Ad hoc网络中双向快速字符串匹配算法

张 莹[1] 徐 剑[1,2] 常桂然[1] 贾 杰[1]
(东北大学信息科学与工程学院 沈阳 110004)[1] (东北大学软件学院 沈阳 110004)[2]

摘 要 网络入侵检测系统的原始AC算法采用单向匹配,由于样本数量增加使得比对时间延长,因此提出了一种高效的多模式匹配算法——双向快速字符串匹配算法,该算法采用有限自动机、正反双向匹配的方式,与单向匹配算法相比,提高入侵检测速度3倍左右。对该算法进行了性能分析,并将其与已有算法进行性能比较。仿真实验结果表明,提出的Re-AC算法比其他算法有更好的优势,能够提高Ad hoc网络入侵检测的效率。

关键词 Ad hoc网络,入侵检测,模式匹配,Re-AC算法
中图法分类号 TP309 **文献标识码** A

Two-way Fast String Matching Algorithm in Ad hoc Networks

ZHANG Ying[1] XU Jian[1,2] CHANG Gui-ran[1] JIA Jie[1]
(School of Information Science and Engineering, Northeastern University, Shenyang 110004, China)[1]
(Software College, Northeastern University, Shenyang 110004, China)[2]

Abstract In a network intrusion detection system, the original AC algorithm adopts one-way matching. The comparing time will increase as the number of samples increases. This paper presented an efficient multi-pattern matching algorithm, a bi-directional fast string matching algorithm. The algorithm uses finite automata and forward-backward two-way matching. Compared with the original one-way matching algorithm, the intrusion detection rate is increased by 3 times. The performance of the algorithm was analyzed and compared with other algorithms. Simulation results show that this algorithm can improve the efficiency and detection rate.

Keywords Ad hoc network, Intrusion detection, Pattern matching, Re-AC algorithm

Ad hoc网络[1]是一种使用无线通信技术的无组织对等网络。相对于有线网络,因其无中心、动态拓扑、自组织、多跳路由等特点,Ad hoc网络更容易受到各种攻击。由于这种网络的特点和广泛应用,使得Ad hoc网络的安全问题尤为突出。Ad hoc网络安全保障主要从路由安全、密钥管理及入侵检测系统等3个方面来解决[2-4]。由于Ad hoc网络的特殊性,虽然传统入侵检测技术的研究已经取得一定的成果,但其无法直接应用于移动Ad hoc网络中。

入侵检测,顾名思义就是发觉入侵的行为。它通过在计算机网络或计算机系统中的若干关键点收集信息并对其分析,从中发现网络或系统中是否有违反安全策略的行为和被攻击的迹象。入侵检测系统根据不同的分类标准有着不同的分类。根据数据分析方法(即检测方法)的不同,我们可以将入侵检测系统分为两类,即误用检测和异常检测。误用检测(Misuse Detection),又称为基于特征的检测,它是根据已知的攻击行为建立一个特征库,然后去匹配已发生的动作。如果一致,则表明它是一个入侵行为,它的优点是误报率低。误用检测主要采用专家系统、模型推理、状态迁移分析和模式匹配等技术。在基于模式匹配算法的入侵检测系统中,模式匹配的效率决定这类入侵检测系统的性能。

目前,Ad hoc网络中对于入侵检测的研究主要集中在体系结构和检测算法两方面。为提高Ad hoc网络入侵检测系统的效率,本文设计了一个专门针对Ad hoc网络入侵检测系统的字符匹配算法,并在snort wireless中实现。与单向匹配算法相比,实验表明双向匹配算法能提高效率3倍左右。

1 Aho-Corasick算法

1975年,Aho和Corasick提出了基于有限状态自动机的AC算法[5],即采用Aho-Corasick有限自动机,将字符串构成一个集合,可以一次查找多个模式。该算法最早被使用在图书馆的书目查询程序中,取得了很好的效果。后来AC算法被引入入侵检测领域,成为一种经典的多模式字符串匹配算法。后来的一些算法都是在它的基础上改进的。

