International Journal of Computer Applications (0975 – 8887)
Volume 96– No.20, June 2014

Feasible Study on Pattern Matching Algorithms based on Intrusion Detection Systems

Saurabh Tiwari
M.Tech(CSE)Final Year.
ITM University Gwalior (M.P), India

Deepak Motwani
Assistant Professor, CSE/IT Department,
ITM University, Gwalior (M.P), India

ABSTRACT

With the increasing popularity of web services, attackers are paying more attention to hack the web services and the servers on which they run. Thus the need of effective modern intrusion detection system has taken prime importance for any system. At present intrusion detection system (IDS) analysis module uses the pattern matching technology, which is applied in various problems in the field of information security, genetic algorithms and evolutionary algorithm. As one of the solution to intrusion detection problems, pattern matching algorithms shown their advantages. This paper presents a feasibility study on a pattern matching algorithm

General Terms

Intrusion Detection System (IDS), Pattern matching algorithm.

Keywords

Intrusion detection; pattern matching; AC algorithm, WU-Manber algorithm. BM algorithm;

1. INTRODUCTION

The intrusion detection system (IDS) is a tool which has ability to read the detail of the network packets. The intrusion detection system is one of the most significant dynamic security technologies, which can be used in the serious security system creation and the basic service protection. Intrusion detection is the security mechanisms that ensure identification of misuse, and abuse of computer systems by both system insider and external attacker. A network intrusion detection system examines all arriving packets and tries to uncover suspicious patterns known as signatures or rules. These rules determined by a network administrator while the configuration and deployment of the network intrusion detection system depend on the safety measures and network policies of the organization. Remember that an important thing about the Intrusion detection system, it can only detect the malicious/suspicious activity and after detection inform to the administrator but it cannot provide any kind of protection against malicious/suspicious activity.

The basically Intrusion detection system can be categorized into two types: host based intrusion detection system and Network based intrusion detection system this categorization upon the basis of place where the intrusion detection process takes place. An Intrusion detection system that runs on host to detect suspicious activity on that host called host based intrusion detection system and An Intrusion detection system that operates on network data flows are called Network based intrusion detection system [11]. One other important thing is that Intrusion detection system can be exist as software as well as hardware.

The Intrusion detection system has 3 part sensor console and engine. In Sensor-Sensor which generates a security event, Console-Console to monitor events and alerts and control the sensors and Engine-Engine that records events logged by the sensor in a database and uses the rules of the system to generate alerts from security events received

At the heart of all current network intrusion detection systems there is a string matching algorithm. The network intrusion detection system uses the string matching algorithm to match up the payload of the network packet and/or flow alongside the pattern entries of the intrusion detection rules, which are a part of every network having a network intrusion detection system [1, 2].

A typical intrusion-detection system is show in Figure1. The term system (target system) is used here to denote the information system being monitored by the intrusion-detection system. The term audit indicates information provided by a system about its inner workings and behavior. An intrusion-detection system obtains information about an information system to act upon a diagnosis on the security status.

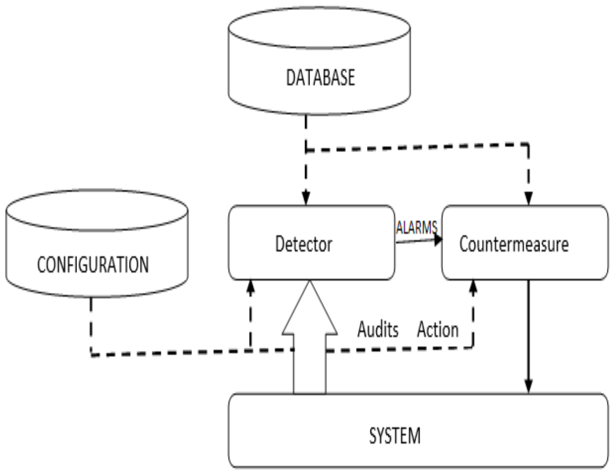

Fig 1 A generalized intrusion detection system

An intrusion-detection system can be conveyed at a very macroscopic level as a detector that processes Information coming from the system to be guarded (Fig. 1).

This detector can also launch probes to activate the audit process. It uses basically three kinds of information: long-standing information associated with the technique used to identify intrusions, configuration information about the in progress state of the system, and audit information illustrating the events that are happening on the system. The role of the

detector is to remove unnecessary information from the audit trail.
A decision is then taken to evaluate the likelihood that these actions or this state can be considered as signs of an intrusion or susceptibility. Then corrective action to either prevent the actions from being executed or change the state of the system back to a secure state can be taken by a countermeasure component.
The remaining paper is divided into following sections. Section 2 deals into pattern matching algorithms, Section 3 gives a brief overview of a single pattern matching algorithms and section 4 gives a brief overview of multi pattern matching algorithms.

