计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2021,30(4):210−215 [doi: 10.15888/j.cnki.csa.007947]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

计算机网络入侵检测系统的多模式匹配算法①

薛　芳[1], 林　丽[2]

[1](集美大学 信息化中心, 厦门 361021)
[2](集美大学 计算机工程学院, 厦门 361021)
通讯作者: 林　丽, E-mail: sometimelin@jmu.edu.cn

摘　要: 为了使网络入侵检测系统能够在高速网络环境中有效的开展工作, 实现计算机网络入侵检测系统的多模式匹配算法优化设计. 首先, 对网络入侵检测的算法与原理进行全面分析. 其次, 对网络入侵检测系统多模式匹配算法的优化思想进行描述, 描述多模式匹配算法, 对算法进行实现, 使模式匹配算法效率得到提高, 以此提高系统检测能力. 通过测试结果表示, 优化后多模式匹配算法能够使网络检测系统的检测性能得到提高.

关键词: 计算机网络; 网络入侵检测; 多模式匹配算法

引用格式: 薛芳,林丽.计算机网络入侵检测系统的多模式匹配算法.计算机系统应用,2021,30(4):210–215. http://www.c-s-a.org.cn/1003-3254/7947.html

Multi-Pattern Matching Algorithm of Computer Network Intrusion Detection System

XUE Fang[1], LIN Li[2]

[1](Informatization Center, Jimei University, Xiamen 361021, China)
[2](Computer Engineering College, Jimei University, Xiamen 361021, China)

Abstract: This study optimizes the multi-pattern matching algorithm for the intrusion detection system in the computer network, so that the system can be in operation in the high-speed environment. First, we comprehensively analyze the algorithm and principle for network intrusion detection. Second, we elaborate the idea of optimizing the multi-pattern matching algorithm for its implementation, so that the algorithm efficiency is increased and then the detection system is improved. In summary, the optimized algorithm can enhance the network detection system.

Key words: computer network; network intrusion detection; multi-pattern matching algorithm

1　引言

随着现代互联网的不断发展, 网络规模持续扩大, 网络的使用也开始发展为全球化的方向. 在此背景下, 网络入侵攻击事件频发. 传统的防火墙技术无法对网络安全性进行保证, 所以需要实现网络入侵检测系统(IDS) 的设计. 网络入侵检测系统指的是主动积极的安全防护技术, 其逐渐发展为网络安全领域研究的重点内容[1]. 在网络入侵检测系统工作过程中, 大部分为被动的监听, 利用关键网段得出网络传输数据包, 通过多种检测分析的方式对数据包进行分析, 从而寻找入侵的痕迹. 在网络入侵检测系统检测的时候, 并不会对网络性能造成影响, 还能够提高网络攻击事件定位的效果. 现代分析网络入侵检测系统的主要方法为基于特征、异常的检测, 由于异常检测过程中的学习时间比较长, 所以一般利用基于模式匹配特征检测[2]. 以此, 本文就对网络入侵检测系统的多模式匹配算法进行全面分析.

2　网络入侵检测

2.1　网络入侵检测的原理

因为传统的安全策略无法有效满足安全的实际需

① 收稿时间: 2020-08-19; 修改时间: 2020-09-15, 2020-11-06; 采用时间: 2020-11-17; csa 在线出版时间: 2021-03-30

求, 从而产生了入侵检测系统, 并且在发展过程中逐渐成为动态安全技术的代表, 使计算机安全领域的发展与研究有所促进. 入侵检测系统为计算机系统与网络中的安全事件检测的技术, 主要包括数据采集、分析与结果处理 3 个功能. 图 1 为入侵检测系统基本的结构.

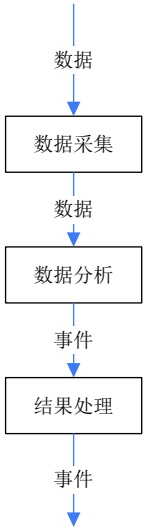

图 1 入侵检测系统的基本结构

在网络入侵检测系统中, 数据分析模块是核心模块, 由于模式匹配原理简单、可扩展性好, 现代网络入侵检测系统利用广泛采用模式匹配的方法进行数据分析. 随着科技的日新月异, 网络的规模也不断扩大, 网络带宽量也逐渐增加, 要求网络入侵检测性能处理的效率得到改进, 否则会导致出现入侵漏报的情况, 也就无法充分发挥网络入侵检测系统的优势.

