International Journal of Computer Applications (0975 – 8887)
Volume 11– No.4, December 2010

# Improving Speed of the Signature Scanner using BMH Algorithm

Sunita Kanaujiya
G.B.T.U., Lucknow
K.I.E.T, Ghaziabad,
U.P, India

Dr. S.P.Tripathi
G.B.T.U., Lucknow
I.E.T., Lucknow,
U.P, India

N.C.Sharma
G.B.T.U., Lucknow
LKCE Ghaziabad
U.P, India

## ABSTRACT
As the importance of data in our computing systems has increased, number of viruses that may harm its integrity has also increased. It has become very important to detect them, so that we can remove them as well as disinfect the infected files. This paper presents an implementation of a fast signature scanner to detect the viruses. To reduce the scanning time, it uses Boyer-Moore-Horspool (BMH), a fast pattern-matching algorithm. It showed the best performance among commonly used pattern matching algorithms like Boyer Moore and Turbo-Boyer-Moore algorithms.

## General Terms
Integrity checking technique, Signature scanning technique, Pattern matching problem.

## Keywords
Boyer-Moore-Horspool (BMH), Boyer-Moore algorithm.

## 1. INTRODUCTION
As the importance of data in our computing systems has increased, number of viruses that may harm its integrity has also increased. It has become very important to detect them, so that we can remove them as well as disinfect the infected files. Therefore any defense system should have a component that detects the presence of any kind of malicious code. There are four basic types of virus detection techniques: Signature Scanning, Integrity Checking, Activity Monitoring and Heuristic method. Each has pros and cons of its own.

This paper analyzes the problem of virus detection using *Signature Scanning Technique* and its reliance on fast pattern matching algorithms. In this technique a pattern i.e. a virus signature, which can be anywhere within the file, is searched for. Pattern matching is the most computationally expensive task that scanner has to perform frequently. Therefore if pattern-matching algorithm used is not fast, it will take time and users may find it annoying. For these reasons, a fast pattern-matching algorithm named Boyer-Moore-Horspool (BMH) has been used to implement the scanner. In comparison with other algorithms like Boyer-Moore and Turbo-Boyer-Moore algorithms BMH proved the fastest pattern-matching algorithm.

In technical terms, a computer virus consists of three parts:

- the infection mechanism,
- the trigger,
- the payload.

The infection mechanism part looks for victims and mostly avoids multiple infections. After that either it overwrites the victim or attaches itself at the end or at the beginning of file. trigger is a specified event when the payload has to be executed. The payload causes some malicious behavior, e.g. corrupting the boot sector of floppy, formatting the hard disk drive or manipulation of files.

Worms are another form of malicious software but unlike viruses, worms are independent programs that can travel across network connections. Therefore worms do not really need to change other programs.

## 2. ANTIVIRUS SOLUTIONS
As the number of viruses increases on daily basis, there is a need of virus detection tools so that they can be eradicated from the system. Generally there are four types of virus detection techniques:

### 2.1 Integrity checking technique
In this technique the integrity checker program generates checkcodes that may be checksums, CRCs, or other secure hashes of the files to be checked for viruses. Now on a regular basis these checksums are again computed and compared against the previously stored checksums. If there is any mismatch between these two, it clearly indicates that the specified file has been modified [3]. Since this technique detects presence of virus by the changes in files, it is capable to detect unknown viruses also.

Besides this, integrity checking technique has several drawbacks. First, the initial calculation of checksums has to be performed on a clean system that does not contain any viruses. If system is not clean, the technique will never detect viruses that have already infected some files. Second, in most cases files may be modified system during execution, which will cause lots of false positives. Users do not always prefer that case.

### 2.2 Signature scanning technique
Signature scanner is the most popular and widely used virus detection tool. It operates by reading data from disk and applying pattern matching operations against a list of known virus patterns. If a match is found for a pattern, presence of

a virus instance is announced [5].

