病毒特征检测中改进的多模式匹配算法*

罗 玺，罗 红，王晓东，刘洪坤
（空军工程大学 电讯工程学院，西安 710077）

摘 要：针对病毒特征检测中码串长度对模式匹配算法性能影响的问题，结合基于码串长度的特征集自适应分类思路，提出了两种改进的多模式精确匹配算法，即 NAC_BM 和 NWM_QS。改进算法通过引入文本窗口的前缀字符块 W_B 增加了跳跃距离，减少了匹配次数，加快了匹配效率。初步实验证明，改进算法在执行时间和速率上优于原算法。

关键词：病毒特征码匹配；多模式匹配算法；AC_BM；WM_QS
中图分类号：TP393.08　　**文献标志码**：A　　**文章编号**：1001-3695(2012)10-3828-03
doi:10.3969/j.issn.1001-3695.2012.10.059

Improved multiple pattern matching algorithm in virus characteristic code detection

LUO Xi, LUO Hong, WANG Xiao-dong, LIU Hong-kun
(*College of Telecommunication Engineering, Air Force Engineering University, Xi'an* 710077, *China*)

Abstract: As to the influence of code length on pattern matching algorithms in feature-based virus detection, this paper proposed two modified precise multi-pattern matching algorithms, NAC_BM and NWM_QS, by combing the feature-based adaptive classification notion based on code length. The advanced algorithms increased the skip distance, decreased the matching number by introducing the prefix character piece of text window, thus improved the matching efficiency. The experimental results show that the novel algorithms are better than the original ones in execution time and speed.

Key words: virus characteristic code matching; multiple pattern matching algorithm; AC_BM; WM_QS

病毒特征码检测技术凭借其快速、高效、精确等特点，被广泛应用于现有各种杀毒软件中[1]。而特征码检测的核心环节就是对模式匹配算法的运用。一般来说，模式匹配算法按照匹配数目分为单模式和多模式，按照匹配精度分为精确和近似[2]。就现有经典算法来看，单模式精确匹配算法大多针对单一病毒特征码进行检测，而多模式精确匹配算法可对病毒特征库进行全面扫描，但特征码过小影响后者性能发挥。本文通过分析和对比现有多模式精确匹配算法，结合病毒特征码检测特点，提出 NAC_BM 和 NWM_QS 两种改进的模式匹配算法。

1 病毒特征码分析

所谓特征码，就是从病毒样本中提取的不超过 64 Byte 且能独一无二地代表病毒特征的十六进制代码。病毒特征码检测就是通过对这些独有的特征指令序列和普通文件的对比匹配，从而判断其是否为病毒体[3]。特征检测法的优点是准确率高、误报率低，缺点是病毒库过大、扫描速度慢等。就目前来看，传统的病毒特征匹配算法选择过于单一，未能针对特征码串的特点进行分类处理，没有完全发挥出算法的性能优势。例如，AC 算法采用自动机数据结构，在特征码长度较长的情况下，构建自动机将会消耗大量内存。而 Wu-Manber 算法中最短特征串长度 L 成为制约其发展的主要因素。因此，本文提出依据码串长度对特征集进行分类的原则，采用改进的 AC_BM 算法和 WM_QS 算法分别实现对分类后病毒特征码的匹配，如图 1 所示。

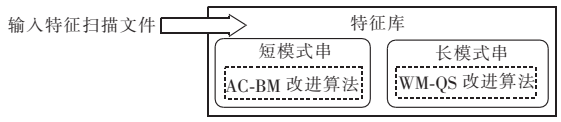

图 1　特征码自适应分类结构

2 多模式精确匹配算法

2.1 AC_BM 算法

AC_BM 算法是在经典多模式精确匹配算法 AC 算法的基础上，结合 BM 算法的两条启发性规则，在比对匹配时尽可能跳过一定的距离，从而减少了匹配次数，有效提高了算法的效率和性能[4,5]。AC_BM 算法采用一种基于自动机的算法，它首先根据模式集构建一个搜索树，然后再对待搜索文本从左至右进行扫描。

