·安全技术· 文章编号：1000—3428(2005)04—0146—02 文献标识码：A 中图分类号：TP393.08

# 多模匹配问题在IDS中的解决

鲁鹏俊，钟亦平，张世永
(复旦大学计算机信息与工程系，上海 200433)

**摘 要**：传统的Snort系统引入单模式匹配速度较快的BM算法进行特定的模式匹配。文献[1]在Snort系统中引入了基于KMP的多模式匹配算法，能有效地提高系统的性能。但是该匹配算法本身存在着局限性，以致于无法完成含有多Content属性选项规则的匹配工作。该文从多content属性选项的规则匹配问题着手，提出了解决方案，从而进一步推进了多模式匹配算法在IDS中的引入。

**关键词**：入侵检测；多模式匹配；状态树

# Solution of Multi-pattern Matching Problem in IDS

LU Pengjun, ZHONG Yiping, ZHANG Shiyong
(Department of Computer Information and Engineering, Fudan University, Shanghai 200433)

**【Abstract】** Traditional Snort system introduces high speed single-pattern matching BM algorithm for specific pattern matching, while document [1] introduces a multi-pattern matching algorithm based on KMP, by which the systematic performance of Snort can be effectively improved. However the algorithm has its limitations too. It is unable to implement the matching work while a rule contains multiple properties. This article aims at the very point mentioned above and gives an effective solution, which largely accelerates the introduction of multi-pattern matching technology in IDS.

**【Key words】** Intrusion detection; Multi-pattern matching; State tree

在基于特征检测的入侵检测系统中，特征匹配的时间开销成为了整个系统的瓶颈。Snort系统采用了单模式匹配较快的BM算法，文献[1]引入了基于KMP的多模式匹配算法大大提高了匹配速度，促进了整个系统的性能。然而，多模算法的引入又给整个系统带来了新的问题——此算法无法完成多Content属性选项的规则匹配问题。从而，导致IDS无法对基于这种规则的入侵手段进行有效的检测，给系统带来了很大的潜在威胁。文献[1]为了弥补这个不足，不得不重新引入效率较低的BM算法，因此也使系统性能大打折扣。本文旨在解决含有多Content属性选项的规则匹配问题。在进行的模拟试验中，我们加入了含有多Content属性选项的规则，重新进行试验发现：比传统的BM算法快了很多，而且解决了文献[1]中的残余问题。

## 1 多模算法的重述及其带来的问题

文献[1]中提到：将Snort系统中规则库内每个Content属性的字符串认定为一个模式串，数据包中数据即为待匹配主串。为了检测主串中是否包含某一Content属性字符串，文献[1]中构造了状态树和状态跳转函数(具体构造方法参照文献[1])。为了便于问题的叙述，我们来看其具体的匹配过程：

假如有模式集G={"abcd", "bce"}，主串S="abced"。参照文献[1]可以建立如图1的状态树。其中S={q0,q1,…,q7}为状态集。实线为接收当前字符的跳转，虚线为不能接收字符的跳转(具体如何构造参照文献[1])。从q0开始，扫描主串碰到字符a，于是自动机转到q1状态，碰到b到q3状态，碰到c到q5状态；继续扫描主串时遇到字符e，由于q5不能接收字符e，根据虚线跳转到状态h(q5)=q4；q4接收字符e，到达q6状态。此时是终止状态，故找到匹配串"bce"，即说明在主串S中存在特定模式"bce"。

算法描述：

```
boolean MutiplyStringMatch(){
q=root;
for (i=0;i<n;i++){// n是主串的长度
  while (q!=fail && g(q ,S[i])==fail)//在i处不匹配
     q=h(q);
  if (q==fail) q=root;
  else
     q=g(q , S[i]);
  if (isElement(q , FinalSet)) //若是到了终止状态说明有串匹配
     return true; }
return false; //无匹配返回
}
```

该算法的时间复杂度是O(Ln+m1+…+mL)，其中L个模式的长度分别为m1，m2，…，mL，主串的长度为n。然而，该算法本身存在着它的不足之处：其只能检测主串是否含有模式集中的某一个模式串。但是，在Snort规则库中，某些规则可能会包含两个或者更多的Content属性选项，此时就需要知道主串是否包含模式集中指定的几个模式串，这就是本文需要解决的多Content属性选项规则匹配问题。