有限状态机是系统所有可能状态的表示,以及该系统可接受的状态转换的信息。有限状态机的处理动作从最初状态开始,然后接受一个输入事件,根据输入事件从当前状态移动

到稿日期:2009-11-26 返修日期:2010-02-05 本文受国家自然科学基金项目(60903159),863国家高技术研究发展项目(2009AA01Z122)资助。

张 莹(1981-),女,博士生,主要研究方向为信息安全,E-mail:engleyy@163.com;徐 剑(1978-),男,博士生,助教,主要研究方向为网络与系统安全;常桂然(1946-),男,博士,教授,博士生导师,主要研究方向为计算机网络、网络与信息安全、无线自组织网络;贾 杰(1980-),女,副教授,主要研究方向为认知网络。

到下一个适当的状态。使用有限状态自动机，将所有样本构成一个有限状态的匹配器。当字符串进入时，可以匹配所有的样本。以字符串元数等于 n 为例，经过 n 次匹配即可产生对所有样本的匹配结果，不会因为样本数量而使比对时间延长，所以 AC 算法的匹配速度最快，适合用于多样本匹配。

Aho-Corasick 算法的基本思想是：有限自动机算法的构建需要 3 个函数：转向函数 *goto*，失败函数 *failure* 和产生函数 *output*，通过这 3 个函数可以构建一个树型有限自动机。这样模式字符串匹配的处理过程就变成了状态转换的处理过程，由“Start”状态开始。在搜索查找阶段，通过这 3 个函数的交叉使用扫描文本，定位出模式在文本中的所有出现位置。构建转向、失败和产生函数包含两个部分：第一部分确定状态和转向函数，第二部分计算失败函数。产生函数的计算则是穿插在第一部分和第二部分中完成。

2 Re-AC 算法

2.1 Re-AC 算法构建的基本思想

本文提出一种新的字符串匹配算法 Re-AC 算法。在原始的 AC 算法中，匹配方式是由左至右地将待匹配的字符送入有限自动机中，进行单向匹配。本研究以增加反向的有限自动机并采用双向的匹配方式，来加快匹配的效率。

2.2 Re-AC 算法构建的基本步骤

在 Re-AC 算法实现的过程中，构建正反自动机的过程是算法的关键所在。Re-AC 算法主要是通过构建转向函数、失败函数、产生函数来完成正向与反向自动机的建立，然后确定匹配起始点以及匹配区间来匹配关键字。

2.2.1 *转向函数以及初步产生函数*

(1)算法思想

首先将待匹配的文本分别以正向和反向输入到模式集 P 中。转移函数 $g(s,a)\rightarrow s$ 为一个映射，其建立过程为：逐个取出模式集 P 中字符串的每个字符，从状态 0 出发，由当前状态和新取出的字符决定下一状态。如果有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加入的状态，将新加入的状态赋为当前状态。处理完模式集 P 中的所有字符串后，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。

初步的产生函数 *output* 第一步是在构造转移函数 g 时，每处理完一个字符串，则将该字符串加入到当前状态 s 的产生函数中，第二次最终的构造要等到失效函数构建完成后。下面以关键字{he,she,his,hers}为例描述转向函数的构建思想。

① 正向转向函数以及初步产生函数

转向函数是将关键字依序输入，每个关键字以一条指示的路径代表，考察关键字元的顺序与重复性，增加新的状态与路径，完成一个固定的指示树。在状态 0 上除了字元 h 或者 s 之外，增加一条回环回到状态 0，并且产生初步的产生函数。