2. PATTERN MATCHING ALGORITHM

Pattern matching, that is character string matching, to find a certain character string in the target character string. Pattern matching algorithms compare the set of strings in the rule set to the data seen in the pockets that flows over the network. If the pattern string come into views in target string once or more than a few times, such case is described successful match, or described unsuccessful match. Based on difference in mode series of numbers which is gotten after retrieving text T, there are two types of pattern matching: Single pattern matching and multiple pattern matching [3,4].In single pattern matching only one pattern string can be matched by the algorithm and in multi pattern matching several pattern string can be matched by the algorithm.

Famous pattern-matching algorithms comprise Knuth–Morris–Pratt (KMP) [5], Boyer–Moore (BM) [6], Wu–Manber (WM) [7], and Aho–Corasick (AC) [9]. The KMP and BM algorithms are capable for single-pattern matching, but are not appropriate for matching multiple patterns. The WM algorithm is revision of the BM algorithm to multiple patterns. The AC algorithm pre-processes the patterns and makes a finite automaton that can match multiple patterns at the same time. The AC algorithm may require an enormous amount of memory space to do so [4]. A simple implementation of the AC algorithm is to build a two-dimensional state transition table for the finite automaton. Such an implementation needs a massive amount of memory space when the whole pattern length is large. A number of schemes had been suggested to lessen the memory requirement

3. SOME FAMOUS SINGLE PATTERN MATCHING ALGORITHM

3.1 Knuth-Morris-Pratt Algorithm (KMP)

This algorithm was introduced by Don Knuth, Jim Morris, and Vaughan Pratt on the basis of analysis of BF (Brute force) algorithm [5,12]. It is quite similar to the Brute Force approach regarding scanning the text left to right, however, we are now using information from the previously compared characters in order to determine the maximum possible shift of the pattern to the right. The idea is to avoid comparisons with elements from the text *T* that have previously been compared with some elements of the pattern *P*. In order to achieve this task, KMP pre-processes the pattern to find matches of prefixes of the pattern with the pattern itself. The pre-calculation is done in time O(m) and is called the next function F[j]. This function is an array that represents the size of the largest prefix of P[0…j] .The KMP algorithm states that we can shift the pattern in order to avoid redundant comparisons is namely the length of the next function.

3.2 Boyer-Moore (BM) Algorithm

Developed in 1977 by Bob Boyer and J Strother Moore, the Boyer-Moore algorithm [6,12 15] shares some similarities with the KMP algorithm, but at the same time introduces three novel ideas in order to approach the pattern matching task more efficiently. The fundamental ideas are:

- Matching the pattern backwards (right to left scan)
- The bad character shift rule
- The good suffix shift rule

This is an algorithm that searches for the location of the first occurrence of pattern in the text. This algorithm has the property that, in most cases, not the entire first i characters of the text are examined. Further, the number of characters actually examined decreases as a function of the number of characters in pattern.

To summarize, the Boyer-Moore algorithm works by calculating two pre-processing steps to determine the next appropriate alignment after each mismatch. In the first step calculate positions ahead of the current position to begin the next search (bad character shift rule). The second step makes a similar computation indicating how many characters were matched successfully before a mismatch (good suffix shift rule). [4]

3.3 Boyer-Moore-Horspool (BMH) algorithm

Although the Boyer-Moore algorithm is very fast in practice, there have been numerous improvements made to it. The first notable improvement was made by Horspool.Horspool's variant simplifies the Boyer-Moore algorithm greatly while also making it faster in general. Horspool noted the bad character shift was usually the longest shift and considered a medication that allows us to omit the second more complex shift table. Horspool proposed Boyer-Moore-Horspool algorithm in 1980 [12,15]
This is a simplified form of the Boyer-Moore algorithm, whole matching process is same as BM algorithm the only difference is BMH algorithm uses only bad-character shift skip good suffix table because pre-processing good suffix rule is tough thus . BMH algorithm is the best in terms of average case performance for all alphabet sizes and almost all pattern lengths. It is relatively simple to implement. Though its worst case time complexity is O (NM), it has better average performance than BM, both experimentally and analytically. Also, whatever the size of target text, it is better to use BMH because of its low space complexity. Since a virus scanner should be very fast and economical, as well as it should be able to handle very large files also, without being slow, hence for this purpose BMH algorithm appears to be the best-choice algorithm.

