IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 2 Issue 3, March 2015.
www.ijiset.com
ISSN 2348 – 7968

Review Paper for Backward Hashing and Automation Tracking For Virus Scanning

Mital K Panchal, Bakul Panchal

Information Technology, L. D college of engineering
Ahmedabad , Gujarat 380013 India

Assistant Professor Information Technology, L. D college of engineering,
Ahmedabad , Gujarat 380013 India.

Abstract

Scanning of virus involves computationally rigorous string matching compared to a great number involving signatures of various attribute. It's a challenge to select the matching algorithms from a variety of matching signatures. We propose an approach that is more hybrid and the partitions of signatures into short and long ones in the open source called ClamAV for scanning of virus. An algorithm upgraded from the Wu-Manber algorithm, also known as the Backward Hashing algorithm, which is responsible for only lengthy patterns to lengthen the average skip distance. Next to come up is Aho-Corasick algorithm scans for only short patterns to shorten the automaton sizes. The algorithm experts mentioned utilizes the bad-block heuristic to capitalize long shift distance and lessen the verification frequency, results in much faster compared to the original WM execution in ClamAV. The latter enhances the AC performance by almost 50% because of better cache locality. For the string matching performance we also rank the factors to indicate their importance.

Keywords: *Data mining, ClamAV, hybrid approach, backward hashing.*

1. Introduction

The last decade has experienced a revolution in information availability and exchange of it through internet. In the same strength more business as well as organizations began to collect data related to their own operations, while the database technologist have been seeking efficient way of retrieving, manipulating and storing data, which the community learning on machines focused on techniques which used to learn develop, and gain information from the data. Data Mining is the process of analyzing data from different perspectives and summarizing it into productive information. Data mining is the outcome of load transaction data, transform and extract data to the data storing system also known as data warehouse system or mechanism which also manages and stores data in a multiple dimension database system, by using application software analyze the data, provide data access to business analysts and information technology experts, shows the data in a fruitful format, example a graph or table. Data mining involves the anomaly detection, regression, association, regression, classification, rule learning, summarization and clustering.

2. Issues in hybrid approach of malware detection

The data mining process is to be consisted of five steps.

1. Problem statement and formulation of hypothesis
2. Data collection
3. Data preprocessing
4. Model estimation
5. Model interpretation

ISSN 2348 – 7968

Objectives for applying hybrid approach in malware detection

- Faster malware signatures identification process
- Implementation of hybrid approach thru data mining on large volume of data
- Protect user data
- Protect system resources (including the network)
- Provide application isolation Features
- Exceptional security at the Operating System level with the use of Linux kernel
- Use of Sandbox application is compulsory for all applications
- Secure inter-process communication
- Application signing
- Application oriented and granting users permissions
- Securing signatures for future references

2.1. Malware signature detection from a file

A Malware detector accepts the obscure version of Malware and eliminates the obfuscation carried on the program and produces the normalized executables .Thus it can be said that the Malware detector increases the detection rate of the program. Malware M is taken and passed through the tool called detector. After detection the signature of this Malware is extracted and compared with the signatures of canonical form [5]. Maximum length of matching signature of canonical form with the malware is considered and the new signature of the canonical form is stored in the signature database for future comparisons. Let us consider a Malware 'M' and ci ε C is canonical form of the malware M.

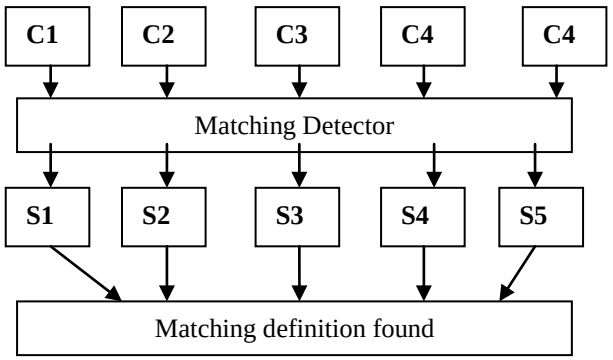

Fig 1 Malware signature detection.

Once we found and detect a definition as a malware definition, we will add it to the database for all future detections of malware signatures in all input way. Here , once we detect the new virus signature, we add it to the database at an appropriate index. Apart from these, when we get a filename as a definition, we match it with the signatures available in the database, it it exists (matches with the stored definition), it will be declared as malware without putting it into malware detection process. Otherwise, it should pass through the process of malware detector tool.

When we are getting a filename and matching it with the available definitions of database, it's a process that is contiguous and time consuming. It is because we may have lacs of definitions in our database at once. To make this process easy and fast, we have read several research papers and taken 2 as base papers. Based on which the following algorithm is explained

2.2 Algorithm adopted for faster searching of patterns from database

The basic idea of the Boyer-Moore string-matching algorithm [BM77] is as mentioned. Lets assume that the pattern is of length n. We start by comparing the last character of the pattern against n, the n'th character of the text. If there is a mismatch (and in most texts the chances of a mismatch is much greater than the chances of a good match), then we come to conclusion that the rightists occurrence of tm in the pattern and shift accordingly.