首先输入{he}，可以得到转向函数，如图 1 所示，并且产生状态 2 的初步产生函数，如表 1 所列。

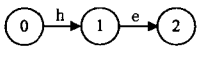

图 1　正向输入{he}的转向函数

表 1　正向状态 2 的初步产生函数

State	Output(state)
2	{he}

再输入{she}，可以得到转向函数，如图 2 所示，并且产生状态 5 的初步产生函数，如表 2 所列。

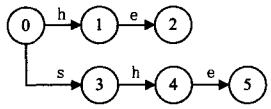

图 2　正向输入{she}的转向函数

表 2　正向状态 5 的初步产生函数

State	Output(state)
2	{he}
5	{she}

再输入{his}，可以得到转向函数，如图 3 所示，并且产生状态 7 的初步产生函数，如表 3 所列。

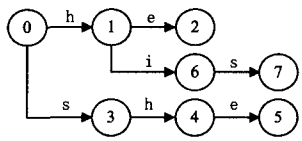

图 3　正向输入{his}的转向函数

表 3　正向状态 7 的初步产生函数

State	Output(state)
2	{he}
5	{she}
7	{his}

最后输入{hers}，可以得到完整的转向函数，如图 4 所示，再增加一个回圈状态 0，除了字元 e、s 之外，其他都会回到状态 0，并且产生状态 9 的初步产生函数，如表 4 所列。

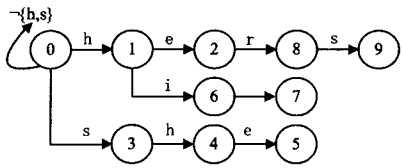

图 4　正向输入{hers}的转向函数

表 4　正向状态 9 的初步产生函数

State	Output(state)
2	{he}
5	{she}
7	{his}
9	{hers}

②反向转向函数以及初步产生函数

首先输入{he}，将其逆转为{eh}，可以得到转向函数，如图 5 所示，并且产生状态 2 的初步产生函数，如表 5 所列。

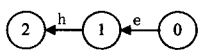

图 5　反向输入{he}的转向函数

表 5　反向状态 2 的初步产生函数

State	Output(state)
2	{he}

再输入{she}，将其逆转为{ehs}，可以得到转向函数，如

图 6 所示，并且产生状态 3 的产生函数，如表 6 所列。

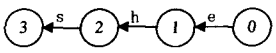

图 6 反向输入{she}的转向函数

表 6　反向状态 3 的产生函数

State	Output(state)
2	{he}
3	{she}

再输入{his}，将其逆转为{sih}，可以得到转向函数，如图 7 所示，并且产生状态 6 的初步产生函数，如表 7 所列。

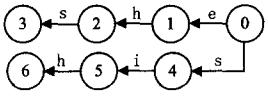

图 7　反向输入{his}的转向函数

表 7　反向状态 6 的产生函数

State	Output(state)
2	{he}
3	{she}
6	{his}

最后输入{hers}，将其逆转为{sreh}，可以得到完整的转向函数，如图 8 所示，再增加一个回圈状态 0，除了字元 e、s 之外，其他都会回到状态 0，并且产生状态 9 的初步产生函数，如表 8 所列。

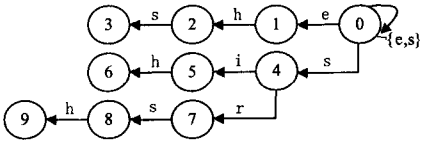

图 8　反向输入{hers}的转向函数

表 8　反向状态 9 的产生函数

State	Output(state)
2	{he}
3	{she}
6	{his}
9	{hers}

在具体实现中，用二位数组表示自动机的转换关系。首先以每个关键字为一组加入二位数组中。加入一个字母后，状态随之加 1，直至将整个关键字完全加入自动机中，再将此关键字作为初步的产生函数，加入输出链表。待失败函数构建后，完整输出。

2.2.2　失败函数

(1)算法思想

失败函数 f 用来指明当某个模式与文本匹配不成功时应处理的下一状态。失败函数的构造方法为：所有第一层状态的失败函数为 0，如 $f(1)=f(4)=0$；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 $f(s)=g(f(s^*),a)$，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。如在反向失败函数中，$f(8)=g(f(7),h)=g(0,h)=1$，$f(9)=g(f(8),e)=g(1,e)=2$。

① 正向失败函数的建立

失败函数首先需要计算出各个状态的深度，再设定深度 1 的失败函数为 0，而其他各状态 S 的失败函数是由其所属深度减 1 的每个状态 r 与每个输入字元 a 作状态转换，当 $g(r,a)=S$ 时，则做：