Scanner is fast and easy to use, but to maintain its effectiveness, the list of patterns must be kept up-to-date. However pattern files are easy to update. Moreover, though many new viruses are discovered every week, only few become disastrous and widespread [6]. Therefore, an old signature database will still work for most cases. There are many advantages of scanners, which include its scanning speed that can be further increased. Another benefit of scanning is that it can also be used for detecting other malicious softwares like worms, logic bombs, trojan horse code etc. All that we have to do is find their signatures and add them to the database. For these reasons, this technique is widely used to detect the majority of viruses.

Cohen says that signature scanning technique is not a good approach against computer viruses because scanning cannot find new viruses before their patterns are known, nor will such methods work against polymorphic viruses. He tries to demonstrate that checksumming is the most cost-effective approach to virus protection. But this argument is not so convincing, because scanners are simpler, more economic and flexible and their still widespread use affirms it. Scanners provide exact nature of infection and other related details, which integrity checker cannot provide.

## 2.3 Activity monitoring technique

In this technique, some programs are used to monitor the behavior of other programs that are executed. These monitoring programs are called behavior monitors and reside in main memory. In case if a normal program tries to modify interrupt table, boot sector, partition table, or performs some other suspicious activities, behavior monitors raise an alarm or perform some action to prevent it [4].

This technique maintains the database of all behaviors that viruses are supposed to have. But the new viruses may use another method to infect system that is not listed in the database and in that case it will never help in finding the virus. Sometimes some normal TSRs perform actions that have been called suspicious, which is likely to give the false positives. Also some viruses can avoid this defense by getting activated earlier in boot sequence than the behavior monitors. Furthermore, these monitors can be modified by viruses if machines does not have hardware memory protection.

## 2.4 Heuristics scanner

Heuristic scanner can find unknown viruses and threats, because it looks at characteristics of a file. Also it checks the code statically and dynamically and determines the chances of infection. Sometimes it can find many new viruses before they get chance to execute. This technique is working on the probabilities of a file being infected. Therefore sometimes an innocent file may be placed in the list of infected files.

## 3. PATTERN MATCHING ALGORITHM

In signature scanning technique pattern matching is a bottleneck and performance is important. To improve the performance of system, an efficient fast pattern-matching algorithm could be used to search file for a set of signatures. For these reasons this virus detection tool uses a fast pattern-matching algorithm Boyer-Moore-Horspool (BMH). It has better performance than the Boyer-Moore algorithm, which is normally used in some popular virus detection softwares and much better than an efficient sequential pattern searching algorithm.

## 3.1 Pattern matching problem

Let there is a text ***T*** of length ***n*** and a pattern ***P*** of length ***m***. The problem is to find the location of pattern ***P*** in text ***T***, or there exists that ***P*** in ***T*** or not.

## 3.2 The Boyer-Moore-Horspool algorithm

The Boyer-Moore algorithm and Turbo Boyer-Moore algorithm are commonly used and considered to be the most efficient pattern-matching algorithms. Both these methods are preprocessed and kept in tables [7].

Though theoretically, it seems advantageous to use BM algorithm, there are many computational steps that are very costly. Therefore costly computational steps nullify the cost saved by skipping characters. This also holds true for the function, which computes the size comparisons in the skip tables.

So Horspool proposed Boyer-Moore-Horspool algorithm, which is a simplified form of the Boyer-Moore algorithm [8]. It uses only bad-character skip table. Baeza-Yates showed that the BMH algorithm is the best in terms of average case performance for all alphabet sizes and almost all pattern lengths.

Among all pattern-matching algorithms tested here, the BMH algorithm showed the best performance in time. It is relatively simple to implement. Though it's 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.

### *3.2.1 Implementation*

To implement the BMH algorithm, first two position indicators are required, ***j*** is set up for 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 letter-by-letter comparison from letter $P_{m-1}$, these comparisons are performed between text $T_k$ and pattern $P_j$. After each successful comparison both $j$ and $k$ are decremented. It continues as long as characters match and as long as there remains uncompared 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 $d$ 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.

