

一种面向入侵检测的改进 AC 算法

徐 红, 秦志光

(电子科技大学 计算机科学与工程学院, 四川 成都 610054)

摘 要: 在研究入侵检测系统中 AC 算法应用 3 的基础上, 提出一种改进的 AC 算法——双重压缩 AC 算法. 在算法中设计了新的 AC 状态自动机存储空间压缩方法, 并给出与之对应的匹配流程. 在 Snort 中进行测试, 测试结果显示双重压缩 AC 算法的空间和时间性能比基本 AC 算法分别提升了 52.2% 和 73.5%.

关键词: 入侵检测; 模式匹配; AC 算法

中图分类号: TP309

文献标识码: A

文章编号: 1000-7180(2010)11-0109-04

An Improved AC Algorithm for Intrusion Detection

XU Hong, QIN Zhi-guang

(School of Computer Science and Engineering, UEST, Chengdu 610054, China)

Abstract: On the basis of analyzing AC algorithm in IDS, this paper presents an improved AC algorithm named double compression AC algorithm (DCAC), describes it's new state automaton storage compression and corresponding matching process. DCAC has been tested in Snort. The results show that DCAC can averagely enhance the performance of Snort by 52.2% in terms of memory usage and by 73.5% in terms of processing time compared to the standard AC algorithm.

Key words: intrusion detection system; pattern matching; Aho-Corasick (AC) algorithm

1 引言

入侵检测系统(IDS)作为维护网络安全性的重要技术之一, 已经得到了广泛的应用. 模式匹配是 IDS 的关键技术之一, 模式匹配算法与 IDS 的性能息息相关. 随着网络速度和网络检测工作量的不断增长, IDS 要求模式匹配算法更加高效, 高效的模式匹配算法设计成为网络安全的研究热点之一^[1].

模式匹配是指在文本串 T 中查找模式串 P 的过程, 若找到则称为匹配成功, 否则匹配失败. 在 IDS 中, T 对应系统接收到的数据包或会话中的数据, P 对应入侵特征库中的入侵模式. 模式匹配按照匹配模式的数目分为单模式匹配和多模式匹配两类, 由于多模式匹配通过一次扫描可以找到多个模式的所有出现, 较之单模式匹配更易应对不断扩大的 IDS 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, 最好情况或最坏情况下, AC 算法模式匹配的时间复杂度都是 $O(n)$. AC 有限状态自动机的存储占用了大量的内存资源, 降低了算法的 cache 性能, 巨大的存储开销是影响 AC 算法性能的重要因素. 一些研究者基于优化 AC 有限状态自动机存储空间, 提出了相关的改进方法以提升 AC 算法的性能^[2-3].

文中从空间开销的角度分析基本 AC 算法及其改进, 提出了一种降低空间开销的改进 AC 算法——双重压缩 AC 算法(Double Compression AC Algorithm, DCAC).

2 AC 算法

2.1 基本 AC 算法

Aho, A. V. 和 M. J. Corasick 于 1975 年提出了 AC 算法^[4]. 算法运行过程包含预处理和匹配两个阶段, 算法先对入侵特征模式串集合进行预处理, 构

建有限状态自动机 FSA(finite state automata), 然后依据该 FSA 对文本串 T 扫描一次就可以找出与其匹配的所有模式串。

在预处理阶段, 算法首先构造 3 个函数: 转移函数 $goto(g)$ 、失效函数 $fail(f)$ 和输出函数 $output$ 。转移函数的作用是指明在当前状态读入一个文本字符后应到达的新状态, 它的构造过程是: 从状态 0 出发, 由当前状态和从入侵特征模式串中新取出的字符确定下一状态。如果存在从当前状态出发并标注该字符的矢线, 则将矢线所指的状态赋为当前状态, 否则, 添加一个标号比已有状态标号大 1 的新状态, 并用一条矢线从当前状态指向新状态, 并将新状态赋为当前状态。所有的模式串处理完后, 再画一条从 0 状态到 0 状态的自返线, 标注的字符为不能从 0 开始的字符集。这样, 每个模式都可以由一条从初始状态出发的路径标识出来。