①$state=f(r)$；

②$g(state,a)=f(S)$。如果 $g(state,a)=$ fail，再执行 $state\leftarrow f(S)$，直到 $state$ 的数值得到 $g(state,a)\neq$ fail。

首先计算出各个状态的深度，如表 9 所列，各状态的失败函数如表 10 所列。

表 9　正向各状态的深度

State	1	2	3	4	5	6	7	8	9
Depth	1	2	1	2	3	2	3	3	4

表 10　正向各状态的失败函数

State	1	2	3	4	5	6	7	8	9
f(state)	0	0	0	1	2	0	3	0	3

② 反向失败函数的建立

首先计算出各状态的深度，如表 11 所列，而各状态的失败函数如表 12 所列。

表 11　反向各状态的深度

State	1	2	3	4	5	6	7	8	9
Depth	1	2	3	1	2	3	2	3	4

表 12　反向各状态的失败函数

State	1	2	3	4	5	6	7	8	9
f(state)	0	0	4	0	0	0	0	1	2

2.2.3　完整的产生函数

(1)算法思想

产生函数 *output* 的作用是在匹配过程中输出已经匹配的字符串。*output* 函数的构造分两步，第一步是在构造转向函数 g 时，每处理完一个字符串，则将该字符串加入到当前状态 s 的产生函数中，如 $output(2)=$ {he}，$output(9)=$ {hers}。第二步是构造失败函数后，若 $f(s)=s'$，则 $output(s)=output(s)\cup output(s')$，如 $output(9)=output(9)\cup output(2)=$ {*hers*,*he*}，因为 $f(9)=2=s'$。

① 正向完整的产生函数

在失败函数的计算期间，同时更新产生函数。当确定 f(S)=S'时，合并状态 S 以及状态 S'的输出结果，以构成完整的产生函数，完整的产生函数如表 13 所列。

表 13　正向完整产生函数

State	*Output(state)*
2	{*he*}
5	{*she*}
7	{*his*}
9	{*hers*,*he*}

② 反向完整的产生函数

经过构建完整的转向函数、失败函数，可以得到完整的产生函数，如表 14 所列。

表 14　反向完整产生函数

State	*Output(state)*
2	{*he*}
3	{*she*}
6	{*his*}
9	{*hers*,*he*}

2.3 Re-AC 算法的实现

(1) 匹配起始点的确定

先以字符中心点为基础，并以转向函数的最大深度即关键字中最长的字符作为双向匹配的重复区域。将此区域平分设定在中心点上，考察最大关键字正好覆盖此区域的情况。即如果中间重复区域是一个关键字，则能够检查出来。防止从中心点向两侧查找时，会漏掉正好处于中间的关键字。

这样，正向自动机向右查找进行匹配，构造的反向自动机向左查找进行匹配。一旦一方找到与关键字相同的文本内容，立即按照处理办法进行报警或者输出。

(2) 匹配过程

表 15　正、反双向的 Next move Function

positive State	positive input symbol	positive Next state	reverse State	reverse Input symbol	reverse Next state
	h	1		e	1
0	S	3	1	s	4
	Others	0		others	0
	e	2		h	2
	i	6	1	e	1
1	h	1		s	4
	s	3		others	0
	others	0		s	3
	r	8	2	e	1
2	h	1		others	0
	s	3		i	5
	others	0		r	7
	h	4	3	e	1
3	s	3		s	4
	others	0		others	0
	e	5		i	5
	i	6		r	7
4	h	1	4	e	1
	s	3		s	4
	others	0		others	0
	r	8		h	6
5	h	1	5	e	1
	s	3		s	4
	others	0		others	0
6	s	7		e	1
	h	1	6	s	4
	others	0		others	0
7	s	4	7	e	8
	h	3	7	s	4
	others	0		others	0
8	h	9		h	9
	s	1	8	e	1
	others	0		s	4
9	s	4		others	0
	h	3		s	3
	others	0	9	e	1
				others	0

