ISSN 1000-1239/CN 11-1777/TP
48(8): 1381-1389, 2011

对模式串匹配算法 WuManber 的复杂度攻击

张 宇[1,2] 刘 萍[1] 刘燕兵[1,2] 谭建龙[1] 郭 莉[1]
[1](中国科学院计算技术研究所 北京 100190)
[2](中国科学院研究生院 北京 100049)
(zhangyu2010@software.ict.ac.cn)

Algorithmic Complexity Attacks Against WuManber

Zhang Yu[1,2], Liu Ping[1], Liu Yanbing[1,2], Tan Jianlong[1], and Guo Li[1]
[1](*Institute of Computing Technology, Chinese Academy of Sciences, Beijing* 100190)
[2](*Graduate University of Chinese Academy of Sciences, Beijing* 100049)

Abstract Pattern matching is one of the fundamental problems in computer science. It is the key problem of many important scopes such as network information security, information retrieval and filtration, computational biology, etc. A great number of security problems have arisen with the wide application of pattern matching, especially in network information security systems, such as intrusion detection and prevention systems, anti-virus systems, anti-spam systems, firewall, etc. A method of algorithmic complexity attacks against WuManber which is a classical multi-patterrn algorithm, and the optimal solution of creating the attacking data are presented in this article. Compared with random data and the data from network, the attacking data can greatly reduce the searching speed of WuManber. Experiments on random data sets show that the larger character set, the better attacking performance. And experiments on the data from the network show that the attacking data can reduce WuManber searching speed by more than 50%. It is found that if a small part of the pattern set is known, the attack data can be created. Defensively speaking, it is important that the pattern set must be kept secret. This article also provides some strategies of improving the security of network information security systems. The attacking data can also be used in checking the system security.

Key words algorithmic complexity attacks; WuManber algorithm; pattern matching; security of algorithm; intrusion detection

摘 要 模式匹配问题是计算机科学的基础问题之一,是网络信息安全、信息检索与过滤、计算生物学等众多领域的核心问题.模式匹配技术在网络信息安全领域的广泛应用,导致了许多安全问题.WuManber 算法是一种经典的多模式匹配算法,通过对 WuManber 算法实现原理的分析,给出了一种对 WuManber 算法进行复杂度攻击的方法,并对攻击数据的构造问题给出了问题描述和最优求解.实验表明,WuManber 算法检测攻击数据的速度明显慢于检测随机数据和网络真实数据的速度,并发现只需已知少量的模式串,就可以构造有效的攻击数据.根据攻击数据的构造方法,在给出攻击方法的同时,也给出了防守方面的建议,可以有效地提高使用 WuManber 算法系统的安全性.

关键词 算法复杂度攻击;WuManber 算法;模式匹配;算法安全性;入侵检测

中图法分类号 TP393.08

收稿日期:2011-04-07;**修回日期**:2011-06-10

基金项目:国家自然科学基金项目(61070026);国家"九七三"重点基础研究计划基金项目(2007CB311100);国家"242"信息安全计划基金项目(2010A018)

多模式串匹配问题是计算机科学领域的一个经典问题,对它的研究已经持续了几十年,已有的算法也有几百个.多模式串匹配算法广泛地应用在信息检索、信息过滤、病毒检测、入侵检测、计算生物学等方面.近年来,随着网络信息安全应用的发展,多模式串匹配技术作为其核心技术之一,也得到了很大的发展,尤其是在大规模、超大规模的串匹配、模糊匹配、正则匹配、硬件匹配技术等方面,都随着应用的强烈需求而迅速发展.

伴随着众多领域上的各种应用,特别是众多的网络信息安全领域的应用,各种安全问题接踵而来.对应用系统级别而言,针对性的攻击和对抗无时不在,在这个层面上的研究从理论到实战都有很多.而在作为系统核心的算法级别层面,跟随着系统也受到了攻击和对抗,但是对于算法安全性的相关研究工作一直没有受到重视,在实战中更是处于处处被动的局面.

本文力图在模式串匹配方面,以经典的多模式串匹配算法 WuManber 为例,研究对其进行算法复杂度的攻击,着重在攻击数据的构造方面,给出问题的定义和描述,同时给出求解方法.当然,作为矛和盾的 2 个方面,在研究怎样可以构造攻击的同时,我们也给出了防守的建议.

1 算法安全性研究现状

多模式串匹配技术通常用在各种以规则过滤为核心的各种过滤系统中.规则的表现形式有多种,使用的规则匹配技术也有多种,多模式串匹配技术是其一.为了使系统达到较高的性能,在系统的不同应用环境中,会使用一种或者多种不同原理的匹配算法,以便满足系统的各方面要求,尤其是匹配速度的要求.但是,近年来,很多的研究表明:算法(例如模式串匹配算法、包分类算法、Hash 算法等)在安全性方面也面临着挑战,这种挑战来主要自于算法自身和应用环境两方面.

从算法本身来说,安全性的问题存在于算法复杂度中.文献[1]是最早从这个方面进行讨论的.它利用系统中的数据结构特征,以及算法的最差时间复杂度和平均时间复杂度的差别,构造数据对算法进行攻击,从而实现了一类具有低带宽(low-bandwidth)特点的对系统实施 DoS 攻击的方法.文章中以 Hash 算法为例,进行了详细的说明,并分析了 Squid Web cache,DJB DNS server 和 Perl 语言解析 3 个系统中的 Hash 表的脆弱性.最后,作者在 Bro 系统上进行了攻击实验,表明了此方法的有效性:在 6 min 的时间内可以耗尽系统的全部 CPU 资源,同时使得系统的丢包率达到 71%.