To implement the BMH algorithm, first two position indicators are required, j is set up in the pattern and k is set for the target text. The first letter of pattern P is aligned under the first letter of the target text T**.** It is similar to a window on the text that allows us to see only m characters, the pattern length. Later, this window will shift to the right, to allow us to view other positions. Another position Indicator i is initialized to m-1, and will record the location of the rightmost text position seen through the window. We start a letter-by-letter comparison from letter Pm-1; these comparisons are performed between text Tk and pattern Pj.

After each successful comparison, both j and k are decremented. It continues as long as characters match and as long as there remains un-compared characters in the pattern. P. j=-1 implies that all characters in the pattern have been matched and we have found an occurrence of the pattern in the text. Whether a match is detected or not, the window is shifted a certain predetermined distance to the right: the position indicator i is incremented by d, j is set to m-1 and k to i. This process is repeated until we reach the end of the text.

3.4 Boyer-Moore-Horspool-Sunday (BMHS) algorithm

At 1990 Sunday proposed an improved variant of BMH algorithm which is come to known as BMHS (Boyer-Moore-Horspool-Sunday) algorithm [4.12,15]. The basic principle of BHMS algorithm is similar to BMH algorithm. It also uses only bad character rule to calculate right shift distance in case of any mismatch occurred, but the method of calculating the right shift distance is different from BMH algorithm. In this algorithm the bad character rule uses the next a character to decide the right shift distance. Obviously performance of the BMHS algorithm is better than BMH algorithm having time complexity O (n/m+1) in the best case. [12].

3.5 Improved BHMS algorithms

JingboYoun, Jinsong Yang and Shunli Ding by analyzing and comparing above several algorithms proposed based on BHMS an improved version of BHMS that is come to known as improved BHMS(IBHMS) algorithm in 11th international symposium on distributed computing and application to business, Engineering &Science in 2012 [15]. The proposed improved BHMS algorithm perform matching from right to left and if any mismatch occurs, then this algorithm calculate rightmost shift distance of pattern P and do accordingly. The IBHMS algorithm uses two types of bad character rule to implement algorithms: single character function Badcharacter1(x) and double character function Badcharacter2 (x, y).

The IBHMS algorithm has mainly two steps: first is pre-processing in this stage values of badchar1(x) and badchar2(x, y) to be calculated and second step is matching in which matching is performed accordingly to the algorithmic procedure. It also uses a 2-D array to save shift distance. The time complexity of the IBHMS algorithm in the best case is O(n/m+2) which is much better than three other algorithm BM, BMH and BMHS algorithms.

The following table gives the comprehensive study of above discussed single pattern matching algorithm.

TABLE 1.0

Algorithm Name	Author and Year	Time Complexity(best case)	Key Characteristic	Comparison Order
KMP	D.KnuthJ.Morris And V.Pratt 1974	O(n+k)	Use the notation of the string’s border therefore increase the performance	Left to right
BM	Bob.Boyer and J.Strother.Moore 1977	O(mn)	Use both good suffix and bad character rule	Right to left
BMH	Horspool 1980	O(n/m)	Use only bad character rule	Right to left
BMHS	Sunday 1990	O(n/m+1)	Use only bad character rule, but badchar consider the next a character to determine the right shift distance	Right to left
IBMHS	JingboYuan,Jinsong Yang and Shunli Ding	O(n/m+2)	Uses two bad character rule: Single character badchar1 (x) and double character function badchar2 (x,y)	Right to left

4. SOME FAMOUS MULTI PATTERN MATCHING ALGORITHM

4.1 Aho-Corasick (AC) Algorithm

AC algorithm was first introduced in 1975 by Alfred Aho and Margaret Corasick and AC algorithm is an extension of KMP algorithm that is a single pattern matching algorithm towards multi pattern matching by combining finite state automata [4,9.12,13]. Two variants of AC algorithm exist first is deterministic AC algorithms and nondeterministic AC algorithm, the basic difference between deterministic and non-deterministic AC algorithm is: deterministic method requires high memory while non-deterministic requires less memory in comparison to deterministic method. AC algorithm performs matching in two steps s is pre-processing and matching step. In pre-processing step construct the finite state automation tree and then this finite state automation tree is used in matching step. This is one of the most widely used algorithms for multiple string matching. This is a kind of “dictionary-matching” algorithm which finds the various strings from a finite set of strings in a text that needs to be inspected. This algorithm can work simultaneously for all the strings in the set and locate these in the text, that is, this algorithm can match all the strings “at once” thus reducing the time taken for the searching. All the other algorithms that we have discussed above match only one pattern at a time. But this algorithm matches all the patterns at once. So this algorithm is better suited for a network intrusion detection, which has to match all the patterns from the rules in the incoming traffic. This algorithm, as well, works on the same concepts of aligning the text with the pattern (in this case the suffix tree) and window shifting. This algorithm also uses the concept of shifting the window as much as possible to remove unnecessary comparison. Operation of AC algorithm pattern matching is controlled by three functions, goto function, failure function, and output function.