在数据包进入匹配判断程序后，先将字符串全部转换为大写字母，之后以字符串长度及关键字长度决定搜索的起始位置。开始匹配后，将字符串放入自动机做状态转换。发现该状态匹配成功时，查看是否有大小写的设定。若有，则会根据 index 值找到原始字符串相同的位置，再次做匹配。完成匹配后，再经过重复判断输出。若无大小写设定，则可直接经过重复判断后输出。

从状态 0 出发，每取出文本串中的一个字符，利用 g 和 f 函数进入下一状态。当某个状态的 *output* 函数不为空时，输出其值，表示在文本串中找到该字符串。完成正、反两个方向的有限自动机后，依照其转向函数、失败函数则可以得到各自的 Next move Function，用以减少状态转换，增加匹配速度，如表 15 所列。

3　算法性能分析与比较

(1)实例比较

以发现第一例攻击特征为基准，用一个实例对原始 Aho-Corasick 算法以及改进后的 Re-AC 算法进行分析比较。

①单向匹配算法

以原始 Aho-Corasick 算法作为单向算法加以说明。假设输入的字符串在 text. txt 文件中为“esrushersu”，关键字分别为 he，she，his，hers。首先构造关键字的正向有限自动机，构造结果如图 9 所示。

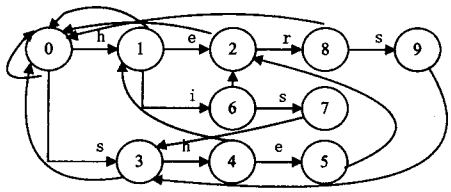

图 9　正向完整有限自动机

然后从左至右进行单向匹配，过程如下：

程序 1：输入字元 e，状态转移 0→0 。

程序 2：输入字元 s，状态转移 0→3。

程序 3：输入字元 r，状态转移 3→0。

程序 4：输入字元 u，状态转移 0→0。

程序 5：输入字元 s，状态转移 0→3。

程序 6：输入字元 h，状态转移 3→4。

程序 7：输入字元 3，状态转移 4→5，输出{she，he}。

②双向匹配程序

以 Re-AC 算法作为双向算法加以说明。假设输入的字符串在 text. txt 文件中为“esrushersu”，关键字分别为 he，she，his，hers。相比于单向匹配程序，双向匹配程序需要增加一个反向的自动机，如图 10 所示。

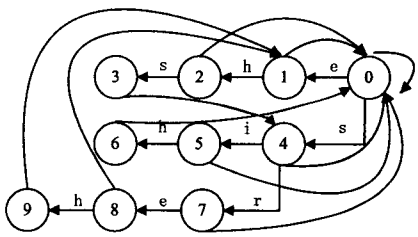

图 10　完整反向有限自动机

在匹配过程中，先以字符中心点为基础，并以关键字中最长的字符“hers”作为双向匹配的重复区域。将此区域平分设定在中心点上，考察最大关键字正好覆盖此区域的情况。此字符串中心区域不包含关键字，所以正向自动机从字母“u”向右进行匹配，反向自动机从字母“e”向左进行匹配。双向匹配的完整构架如图 11 所示。

双向匹配过程如下：

程序 1：正向输入字元 u，状态转移 0→0。反向输入字元

e，状态转移 0→1。

程序 2：正向输入字元 s，状态转移 0→3。反向输入字元 h，状态转移 1→2，输出{he}。

(2) 比较结果

在单向匹配程序中发现第一例攻击特征需要花费 7 个程序，而在双向匹配中只需花费 2 个程序即可发现第一例攻击特征。以此例来看，双向匹配程序比单向匹配程序快 3 倍左右。

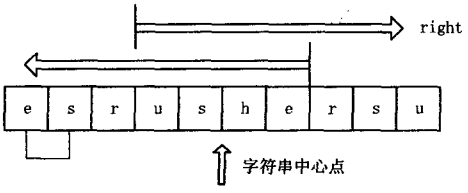

图 11 双向匹配的完整构架

4 实验模拟

针对本文提出的算法使用虚拟机模拟攻击环境进行测试。在一台装有 Snort 的虚拟机上进行数据包测试，并将此台机器当作被攻击方，另一台虚拟机当作攻击方进行 Ping 测试。读取攻击数据包或模拟攻击行为，且分别以不同的规则数、算法测试。Snort 主机接收数据包后分析数据包内容，并使用 Performance Monitor 统计出整体入侵检测性能。入侵检测系统的设备规格如表 16 所列。使用的规则组是由 Snort 官方网站上下载的 snortrules-snapshot-2.7.tar.gz。

