An improved multiple patterns matching algorithm for intrusion detection

Zhengqiang
School of Computer Science & Technology, Shandong University of Technology
Zibo,China
e-mail: our.net@163.com

Abstract-**Pattern matching algorithm is one of the Core algorithms in the detection engine of the intrusion prevention system. Efficiency of the intrusion prevention system is determined by pattern matching algorithm. A survey of the pattern matching algorithm is described in this thesis. The Wu-Manber algorithm which is one of the multi-pattern matching algorithm is explained in detail and the improvement of the Wu-Manber algorithm is presented to improve the efficiency. By dividing the pattern group into two subgroups and dealing with the two subgroups in different methods, the QWM algorithm enhanced the efficiency of pattern matching. Experimental results show that when pattern group contains a pattern that is less than three bytes, the performance of the improved Wu-Manber algorithm is much better than the traditional Wu-Manber algorithm.**

Keywords-intrusion detection system; multi-pattern matching; BM algorithm; Wu-Manber algorithm; QWM Algorithm

1. INTRODUCTION

The rapid development of network technology has brought great convenience to people, yet also great security concern. Constant unsanctioned intrusion of the computer network has left important information stolen or even led to the paralysis of the network system. This has caused huge economic losses for many countries and companies or even endangered the security of the country and the region. The current security technology is mainly divided into prevention technology and detection technology. Traditional safety strategies such as firewalls and VPN mainly focus on protection, which is always lagging behind in the face of attack. Network Intrusion Detection System (NIDS), as one of the important technologies to improve network system security, has been widely used. Typical NIDS describes known aggression with a series of rules (characteristics). According to statistics, at present about 95% of intrusion detection is based on feature matching. Therefore, the performance of the pattern matching algorithm will have a direct impact on the efficiency of intrusion detection system.

This paper mainly focuses on commonly used analysis of the multi-pattern matching algorithm "Wu-Manber algorithm". The short mode which affects the performance of the algorithm is separated and an improved algorithm is proposed.

978-1-4244-6585-9/10/$26.00 ©2010 IEEE

2. WUMANBER ALGORITHM, MWM ALGORITHM

Pattern matching: It refers to the process of looking for the first or multiple occurrences of the pattern string P= $P_1P_2...Pm$ with a given length of m in a given pattern string $T = T_1T_2...T_n$ with a length of n. Here T_i ($1 \le I \le n$), P_j($1 \le J \le m$) $\in \sum$(character set). When P occurs for one or more times in T, it is said to be a successful match; otherwise, a failed match. Single pattern matching algorithm: The algorithm which can only do one pattern matching per time in the target matching string. Multiple pattern matching algorithm: The algorithm which can do multiple pattern string matching in the target string simultaneously.

There are many typical matching algorithm in intrusion detection system such as Boyer-Moore,Aho-Corasick[1,2],Modified Wu-Manber and so on.Boyer-Moore[3] algorithm solve the issue of single pattern matching, Aho-Corasick algorithm and Wu-Manber algorithm solve the issue of multiple pattern algorithm.

2.1Wu-Manber algorithm

Wu-Manber algorithm [4,5] is derived from BM algorithm in a multi - pattern matching，and it is a practical and fast multi - pattern matching algorithm. In the multiple-pattern matching, with the increase in the number of patterns, each character appears in the back-end of model with correspondingly increased probability, and gradually reduce the distance between the end of string, its mobile distance decreases, so the BM algorithm efficiency has been greatly reduced in a multi-pattern matching. Wu-Manber algorithm use block B with extended bad character moving rule to resolve this problem, also use a hash table to filter the matching stage which should match the mode matching algorithm, to reduce time and improve efficiency.

Firstly, Wu-Manber algorithm pretreats the pattern string collection. Preprocessing stage will establish three tables: SHIFT table, HASH table and the PREFIX table. SHIFT table stores character's moving distance when it appears in pattern string including all the characters in character set. When computing moving distance, it follows two principles:

1)If string block WB in the windows does not appears in any pattern strings, then SHIFT [h]=m-B+1,here，h is the hash value for block W_B.

2)If block W_B appears in some pattern strings, and the right position in all patterns is q, then SHIFT[h]=m-q.HASH

table stores the last blocks of pattern string which have same hash values. PREFIX table stores the first blocks' hash value of pattern string which have the same last block hash value. Matching stage is to use these three tables to complete the scan and search for text matching process

The main process of matching algorithm:

1) Compute the shortest length of pattern strings between all models, marked as m, and only consider the first m-character string of each models, in other words, m is the size of matching window.

