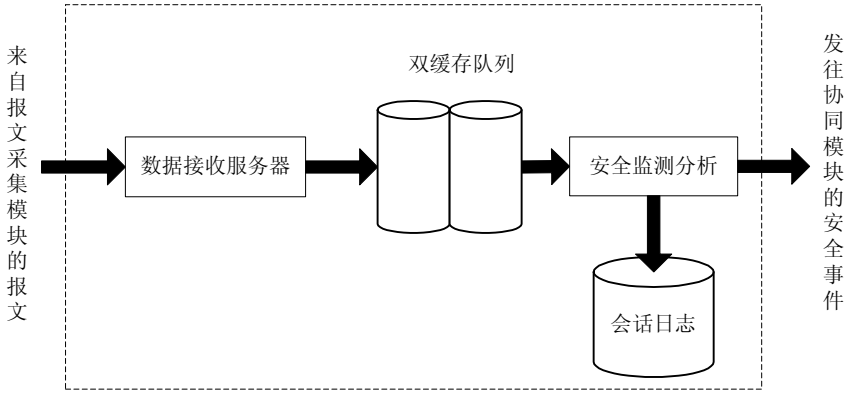

图2 分析模块的体系结构

实验根据报文匹配长度的不同分别对改进的互斥机制与传统的并发锁机制进行了测试，测试结果如下：

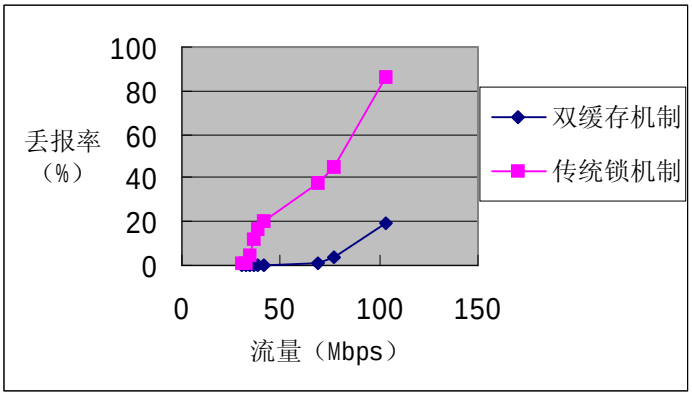

图3 400字节作包头匹配情况两种互斥机制丢报率对比

图 3 显示了平均包长为 400 字节的情况下，对报文的 40 到 120 字节进行模式匹配下两种互斥机制的丢报率情况。

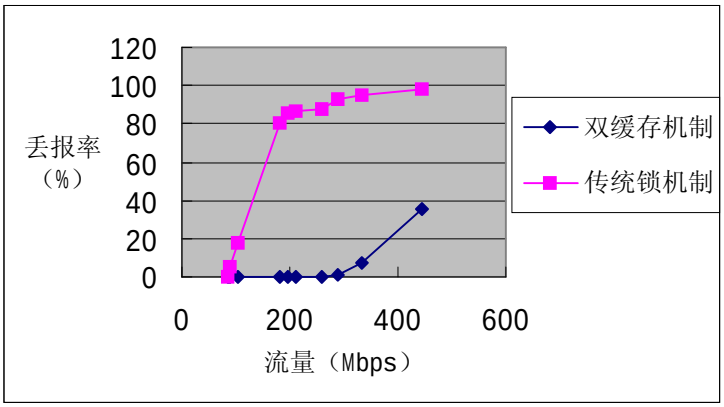

图4 1500字节做包头匹配情况下两种互斥机制的丢报率对比

图 4 显示了平均包长为 1500 字节的情况下，对报文的 40 到 120 字节进行模式匹配下两种互斥机制的丢报率情况。

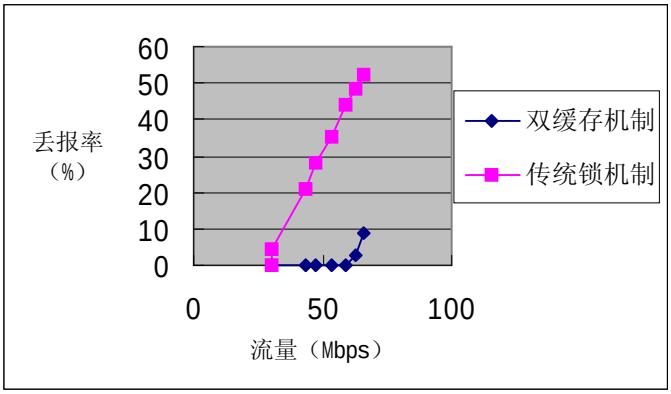

图5 1500字节做全文匹配两种互斥机制的丢报率对比

图 5 显示了平均包长为 1500 字节的情况下，对报文的 40 到 1500 字节进行全文模式匹配下两种互斥机制的丢报率情况。

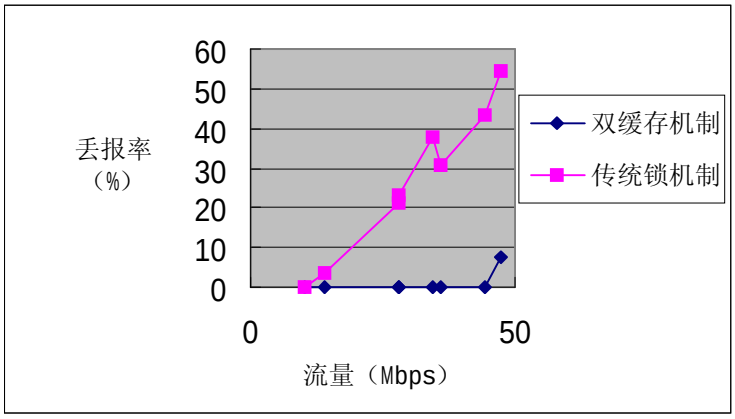

图6 400字节做全文匹配两种互斥机制的丢报率对比

图 6 显示了平均包长为 400 字节的情况下，对报文的 40 到 1500 字节进行全文模式匹配下两种互斥机制的丢报率情况。

从实验结果可以很明显的看出来，改进后的双缓冲区互斥机制在丢报率上远远低于传统的并发锁机制，并且待处理报文的流量越大，两种机制间的性能差异就越大。

五．总结

本文对基于锁机制的读写缓冲区的互斥机制进行了改进，利用双缓存队列将读写进程合理地分配在两个不同的内存队列中同时运行，虽然读进程获取的数据在通过双缓存队列之后会产生一定的延时，但在高速主干网络的大流量背景下，这种延时是可以忽略的。本文所论及的双缓冲区互斥机制解决了传统的加锁机制中资源利用率不高的问题。同时利用写进程优先级高于读进程的设计可以避免频繁的加解锁操作，降低了互斥机制的丢包率，大大提高了 IDS 系统的处理性能。

参考文献：

【1】 Raghu Ramakishnan，Johennes Gehrke. DataBase Management Systems[M] 北京 清华大学出版社，2000 521－594

【2】 （美）Barry Wilkinson Michael Allen，（译）陆鑫达等 《并行程序设计》[M/CD] 机械工业出版社 2002-1-1

【3】 龚俭，董庆，陆晟 《面向入侵检测的网络安全监测实现模型》[J] 微型计算机 22 卷第 2 期

【4】 Lee et al., 1999a] W. Lee, S. J. Stolfo, and K. W. Mok. *A data mining framework for building intrusion detection models.* [J]In Proceedings of the 1999 IEEE Symposium on Security and Privacy, May 1999.