表 16 入侵检测系统的设备规格

	Snort 主机	模拟攻击主机
OS	Fedora Core 7(kernel 2.6)	Red Hat Linux 9
Memory	1024MB	512MB
Hard	8G	10G
Ethernet	Bridged	Bridged

本实验选择经典的 3 种匹配算法作对比。在图表中，Default 代表 Snort 的默认算法 Boyer-Moore[6] 算法，Re-AC 代表本文提出的双向快速字符串匹配算法，AC 代表 Aho-Corasick 算法，MWM 代表 Modified Wu-Manber[7] 算法。

4.1 接收相同数据包的测试

使用读取数据包进行测试统计，结果显示 4 种算法均能准确测出 Alert 数值，以及数据包所用的协议，达到了有效性的要求。为了取得有效值，因为所测得的数值有浮动的情况，所以以测试 10 次取其平均值的方法得到实验数据。

第一次测试所用数据包的大小为 325.2MB，共有 1532453 个包，如图 12、图 13、图 14 所示。

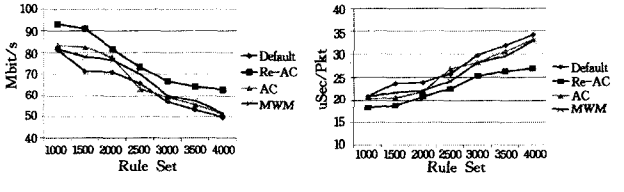

图 12 每秒检测的数据包大小 图 13 每个数据包花费的检测时间

第二次测试所用数据包的大小为 310.3MB，一共有 1521919 个包。如图 15、图 16、图 17 所示。

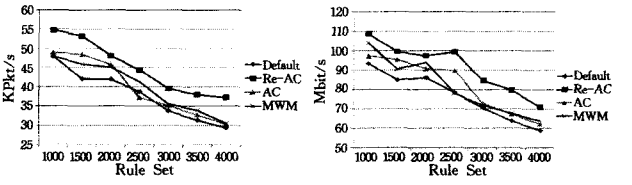

图 14 每秒检测的数据包数量

图 15 每秒检测数据包大小

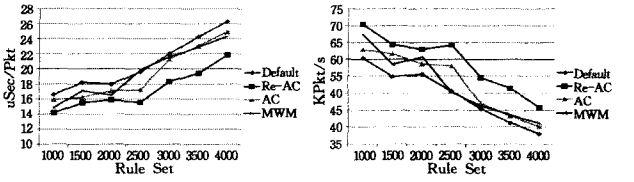

图 16 每个数据包花费的检测时间 图 17 每秒检测的数据包数量

通过对数据包不同的测试可以看出，Re-AC 算法在速度增加方面有明显优势，尤其对于单向 AC 效率提高不少。

4.2 PING 测试

PING 原本是用来检查网络是否通畅或者网络连接速度的命令，它的工作原理是利用发出 ICMP type 8 的 Echo request 给对方主机。当对方收到 request 这个数据包后，IP 层会发出一个终端信号给操作系统，请系统回送一个 type0 的 Echo Replay。当原来系统接收了这个回应之后，同样的 IP 层会中断操作系统，并由操作系统将此回应信息传达给每一个曾经在系统中发出 ICMP Echo Request 的行程。

利用此工作原理，在瞬间送出大尺寸或者大量的 PING 数据包时，可以制造出类似的 DoS 攻击效果。本测试在攻击端使用 PING 程序，以最快的速度发送 30 次大小为 60000byte 的数据包，其中数据包传输协议 ICMP 占 100%。Snort 以范围 1800 数据包作为接收，取得有效值。同样，因为所测的数值有浮动的情况，所以以 10 次取平均值的方式得到实验数据，如图 18、图 19、图 20 所示。