在认识了算法复杂度和系统脆弱性(可攻击)之间的关系后,很多文献开始针对不同环境下的此类问题提出了不同的解决方法.文献[2]是从 NIDS 系统中使用的规则匹配方法来入手分析的,以 Snort 系统为例,它的规则特征匹配算法存在回溯式的缺陷,使得系统性能在某些数据报文下呈指数级下降.文章提出了使用增加记忆点的新算法避免了大量回溯,并针对网络应用特点给出了 3 个优化策略,有效缓解了问题.文献[3]研究了包分类算法的问题,根据算法在匹配规则上的不均衡(skewed)现象,提出了新的算法,文中对新算法在攻击和防御方面的表现进行了分析,最后通过实验给出了其在系统总体性能、预处理时间复杂度、空间复杂度、并行处理等方面的分析和结果.文献[4]设计了新算法 LDM,分析证明:LDM 算法的最差、最好、平均时间复杂度分别达到了理论最好结果:$O(n)$,$O(n/m)$,$O(n(\log \sigma m)/m)$.若从算法复杂度攻击角度而言,使用此算法应该是最好的选择.

在系统级上,怎样快速有效地发现系统规则上的漏洞(系统的脆弱性分析)从而进一步构造攻击数据也是众多研究人员关心的问题.文献[5]中发现:在 Snort 系统中,所有的攻击实例都可以从规则集中的若干规则转变而来.文献[6]中的 SFET 工具是针对 Snort 规则集的 flowbit 缺陷而设计的,它不仅可以发现给定规则集的弱点,而且可以生成新的规则以阻止那些基于旧规则生成的攻击实例.文献[7]中指出:对于 NIDS 系统,如果规则集已知,则比较容易构造实例对系统进行攻击,使得它的服务质量下降,或者报警过多,文中给出了逆向工程的方法和工具,可以分析其规则匹配的过程,并且构造攻击数据以规避系统检测,或者构造非恶意数据以使系统过报警.其有意义的结论是:即使刻意对系统规则进行保密,系统的服务安全性也不能完全保证.文献[8]针对深度包检测(DPI)系统中的正则表达式匹配算法,提出了一种评估攻击数据恶意程度的方法,并在此基础上结合 cache 参数、算法数据结构的存储实现方式等,提出了可针对 NFA 和 DFA 的匹配算法方便构造攻击流量的方法,它也可以用来对 DPI 系统进行性能评测.

2　WuManber 算法

WuManber 算法[9]是一种快速的多模式串匹配算法,它是 Boyer-Moore 算法的一种派生形式. 它采用了 Boyer-Moore 算法的框架,使用长度为 b 的字符块 B(block character)而不是单个字符来计算坏字符(bad-character)的距离表 SHIFT. 此外,在进行匹配时,它使用散列表 HASH 选择模式串集合中的一个子集与当前文本进行匹配验证,减少无谓的运算. WuManber 算法的执行时间主要依赖模式串集合中最短的模式串长度,它不会随着模式串集合大小的增加而成比例增长,其时间要远少于使用每一个模式串和 Boyer-Moore 算法对文本进行匹配的时间总和.

WuManber 算法在预处理阶段,主要构造 SHIFT 表和 HASH 表,一个示例如图 1 所示. 在匹配阶段,求出当前匹配窗口内文本片断的末尾块字符的散列值 $h(B)$,如果该散列值对应的转移距离 $SHIFT[h(B)]$大于 0,则当前窗口内文本片断不匹配,将匹配窗口向右移动 $SHIFT[h(B)]$位;否则当前窗口内的文本可能和某个模式串匹配,利用散列值 $h(B)$选取模式串的一个子集 $HASH[h(B)]$,逐个与文本进行验证,以得到最终的匹配成功结果.

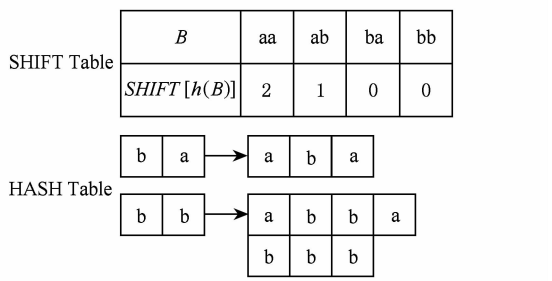


Fig. 1　WuManber *SHIFT* table ande WuManber *HASH* table for pattern set {aba, abba, bbb}. Character size is 2. $b=2$.

图 1　根据模式串集合{aba,abba,bbb}在字符块长度为 2,字符集大小为 2 时构造的 SHIFT 表和 HASH 表

在串匹配算法的设计和分析中,都是以"模式串和文本中的字符都是独立出现并且等概率分布的"为前提. 在这种前提下,WuManber 算法的匹配时间复杂度分析如下:

最好时间复杂度 $O(b\times n/l_{\min})$;

平均时间复杂度 $O(b\times n/l_{\min})$①;

最差时间复杂度 $O(b\times n)$.