失效函数用来指明在某个状态下, 读入的字符不匹配时应转移到的状态。失效函数是逐层构造的。所有第一层状态的失效函数为 0; 对于非第一层状态 s , 若其父状态为 r (即存在字符 a , 使 $g(r, a) = s$), 则 $f(s) = g(f(r), a)$, 其中状态 s^* 为追溯 s 的祖先状态得到的第一个使 $g(f(s^*), a)$ 存在的状态。

输出函数的作用是在匹配过程中输出已经匹配的模式串。该函数的构造分两步: (1) 在构造转移函数时每处理完一个模式串, 则将该模式串加入到当前状态的输出函数中; (2) 构造失效函数 f 时, 若 $f(s) = s'$, 则 $output(s) = output(s) \cup output(s')$ 。例如, 模式集 $P = \{he, she, his, hers\}$ 的转移函数、失效函数和输出函数如图 1 所示。

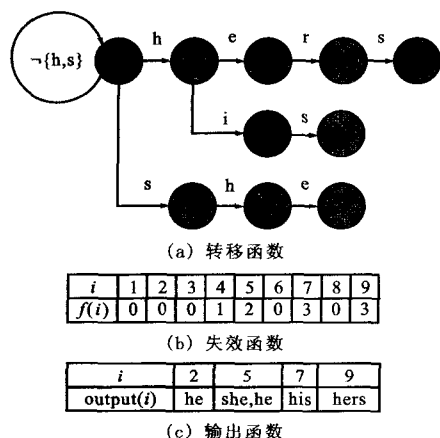


图 1 模式集 P 对应的有限状态自动机

在此基础上 Aho, A. V. 和 M. J. Corasick 将失效

函数与转移函数融合构建了 next move 函数, 使得对文本进行匹配时对每个输入字符仅进行一次状态转移, 提高了算法的运算效率。AC 算法根据 next move 函数构造了一个确定性有限状态自动机 (DFA), 根据这个 DFA 和输出函数就可以进行匹配。

2.2 AC 算法的改进

基本 AC 算法通常用二维矩阵存储有限状态自动机, 矩阵的行表示自动机的状态, 列表示输入的字符, 矩阵的元素表示在当前状态和输入字符情况下将要跳转到的下一个状态。如果入侵特征模式串长度为 M , 则对应了自动机的 M 个状态, 并且对于每个状态都可能有的 256 种不同的输入字符, 则矩阵的容量为 $256 \times M$ 。因此, 对于拥有大量入侵特征模式的 IDS, AC 自动机的存储占用了大量的存储空间, 从而直接影响着算法的 cache 性能。而 AC 自动机的每个状态对应的下一个有效转移状态通常只有很少几个, 使得存储自动机的二维矩阵的很多元素是 0, 这样就浪费了大量的存储空间。为了尽可能减少无谓的空间开销, 人们提出了一些方法以改进 AC 算法。

Norton^[5] 基于稀疏向量压缩的思想, 提出了采用 banded-row format 存储格式对存储自动机的矩阵的每一行数据进行压缩。设向量为 $Vector = (0, 0, 0, 0, 2, 9, 0, 0, 0, 0, 6, 0, 7, 0, 0, 0, 0, 0, 0, 0)$, 采用 banded-row format 压缩后 $Vectorb = (9, 5, 2, 9, 0, 0, 0, 6, 0, 7)$ 。这种储存格式由三部分组成: 其中 “2, 9, 0, 0, 0, 0, 6, 0, 7” 记录了原向量中第一个非 0 元素到最后一个非 0 元素的所有元素, 这些元素被称为一个带; “9” 表示组成这个带的元素个数; “5” 表示这个带在原向量中的起始位置。这种存储格式便于随机访问元素, 可是它仍含有较多的元素 0。

用 banded-row format 存储稀疏向量, 只能对带两端的 0 元素进行压缩, 而对于带里的 0 元素却无能为力。为了尽可能地压缩向量, 可以采用 sparse-bands format^[6] 存储稀疏向量。它的思路是: 当带里的两个非 0 元素间的 0 元素个数超过指定的阈值时就将带分段, 以压缩这些 0 元素。假设阈值为 2, 将向量 $Vector$ 用 sparse-bands format 压缩后 $Vector = (2, 2, 5, 2, 9, 3, 11, 6, 0, 7)$ 。这种格式各元素的含义是: 第一个 “2” 表示带分段数量; 第二个 “2” 表示第一个段包含的元素数量; “5” 表示这个段在原向量中的起始位置; “2, 9” 即对应该段包含的元素。后面的数据分别对应着第二个段的元素数量、段起始位

置和元素.