模式匹配算法应用于入侵检测领域, 是将攻击模式库中的攻击模式与待检测网络数据进行匹配, 若匹配成功则系统判断出现了网络攻击. 目前的模式匹配算法分为单模式和多模式匹配, 其中典型单模式匹配算法包括 KMP 算法和 BM 算法, 多模式匹配算法包括 AC 算法和 AC-BM 算法[3].

2.2 传统网络入侵检测算法

在 1977 年, Boyer 和 Moore 提出了 BM 算法[4], 促进了模式匹配算法的发展. 此算法在进行匹配时包含两个并行算法, 坏字符和好后缀算法, 目的是让模式串每次向右移动尽可能大的距离.

多模式匹配即在一个文本串中同时查找多个模式串, 较之单模式匹配更易应对不断扩大的入侵特征库, 因此现今主流的 IDS 使用的基本都是多模式匹配算法. AC (Aho-Corasick) 算法作为最经典的多模式匹配算法被许多 IDS 采用, 该算法包含预处理和匹配两个阶段, 将待匹配的入侵特征模式串转换为树状有限状态自动机, 然后进行扫描匹配.

BM 算法是一种性能较好的模式匹配算法, 该算法在不匹配的情况下可以产生跳跃, 从而减少匹配次数, 但无法满足日益复杂的网络入侵类型; AC 算法记录的自动机耗费了大量的存储空间. 此外, AC 与 BM 网络入侵检测算法具有较大的漏配量和无效配的情况, 降低系统的检测精准率与匹配速度, 所以就要对网络入侵检测算法进行改进[5].

3 检测系统模式匹配算法优化

3.1 单模式匹配算法的改进

1980 年, Horspool 提出了改进的 BM 算法[6], 也就是 BMH 算法. 简化了 BM 算法, 执行方便, 效率也有所提高. 有 n 长度的文本字符串 T 与 m 长度的模式字符串 P. 在改进 BM 算法过程中, 不匹配过程中的正文与模式移动距离有所扩大. 它不再像 BM 算法一样关注失配的字符, 它的关注的焦点在于匹配文本每一次匹配失败的最后一个字符 X, 根据这个字符 X 是否在模板出现过来决定跳跃的步数, 否则跳跃模板的长度. 如果字符 X 不在模式 P 中, 则跳跃的步数为模式 P 的长度, 字符 X 在模式 P 中, 跳跃的步数为字符 X 距离离尾部最近的字符 X 的距离 (不包括最后一个字符).

$$dist[X]=\left\{\begin{array}{l} m;X字符在P中未出现 \\ m-j;X字符在P中出现 \end{array}\right\}$$
$$j=\max\{j|P[j]=x,1\le j\le m-1\} \quad (1)$$

利用 $dist$ 函数的分析表示, $dist[T[j]]\le m$. 以此看出来, 模式在所设置的长度中最大距离的对右移动. 模式正文匹配的结构详见图 2, 在$T[j]\ne P[j]$时, 此模式通过 BM 算法在$dist[T[j]]$个位置中移动. 利用正文的$i+dist[T[j]]$位置实现重新匹配检查, 忽视 $T[i+v+1]$模式串的检查. 如果模式中没有$T[i+v+1]$, 那么使模式 P 对右移动距离为 $m+1$, 不对$T[i+v+1]$进行匹配检查[7].

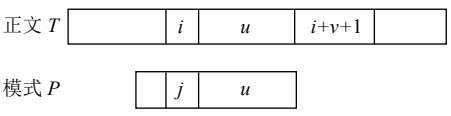

图 2 模式正文匹配示例

本文改进算法的主要思想为:

首先, 假设变量 k 为模式第一次与最后一次的匹配字符, 将 BM 算法作为基础, 在出现不匹配的时候, 能够实现 $T[k+1]$ 的判断. 如果出现 $T[k+1]$, 那么模式移动距离假设为 L, 规定 $L=\max\{dist[T[k+1]],dist[T[j]]\}+1$, 其中的变量 i 和前文相同, 词表达式的含义就是对 $T[k+1]$ 和 $dist[T[j]]$ 的大小对比, 实现正文与模式最大移动距离的进行接下来的匹配. 通过 L 的赋值表示, 在对是否存在 $T[k+1]$ 模式判断过程中, 能够提高下一次匹配的移动模式距离. 以此匹配移动模式, 假如完全匹配, 表示已经成功地进行匹配. 如果没有完全匹配, 移动模式就不发生改变, 直到正文结束[8].