3　攻击数据的构造问题

由 WuManber 算法的原理可以看出,算法的匹配时间主要消耗在 2 个方面:一个是计算当前窗口的散列值,另外一个是当存在一个可能的匹配时($SHIFT$ 值为 0 时),对相应 HASH 表项后链接的多个模式串进行一一验证. 显然,后面的一个耗时要远远大于前面一个. 因此,在攻击数据的构造中,主要考虑使用后面的一个因素,将验证代价作为衡量构造数据问题的一个优化目标.

3.1　符号定义及问题描述

在后面的描述中,使用符号如下:字符集为 Σ,$|\Sigma|=\sigma$,模式串集合为 $P=\{p_1,p_2,\cdots,p_r\}$,模式串的个数为 $|P|=r$,模式串的最短长度为 $l_{\min}$,文本 T 的长度为 n,字符块 B 的长度为 b. 长度为 $b-1$ 的字符串集合记为 N. 当前搜索窗口的最后 b 个字符为 B',$HASH[h(B')]$对应了一个至少有一个模式串的候选模式串链表,它的长度记为 w.

攻击数据的构造问题:对于给定的模式串集合 P,构造长度为 n 的文本 $T=t_1t_2\cdots t_n$,$t_i\in\Sigma$,使得 WuManber 算法在扫描文本 T 的过程中,$\sum w$ 的值最大.

攻击数据的构造问题实际是寻找一个字符序列,使得在其上扫描的过程中,尽可能多地在 HASH 链表中进行验证过程. 验证过程的代价过高,足以使得算法匹配速度下降到一个不可接受的程度,从而形成对算法的攻击.

3.2　问题分析及转化

对于任意 $k\in[b,n]$,所有长度为 k 的文本组成的集合记为 U_k,$\bigcup U_k$,$k\in[b,n]$记为 U. 按照文本长度、

① WuManber 算法的平均跳跃距离(ASD)决定它的平均时间复杂度 $O(b\times n/ASD)$,$ASD=(l_{\min}-b+1)\left(1-\frac{l_{\min}-b+1)r}{2h}\right)$,当 $b\ll l_{\min}$,$h\gg r\times l_{\min}$时,$ASD=l_{\min}$,其中 r 为模式串个数,h 为 $SHIFT$ 表的大小. 所以 WuManber 算法的平均时间复杂度是 $O(b\times n/l_{\min})$. 详细分析见文献[10].

文本结尾的字符块和跳跃距离3个条件,U 可分为若干个子集,任一个子集记为 $Z_{k,\pi,s}$,$k\in[b,n]$,$\pi\in\mathbb{N}$,$s\in[1,l_{\min}-b+1]$,它表示长度为 k 以 π 结尾并且当前需要跳跃 s 个字符的文本集合. 设构造字符串的方法 $Link1(\pi,\theta)$,$\pi\in\mathbb{N}$,$\theta\in\Sigma$,当 $b>2$ 时,表示 π 的后 $b-2$ 个字符加上 θ 组成字符串,例如:$b=3$,$\pi=ab$,$\theta=c$ 则 $Link1(\pi,\theta)=bc$,当 $b=2$ 时,$Link1(\pi,\theta)=\theta$,$\pi\in\mathbb{N}$,$\theta\in\Sigma$. 设构造字符串的方法 $Link2(\pi,\theta)$,$\pi\in\mathbb{N}$,$\theta\in\Sigma$,表示以 π 开头 θ 结尾长度为 b 的字符串,例如:$b=3$,$\pi=ab$,$\theta=c$ 则 $Link2(\pi,\theta)=abc$.

从上述定义可以看出,集合 U 满足以下3条性质:

性质 1. 对于任意 $k\in[b,n]$,$\bigcup Z_{k,\pi,s}=U_k$,$\pi\in\mathbb{N}$,$s\in[1,l_{\min}-b+1]$.

性质 2. $X\cap Y=\varnothing$,$X,Y\in U$,$X\neq Y$.

性质 3. 对于任意 $k\in[b,n-1]$,$Z_{k,\pi,s}$,$\pi\in\mathbb{N}$,$s\in[1,l_{\min}-b+1]$中的每个文本最后增加一个字符 $\theta\in\Sigma$ 后会变为 $Z_{k+1,\pi',s'}$,$\pi'\in\mathbb{N}$,$s'\in[1,l_{\min}-b+1]$ 的子集,s'可通过式(1)计算得出.

攻击数据的构造,本质上就是在文本的任意位置如何决定和决定放哪个字符,最终组成的文本字符串在匹配时可以达到设定的最大代价目标. 该问题可以转化为有向无回路图求最长路径的问题. 此有向无回路图 G 的顶点 v 和边 e 的定义如下.