通过采用 banded-row format 和 sparse-bands format 格式存储 AC 自动机,压缩了 AC 自动机的存储空间,有效地改进了 AC 算法的 cache 性能.

3 双重压缩 AC 算法

基本 AC 算法耗费大量存储空间,基于 banded-row format 和 sparse-bands format 的改进方法实现了对存储空间的压缩,然而它们只是对矩阵的行进行压缩,没有从列的角度去考虑.存储 AC 自动机的矩阵有 256 列,代表了 256 个 ASCII 码.在实际应用过程中,入侵特征模式库并未涵盖这所有的 256 个 ASCII 码字符,而往往只是涉及到了少量的字符,导致了这个矩阵中大部分 ASCII 码字符对应的列为全 0 列;与此同时,在矩阵的各行中还存在许多可压缩的 0 元素,它们无谓地占据了大量的存储空间,降低了 AC 算法的 cache 性能.

文中基于压缩矩阵行和列的思想,提出双重压缩 AC 算法,在矩阵列的方向压缩全 0 列.对于非全 0 列中可压缩的 0 元素,在矩阵行的方向对其进行压缩.双重压缩 AC 算法的运行过程可分为预处理和匹配两个阶段.

3.1 双重压缩 AC 算法预处理

预处理要完成的任务是依据入侵特征模式库构建 AC 自动机,并按双重压缩 AC 算法的压缩思想对存储 AC 自动机的矩阵进行压缩.预处理分为三个阶段:

第一个阶段按基本 AC 算法的方法构造 AC 自动机,将 AC 自动机用二维矩阵存储.后两个阶段分别从列和行的角度压缩这个矩阵.

第二个阶段压缩矩阵的全 0 列,其流程是:

(1) 查询矩阵中的全 0 列,将全 0 列对应的 ASCII 码字符记录在未用字符表中;

(2) 删除矩阵中所有的全 0 列.

第三个阶段:对前一个阶段处理后的矩阵的各行用 sparse-bands format 格式进行压缩,其流程是:

(1) 遍历矩阵各行,根据设定的阈值将行向量分段,记录各段的元素;

(2) 计算各段的长度以及起始位置;

(3) 按 sparse-bands format 格式存储处理后的各行数据.

3.2 双重压缩 AC 算法匹配

匹配时用经过预处理后建立的 AC 自动机扫描文本,以查找是否存在匹配的模式串,匹配流程是:

(1) 从待匹配文本 T 中新读取一个字符 CHAR;

(2) 在未用字符表中查找新读入的字符,如果在其中找到 CHAR,就置当前状态 current state 为 0,流程转至(1);否则继续下面的操作.

(3) 将 CHAR 减去未用字符表中 ASCII 码小于 CHAR 的字符的数量;

(4) 将(3)的计算结果根据 sparse-bands format 格式的定义查找到对应的状态,并将其置为 current state;

(5) 根据输出函数 output 判断是否有匹配存在;

(6) 如果文本读取完毕就退出,否则返回(1).

4 实验评估

为了测试文中提出的双重压缩 AC 算法的性能,并与基本 AC 算法及其改进相比较,将 Snort 作为测试用入侵检测系统. Snort 版本为 2.8.2.2,规则为 snortrules-snapshot-2.8. 实验在 Windows XP 下进行,编译工具为 VC6.0. 硬件环境为 CPU: Intel E2160, 1.8GHz, 内存 1GB.

测试数据采用麻省理工学院林肯实验室提供的 1999DARPA 入侵检测测试数据集. 数据集取自一个虚拟的美国空军基地的 7 个星期的网络流量,其中包含大量的正常网络流量和各种攻击,具有很强的代表性. 文中选取第四周星期一至星期五的输出数据 outside.tcpdump 作为测试数据.