This algorithm can be written this way:

```
while i < n  do  // a window is defined
        j=m-1
        k=i
        while  j >= 0 and  T[k] = P[j]  do
               j - -
               k - -
        end while
        if   j < 0  then
             report the occurrence of the pattern
        end if
        i = i + D[t_i]    //shift the window on the right
End while
```

To determine the shift distance $d$, the pattern is preprocessed: for each character $a \in A$, and a distance $d_s$ is computed. Where $A$ is the alphabet size. All the preprocessing steps can be written as:

```
For ∀a ∈ A do
     D [a]=m
End for
For i=0 to m-2 do
   D [P_i]=m- (i+1)
End for
```

## 4. DESCRIPTION OF THE SIGNATURE SCANNER

The tool consists of two main components: a Signature database and an Engine that scans files for viruses against signatures kept in the database. They are complementary to each other and cannot work independently.

### 4.1 Implementation

The first step to implement signature scanner is to build the signature database of all the viruses known till today. While in the second step, actual virus searching is done for the viruses stored in signature database.

#### *4.1.1 Signature database*

It is a database of uniquely identifiable signatures that a virus contains. The signature for an executable virus typically is a series of machine code bytes that a virus always contains. In this every virus record has following fields:

1. Virus signature in HEX
2. Type of virus (B: Boot sector, P: partition table, F: file virus)
3. Description of the virus

Whenever a new virus appears, database can be updated through a data entry program. It first asks user to enter the signature. It has to be in HEX and without commas and blank spaces. Next the type of virus has to be entered and finally the description of virus can be entered. The virus description includes the virus name, properties, comments, etc. After all the data has been verified, it is saved to the signature database.

#### *4.1.2 Virus detection engine*

It scans boot sector, partition table and files of all types for viruses. The scanner starts by reading information about viruses from the signature file. Now it knows the particular sequence of code and is looking for an exact match, which will identify the code as a virus.
To keep the scanning speed fast, Boyer-Moore-Horspool algorithm has been used, which is a very fast exact pattern-matching algorithm. For efficiency point of view, an array stores the first nibble of all signatures, and if the first nibble matches while scanning the executables, then only further matching with the respective signature is done.

The database file contains, besides the signature and description of the virus, its type as well. That is, whether it is a Boot sector, Partition or a File type of virus. Depending on the value of the type field, an array of structures representing these virus types is created to represent each virus record. If user wants to check all EXE files in C drive, signatures of all the file viruses are dumped into the array of structures. After that signature by signature all executable files are searched for. The same procedure is followed for boot sector and partition table also.

Every file will be scanned from first byte to last byte against the signature database. If certain anomalous patterns are detected, it will notify the user.

## 5 PERFORMANCE ANALYSES

### 5.1 Measures

All algorithms have been implemented in C. The target text was divided into slices of 1024 characters, except the last one, which might have fewer characters. All measurements were done in an incremental manner growing in steps from the size of one slice to the whole target size. All algorithms were tested for the various patterns.

### 5.2 Search for boot sector viruses

The first test was done in boot sector using the boot sector virus signatures. In this case size of target text was 512 bytes. The difference in performance of all BM and its variants was subtle, because of the smaller size of target text. But BMH is certainly faster than the sequential algorithm.

## 5.3 Search for partition table viruses

In the second test partition table of hard disk was searched for partition table type virus signatures. Since here also the target text size was 512 bytes, the same as that of the boot sector, so same results were obtained.