顶点 v:将上述 $Z_{k,\pi,s}$,$k\in[l_{\min},n]$,$\pi\in\mathbb{N}$,$s\in[1,l_{\min}-b+1]$转化为 G 中的节点,定义为 $v_{k,\pi,s}$,与 $Z_{k,\pi,s}$一一对应;

边 e 和 e 上的权值 c:对于一个顶点 $v_{k,\pi,s}$,对任意一个字符 $\theta\in\Sigma$,通过式(2)计算出 π',s',存在一条从顶点 $v_{k,\pi,s}$ 指向 $v_{k+1,\pi',s'}$ 的边,记为 $e(v_{k,\pi,s},v_{k+1,\pi',s'})$,其上的权值记为 $c(e(v_{k,\pi,s},v_{k+1,\pi',s'}))$,可通过式(3)计算得出,将该路径标记为 θ.

$$s'=\begin{cases}SHIFT[h(Link2(\pi,\theta))],\\ \quad \text{if } s=1 \text{ 且 } SHIFT[h(Link2(\pi,\theta))]\neq 0;\\ 1,\ \text{if } s=1 \text{ 且 } SHIFT[h(Link2(\pi,\theta))]=0;\\ s-1,\ \text{else.}\end{cases}\tag{1}$$

$$\pi'=Link1(\pi,\theta).\tag{2}$$

$$c(e(v_{k,\pi,s},v_{k+1,\pi',s'}))=\begin{cases}HASH[h(Link2(\pi,\theta))],\\ \quad \text{if } s=1 \text{ and } SHIFT[h(Link2(\pi,\theta))]=0;\\ 0,\ \text{else.}\end{cases}\tag{3}$$

为了算法表述的方便,在上述图的基础上增加一个起点 V_{source} 和一个终点 V_{sink}:V_{source} 指向全部 $v_{B,\pi,SHIFT[h(\pi)]}$,$\theta\in\Sigma$,$\pi\in\mathbb{N}$,边上的权值为 $HASH[h(Link2(\pi,\theta))]$,标记为 $Link2(\pi,\theta)$;全部 $v_{n,\pi,s}$,$\pi\in\mathbb{N}$,节点 $s\in[1,l_{\min}-b+1]$都指向 V_{sink},边上的权值全部为0,不需要标记.

攻击数据的构造问题可以转化为:在如上述方法构造的有向无回路图 G 中,寻找一条从起点 V_{source} 到终点 V_{sink} 的路径,其边上的权重最大.

3.3 攻击数据构造问题的求解算法——SPEAR

Pseudo-code of SPEAR:

Input{$n,b,\sigma,SHIFT[],HASH[]$};

Output{$T=t_1t_2\cdots t_n$}.

1) Add node $v_{k,\pi,s}$, $k\in[b,n]$, $\pi\in\mathbb{N}$, $s\in[1,l_{\min}-b+1]$ to G;

2) Add edge $e(v_{k,\pi,s},v_{k+1,\pi',s'})$, $k\in[b,n-1]$, $\pi\in\mathbb{N}$, $s\in[1,l_{\min}-b+1]$ to G according to the formula (1) and (2);

3) Calculate $c(e(v_{k,\pi,s},v_{k+1,\pi',s'}))$ according to the formula (3);

4) Add starting node and ending node to G with there edges;

5) Find the longest path $L=l_1l_2\cdots l_{n-l_{\min}+b}$ according to Dijkstra Algorithm;

6) $t_{l_{\min}-b+1}t_{l_{\min}-b+2}\cdots t_n=l_1l_2\cdots l_{n-l_{\min}+b}$ and $t_i=\forall\alpha,\alpha\in\Sigma,i=1,2,\cdots,l_{\min}-b$.

当攻击数据的构造问题转化为在一个有向无环图中寻找最长路径时,就可以使用经典的 Dijkstra 算法来求解,此处不阐述. 需要注意的是:求解的权值最大路径是构造攻击数据的一个“痕迹”,真正的攻击数据文本需要按照路径上附加标识的符号信息具体地逐一生成.

例 1. 对于模式串集合{aba,abba,bbb},在 $b=2$,$\sigma=2$,$n=6$ 时,图1是 WuManber 算法预处理阶段产生的 SHIFT 表、HASH 表,图2为根据 SPEAR 算法构造图 G.

由于 WuManber 算法是从第 $l_{\min}-b+1=2$ 个字符开始计算第1个字符块的,在这之前的字符可在字符集中任意选取,这里假设取 a. 在图2中,根据权值最大的路径(图2中粗线)上面的符号信息,可以依次得到字符串 bbbbb. 前面加上字符 a,最终构造出的长度为6的攻击数据为:abbbbb.

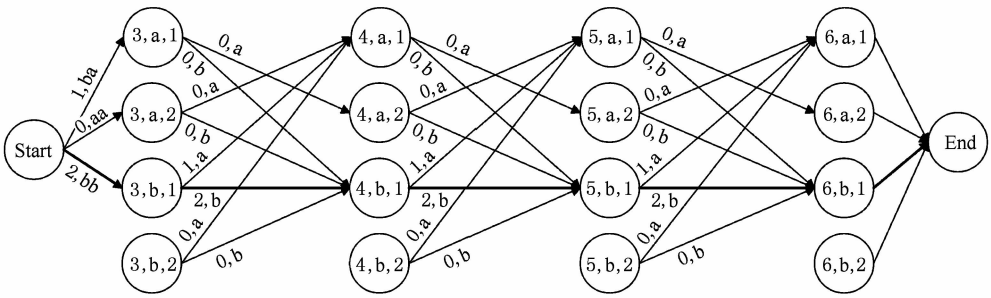


Fig. 2　The graph G according to SPEAR.