假设模式集 P 为 {abcde, abdeb, abce, abac}，文本串 T = {adabcedaacdbfbbab}。AC_BM 算法匹配过程如图 2 所示。

具体过程为：首先取模式串集最小长度记为 L，模式 P 为 {abcde, abdeb, abce, abac}，则 $L=4$。移动前先将模式集树中最短的模式"abce"与文本字符串右对齐，从左至右检查字符是否匹配。结果发现字符文本串"b"≠"a"，根据坏字符原则，模

收稿日期：2012-03-15；**修回日期**：2012-04-25　　**基金项目**：陕西省自然科学基础研究基金资助项目(2010JM8004)

作者简介：罗玺(1988-)，男，陕西西安人，硕士研究生，主要研究方向为网络安全(luoxi19887302@126.com)；罗红(1963-)，男，副教授，博士，主要研究方向为网络安全、网格计算；王晓东(1974-)，男，讲师，博士，主要研究方向为信息安全；刘洪坤(1968-)，男，副教授，博士，主要研究方向为信息安全.

式集树向左移一位;第二次匹配,检查发现"f" ≠ "a",模式树左移 L;第三次匹配模式集树左移 2 位;第四次匹配模式集树左移 2 位;第五次匹配模式树左移 2 位,匹配成功。

AC_BM 算法通过对多模式经典算法 AC 和 BM 的融合改进,运用两个启发性规则解决了文本串搜索无跳跃问题,减少了匹配次数,提升了算法速率。但是由于搜索树的大小与字符集的大小有关,因此 AC_BM 算法一般需要较大的内存资源,特别是当模式集长度较长时,算法需要的内存会迅速膨胀,无法满足大模式集的匹配检测,因此不适合于长特征码的匹配检测。

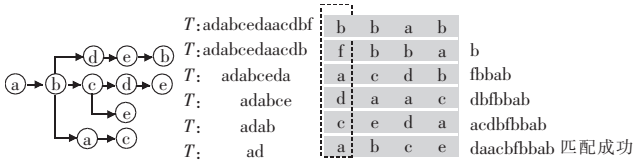

图 2　AC_BM 算法匹配过程

2.2　WM_QS 算法

WM_QS 算法是人们通过对 Wu-Manber 经典算法的理解和认识,结合 QS 算法的特点,充分考虑了原有窗口后的字符块 W_B 来计算当前窗口的移动距离(B 通常取 2 或 3)[6,7]。在预处理阶段,WM_QS 算法通过建立三个表,即 SHIFT、HASH 和 PREFIX 表来完成对算法的先期处理。SHIFT 表用来计算字符跳跃距离;HASH 存储着指向相同后缀的模式串的指针;PREFIX 表用来存储尾块字符散列值相同的模式串的首块字符散列值。WM_QS 算法执行过程如图 3 所示。

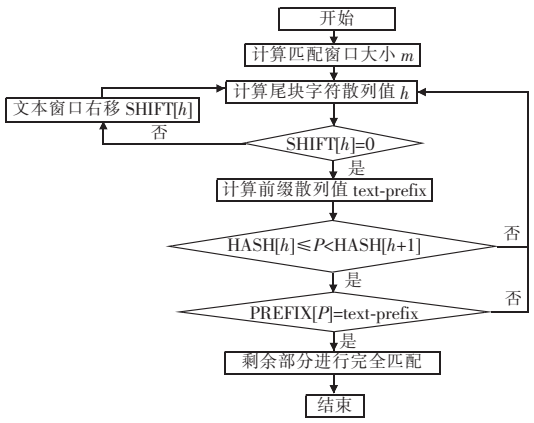

图 3　WM_QS 算法流程

WM_QS 算法的 HASH 表和 PREFIX 表结构同经典 Wu-Manber 算法,其 SHIFT 表描述如下:由于每次跳跃距离至少为 1,在计算跳跃距离时可以考虑当前窗口的紧邻后一个字符带来的信息。将该字符和当前窗口的最后 $B-1$ 个字符一起考虑,作为获得跳跃距离的根据。设该字符块为 $K_B = \{k_1 \cdots k_b\}$,针对 K_B 在关键字集合中的出现情况,按照如下规则计算跳跃距离:

a)K_B 作为子串出现在某些关键字中。设 K_B 在所有关键字中最右的出现位置为 q,此时可以安全的跳跃字符数为 $m-q$,即 SHIFT[i] $= m-q$。

b)K_B 不出现在任何关键字中,但是其某一长度的后缀出现在某些关键字的前缀中。设该后缀的长度为 $L(0<L<B)$,此时可安全的跳跃 $m-L+1$ 个字符,即 SHIFT[i] $= m-L+1$。

c)否则,可以跳跃 $m+1$ 个字符,即 SHIFT[i] $= m+1$。

3　改进的模式匹配算法

3.1　改进的 AC_BM 算法

AC_BM 算法在 AC 算法的基础上,通过对 BM 算法的两条启发性规则运用,大大减少了匹配的次数,提高了效率。但是从算法的思路上来看,由于最短模式串 L 的存在,且文本串是从最短模式串内第一个字符开始计算模式串树的移动位置,算法的跳跃距离仍未达到最大值,当文本字符不存在于模式串中,其最大跳跃距离为 L。因此考虑借助 QS 思想引进字符块 W_B,即同时检查文本窗口内的第一个字符和前一个字符在模式集中的位置来计算模式树左移距离,其最大跳跃距离可达到 $L+1$。假设文本串长度为 M,最短模式串长度为 L,则令文本窗口内的第一个字符与其前一个字符为 K_{M-L+1} 和 K_{M-L},移动距离规则定义如下:

a)若 K_{M-L} 不在模式集中,则模式集树的偏移量为模式集中最短模式串的长度 $L+1$。

b)若前缀字符块 W_B 不在模式集中,则模式集树的偏移量为模式集中最短模式串的长度 L。

c)若前缀字符 W_B 是某一模式串的子串,则将模式集树移动到子串首次出现的位置。

下面是改进算法 NAC_BM 的匹配过程,如图 4 所示。假设模式集 P 为 {abcde, abdeb, abce, abac},文本串 T = {adabcedaacdbfbbab}。

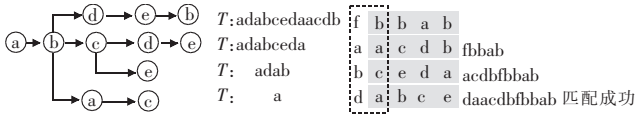

图 4　NAC_BM 算法匹配过程

改进算法的具体实现过程为:第一次匹配,考察紧邻文本窗口的前一个字符"f",检查发现未在模式集中出现,根据移动距离规则 a),模式树左移 $L+1$;第二次匹配,检查前缀字符块"aa",发现未在模式集中出现,根据移动距离规则 b),模式树左移 L;第三次匹配发现"bc"在模式集中出现,即前缀字符是模式串"abce"的子串,则将模式集树移动到子串首次出现的位置即左移 2 位,匹配成功。

由于 NAC_BM 算法在匹配比对中,引用了前缀字符块 W_B,进一步增大了模式树跳跃距离,因此可以明显发现同样的文本输入,改进的算法仅需要四次即可匹配成功,而原算法则需要六次。倘若文本输入和模式集更为复杂,NAC_BM 算法效率会更为明显。NAC_BM 算法适用于分类后的病毒特征码检测,即特征库中长度较短的特征码串,这样既避免了 AC_BM 算法由于特征集过长导致的自动机树引起内存膨胀问题,同时也最大限度地发挥了算法性能。

3.2　改进的 WM_QS 算法

WM_QS 算法在 Wu-Manber 的基础上,结合 QS 算法,充分考虑了紧邻文本窗口的下一个字符和模式集的关系,从而减少了匹配次数,提高了效率。但作为病毒特征码检测,WM_QS 算法仍存在一些问题,主要体现在以下几个方面:

a)WM_QS 算法主要是普遍增大了文本窗口内的跳跃距离,由原来保守的最大移动距离 $m-B+1$ 扩大为 m,且通过新的规则定义来计算移动距离,但当文本匹配时无从明显判断,极有可能继续跳过进行下一次匹配。导致的原因是其仅仅考