2.2 Algorithm as per base paper

- *Compute a hash value h based on the current B characters from the text (starting with tm B +1. . . tm).*

ISSN 2348 – 7968

- *Check the value of SHIFT[h]: if it is > 0, shift the text and go back to 1; otherwise, go to 3.*
- *Compute the hash value of the prefix of the text (starting m characters to the left of the current position); call it text_prefix.*
- *Check for each p, HASH [h] p < HASH [h +1] whether PREFIX[p] = text_prefix. When they are equal, check the actual pattern (given by PAT_POINT[p]) against the text directly.*

Table 1: Symbol descriptions

Symbol	***Description***
M	The length of the search window (also the shortest pattern length)
X	The block in the suffix of the search to be hashed into the shift table
B	The length of X i.e. block size
∑	The set of characters (We assume $\|\sum\|$ = 256 in the text. i.e. the number of values in a byte).

3. Approach for the new algorithm

3.1 Why do we need a new algorithm?

Wu-Manber algorithm has wide scope over other algorithm when multiple string matching process is concerned. But when implemented practically, we realize the following limitations.

1. There are information and operations that has redundancy.
2. There is a very less use of prefix table. So using it serves a very small or no purpose.
3. We need to traverse the whole list of link. These turn the algorithm more complex and affect the performance of it.

Because of these limitations we found, we have tried to user Prefix table to filter the patterns. Filtering based on Addresses will avoid traversing the whole records of database

After implementing the following algorithm, we have found a better performance than the original one.

3.2 Additions and changes in Algorithm

We have added one more SHIFT table to the above algorithm. Considering the size of the character block B, rather than simply a character, block transfer characters are used till date. Mostly B=2 or 3, SHIFT build an index for all the possible characters the length is B. This is the reason why the size of SHIFT is the permissible in permutations of B characters. The SHIFT value decides the in-motion distance of a string of some certain B-characters in the text, mentioned as the vector between the far right the certain B-characters and the tail of all patterns. Suppose X is B-length n character block of the current calculation and its hash value is i, let's take two cases: Firstly, X disappear in any of the string pattern, algorithm of the current text moves distance m-B +1 characters position in string which are matching, so we store m-B +1 in SHIFT [i]. Second, X string appears in some modes, in this case, the algorithm matches the rightmost position X that appears in the pattern string.

3.3 Performance

Rough Analysis of the Running Time

We are presenting an estimate for the running time of this algorithm assuming that both the pattern and the strings are randomly entered with uniform distribution. In practice, Strings and patterns aren't random, they gives a rough estimate idea about the performance of the algorithm. We confirm that the running time expected is less than linear in the size of the text as compared to the algorithm given in base paper.

ISSN 2348 – 7968

Let N be the size of the text, P the number of patterns, m the size of one pattern, M = mP the total size of all patterns, and assume that N ≥ M. Let c be the size of the alphabet. We define the size of the block used to address the SHIFT table as B = Logc 2*M*. The SHIFT table contains all possible strings of size b, so there are $c^b = c > 2Mc$ entries in the SHIFT table. The SHIFT table is constructed in time O (M) because each substring of size B of any pattern is considered once and it takes constant time on the average to consider it. We divide the scanning time into two cases. The first case is when the SHIFT value. The second case is when we consult the hash table for the value.

The second shift table is added to increase the performance where expensive exact string comparison is avoided quite often, which leads this algorithm towards a faster multi-pattern matching algorithm

4. Conclusions

This paper proposed a method for backward hashing in a hybrid approach where we have proposed to find malicious signatures from a computer using data mining in an effective way. Because this approach is a mixture of multiple theories suggested by many of the programmer before, we will implement it to generate satisfactory results. After that the formal description of the program to obtain malicious signatures can be said appropriate for the purpose. We have adopted several detection method that suits to the purpose.

6. References

[1] Data Mining: Concepts and Techniques. By : Jiawei Han and Micheline Kamber.

[2] Computer Software and Applications Conference - Workshops and Fast Abstracts - (COMPSAC'04) - Volume 02, pp. 41–42, 2004.

[3] [api] "Wikipedia CLAMAV article. http://en.wikipedia.org/wiki/Clam_AntiVirus

[4]J.Xu, A.H.Sung, P.Chavez, S.Mukkamala,"Polymorphic Malicious Executable Scanner by API Sequence Analysis", Proceeding of 4th IEEE Symposium of International Conference on Hybrid Intelligent Systems (HIS '04).

[AC75] Aho, A. V., and M. J. Corasick, ''Efficient string matching:an aid to bibliographic search,'' Communications of the ACM 18 (June 1975), pp. 333 340.

[BM77] Boyer R. S., and J. S. Moore, ''A fast string searching algorithm,'' Communications of the ACM 20 (October 1977), pp. 762 772.

First Author Mrs Mital Panchal holds the degree in BE IT with over 6 years of experience as an assistant professor in Govt Engg. college on various IT subjects. Current research interest is just shown in the paper which is about to publish.

Second Author Mr. Bakul Panchal holds the degree in ME IT with over 11 years of experience as an assistant professor in Govt. Engg. College.