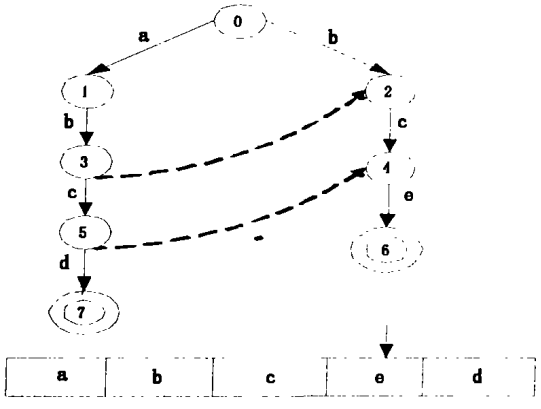


图1 多模匹配实例

**作者简介**：鲁鹏俊(1980－)，男，硕士生，研究方向：计算机网络安全；钟亦平，副教授；张世永，教授、博导

**定稿日期**：2004-01-18 **E-mail**：abbott_2000@ctang.com

## 2 多Content选项规则问题的解决

在解决这个问题之前看一下怎样利用状态树来一次让所有模式都和主串相匹配。换句话说，就是让主串中包含模式集中所有的模式串，这时才算得上匹配，否则就不匹配。由于需要解决的是一条规则中的多个Content选项。因此可以先将重复串和被另外的串所包含的字串删掉。例如：在一规则中若有{ba , abc , bab , babb , bbc}，那么显然{bab,ba}是不必要的。因为若包含了模式babb,那么肯定包含模式bab和模式ba。这一点应该在规则维护的时候就应该解决了的问题。根据文献[1]中的算法，可以建立如图2的一颗状态树。

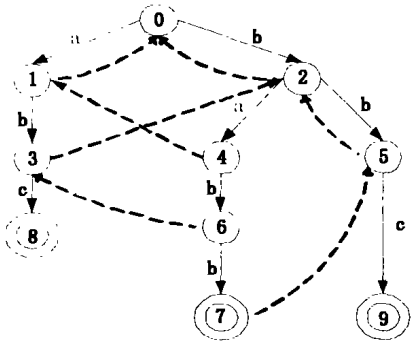


图2 P={abc,babb,bbc}的状态树

由第1节中的算法可知，一旦到达了某个终结状态，那么其对应的模式串肯定是在众多模式中首先被匹配的。我们可将这个模式从所有模式中去掉；在匹配过程结束时，若所有模式都被去掉，那么我们就说发生了匹配。现在，我们的策略就是：建立一个标志数组用以标志该模式是否已经被匹配；若到达某个终结状态，那么就将其对应模式的标志位置位。当整个主串被扫描完时，若有某个标志位没有被置位，那么就失配。为了知道我们的终结状态所对应的是哪一个模式串，需要在状态节点中增加一个标识字段，这点是在建树时就可以保证的。

为了阐述如何解决多Content属性选项问题，我们将举例来说明。不妨设有3条规则C1,C2,C3。其中C1包含两个Content属性选项p1,p2; C2包含两个Content属性选项p3,p4; C3包含两个Content属性选项p5,p6；按照文献[1]的方法创建状态树（给出其示意图3）。同时构造一个6位的2进制数A=$(000000)_2$，并令b1=$(000011)_2$，b2=$(001100)_2$,b3=$(110000)_2$；按照第1节中的算法对所有模式串进行匹配，只是当到达某终结状态Pi时，将A的第i位置1，并且回到q0状态，直至所有主串字符扫描完成为止。不妨假设主串中包含p2,p3,p4模式，那么此时A=$(001110)_2$。如图4所示。

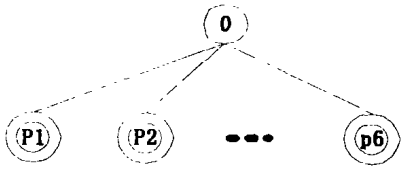


图3 6个属性的状态树简图

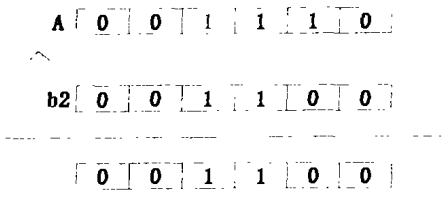


图4 A与b2的与运算

我们的目的就是要看主串包含的模式集是否能够包容某一规则中的全部规则。因此将A分别和bi作“与”运算，看结果是否与bi相等。若相等，说明主串与第i条规则相匹配；否则，就不和第i条规则相匹配。在上面的例子中由于A^b2=b2，因此主串中包含规则2。于是，便可按照规则2进行报警或者日志记录等行为处理。