虑当前文本窗口的 $B-1$ 个字符和下一个字符，算法虽然加大了跳跃距离，但是容易发生漏检现象。

b）由于病毒特征码串的长度普遍偏长，因此在运用 WM_QS 算法时就不可避免地存在 HASH 表链过长的情况，即使对跳跃距离进行了改进，但进入精确匹配后，仍要面对大量具有相同后缀的模式串匹配问题，而 WM_QS 在进入精确匹配后采用 BF 算法，执行效率偏低。

c）针对病毒码串的特点，算法的最短模式长度 minlenP 势必要加大，一般都在 10 个字符以上，因此原有的 B 值取 2 或 3 就会加大 PREFIX 表前缀比较的负担，造成算法的匹配效率急速下降。

针对上述三点不足对 WM_QS 进行改进，改进算法 NWM_QS 仍分为预处理阶段和匹配阶段。在预处理阶段，仍然采用 SHIFT、HASH 和 PREFIX 三个表，但对 SHIFT 表的移动距离重新定义，即在计算跳跃距离时考虑当前窗口的 B 个字符和紧邻文本窗口的后一个字符带来的信息作为获得跳跃距离的根据。当文本窗口内的 $B+1$ 个字符在模式串中出现时，取其最大值计算移动距离，当 SHIFT[i] = 0 时进入 HASH 表继续匹配；若未出现，则移动最大距离 m。假设模式集 $P=\{p1,p2,p3\}$，其中 $p1=\{\text{father}\}$，$p2=\{\text{wife}\}$，$p3=\{\text{today}\}$，文本串 T 为 {rwdsfkwifeosfatherhavetoday}，匹配过程如表 1 所示，SHIFT 表如表 2 所示。

表 1 NWM_QS 算法匹配过程

r w d s f k w i f e o s f a t h e r h a v e t o d a y
shift(ds,sf) = 4 ↑
shift(wi) = 2 ↑匹配成功 $p2$，右移一位
shift(eo,os) = 4 ↑
shift(at) = 1 ↑匹配成功 $p1$，右移一位
shift(he,er) = 4 ↑
shift(av,ve) = 4 ↑
shift(od) = 1 ↑匹配成功 $p1$

表 2 NWM_QS 算法 SHIFT 表

h	1	2	3	4	5	6	7	8	9	10
字符块	fa	at	th	wi	if	fe	to	od	da	else
shift(h)	2	1	0	2	1	0	2	1	0	4

NWM_QS 的实现步骤是：首先比较文本窗口 m 内的字符，若窗口后 B 位即 $T[m-B]\cdots T[m]$ 在 SHIFT 中，则计算其移动距离；若窗口后 B 位不在 SHIFT 中，比较紧邻文本窗口的后一位即 $T[m-B+1]\cdots T[m+1]$ 的情况，并计算移动距离。如案例所示：$m=4$，$B=2$，先比较"rwds"，发现"ds"不在 SHIFT 中，考虑后一位"sf"，不在，移动 4，考虑"fkwi"，SHIFT[wi] = 2，移动 2，SHIFT[fe] = 0，转到 HASH 表查找指针，发现有模式串和文本串后缀一致，转到 PREFIX 表查看其前缀散列值是否相同，相同则匹配，文本窗口右移 1。

在大规模特征码匹配阶段，由于特征码串过长，为避免 PREFIX 表过长应考虑适当增加 B 值，当 $m>10$，可考虑使字符块 $B=5$。同时，对于进入精确匹配阶段后前缀和后缀匹配问题，可运用 BM 算法加速算法的效率。NWM_QS 算法在原有算法的基础上，采用 $B+1$ 个字符对文本串进行搜索匹配，进一步加大了字符的移动距离，同时有效避免了原有算法的漏检现象。由于 WM 算法的效率取决于最短模式串长度，因此在病毒特征码检测中，过滤掉特征库中的短码串，让 NWM_QS 算法仅处理其适合的特征集范围，大大降低了算法中 m 值为 1 或 2 时造成的冗余匹配，充分发挥了算法的实际效能。