图 2　根据 SPEAR 算法构造的图 G

4　实验与分析

实验的目的有 3 个:一是测试 SPEAR 算法本身的性能;二是测试 SPEAR 算法产生的攻击数据的攻击效果,分为随机数据和真实数据 2 个方面;三是分析影响攻击效果的因素. 实验环境为 Red Hat Enterprise Linux Server Release 5.2 操作系统,系统配置为 Quad-Core AMD Opteron™ Processor 2376处理器,16 GB 内存. 实验结果中的所有时间数据都是通过 Linux 下的"time"命令测量得到的.

4.1　SPEAR 算法的性能

在本组实验中,我们取 $n=1\,024$,$b=2$,$r=1\,000$,改变 σ 和 $l_{\min}$ 的值构造随机模式串,测试 SPEAR 算法的性能. 实验结果如表 1. 可以看出:构造时间与 σ 和 $l_{\min}$ 都有正相关性,其中 σ 增大导致构造时间增加的幅度大于 $l_{\min}$ 增大导致构造时间增加的幅度.

Table 1　Comparison of Performance with Different Character Sets and the Length of the Shortest Pattern

表 1　字符集大小和最短模式串长度不同时的性能比较

$\sigma,l_{\min}$	Time/s
32,10	0.088
32,100	0.169
32,1 000	0.242
64,10	0.469
128,10	1.664
256,10	6.286

4.2　攻击效果

为了评价 SPEAR 算法生成的攻击数据对 WuManber 算法的攻击效果,在相同的环境下,我们分别让 WuManber 算法检测攻击数据构成的文本(Worst Text)和随机生成的文本(Random Text),通过检测速度的变化(Speed Reduced),来分析攻击效果. 通过式(4)来计算 Speed Reduced.

$$\text{Speed Reduced}=1-\frac{\text{WorstTextSpeed}}{\text{RandomTextSpeed}}. \quad (4)$$

随机数据下的攻击效果:随机数据包括随机的模式串集合数据和随机文本数据,它们都是在"模式串和文本中的字符都是独立出现并且等概率分布的"的前提下构造的,在它们上面的实验结果可以认为是算法的平均性能. 因此,本组实验中,攻击数据采用 SPEAR 算法在随机模式串集合上构造;正常数据使用随机生成的文本数据.

影响 WuManber 算法匹配速度的主要因素是 σ,r 和 $l_{\min}$. 我们分别进行 3 组测试,在 $n=1\,024$,$b=2$ 的前提下分别改变 σ,r 和 $l_{\min}$,观察算法受到攻击时的表现. 实验结果如图 3~5.

从图 3 中可以看出,随着 σ 增大,Speed Reduced 增大,且幅度很大. 当 $\sigma\geqslant 64$ 时,Speed Reduced 大于 80%,攻击效果明显;当 σ 较小时,攻击效果不明显,因为 WuManber 算法本身比较适合大字符集的情况,小字符集时性能本来就很差,所以构造的攻击数据也不会导致性能的大幅度下降. 从图 4 中可以看出,随着 r 增大,Speed Reduced 有减小的趋势,但幅度很小. 从图 5 中可以看出,随着 m 增大,Speed Reduced 有增大的趋势,但幅度也很小.

真实数据下的攻击效果:在上面的实验中,测试数据中用到的模式串和文本都是随机产生的,和真实数据还有一定差别,因为这些数据在字符集上是均匀分布的,因此最好的测试是用真实数据.

在本次实验中,我们采用了 2 组真实的模式串集合数据,一组是从 Snort 规则中提取的,一组是从 ClamAntiVirus 病毒库中提取的:

Snort 是一个开放源码的 NIDS,它的最新版本可以在网站 http://www.snort.org/dl/下获得. 我们使用了 Snort 2.8.3 版本,从它的规则中选取长度大于 4 B 的关键词共 5 000 个作为模式串集合.

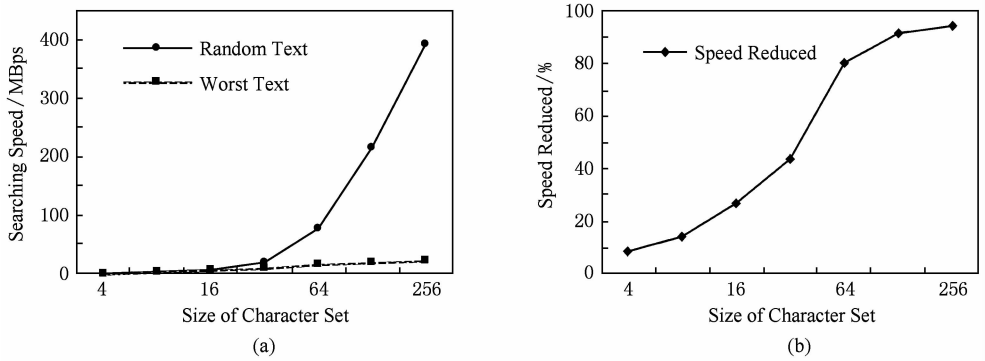


Fig. 3 Comparison of attacking performance with different size of character set (Pattern set size is 1 000, the length of the shortest pattern is 10).