2)According to the first m-character string of each models, compute the hash value (h) for the last block character T[m-B+1,… ,m];

3) Retrieve the value of the SHIFT [h], if SHIFT [h]>0, then the windows will move right with SHIFT [h] ,and back to step 2,else go to step 4;

4) Calculate the hash value of the corresponding "prefix" text window, marked as text_ prefix;

5) For each P value which meets HASH[h] ≤ P<HASH[h+1], test where there PREFIX[P]=text_prefix. If true, String text and pattern model exactly match.

Table Shift Calculated as follows:

1） with m-b +1 complete shift [] table

2） For each pattern string

```
{// calculate the effective distance for each text block
    for (i=m-1;i>=(B-1);i--)
    {
      hash = hashBlock ([Pattern [i],…,Pattern[i-B+1]);
    if(Shift[hash]>=m-1-i)
    Shift[hash]=m-1-i;
    }
}
```

Wu-Manber Algorithm the search process is as follows:

```
while (text < textend)
{
  hash=hashBlock(text); //Calculate the hash value of the current text block
   s=Shift [ hash]; //get the jump distance by search the bad character shift table of text block
   if(s==0)
   { //current character block appears in the end of the pattern stings
  text_prefix=prefixBlock(text); //calculate the prefix of text block
   p=Hash[hash]; //get all initial positions of all patterns which match current text
  while (p)
  { // vertify if the patter string of the subsets matches;
   if (text_prefix ! = Prefix[p]) continue;
  //report the results if they mathc exactly
  }
 s=1;
}
text+=s; //goto the next match point
}
```

The Time complexity of Wu-Manber algorithm is O(Bn / m) in the average case. Here, B is the length of the block, n is the length of string text, m is the shortest length of string model. The algorithm is sensitive to the shortest length of string model, The max value of SHIFT table is limited by the shortest length. If the minimum length is small, the moving distance is also small, which has a great influence on the matching efficiency.

2.2 MWM Algorithm

Document [6] achieves Modified Wu-Manber algorithm (MWM for short), a variant of Wu-Manber algorithm, in snort2.6. 0. MWM uses a standard 1- or 2- byte bad character jump table and a fixed 2-byte prefix Hash table. It is worth pointing out that the biggest advantage of the various algorithms based on Boyer-Moore jump is that character comparison can be achieved by jumping. However, its performance is highly dependent on the length of the minimum model for the reason that the maximum jump range of the model cannot exceed this value, or possible matches might be missed. The longer the length of the minimum model, the more effective the jumps of the bad characters will be, thus the faster the speed of detecting.

Analysis on snort set of rules found that many rules of most of the rule groups are multi-byte mode (here refers to mode equals to or greater than 3 bytes in length, similar hereafter), but it also contains 1-or 2- byte mode of rule. Although these 1or 2- byte modes are not significant in numbers, they limit the maximum jump range of the whole model set to be 1, that is, no jump at all, thereby reduces the performance of Wu-Manber algorithm greatly. Here in this article a new type of Wu-Manber multi-pattern matching algorithm called QWM (QuickWu-Manber) is proposed. QWM divides the modes into groups and matches 1 or 2-byte modes by bitmaps so that the length of the minimum mode is greater than 2 and improves the performance of the algorithm greatly.

3. QWM ALGORITHM DESIGN

The main method to improve Wu-Manber algorithm is to increase the distance of matched windows and to reduce the number of character comparisons. Improved QWM algorithm divides the modes into groups and matches 1 or 2-byte modes by bitmaps so that the length of the minimum mode is greater than 2 and improves the performance of the algorithm greatly.

When checking each packet, pattern matching algorithms will report that a packet may contain attack as long as the current character string matches with certain model in the model groups. It will no longer check the remaining characters in the package, and immediately begin to check the next packet. Compared with the long mode, there's a larger possibility of being matched for the short mode. Therefore, QWM algorithm starts to match packet with short mode group and then with long patterns group. Once a packet is found to contain character strings matching the mode it will start the inspection of the next packet immediately.