A Go to Function The go to function is basically a trie (a prefix tree used to hold an associative array of strings) of the patterns we are searching for.

A Failure Function This function tells the algorithm what to do when there is no suitable match. It maps a state into a new state and is invoked when the go to function fails.

An Output Function This output function defines a set of keywords for every state of the DFA. Moreover, given a state and an input symbol, there is only one transition state that the machine can jump to.
The main difference between the Boyer-Moore algorithm and Aho-Corasick algorithm is that instead of sliding a single pattern along the text, this algorithm slides a tree of patterns (the suffix tree) while performing the shifting techniques.

4.2 Aho-Corasick_Boyer-Moore(AC-BM)algorithm

AC-BM algorithm is an improvement and extended version of AC algorithm was introduced by Jang Jong in 1993 by using the core idea of the BM algorithm [4,12].Almost whole matching process of AC-BM algorithm is same as AC algorithm the only difference is AC-BM algorithm uses bad character and good suffix rule, if any mismatch occurred instead of fail() function thus AC-BM algorithm speed-up of the matching process therefore AC-BM algorithm is better than AC algorithm.

The AC-BM algorithm includes three important parts: Construction of tree automaton of the AC pattern on the basis of the pattern string set, shift function and searching text strings with AC automaton and shift function [10].

1. With the help of pattern string set constructs pattern tree automaton. The automaton of the pattern tree constructs the prefix based on character string and the same prefixes will be used as the root nodes of the tree. In matching the search, the character P [m-1] on the right end of the shortest pattern string in the automation pattern tree should have to be aligned with the rightmost character T [n-1] of the text string.

2. In matching search adopt the direction from rear to front Character comparison is conducted from the left to the right (the characters are compared, that is one by one and by the levels in direction from the root characters of the pattern tree automaton to the nodes). A pre-processing may be conducted before comparison in order to avoid unnecessary comparisons.
3. Use the shift character rules that are: bad characters and good suffixes. Bad character shifts: when the character in the pattern tree is inconsistent with the character Q of the text string, shifts the text string to the next position with Q occurring and align the Q in the pattern tree with the Q in the text string; if the character Q does not present in the pattern tree, shift the text string right side for the length of the smallest character string in the pattern string set. Good suffixes shift rule: when the character in the pattern tree is inconsistent with the character Q of the text string, shifts automaton to the next position in the pattern tree the earliest matching with the text string. In the process of shifting, the automaton does not shift more than the shortest pattern string in the pattern string set

4.3 Wu-Manber (WM) Algorithm

WU-Manber proposed by Sun Wu and UdiManber in 1994 by using the concept of the bad character role of BM algorithm therefore it is an extension of the BM algorithm towards multi pattern matching. WM algorithms two core mechanisms, the filter mechanism based on hash technology and the block character shift mechanism based on the bad character shift technology from BM algorithm.[7,16]
By calculating the hash value of the suffix block character in patterns, the patterns with same suffix are linked in a list. The entry list is stored in a hash table. When searching a text, we calculate the hash value of the block character inside the current match window, and look up the hash table to get the entry of the patterns that contain the same block character as their suffix. Obviously, these patterns are possible matching strings. Symbol "*B*" is used to represent the size of the block character. It usually is 2 or 3. In this paper, we suppose "B = 2".
When a matching is finished, the match window should right shift. The size of match window is determined by the length of the shortest pattern. Here we suppose it is "m". The bad character shift technology is used for the match window shift. The only difference is that the single character is replaced by the block character. The bad character shift technology insures the match window can shift to the best of its abilities. It quickens the pattern matching very much.
The following table gives the comprehensive study about above discussed multi pattern matching algorithms

TABLE 1.1

Algorithm	Author and year	Time complexity	Key characteristic	Comparison order
AC	Aho A.V and Corasick M.J 1975	Linear	Based on Finite automata	In two steps: preprocessing and matching
AC-BM	Jang Jong 1993	-	Based on AC algorithm and using the leaf idea of BM algorithm	In two steps: preprocessing and matching
WU-Manber	Sun Wu and UdiManber 1994	Sub linear	Use the badchar rule of BM algorithm and calculate the shift distance from a block of character in the suffix the search windows	In two steps: preprocessing and matching