图 3 字符集大小不同时的攻击效果(模式串规模为 1 000,最短模式长度为 10)

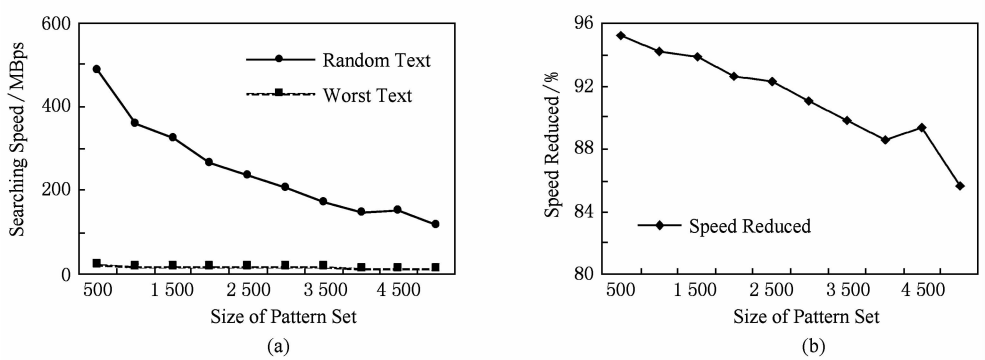


Fig. 4 Comparison of attacking performance with different size of pattern set (Character set size is 256, the length of the shortest pattern is 10).

图 4 模式串规模不同时的攻击效果(字符集大小为 256,最短模式长度为 10)

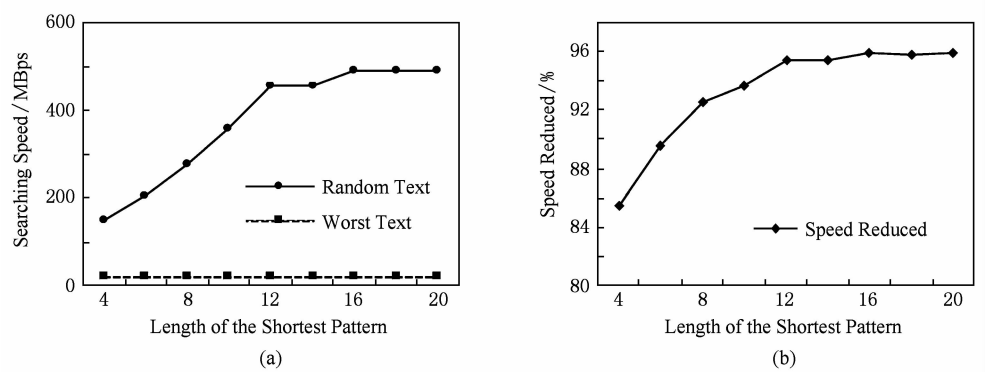


Fig. 5 Comparison of attacking performance with different length of the shortest pattern (Character set size is 256, pattern set size is 1 000).

图 5 最短模式串长度不同时的攻击效果(字符集大小为 256,模式串规模为 1 000)

ClamAntiVirus 是一个开放源码、免费授权的防病毒软件,它的病毒库是随时更新的,最新版本可以在 http://www.clamav.net/binary.html#pagestart 下获得. 我们使用了 ClamWinFree AntiVirus 0.90 版本,从它的病毒库中提取了长度大于 10 B 的关键词共 20 000 个作为模式串. 因为其中许多的病毒签名中包含各种通配符,不适于在精确串匹配中使用,因此我们提取了这种签名中最长片断.

测试使用的正常数据来自于 MIT 公开的一组真实的网络数据,它的原始目的是用来对网络入侵检测系统进行评估. 数据集可以在 http://www.ll.mit.edu/IST/ideval/下获得. 我们使用了 mit_1999_

training_week1_friday_inside.dat 的全部数据在 64 MB 左右.对于不同模式串集合,使用本文介绍的构造攻击数据的方法构造长度为 1 KB 的攻击数据,重复使用这一数据构造长度为 64 MB 的文本(Worst Text).用 WuManber 算法分别匹配这 2 组文本,比较速度变化,其中 $b=2,\sigma=256$.实验结果如表 2 所示.从实验中可以看出在真实环境下,攻击数据使 WuManber 算法下降的速度达到 50%以上.

Table 2　Searching Speed Comparison between the Worst Text and Real Network Data

表 2　使用真实数据的攻击效果

Pattern Set	MIT Speed/MBps	Worst Text Speed/MBps	Speed Reduced/%
Snort	74.702	23.414	68.66
ClamAntiVirus	50.605	22.241	56.05

4.3　影响攻击效果的因素分析

根据 WuManber 算法的原理,当扫描过程中产生的匹配较少时,需要时间较短,反之则较长.SPEAR 算法构造的 Worst Text 是使 WuManber 算法产生最多的匹配,所以影响攻击效果的主要因素就是检测 Random Text 匹配次数.综合 3 组实验可以得出,影响攻击效果的主要因素是 σ,σ 越大时攻击效果越明显,另外当 r 较小、$l_{\min}$ 较大时,攻击更加明显.结合上述的几组实验数据,得出如下 4 个结论:

结论 1. 在已知的 3 个因素中,攻击效果受字符集大小的影响最大.

结论 2. 随着字符集的增大,攻击效果越来越好,即攻击造成的 WuManber 算法匹配速度下降得越来越明显;

结论 3. 随着模式串规模的增大,攻击效果越来越差,即攻击造成的 WuManber 算法匹配速度下降得越来越不明显.