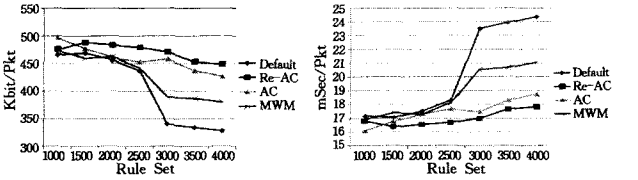

图 18 每秒检测数据包的大小 图 19 每个数据包花费的检测时间

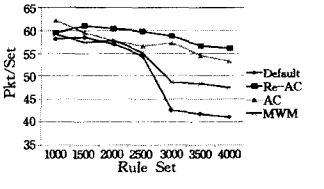

图 20 每秒检测的数据包数量

通过 PING 测试可以看出，随着规则数的增加，Re-AC 效率下降趋势减缓，并且高于其他的匹配算法。

结束语 本文设计了一个双向式平行处理的 Aho-Corasick 算法，即 Re-AC 算法，并以多种情况的数据包进行网络效率测试。由测试结果证实 Re-AC 算法在包含大量的网络数据包检测上，效率比其他算法有更好的优势，能够提高 Ad hoc 网络入侵检测的效率。

参 考 文 献

[1] Nee Rajk F. Security considerations in Ad hoc sensor networks

[J]. Ad hoc Networks,2005,1(3):69-89
[2] Cevics S, et al. On communication security in wireless Ad hoc sensor network[A]//Eleventh IEEE International Workshops on Enabling Technologies:infrastructure for collaborative enterprises(WETICE'O2) [C]. Pittsburgh,Pennsylvania,2002:10-12
[3] Ren xiuli. Security methods for wireless sensor networks [A]// Proceedings of the 2006 IEEE International Conference on Mechatronics and Automation[C]. 2006:1925-1930
[4] Ganesan P, et al. Analyzing and modeling encryption overhead for sensor network nodes [A]//Proceedings of the 2nd ACM International Conference on Wireless Sensor Networks and Applications [C]. San Diego,2003:151-159
[5] http://www-igm. univ-mlv. fr/~lecroq/string/node12. html
[6] Liu R T, Huang N F, Chen C H, et al. A Fast String-Matching Algorithm for Network Processor-based Intrusion Detection System[J]. ACM Transactions on Embedded Computing System,2004,3(3):614-633
[7] Roesch M. Snort-lightweight Intrusion Detection for Networks [A]//Proceeding of the 13th System Administration Conference [C]. USENIX,1999:4-5

(上接第 37 页)

对隐式多项式曲线进行分析,基本概况了目前所有的拟合算法,可以看出,Min-Max 算法很精确地拟合了数据点形状,而且非常稳定,只是需要进一步研究 3L 集合的权值参数调整问题,Min-Max 等隐式多项式曲线拟合算法抛弃了需要迭代的优化算法,仅仅求解一个线性方程组就可以确定隐式多项式曲线方程系数,应该说已经趋于成熟。可以预见,将这种建模思想应用到各种数据点集合中必将带来很大的发展空间。