5. CONCLUSION

Over the past few decades, there have been many solutions presented, tested, and implemented to detect signatures which containing attack definitions. In order to ensure advance security of computers, intrusion detection system is requisite to work with high competence. This survey identifies a number of promising algorithms and provides an overview of recent developments in the single keyword string matching as well as multiple strings matching for NIDS. IDS find the Intrusion using known attack patterns called signatures. Every IDS will have large number of signatures. If Pattern matching algorithm is time-consuming, the IDS attack reaction time will be very high. The existing efficient algorithms such as Boyer Moore (BM), Aho-Coarasick (AC) does not get better throughput of IDS. In future the proposed solution of the given problem is to Improves the detection speed or attack-response time, system should be capable of processing more number of signatures on every packet payload, The attack response time should be less when Compared with existing algorithm like AC,BM and it Should use less memory (Space complexity is low).

6. ACKNOWLEDGEMENT

We thank the ITM University Gwalior and our guide Mr. Deepak Motwani Assistant Professor, CSE/IT Department ITM University Gwalior for the Motivation and Encouragement to make this research work possible.

7. REFERENCES

[1] Anagnostakis K G, Markatos E P, Antonatos S, Polychronakis M, 2003 A domainspecificstring matching algorithm for intrusion detection, Proceedings of the 18th IFIP International Information Security Conference (SEC2003).Athens,Greece: Kluwer AcadamicPubhishers,.

[2] Bi Kun., Gu. Nai-jie., Tu Kun., Liu.Xiao-hu.and Liu. Gang, 2005 A PracticalDistributed String Matching Algorithm Architecture and Implementation Proceeding of world academy of science, engineering and technology.

[3] Cheng Yu–qing, Mei Deng–Hua. 2009 The improved of IntrusionDetection System BM pattern matching algorithm [J].Computer Technology and Development, 2009 **19** (3) :172–174. (In Chinese).

[4]QIN Hai-sheng, LI Xin-hua, WEI Hai-lan and LI Jun-hui 2011Research and optimization of Pattern Matching Algorithm Based on Instrusion Detection System IEEE ,Pp 258-261.

[5] D. E. Knuth, J. H. Morris, and V. R. Pratt, 1974. Fast pattern matching instrings, Stanford University, Stanford, CA, TR CS-74-440.

[6] R. S. Boyer and J. S. Moore, Oct. 1977 A fast string searching algorithm, . ACM, vol. 20, pp. 762–772,.

[7] S. Wu and U. Manber, USA, May 1994 A fast algorithm for multi patterns searching, Technical Report TR 94-17, University of Arizona at Tuscon,.

[8] AHO, A. 1980. "Pattern Matching in Strings," in Formal Language Theory: Perspectives and Open Problems, ed. R. Book, pp. 325-47. London: Academic Press

[9] A. V. Aho and M. J. Corasick, Jun. 1975, Efficient string matching: An aid tobibliographic search," Commun.ACM, vol. 18, pp. 333–340,

[10] Zhengcai Wang, Xiaofeng Wang and Zhengyi Tang may 2013 Analysis and Improvement of AC-BM Algorithm in Intrusion Detection System" journal of networks, vol. 8, no. 5.

[11] Martuza Ahmed, Rima Pal, Md. MojammelHossain, Md. Abu Naser Bikas, and Md. KhaladHasan 2009 A comparative study on the currently existing intrusion detection systems 2009 International Association of Computer Science and Information Technology - Spring Conference,IEEE.Pp.151-154.

[12] Pei-fei Wu, Hai-juanShen 2012 The Research and Amelioration of Pattern-matching Algorithm in IntrusionDetection System IEEE 14th International Conference on High Performance Computing and Communications,Pp.1712-1715.

[13]AkhtarRasool, Amrita Tiwari, Gunjan Singla, Nilay Khare 2012 String Matching Methodologies: A Comparative Analysis International Journal of Computer Science and Information Technologies (IJCSIT), Vol. 3 (2) ,Pp.3394 – 3397.

[14]Zeeshan Ahmed Khan, R.K Pateriya July 2012 Multiple Pattern String Matching Methodologies: A Comparative Analysis International Journal of Scientific and Research Publications, Volume 2, Issue 7,

[15]Jingbo Yuan, Jinsong Yang and ShunliDing 2012 An Improved Pattern Matching Algorithm Based on BHMS 11th International symposium on Disributed Computing and Application to Business ,Engineering & science,IEEE-,Pp 441-445

[16]Cheng Ke-qin,DengLin,WangHui 2013 An improved multi-pattern matching algorithm in intrusion detection Fifth Conference on Measuring and Mechatronic Automation,IEEE-,Pp 203-205.