结论 4. 当模式串规模较小、模式串长度较大时,攻击效果会更加明显.

5　算法复杂度攻击的对策

5.1　防止模式串信息的泄漏

SPEAR 算法是在已知全部模式串集合的基础上构造攻击数据,使得算法性能快速下降.为了防止这种恶意攻击,有效的办法是严密保护模式串集合的信息,防止其泄漏.

更进一步的实验让我们认识到:模式串集合中,不同模式串的危害程度是不一样的.也就是说,存在一个模式串子集,其上构造的攻击数据与在全部模式串集合上构造的攻击数据完全相同,我们称该子集为最差模式串子集(worst pattern subset, WPS),显然这个子集与匹配算法和模式串集合有关.对于 WuManber 算法,寻找这个子集并不是一件困难的事情:使用 SPEAR 算法产生一个攻击文本数据,在使用此文本数据进行扫描匹配过程中,当 SHIFT 值为 0 时标记所有需要验证的模式串,即可以获得该模式串集合的最差模式串子集.

表 3 记录了在模式串规模不同的情况下,最差模式串子集大小的变化情况.可以看出最差模式串子集远远小于整个模式串集合.WPS 的存在可以使得应用 WuManber 算法的守方注意 2 点:

1) 要注意保管模式串集合特别是 WPS 的信息不被泄漏.如果攻击者盗取了一个模式串子集,并且这个子集包含了 WPS,那么攻击者就能构造出使 WuManber 算法匹配模式串最多的攻击数据.

2) 当发现有一部分模式串已经泄露出去,则需要对系统进行评估,计算可能产生的危害.可以使用 SPEAR 算法根据泄露出去的模式串集合来构造攻击数据,如果攻击效果已经超出了可容忍的范围,则需要采取相应的对策.

Table 3　The Size of Worst Pattern Subset with Different Size of Pattern Set

表 3　模式串规模不同时的最差模式串子集大小

Size of Pattern Set	Size of Worst Pattern Subet
1 000	82
2 000	125
3 000	17
4 000	23
5 000	20

5.2　使用抗攻击算法

SPEAR 算法是根据 WuManber 算法的特点,构造使扫描过程中匹配最多的攻击数据,从而导致匹配速度的下降.反之,可以通过改进 WuManber 算法的若干实现来减少此类攻击到来时的危害,例如:采用较好的散列函数,减小冲突,可以有效减小扫描过程中的验证次数.

作为防御者,可以根据自己 NIDS 的匹配算法和模式串集合,找到最差模式串子集,它跟模式串集合和匹配算法都有关系.在特定需求下,我们无法

改变模式串集合,但是可以通过改变算法,找到这样一种匹配算法,使得在模式串集合不变的情况下,增大最差模式串子集的大小.这样可以有效地提高NIDS的安全性,在泄露等量的模式串的情况下,减小可能对系统带来的危害.

6 总 结

本文对经典多模式串匹配算法 WuManber 进行了复杂度攻击研究,给出了攻击数据构造问题,并找到了最优的求解算法.实验结果表明:通过此方法构造的攻击数据主要跟 WuManber 算法的字符集大小有关,字符集越大,攻击效果越显著,通常可以使 WuManber 算法的速度下降 80%以上.此外,需要注意的是,攻击者不需要获取全部的模式串信息,而只需要获知数量很少的最差模式串子集(WPS)信息,即可构造出具有同等攻击效果的攻击数据.

本文的研究只是模式串匹配算法安全性研究的一个初步试探,串匹配算法原理众多,实际使用中也有很多不同的实现方式,应用环境也不尽相同,这些都和算法的应用是否安全相关.未来的工作可能会在如下的方面展开:

1) 抗攻击算法的设计.算法的安全性和它的匹配效率是相关的,通常是反相关.在设计新算法时,要在安全性和速度之间进行有效的取舍.而在实际工程应用中,这也许是一个动态的过程.

2) 算法安全性评估.作为应用算法的工程人员,是否可以通过一个有效的评估,得知算法的安全性指标,来帮助应用系统决策核心匹配算法的安全使用.

3) 网络环境下的算法安全性.模式串匹配算法应用在网络应用中,具有了很多网络特点,例如:在任意网络包负载位置开始匹配,频繁的小网络包等.这些是否对其安全性有影响,以及怎样评估和验证也是需要解决的.

参 考 文 献

[1] Crosby S A, Wallach D S. Denial of service via algorithmic complexity attacks [C] //Proc of the 12th USENIX Security Symp. Berkeley: USENIX, 2003: 29-44

[2] Smith R, Estan C, Jha S. Backtracking algorithmic complexity attacks against a NIDS [C] //Proc of the 22nd Annual Computer Security Applications Conf. Washington, DC: IEEE Computer Society, 2006: 89-98

[3] El-Atawy A, Samak T, Al-Shaer E, et al. On using online traffic statistical matching for optimizing packet filtering performance [C] //Proc of In IEEE INFOCOM'07. Los Alamitos: IEEE Communication Society, 2007: 866-874

[4] He Longtao, Fang Binxing, Yu Xiangzhan. A time optimal exact string matching algorithm [J]. Journal of Software, 2005, 16(5): 676-683 (in Chinese)
(贺龙涛, 方滨兴, 余翔湛. 一种时间复杂度最优的精确串匹配算法[J]. 软件学报, 2005, 16(5): 676-683)