参 考 文 献

[1] Bajcsy R, Solina F. Three dimensional object representations revisited[C]//Proc. Inter. Conf. Computer Vision. London,1987: 231-240
[2] Khotanzad A, Hong Y H. Rotation and Scale Invariant Features for Texture Classification[C]//Proc. of Robotics and Automation. Santa Barbara, USA,1987:16-17
[3] Unel M, Wolovich W A. On the construction of complete sets of geometric invariants for algebraic curves[J]. Advances in applied mathematics,2000,24:65-87
[4] Lei Z. Implicit polynomial shape modeling and recognition, and application to image and video database[D]. Brown University,1997
[5] Khan N. Silhouette-Based 2D-3D Pose Estimation using implicit algebraic surfaces[D]. Saarbrucken:Saarland University,2007
[6] Lebmeir P, Richter-Gebert J. Rotations translations and symmetry detection for complexified curves[J]. Computer Aided Geometric Design,2008,25:707-719
[7] 吴刚,李道伦. 基于隐含多项式曲线仿射不变量的目标识别[J]. 电子学报,2004,32(12):1987-1991
[8] 吴刚. 隐含多项式曲线曲面拟合次数的确定研究[J]. 计算机研究与发展,2007,44(1):148-153
[9] Sampson P D. Fitting conic section to very scattered data:an interactive improvement of the book stein algorithm[J]. Computer Vision, Graphics, and Image Processing,1982,18(1):97-108
[10] Keren D, Cooper D. Describing complicated objects by implicit polynomial[J]. IEEE Trans. On PAMI,1994,16(1):38-53
[11] Taubin G, Cukirman F, Sullivan S, et al. Parameterized families of polynomials for bounded algebraic curve and surface fitting [J]. IEEE Transactions on pattern analysis and machine intelligence,1994,16(3):286-303
[12] Sullivan S, Sandford L, Ponce J. Using geometric distance fits for 3D object modeling and recognition[J]. IEEE Trans. PAMI, 1994,16(12):1183-1196
[13] Keren D, Gotsman C. Fitting curves and surfaces with constrained implicit polynomials[J]. IEEE Trans. PAMI,1999,21(1):31-41
[14] Blane M M, Lei Z. The 3L algorithm for fitting implicit polynomial curves and surfaces to data[J]. IEEE Trans. PAMI,2000, 22(3):298-313
[15] Hezler A, Barzohar M, Malah D. Robust fitting of implicit polynomials with quantized coefficients[C]//Proceedings of the international conference on Patten Recognition. 2000,3:290-293
[16] Sahin T, Unel M. Globally stabilized 3L curve fitting. Image Analysis and Recognition[M]. Heidelberg:Springer Berlin,2007: 289-300
[17] Tasdizen T, Tarel J-P. Improving the stability of algebraic curves for application[J]. IEEE Trans. image processing,2000,9 (3):405-416
[18] Zhu Chun-gang, Wang Ren-hong. Least Squares Fitting of Piecewise Algebraic Curves[J]. Mathematical Problems in Engineering,2007,10:11
[19] Heizer A, Barzohar M, malah D. Stable fitting of 2D Curves and 3D surfaces by implicit polynomials[J]. IEEE Trans. PAMI, 2004,26(10):1283-1294
[20] Unsalan C. A new robust and fast implicit polynomial fitting technique[C]//Proceedings of MVIP 1999. 1999,9:15-20
[21] Blane M M, Lei Z. The 3L algorithm for fitting implicit polynomial curves and surfaces to data[J]. IEEE Trans. PAMI,2000, 22(3):298-313
[22] Yalcin H, Unel M, Wolovich W. Implicization of parametric curves by matrix annihilation[J]. International Journal of Computer Vision,2003,54:105-115
[23] Sener S, Unel M. Affine invariant fitting of algebraic curves using Fourier descriptors[J]. Pattern Analysis Application,2005, 8:72-83
[24] Unsalan C. A new robust and fast implicit polynomial fitting technique[C]//Proceedings of MVIP 1999. 1999,9:15-20
[25] Guennebaud, Gael, Gross, et al. Algebraic point set surfaces[J]. ACM Transactions on Graphics,2007,26(3):231-239
[26] Redding N J. Implicit polynomials, orthogonal distance regression, and the closest point on a curve[J]. IEEE Trans. PAMI, 2000,22(2):191-199
[27] Coutinho D F, de Souza C E, Trofino A. Regional stability analysis of implicit polynomial systems[C]//Proceedings of the 45th IEEE Conference on Decision & Control. San Diego,2006:13-15
[28] Zheng B, Takamatsu J, Ikeuchi K. Adaptively determining degrees of implicit polynomial curves and surfaces[C]//ACCV (2). 2007:189-300
[29] Diaz P, Sonia. Partial degree formulae for rational algebraic surfaces[C]//Proceedings of the 2005 International Symposium on Symbolic and Algebraic Computation. 2005:301-308
[30] Sahin T, Unel M. Stable Algebraic Surfaces for 3D Object Representation[J]. Journal Math Imaging Vision,2008,32:127-137