值得一提的是，当把所有规则不加区分时A的位数将会达到千位以上。这是一个相当大的大数，一个相当好的解决方案是先将这些具有多Content属性的规则分离出来加以考虑。同时注意到：前而提到的一个标志位对应着一个Content属性(也即一个模式)。为了方便处理，需要将这些规则进行充分的分组而加以处理，这些在IDS初始化的过程中就可以完成。

举个很简单的例子：如果有几个规则总共的Content属性个数为36个，那么一个32位的整数肯定不够；此时，需用两个32位数A1和A2来进行处理。我们分组的策略是：让任何一个规则的多个Content属性对应的标志位都分布在一个整数中。在这个例子中，若有某一个规则包含Content属性P32和P33，那么需要调整。因为第32位属于A1，而第33位属于A2。可以将其调整，让这两个属性对应于A2的第1和第2位。这种调整看起来比较复杂，但是从现实的角度来考虑其工作量并不是很大。而且其只是在规则库的更新或者建立时才会调整的，在更新时也无须大幅度调整，其维护还是非常容易的。

经过上述方法的引入，基于规则匹配的入侵检测系统可以彻底地去掉BM算法，而引入多模式算法让系统高速有效地运行。根据前而的叙述和文献[1]的讨论，可以将Snort系统的检测器组织成如图5的结构。

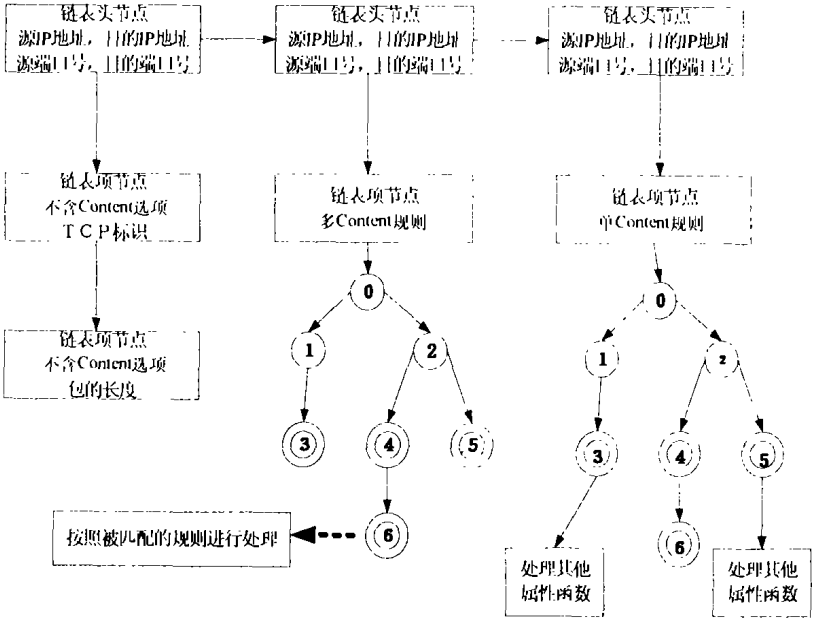


图5 经过变换后的Snort结构组织图

(下转第151页)

模型中包括mkdir建立的目录文件vul_file、对一般用户保密、属于root的文件target_file以及两个进程attack和mkdir。NuSMV原始的反例输出有13次状态转换，过滤后有3步，下面是详细描述：

安全约束：target_file的所有者不能为attack。

初始状态：vul_file不存在，target_file所有者为root，target_file对别的用户既不可写也不可执行。

攻击步骤：

步骤序号：1

进程：mkdir

操作：creat（建立目录文件vul_file）

改变的状态：vul_file存在，vul_file所有者为root。

步骤序号：2

进程：attack

操作：link（将vul_file链接到target_file）

改变的状态：vul_file所有者为attack，vul_file为符号链接文件，target_file被链接到vul_file。

步骤序号：3

进程：mkdir

操作：chowner（改变vul_file所有者为attack）

改变的状态：target_file的所有者为attack。

## 5 结论

本文通过使用NuSMV验证系统行为模型进行脆弱性分析。这种方法不依赖于和产生脆弱性原因有关的专家知识和经验总结，并具有识别已知和未知脆弱性的能力。文中提供的例子展示了这种在系统行为模型中并未包含关于脆弱性知识的情况下NuSMV给出了利用脆弱性进行攻击的步骤。