全面考虑从右到左的正文与模式匹配, 模式右边的字符匹配大于左边. 假如具有长模式的情况, 和左边不匹配时候进行对比, 需要的时间成本量比较大. 为了方便算法的实现, 在本文中实现针对性方案的设计. 在算法改进之前对比正文位置字符与模式首字符, 如果不匹配直接判断 $T[k+1]$, 继续下个匹配过程. 之后根据以上改进算法实现匹配对比, 降低不必要匹配过程, 从而节约匹配时间[9].

3.2 多模式匹配算法的改进

网络入侵检测系统能够深入检测执行数据包, 并且全面扫描负载或者已经定义规则集的匹配模式串, 检测是否具备入侵检测事件[10].

3.2.1 分析算法的后缀函数

在 BM 算法中, 在匹配过程中比较了一个模式的长度, 而且存在之前匹配的结果, 在后面的匹配过程中被“遗忘”的情况, 模式串长度越长效率越低. 本文使改进的 BMH 算法和传统 BM 算法实现实验对比, 此实验主要包括的测试指标为 BM 与改进 BMH 算法字符数量、运行的时间和 BM 算法字符比较数量和使用后缀规则数量.

因为入侵检测实现自检, 其模式串的长度设置为 20–30 字符. 本文通过随机的选择开展实验, BM 与改进 BMH 算法的运行时间比详见表 1. 通过表可以看出来, 两种算法的总字符数量是相同的, 但是基于时间复杂度中, 改进 BMH 的性能更加良好.

3.2.2 算法描述

基于传统 BM 算法, 充分考虑文本串中的字符 $T[j]$ 并不会出现在模式串中, 如果其下个字符 $T[j+1]$ 和模式串中的首字符 $P[1]$ 相同, 也就是 $T[j+1]=P[1]$, 以此创建滑动距离函数 2, 简化判断的过程, 使比较次数降低, 提高匹配的效率. 在分析传统 BM 模式匹配算法过程中, 模式串中出现重复字符的时候会导致指针回溯的问题, 那么在本文函数中使用“取子串”的方法能够避免出现指针回溯的问题[11]. 算法流程如算法 1.

表 1 BM 与改进 BMH 算法运行的总时间对比 (单位: s)

模式串长度	BM	BMH
5	24.5211	18.9525
19	24.5211	18.4152
11	28.5242	21.6251
15	32.5264	24.6254
20	36.3521	25.6255
22	39.5422	26.9452
26	44.6254	30.6242
28	50.6254	30.3642
30	52.625	35.9142
35	61.254	40.2541

算法 1. FBM 模式匹配算法

输入: 文本串 T, 模式串 P

输出: 文本串 T 中模式串 P 的出现起始位置

1. 假如此文本串中从位置 j 开始往左一个子串和模式串开展从右到左对比, 如果出现不匹配的情况, 就要调用滑动距离函数 $Slidel(T[j])$;
2. 假如 $T[j]$ 在 P, 在算法过程中重新匹配正文中的 $j+Slidel(T[j])$ 位置;
3. 假如 $T[j]$ 并没有处于 P 中, 那么 $j=j+1$ 并且对滑动距离函数 $Slide2(T[j])$ 调动;
4. 对步骤 3 反复的进行, 直到对匹配位置寻找后才能够进行匹配, 或者匹配的过程失败;
5. 将文本串 T 中的模式串输出 P 的起始位置[12].