双重压缩 AC 算法与其它算法的空间开销对比如图 2 所示,结果显示这些算法中双重压缩 AC 算法的空间开销最小. 双重压缩 AC 算法占用的存储空间为 635.13MB,比占用 1327.96MB 的基本 AC 算法性能提升 52.2%,比占用 688.89MB 的 AC-SPARSEBANDS 算法性能提升 7.8%. 时间性能测试时为避免进程调度引入的干扰,每组取 10 次测试的平均值. 测试数据对比如表 1 所示,可见双重压缩 AC 算法在其中时间性能最优,匹配速度平均比基本 AC 算法提升了 73.5%,比 AC-SPARSEBANDS 算法提升了 17.5%. 综合而言,双重压缩 AC 算法实现了对 Snort 性能的提升.

表 1 snort 中算法时间性能测试结果 (时间/s)

测试数据(outside. tcpdump)	AC-STD	AC-BANDED	AC-SPARSEBANDS	DCAC
Mon(206MB)	155.2300	80.6154	41.0032	28.7887
Tues(287MB)	222.4250	126.6900	63.0658	56.3814
Wed(304MB)	248.0290	147.9350	92.4920	75.8100
Thurs(381MB)	278.0820	161.9540	99.4688	84.7250
Fri(249MB)	238.1780	134.7470	70.4000	56.5940

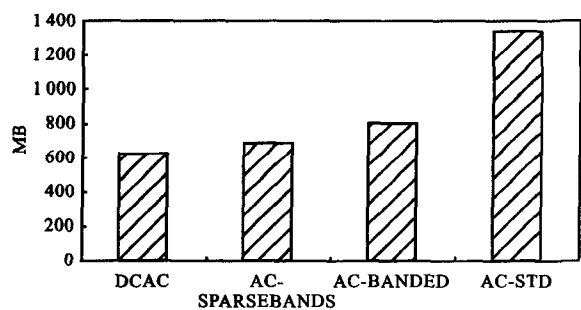


图 2 算法存储 snort 规则集空间开销对比

5 结束语

文中对 AC 算法及其改进在入侵检测系统中的应用进行了研究,针对 AC 自动机大量占用无效存储空间的情况,基于存储空间压缩提出了双重压缩 AC 算法,并将其与 AC 算法及其改进作了对比,结果表明双重压缩 AC 算法在空间性能和时间性能上都更优越.

参考文献:

[1] Tuck N, Sherwood T, Calder B, et al. Deterministic memory-efficient string matching algorithms for intrusion

detection[C]//Proc. of the IEEE INFOCOM conference. Hong Kong, China: Institute of electrical engineers inc., 2004:333-340.

[2] 卢汪节,鞠时光.入侵检测系统中一种改进的 AC 算法[J].计算机工程与应用,2006(15):146-148.

[3] Yu Jianming, Xue Yibo, Li Jun. Memory efficient string matching algorithm for network intrusion management system[J]. Tsinghua Science and Technology, 2007, 12 (5):585-593.

[4] Alfred V Aho, Margaret J Corasik. Efficient string matching:an aid to bibliographic search[J]. Communications of the ACM,1975,18(6):333-340.

[5] Norton M. Optimizing pattern matching for intrusion detection[EB/OL]. [2004-12-10]. <http://www.NIDSresearch.org>.

[6] Martin Roesch. Snort user manual[EB/OL]. [2008-08-28]. <http://www.snort.org/docs/snort-htmanuals/ht-manual-282/>.

作者简介:

徐红女,(1981-),硕士.研究方向为计算机体系结构.
秦志光男,(1956-),教授,博士生导师.研究方向为计算机开放系统与网络安全、信息系统安全、智能交通、电子商务.

作者简介:

邓冬虎男,(1986-),硕士研究生.研究方向为 MIMO 系统空时编码技术.
李宏伟男,(1966-),副教授,硕士生导师.研究方向为移动通信技术.
朱仁飞男,(1986-),硕士研究生.研究方向为雷达信号处理.
于涛男,(1981-),硕士研究生.研究方向为电子与信息对抗.

(上接第 108 页)

[6] Hassibi B, Hochwald B M. High-rate codes that are linear in space and time[J]. IEEE Trans on Information Theory, 2002,48(7):1804-1824.

[7] Antonio Fasano, Sergio Barbarossa. Information lossless full-rate full diversity trace-orthogonal space-time codes[C]// IEEE Signal Processing Magazine. Cannes, 2006:1-5.