In terms of model groups that contain 1 or 2- byte model, first divide that group into two: short model group $P_{shorter}$= { P_1, P_2} and long model group P_{longer}= {P_m, P_{m+1},…, $P_{k|m}$ >2}. When the file of rules is read in, 1- byte mode, 2 - byte mode and multi- byte mode are added to the appropriate mode structures OneBytePat, TwoBytePat and MultiBytePat respectively. For those mode groups without 1 or 2 – byte, there is no further division and only MultiBytePat is constructed. Only those model groups containing 1 or 2- byte models are discussed in this essay.

3.1 The preprocessing phase

Firstly short mode group Pshorter is preprocessed. For every character char that appears in P_1, its corresponding location in bitmap EXISTONE with the size of 256 will be tagged 1. For every character char1char2 that appears in P_2, its corresponding location in bitmap EXISTTWO with the size of 256×256 will be tagged 1. In this way, during the searching stage whether the current character in the text matches a certain pattern in P_1 or P_2 can be quickly determined by whether the value of the corresponding location in the bitmap is 1.

EXISTONE and EXISTTWO can be constructed according to the following method:

$$\text{EXISTONE(char)} = \begin{cases} 1 & char \in p_1 \\ 0 & char \notin p_1 \end{cases}, \quad (1)$$

$$\text{EXISTTWO(char1, char2)} = \begin{cases} 1 & char1char2 \in p_2 \\ 0 & char1char2 \notin p_2 \end{cases} \quad (2)$$

3.2 Pattern and Text Matching Algorithm

3.2.1 The Algorithm Process of Pattern and Text Matching in the Short Pattern Group

1) Take the current character tc in the text as the index to check EXISTONE (tc).

2) Take the two characters tc and tc+1of the current text as index to check EXISTTWO(tc, tc+1).

3) If both the value of EXISTONE(tc) and EXISTTWO(tc, tc+1) are 0, add 1 to the text pointers and turn to step one.

4) If the current pattern is case sensitive, examine whether they match under the case sensitive circumstances. If not, add 1 to the text pointers and turn to step one.

5) Report the match and finish the search.

3.2.2 The Algorithm Process of Pattern and Text Matching in the Long Pattern Group

For the long Pattern group P_{longer}, first of all, sorting the order of the Pattern; and then construct a bad character skip table BCSHIFT and a prefix hash table Hash.

1) Take the current character tc as the index, search for BCSHIFT(tc).

2) If the value of BCSHIFT (tc) is greater than 0, the move the text pointer to the right for BCSHIFT (tc) characters and turn to the first step.

3) Calculate the Hash function values of the current characters tc and tc+1 in the text. Take Hash (tc) as the index to search the Hash table. If the value is 0, add one to the text pointers and start the first step.

4) Examine all of the Patterns that have the same prefix with the current Pattern. If there is no Pattern matches the current text, add 1 to the text pointers and move the step one.

5) If the current pattern is case sensitive, examine whether they match under the case sensitive circumstances. If not, add 1 to the text pointers and turn to step one.

6) Report the match and finish the search.

QWM algorithm treats the short mode first after separates it from the mode groups. The minimum mode length is more than 2, i.e. the maximum skip range is at least 3. Thus the time taken to inspect every packet is shortened significantly and so the performance of pattern matching algorithms is improved.

4. Performance Test and Analysis

In order to test the performance of the new algorithm, an experiment is carried out to compare QWM algorithm and MWM algorithm realized in snort. Experimental environment by Pentium4 PC computer hardware platform and CPU 2.0GHz, memory, 256 MB ; Operating Systems : Red H. The hardware platform of the experimental environment is PC computer with Pentium4 with CPU 2.0GHz and memory 256 MB; operating system: red hatlinux 9.0 under snort 2.6.0 edition. Test data is from DARPA 1999 Intrusion Detection Dataset [7,8,9] developed by MIT Lincoln Laboratory. The dataset is the most comprehensive attack-testing dataset, covering five main categories, namely Probe, DoS, R2L, U2R and Date, and 58 typical attack modes. All the time data in the experimental results are obtained from the means of measurements of "time" command executed 10 times under Linux.