BMH 算法在改进过程中的重点为定义给定模式 $P=P_1P_2\cdots P_m$ 从字母到正整数的映射:

$$Slide: C\rightarrow\{1,2,\cdots,m\}$$

其中, $c\in\sum$(假设$\sum$指的是所有英文字母集合)

$Slide1$ 指的是滑动距离函数设置为 1, 其给出正文中的任意字符 c 在模式中的距离, 具体定义为:

$$Slide1[c]=\begin{cases}-1, & \text{如果任意字符}C\text{不出现}P\text{中}\\ & \text{或者}c=P_j(j=m)\\ & \text{但是}c\neq P_j(1\le j\le m-1)\\ m-j, & j=\max(j:P_j=c,1\le j\le m-i-1)\\ & i\text{指的是模式串失配的位置}\end{cases}\tag{2}$$

$Slide2$ 指的是滑动距离函数 2, 其能够给出正文中会出现的任意字符位置, 具体定义为:

$$Slide2[c]=\begin{cases}-1, & \text{若任意字符}c\neq P_i\\ m-1, & \text{若任意字符}c=p_j\end{cases}\tag{3}$$

P_i为模式串首

3.3 算法实现

预处理阶段在读入规则文件的时候, 使模式组划分成为多字节模式组与单字节模式组, 将两者添加到相应模式组中. 针对单字节模式串组, 预处理阶段不进行处理. 改进算法以多字节模式串组计算前缀和后缀索引、跳跃距离 Shift 表, 计算的方法为: 得出最短模式长度 m, 使其成为匹配窗口大小. 取每个模式最后 B 个字符对 Hash 值计算, B 个字符指的是一个块字符. Shift 表中将字符串中全部块字符在 T 时的移动距离进行计算. 针对每个出现的多字节模式串中字符块, 使二维位图 $EXIST\_P$ 中的位置标记成为 1, 其他标记成为 0. 以此, 在匹配的时候如果查找文本字符块处于位图 $EXIST\_P$ 标记成为 0, 那么此字符串并不会在多字节模式串组任何模式串中出现. 二维图计算方法为:

$$EXIST\_P(char1char2) = \begin{cases} 1, char1char2\text{在模式串中} \\ 0, char1char2\text{不在模式串中} \end{cases} \tag{4}$$

以改进算法思想, 图 3 为改进算法的实现流程. 为了能够有效地简化流程, 数组下标为字符的位置[13].

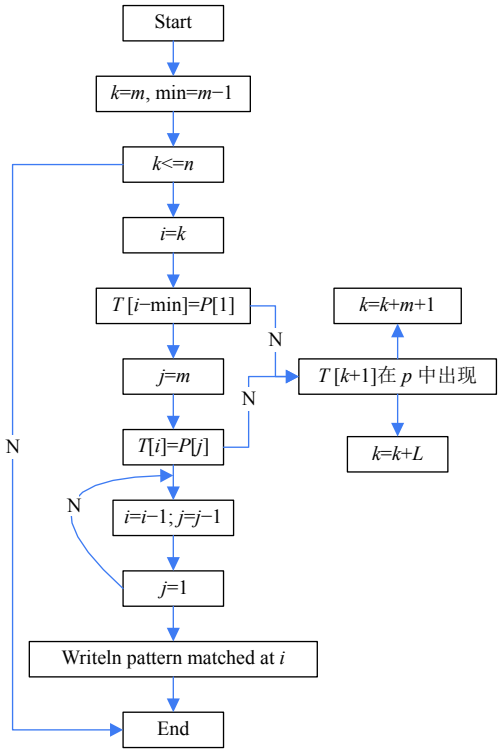

图 3 改进算法的实现流程

具体的流程为:

1) 对 $K \le n$ (n 为待检测字符串 T 的长度) 是否成立判断, 如果 $k<n$, 那么在模式串长度比正文字符串长度要小, 此时就要对其进行匹配检查, 模式和正文对其, 对比正文位置字符和首字符. 假如出现不匹配的情况, 就到最后一步. 假如 $k>n$, 匹配的过程失败, 程序就结束.

2) 从左到右匹配模式和正文, 并且对比, 如果模式字符串和正文字符串进行匹配, 对匹配的位置进行记录, 以此说明能够成功匹配, 结束程序. 假如某位置的字符串出现不匹配的情况, 就进入到步骤 3).

3) 对目前的匹配操作进行判断, 模式中的正文和模式最后一位对应位置是否有下位字符, 以判断结果变量 k 值计算, 之后转到步骤 1)[14].