[5] Rubin S, Jha S, Miller B P. Automatic generation and analysis of NIDS attacks [C] //Proc of the 20th Annual Computer Security Applications Conf. Los Alamitos: IEEE Computer Society, 2004: 28-38

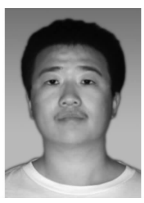
[6] Tran T, Aib I, Al-Shaer E, et al. Evasion attack on stateful signature-based network intrusion detection [R/OL]. Waterloo, Canada: University of Waterloo, 2008. [2011-02-10]. http://www.cs.uwaterloo.ca/research/tr/2008/CS-2008-18.pdf

[7] Kruegel C, Mutz D, et al. Reverse engineering of network signatures [C] //Proc of the AusCERT Asia Pacific Information Technology Security Conf. Gold Coast, Australia: AusCERT, 2005

[8] Becchi M, Franklin M, Crowley P. A workload for evaluating deep packet inspection architectures [C] //Proc of the 2008 IEEE Int Symp on Workload Characterization (IISWC). Piscataway, NJ: IEEE, 2008: 79-89

[9] Wu S, Manber U. A fast algorithm for multi-pattern searching [R]. Arizona: The Computer Science Department of the University of Arizona, 1994

[10] Liu Yanbing. Research of string matching algorithm optimization [D]. Beijing: Institute of Computing Technology, Chinese Academy of Science, 2006 (in Chinese)
(刘燕兵. 串匹配算法优化技术研究[D]. 北京: 中国科学院计算技术研究所, 2006)



Zhang Yu, born in 1987. Master candidate since 2009 in the Institute of Computing Technology, Chinese Academy of Sciences. His current interests include string matching algorithms and network information security(zhangyu2010@software.ict.ac.cn).

Liu Ping, born in 1972. Assistant professor at the Institute of Computing Technology, Chinese Academy of Sciences. Member of China Computer Federation. Her main research interests include information security, data stream processing and string matching algorithms(liuping@ict.ac.cn).

Liu Yanbing, born in 1981. PhD candidate and assistant professor at Institute of Computing Technology, Chinese Academy of Sciences. His main research interests include string processing algorithms and information security(liuyanbing@software.ict.ac.cn).

Tan Jianlong, born in 1974. PhD. Associate professor and master supervisor at the Institue of Computing Technology, Chinese Academy of Sciences. His main research interests include algorithm design, data stream management and information security (tan@ict.ac.cn).

Guo Li, born in 1969. Senior engineer and master supervisor at the Institue of Computing Technology, Chinese Academy of Sciences. Member of China Computer Federation. Her main research interests include information security and data stream processing (guoli@ict.ac.cn).

2012 年《计算机研究与发展》专题征文通知

——“可信编译的理论与关键技术”

编译器是重要的系统软件之一,编译器的可信性对于整个计算机系统而言具有非常关键的意义.可信编译是指编译器在保证编译正确的同时提供相应的机制保证编译对象的可信性,对可信编译理论和技术的研究具有重要理论意义和实用前景.近年来,国内外相关学者开展了大量研究,也取得了很多重要研究成果.为此,《计算机研究与发展》将于 2012 年出版一个“可信编译的理论与关键技术”专题,报道可信编译领域国内外科技工作者所取得的研究成果,提炼可信编译有待解决的关键问题,同时展望可信编译未来的研究方向,以推动我国可信编译的科学研究和工程应用.

本期“可信编译的理论与关键技术”专题将面向国内外征集论文,欢迎广大学者、专家、工程技术人员积极投稿,现将专题论文征集的有关事项通知如下.

征文范围 包括(但不限于)以下内容:

① 编译器的可信性分析与证明
② 可信编译器的设计与实现
③ 编译过程中的代码分析与验证
④ 基于定理证明的程序验证理论与方法
⑤ 可信编程语言的设计与构造
⑥ 基于编译的程序动态可信性保障技术
⑦ 基于编译器的辅助软件测试方法与技术
⑧ 可信编译器在嵌入式领域中的实践与应用
⑨ 可信编译的理论与关键技术综述

投稿要求:

① 来稿应属于作者的科研成果,数据真实可靠,具有重要的学术价值与推广应用价值,未在国内外公开发行的刊物或会议上发表或宣读过.
② 论文一般不超过 8 页,一律用 word 格式排版,论文格式体例参考近期出版的《计算机研究与发展》的要求(http://crad.ict.ac.cn/).
③ 论文通过专辑投稿信箱(compiler2012@126.com)发送电子稿,投稿时提供作者的联系方式(Excel 文档,按顺序包含:作者、论文标题、联系人、email、电话(手机)、通信地址、邮编).

重要时间:

截稿日期:2011 年 10 月 30 日
结果通知日期:2012 年 2 月 28 日

特约编辑:

何炎祥　武汉大学计算机学院,软件工程国家重点实验室　yxhe@whu.edu.cn
吴　伟　软件工程国家重点实验室,武汉大学计算机学院　whuwuwei@126.com

专辑投稿信箱:compiler2012@126.com
通信地址:湖北省,武汉市,武汉大学计算机学院　邮编:430072
联 系 人:吴　伟
电　　话:13437166157