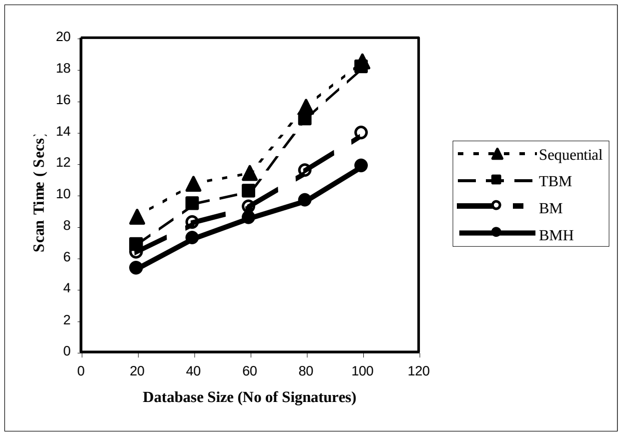
## 5.4 Search for file type viruses

In this test, 1127 files occupying 1.5 GB space were searched. Table 5.1 and figure 5.1 shows the performance of all the algorithms by varying database size.

**Table 5.1 Performance on the basis of Signature Database Size**

| Database Size (No. of Patterns) | Sequential Algorithm (Sec) | TBM (Sec) | BM (Sec) | BMH (Sec) |
|---|---|---|---|---|
| 20 | 8.6 | 6.8 | 6.3 | 5.3 |
| 40 | 10.7 | 9.4 | 8.2 | 7.2 |
| 60 | 11.4 | 10.2 | 9.2 | 8.5 |
| 80 | 15.6 | 14.8 | 11.5 | 9.6 |
| 100 | 18.5 | 18.1 | 13.9 | 11.8 |

**Figure 5.1 Performance graph on the basis of Signature Database size**



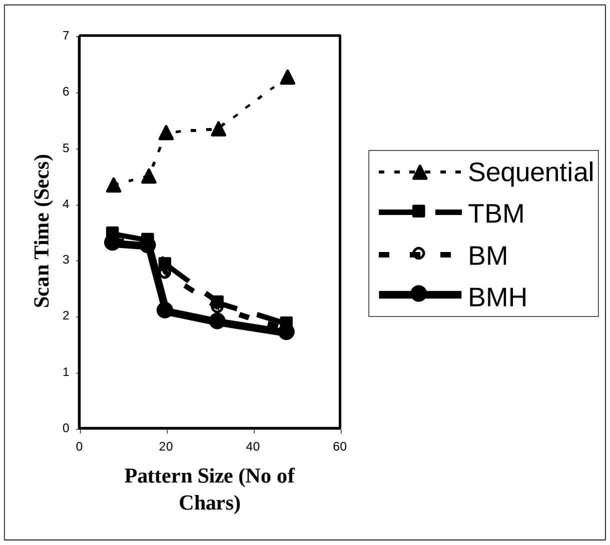
## 5.5 Performance according to pattern size

Sequential algorithm that does not use a skip table to optimize the shift function takes same time for all pattern sizes. This does not apply to the Boyer-Moore algorithm and its variants. In these algorithms, in case of mismatch, larger pattern size means longer the skip shift and therefore, faster the algorithm. Nevertheless, the BMH algorithm is still fastest among all algorithms tested. The overall results of all algorithms tested are presented in table 5.2 and figure 5.2. This table emphasizes the extreme speed of the BMH algorithm and the slight advantage to long patterns. It shows also that the sequential algorithm, compared with the BMH algorithm, performs less well with long patterns than with short ones.

**Table 5.2 Performance according to the pattern size**

| Pattern Size (No of Chars) | Sequential Algorithm (Sec) | TBM (Sec) | BM (Sec) | BMH (Sec) |
|---|---|---|---|---|
| 8 | 4.34 | 3.46 | 3.35 | 3.29 |
| 16 | 4.50 | 3.35 | 3.29 | 3.24 |
| 20 | 5.27 | 2.91 | 2.75 | 2.08 |
| 32 | 5.34 | 2.23 | 2.14 | 1.89 |
| 48 | 6.26 | 1.86 | 1.75 | 1.70 |