处理阶段时间复杂度为 $O(m+\sigma)$, 其中 σ 为 x 与已匹配部分在 P 中的位置, 搜索阶段最坏情况下时间复杂度为 $O(mn)$, 模式字符串的长度大小将影响到时间复杂度, 一般文本字符的平均比较数在 1σ 和 $2/(\sigma+1)$ 之间.

图 4 为改进算法的匹配过程, 算法以二维图判断某字符串是否在模式串中出现, 首先读入字符串“st”, $EXIST\_P(st)$ 为 0, 使窗口向后移动一位. $EXIST\_P(tr)$ 值为 0, 继续使窗口向右移动一位. $EXIST\_P(rc)$ 值为 1, 此处会出现匹配, 通过本文算法实现匹配, 查找 Shift 表, 后续文本串根据同样方法进行匹配.

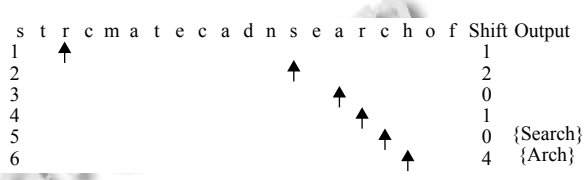

图 4 改进算法的匹配过程

通过上述匹配可以看出来, 使用原本算法匹配的时候, 要计算 9 次 Hash 值并且查找 Shift 表实现跳跃. 改进算法能够以位图判断此字符串是否处于模式串中. 在以上匹配过程中一共计算 6 次 Hash 值, 并且查找 Shift 表实现跳跃.

4 算法的实验和分析

为了对今后算法性能进行校验, 本文用改进后算法和传统 BM 算法实现实验对比. 在 Win 7 环境中实验, 系统配置 2.66 GHz Intel CPU. 实验过程中使用官网中的模式串, 以 Snort 匹配过程, 使所有规则文件规则字符串作为模式组, 在实验过程中依次使用此模式

组对改进算法实现校验. 在查找文本通过 MIT Lincoln 实验室中的 DARPA99 数据集的数据构成, 通过 DARPA99 测试数据集中选择 4 MB 测试数据. 图 5 为在不同最小模式中的算法性能对比. 通过图 5 可知, 在模式串组中具备单字节模式串与两个字节模式串时, 改进算法性能有所提高.

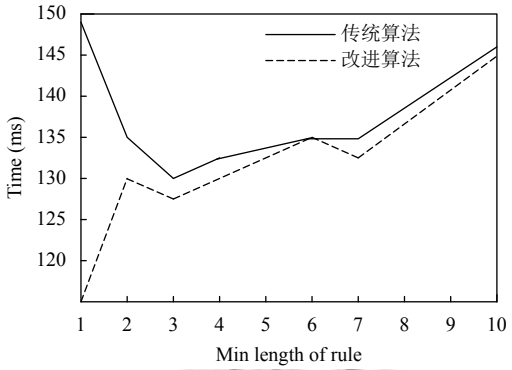

图 5 不同最小模式中的算法性能对比

图 6 为不同模式串集的算法性能, 通过图 6 可知, 在模式串集模式串数量不断增加的过程中, 改进算法与原算法的性能相同. 但是在模式串数量比较少的时候, 改进算法的匹配运行时间比原算法的匹配运行时间要短. 在入侵检测系统中, 匹配规模集中规则数量不超过 400 条, 改进算法的应用价值良好. 增加到 1200 条时, 两种算法应用价值相当. 在 2000 条时, 改进算法使用价值良好. 实际应用情况中, 检测条目越多, 将严重影响网络访问时延, 导致用户上网体验差; 检测条目越少, 匹配到的威胁少, 对网络安全危害增大, 一般检测条目约在 500–800 之间.

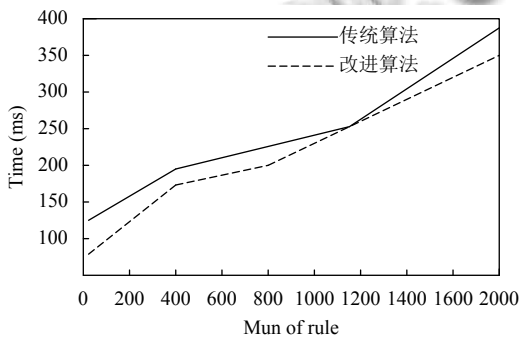

图 6 不同模式串集的算法性能

本文也对改进算法在不同数据包时的情况, 模式串组使用 Snort 规则文件 ftp.riles. 通过图 7 表示, 数据包越大, 改进算法所消耗时间短与原本算法, 能够提高其性能. 本文改进算法的测试平均查询时间为 0.38 s, 传统算法需要 2.16 s, 以此表示能够实现快速查询, 满足用户的实际使用需求.