分析结果可用于很多方面，系统设计者或者管理员可根据分析出的脆弱性产生原因对系统进行重新设计或者配置从而消除这些脆弱性带来的危险，同时这些原因也可作为规则加入诸如COPS或者SATAN等配置扫描器增强其扫描能力，另外，MC产生的反例可以作为模式提供给IDS。

本文中提出的模型实例的规模比现实中的系统小得多，结构也简单得多，因此为使其有真正的实用价值，除了完善已有的文件系统和进程模型外，还要在系统中加入外部网络实体模型，并对某些通信协议具体实现中的安全相关部分进行建模，通过这样的扩展期望可以对绝大多数种类的系统脆弱性进行分析；此外，为使编码过程更系统、更灵活也更方便，还需要开发从对系统模型的高层描述到MC执行机的规范编码的自动转换工具。

### 参考文献


1 Farmer D, Spafford E. The COPS Security Checker System. CSD-TR-993, Department of Computer Science, Purdue University, 1991

2 Baldwin R.Rule Based Analysis of Security Checking. MIT LCS Technical Report No. 401, 1988

3 Clarke E M, Emerson E A, Sistla A P. Automatic Verification of Finite-state Concurrent System Using Temporal Logic Specification. ACM Trans. Programming Languages and Systems,1986,8(2):244-263

4 Ritchey R,Ammann P.Using Model Checking to Analyze Network Vulnerabilities.IEEE Oakland Symposium on Security and Privacy, 2000

5 Ramakrishnan C R,Sekar R.Model-based Analysis of Configuration Vulnerabilities. Journal of Computer Security Archive,2002,10(1-2): 189

6 Kumar S,Spafford E .A Pattern Matching Model for Misuse Intrusion Detection. Proceedings of the 17th National Computer Security Conference, 1994-10:11

7 Tanenbaum A S.Operation Systems Design and Implementation. Prentice-Hall Inc., 1987


(上接第147页)

## 3 模拟试验及其结果

为了检验上面描述的算法，选取SNORT-1.9.1中的规则库作为待测数据集(其中去掉了没有Content选项的规则)，选取一个TXT文本文(10kB)模拟待测数据包。为了便于计时，将每次测试循环100遍。得到的模拟结果如图6所示(此结果在p4机器，ReadHat Linux7.3得到，与文献[1]在同一环境中进行)。其中，横轴是规则条数，括号里面是含有多content属性的规则条数；纵轴是循环100遍的时间，单位是秒。

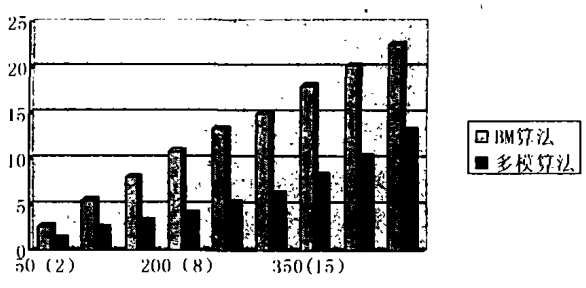


**图6 模拟试验及其结果**

由试验的结果可以得到如下结论：经过修改后的多模算法，无须再次引入BM算法来解决多Content属性选现规则问题。从图6可以清楚地看到其运行速度大概是BM算法的2倍。

## 4 小结

在网络全球化的今天，网络安全变得越来越重要。然而，在网络不断发展的同时，各种入侵手段也越来越高明；因此，一个好的入侵检测系统显得十分重要。本文通过改进多模匹配的不足，使多模匹配算法的适用性更好，从而很好地将其引入到Snort系统中，提高了整个系统的性能。

### 参考文献


1 Lu Pengjun,Zhou Bin. Intrusion Detection System.The 2003 International Conference on Security and Management ,LasVegas, Nevada,USA,2003-06-23

2 Aho A,Corasick M. Efficient String Matching: An Aid to Biliographic Search.Comm. ACM, 1975,18:333

3 Coit C J, Staniford S, Defense S ,Towards Faster String Matching for Intrusion Detection or Exceeding the Speed of Snort. http//www.Silicondefense.com

4 http://www.snort.org

5 严蔚敏，吴伟民.数据结构.北京：清华大学出版社，2002-09-01