**Figure 5.2 Performance graph on the basis of the pattern size**



## 6. CONCLUSION AND FUTURE WORK

Signature scanners are evaluated by two criteria: how fast they scan and how well they detect the viruses. Several techniques are used to keep a handle on performance. First, signatures are classified by the type of infection they represent like boot sector, partition table or file type. Through a process of elimination, when a particular file is scanned, only the signatures that pertain to that file type is used to keep scan times down. For example, a boot sector signature would not be used to scan a file. If a signature matches, with the *Virus Detected* warning, *the name of file, the offset* at which it is found in the file and *the description* of the virus is displayed on the screen.

The implementation demonstrated here provided good results within acceptable time. By carefully using the BMH algorithm, the performance of virus detection system is improved as compared to the performance of most commonly used Boyer Moore pattern searching algorithm-based system. Considering the growing amount of text, that needs to be handled by a virus detection system, the BMH algorithm is worth implementing.

The initialization overhead of reading in the patterns from signature database and storing them in the internal structure of a virus is very small. The scanner has the potential to scan an almost infinite range of different file types. In practice, however, not all file types need to be scanned because some types of file, e.g. ASCII text files are not capable of being virus carriers. It is very easy to use. Simply run the tool and it provides concise results. It has options describing which disk, files, or directories to scan.

This is a prototype implementation of scanner; therefore there is a lot of work that could still be done from both a research point of view and from a commercial point of view. Like more efficient file reading routines can be developed that reads the files directly rather than reading it into intermediary buffers. Currently the signature database has 100 signatures, but to use it in the practical world all the existing virus signatures have to be maintained in the database.

## 7. ACKNOWLEDGMENTS

Every person is different with a treasure of unique potential stored in her/him and this potential is detected first by a good teacher (guide). It is difficult task, but I am going to give it a try, and if any name is not below, I assure that my gratitude is no less them than for those listed below.

I am highly grateful to my guide Prof.(Dr.) S.P.Tripathi (HoD) , Department of Computer Science and Engineering , Institute of Engineering and Technology, Lucknow, whose continued support and valuable suggestion throughout my M.Tech. Education and thesis work .It is my privilege to express my deep sense of gratitude to Prof N.C. Sharma , Additional Director and HoD(IT), Lord Krishna College of Engineering , Ghaziabad , who will always be my mentor in life. I am deeply in debited to the faculty member of K.I.E.T, Ghaziabad, especially Mrs. Nimisha Goel who was always a source of knowledge and inspiration to me. I am grateful to my father Mr. F. Lal and my mother Mrs Savitri for providing innovative and encouraging environment to work on this paper.

## 8. REFERENCES

[1] H.M. Mahmoud, R.T. Smythe, and M. Regnier.Analysis of boyer-moore-horspool string-matching heuristic. Random Structures and Algorithms, 1997, 10(1-2):169-186.

[2] http://www.extremetech.com/article2/0,1697,1154648,00.asp

[3] Morton Swimmer, Baudouin Le Charlier, and Abdelaziz Mounji. "Dynamic Detection and Classification of Computer Viruses Using Behavior Patterns". Proceedings of the 1995 Virus Bulletin Conference. 1995. Page 75

[4] Pathology of computer viruses March-April 2001. http://www.leavenworth.army.mil/milrev/download/English/MarApr01/deal.pdf.

[5] R. Nigel Horspool, Practical Fast Searching in Strings, Software Practice and Experience, 1980, 10: 501-506

[6] Robert S. Boyer and J Strother Moore, A Fast String Search Algorithm, CACM, October 1977, 20(10): 762-772.

[7] www.boll.ch/fortinet/assets/AntiVirus.pdf

[8] www.cs.wright.edu/~pmateti/Courses/499/Viruses/

[9] Yisrael Radai. "Checksumming Techniques for Anti-Viral Purposes". Proceedings of the Fifth International Computer Virus & Security Conference. 1992. Page 783.