4 实验结果

为检验改进后的算法性能，将改进后的算法和原算法进行了实验对比。实验环境为 Debian Linux 操作系统，系统配置为 2 GHz Intel CPU，996 MB 内存。通过提取 ClamAV 开源软件中的病毒特征集进行测试和分析，文本随机产生，大小为 32 MB。两组实验分别从匹配速率和执行时间对不同大小的字符集作了测试，得出如下数据。

图 5 是 BM、AC 及 AC_BM 改进算法的执行速度和模式集大小之间的关系。可以明显地看出，当模式集数量较小时，BM 算法执行速率较慢，而 AC 算法和改进算法执行速率相差不大。随着模式集的增大，改进算法虽然也因模式集大小的增加而下降，但是执行速率明显优于 AC 算法。而 AC 算法由于构建自动机树，且没有跳跃的匹配方式，严重制约了算法的效率，随模式集增大速率下降很快。因此，NAC_BM 算法从一定方面缓解了原有算法的内存膨胀问题。初步实验得出，改进算法较原算法平均速率提升 18% ~37%。

图 6 是 QS、WM_QS 和 WM_QS 改进算法执行时间和模式集大小之间的关系。由实验数据可以得出，QS 算法的执行时间斜率最大，且当模式集到一定数量时，算法会停滞不前。NWM_QS 算法执行时间随模式集的增大而增大，但优于原有算法。其原因就是在不匹配时，增大了算法的窗口长度，减少了不必要的匹配，增加了算法效率。

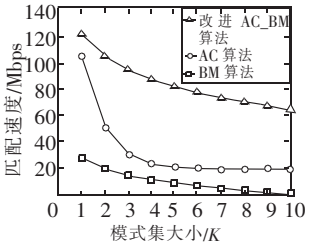

图 5 匹配速度与模式及大小之间的关系

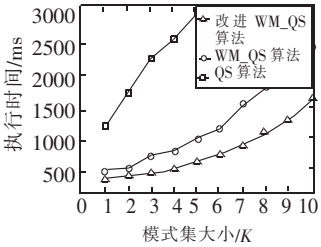

图 6 执行时间与模式及大小之间的关系

5 结束语

病毒特征检测中 70% 的时间用于进行模式匹配，因此提高匹配算法的效率和速度具有重要意义。本文通过对多模式精确匹配算法的深入研究，分别对 AC_BM 和 WM_QS 算法提出改进思路，并将改进算法运用于实际系统中。实验证明，改进的两种算法能够有效提高病毒特征码检测的效率。

参考文献：

[1] 王维，张鹏涛，谭营，等. 一种基于人工免疫和代码相关性的计算机病毒特征提取方法[J]. 计算机学报，2011，34(2)：204-214.

[2] 刘萍，刘燕兵，郭莉，等. 串匹配算法中模式串与文本之间关系的研究[J]. 软件学报，2010，21(7)：1503-1514.

[3] 王海峰. 基于智能特征码的反病毒引擎设计[J]. 计算机工程，2010，36(3)：145-147.

[4] 韩忠秋，刘晓洁，李涛. 一种入侵检测系统的模式匹配算法[J]. 计算机应用研究，2009，26(8)：3033-3035.

[5] 王培凤，李莉. 基于 Aho-Corasick 算法的多模式匹配算法研究[J]. 计算机应用研究，2011，28(4)：1251-1254.

[6] 胡金柱，熊春秀，舒江波. 一种改进的字符串模式匹配算法[J]. 模式识别与人工智能，2010，23(1)：103-106.

[7] 孙晓妍，武东英，祝跃飞，等. Wu_Manber 多模式匹配算法的研究与改进[J]. 计算机工程，2008，34(8)：85-87.