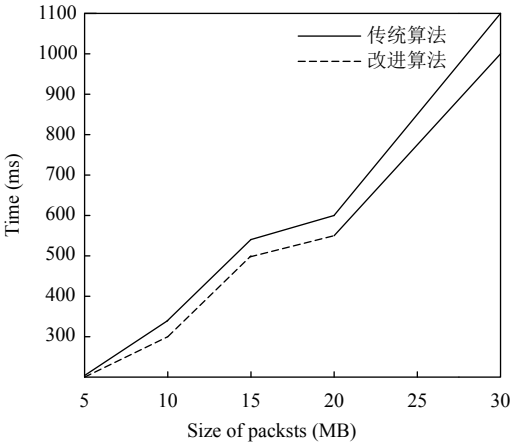

图 7 不同数据包大小的算法性能

通过实验结果表示, 改进之后的 BM 算法性能提高, 匹配效率比传统算法要高. 通过本文中的改进算法能够使入侵检测系统性能得到提高, 实用价值良好.

5 结束语

网络带宽在网络技术持续发展过程中不断增加, 入侵监测系统处理的性能也要相应不断得到提高, 使得大流量网络环境的需求得到满足. 本文对模式匹配算法进行全面的研究, 并且对传统模式匹配算法进行改进. 通过实例测试可以看出来, 改进的多模式匹配算法能够有效满足网络使用过程中的需求, 并且提高系统检测效率. 另外, 改进算法还能够降低冗余移动, 使匹配速度与查找速率得到提高, 对入侵检测系统的性能进行改进.

参考文献

1 于粉娟. 基于多模式匹配算法的计算机网络入侵检测研究. 自动化与仪器仪表, 2018, 15(5): 159–161.
2 周小松, 刘帅. 多网络环境下的差异化入侵特征检测平台的设计与实现. 现代电子技术, 2017, 40(10): 149–152.
3 吕峰, 叶东海, 杨宏, 等. 模糊数据挖掘和遗传算法的网络入侵检测方法. 电子技术与软件工程, 2017, 18(4): 185–186.
4 Boyer RS, Moore JS. A fast string searching algorithm. Communications of the ACM, 1977, 10: 762–772.
5 刘达. 基于朴素贝叶斯分类算法的数据库入侵检测系统.

网络空间安全, 2017, 8(8-9): 32–34.
6 Horspool NR. Practical fast searching in strings. Software Practice and Experience, 1980(6): 50–56.
7 关丽. 基于模式匹配的计算机网络入侵防御系统. 电子制作, 2019, 18(13): 81–82, 96. [doi: 10.3969/j.issn.1006-5059.2019.13.032]
8 鲍海燕. 基于 K-means 算法的入侵检测系统研究. 现代计算机, 2019, 19(23): 9–13. [doi: 10.3969/j.issn.1007-1423.2019.23.002]
9 朱平哲. 网络入侵检测与防护算法系统的实现. 安徽电子信息职业技术学院学报, 2019, 18(3): 7–12. [doi: 10.3969/j.issn.1671-802X.2019.03.002]
10 常刚, 罗作民. 基于聚类算法的入侵检测系统设计. 自动化与仪器仪表, 2018, 15(9): 110–113.
11 吴志福. 基于一次判断双字符比较的模式匹配算法. 科技通报, 2018, 34(4): 240–242, 261.
12 刘建. 基于改进神经网络的网络入侵检测. 科技创新与应用, 2018, 16(2): 11–12, 14.
13 Chen JL, Li QY, Li JQ, *et al*. DOA estimation of MIMO Radar based on covariance matching SL0 algorithm. Radar Science and Technology, 2019, 17(1): 19–24.
14 王子玲, 贾舒宜, 修建娟, 等. 基于人工神经网络的多模型目标跟踪算法. 海军航空工程学院学报, 2019, 34(4): 343–348.