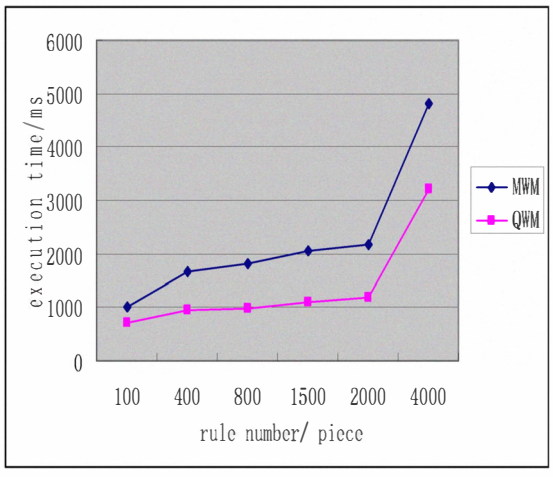

Figure 1 Performance Analysis under Different Rule Numbers

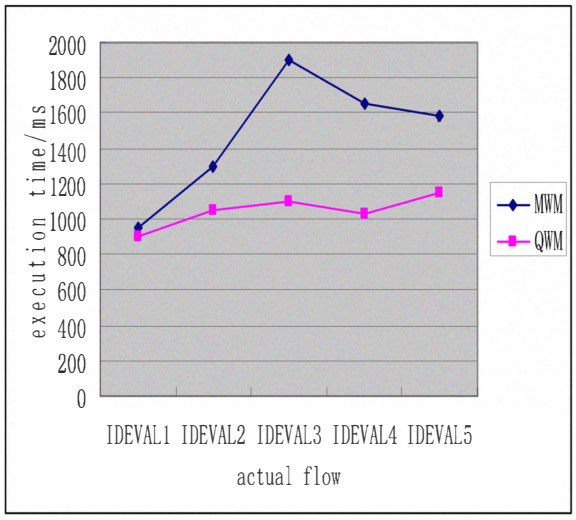

Figure 2 Performance Analysis under Different Actual Flow

The rule sets in figure 1 all contain modes with length of 1 and 2. Figure 1 clearly shows that the execution time of QWM algorithm is significantly lower than that of MWM algorithm and the performance is markedly improved. Figure 2 clearly shows when the experiment uses rule set that contains 1000 rules of modes with the length 1 or 2, FWM algorithm performs better than MWM algorithm to various extent in different real flow detection. But when the model of minimum length is greater than 2, there's only small difference between these two algorithms.

Conclusion

This essay puts forward an improved algorithm on the basis of Modified Wu-Manber algorithm multi- pattern matching algorithm. It divides the mode into groups and uses bitmap match for 1 or 2 – byte modes. This ensures the length of the minimum mode is longer than 2 and the window will have a bigger migration distance no matter the match succeeds or fails. Less matching attempts of the mode set and the text is required thus the efficiency of matching is improved. Experiments show that the performance of the improved algorithm can improve by 41% to the highest.

ACKNOWLEDGEMENT

I would like to express my gratitude to all those who helped me during the writing of this paper. Their encouragement and unwavering support has sustained me through frustration and depression.

Last but not the least, my gratitude also extends to my family who have been assisting, supporting and caring for me all of my life.

REFERENCES

[1] DAI Liu-ling, XIA Yu-ning. A lightweightmultiple stringmatching algorithm computer science and information technolog[C] //Proc of InternationalConference on Computer Science and Information Technology. 2008: 611-615.

[2] BAIYue-bin, KOBAYASHIH. New stringmatching technology for network security[C] //Proc of the 17th International Conference on Advanced InformationNetworking andApplications. Washington DC:IEEE ComputerSociety. 2003: 198-201.

[3] BOYER R S, MOORE J S. A fast string searching algorithm [J].Communications of theACM, 1977, 20(10): 762-772.

[4] WU S, MANBER U. A fast algorithm for multi-pattern searching Tech Rep TR94－17[R]. Tucson : Department of Computer Science, University of Arizona, 1994.

[5] CHEN Yu ,CHEN Guo-Long The Performance Analysis of Wu-Manber Algorithm and its Improvement,Computer Science,2006(6)

[6] Lu wangjie ,Ju shiguang An Optimizing AC Algorithm for Intrusion Detection System Computer Engineering and Applications, 2006, 42(15):146－148.

[7] CAI Xiao-yan, DAI Guan-zhong,YANG Li-bin Improved multiple patterns string matching algorithm ,Journal of Computer Applications,2007(6)

[8] Snort2. 6. 0 [EB/OL]. [2006-12-05]. http: //www. snort. org/d.1

[9] 1999 DARPA intrusion detection evaluation data set[DB /OL].[2007 - 04 - 09]. http: //www. l.1 mit. edu/IST/ideval/data/1999 /1999_data_index. htm.l