Multi-pattern string matching algorithms comparison for intrusion detection system

Article *in* AIP Conference Proceedings · December 2014

DOI: 10.1063/1.4903558

Awsan A. Hasan

13 PUBLICATIONS **66** CITATIONS

SEE PROFILE

Nur'Aini Abdul Rashid

91 PUBLICATIONS **1,583** CITATIONS

SEE PROFILE

Atheer Akram Abdulrazzaq

UNIVERSITY OF INFORMATION TECHNOLOGY & COMMUNICATIONS

29 PUBLICATIONS **80** CITATIONS

SEE PROFILE

Multi-pattern string matching algorithms comparison for intrusion detection system

Awsan A. Hasan, Nur'Aini Abdul Rashid, and Atheer A. Abdulrazzaq

Citation: AIP Conference Proceedings **1635**, 27 (2014); doi: 10.1063/1.4903558
View online: http://dx.doi.org/10.1063/1.4903558
View Table of Contents: http://scitation.aip.org/content/aip/proceeding/aipcp/1635?ver=pdfcov
Published by the AIP Publishing

Multi-Pattern String Matching Algorithms Comparison for Intrusion Detection System

Awsan A. Hasan[a], Nur' Aini Abdul Rashid[b] and Atheer A. Abdulrazzaq[c]

[a,b,c] *School of Computer Sciences, Universiti Sains Malaysia, 11800, Penang, Malaysia*

Abstract. Computer networks are developing exponentially and running at high speeds. With the increasing number of Internet users, computers have become the preferred target for complex attacks that require complex analyses to be detected. The Intrusion detection system (IDS) is created and turned into an important part of any modern network to protect the network from attacks. The IDS relies on string matching algorithms to identify network attacks, but these string matching algorithms consume a considerable amount of IDS processing time, thereby slows down the IDS performance. A new algorithm that can overcome the weakness of the IDS needs to be developed. Improving the multi-pattern matching algorithm ensure that an IDS can work properly and the limitations can be overcome. In this paper, we perform a comparison between our three multi-pattern matching algorithms; MP-KR, MPHQS and MPH-BMH with their corresponding original algorithms Kr, QS and BMH respectively. The experiments show that MPH–QS performs best among the proposed algorithms, followed by MPH–BMH, and MP–KR is the slowest. MPH–QS detects a large number of signature patterns in short time compared to other two algorithms. This finding can prove that the multi-pattern matching algorithms are more efficient in high-speed networks.

Keywords: Intrusion detection system, High speed network, Signature threats, Multi-pattern matching.
PACS: 06.60.Mr, 07.05.Kf

INTRODUCTION

Nowadays, the Internet users and their related data are increasing gradually. The user's data are transferred within many networks in high speed network. Among this revolution, the curiosity of the attacker community is increased to compromise the user's data using many tools such as Trojan, viruses, spy tools and so many other tools. The needs for a system to protect the users' data become imperative. As security is considered an important issue in any modern network, The IDS is created to perform the needed security action along with the Firewall and the Anti-virus. The IDS have two main approaches; signature based and the anomaly based [1]. This paper focus only on the signature based IDS that relies on the string matching algorithms. The string matching is inspecting individual packets in network traffic and determining whether these packets are infected by any intrusion activities. However, these string matching algorithms consume a considerable amount of IDS processing time, thereby slow down the IDS performance [2, 3]. Developing an efficient multi-pattern algorithm can overcome the IDS limitations [4].

In this paper, we compare the performance of our previous proposed multi-pattern string matching algorithms MP-KR, MPH- QS [5] and MPH-BMH [6] against their original algorithms.

STRING MATCHING

String matching is an important algorithm in computer science. It is applied to many fields such as bioinformatics, web search engine and artificial intelligence. The algorithms are also used in network applications such as IDS [7, 8]. There are two types of string matching; single-pattern matching and multi-pattern matching.

International Conference on Quantitative Sciences and Its Applications (ICOQSIA 2014)
AIP Conf. Proc. 1635, 27-34 (2014); doi: 10.1063/1.4903558
© 2014 AIP Publishing LLC 978-0-7354-1274-3/$30.00

Single-Pattern Matching

In single-pattern matching, algorithms search for only one pattern in a text string every time. The search consumes time depending on the shift value technique performed by the algorithm [2, 7]. Single-pattern matching has been widely used in network applications such as detecting spam, viruses, and signature attacks [9]. Examples of single-pattern matching that have been used in IDS are Boyer-Moore algorithm [10] and Boyer-Moore Horspool algorithm [11].

Multi-Pattern Matching

In multi-pattern matching algorithm, a set of patterns are matched with a text string at the same time. The search in this case will be faster than that in single-pattern matching [2, 7, 8]. Examples of multi-pattern matching used in IDS are Aho–Corasick [12] and Wu-Manber [13]. Recently, many network applications such as IDS moved to multi-pattern matching because it can support a fast search for signatures or viruses when a large number of packets are matched against patterns [9]. The multi-pattern matching idea adopted in this paper is inspired by the fact that no single-pattern matching algorithm can properly work in network security applications without lose any security issues [3, 14, 15].

RELATED WORK

There are many existing string matching algorithms that are used in IDS. Aho-crosik (AC) is a multi-pattern string matching algorithm based on finite-state machine. This algorithm is one of the most efficient algorithms that have been applied to Snort IDS. The problem with this algorithm is that it cannot process a large number of incoming packets and a large set of rule patterns properly. It can only process one character input in each transition cycle. Hence, this algorithm slows down Snort IDS performance in high speed networks [2, 16, 17].

Boyer-Moore Horspool algorithm (BMH) is a single pattern matching algorithm. BM is one of the early algorithms used in the detection engine of Snort IDS, which has a sub-linear performance on average. The disadvantage of this algorithm is its inability to work properly with long pattern length, whereas it works fast with a short pattern length. BM is replaced because its speed is ineffective in high-speed networks, where BM does not support multi-pattern matching [11, 18, 7].

Quick search (QS) algorithm [19] is a single pattern matching algorithm. This algorithm is simple to implement and it can work quickly, especially with short and medium pattern lengths. QS performs a few attempts in each comparison, and it exhibits fast shifting of the pattern after each attempt.

Karp-Rabin (KR) algorithm [20] is an algorithm that based on the hash function, which reduces the required time during character comparison [18, 8]. It can work efficiently with very short pattern length, because the KR algorithm does not depend on shift tables to perform pattern shifting [21]. The hash function of KR algorithm is used by many algorithms such as WM algorithm that used in IDS [13].

Wu and Manber (WM) and modified WM algorithms (MWM) [22] are multi-pattern matching. The shift value of WM depends on the pattern length and consequently on the length of the shortest pattern presented. In this case, the short length pattern can reduce the efficiency of WM algorithm [22]. Therefore, MWM algorithm is developed to solve the problems inherited from the original WM algorithm and to process patterns with a short length using reasonable memory space. The drawback of the MWM algorithm is its inability to work properly with a pattern length of less than two bytes, but it can work well with a pattern length of two bytes and above [16, 14, 22].

PERFORM COMPARISONS

Three experiments are performed for 300 seconds for each proposed algorithm against its original algorithm: MP–KR against KR, MPH–QS against QS, and MPH–BMH against BMH. The experiments are performed on an Intel Core i5 2.3 GHz CPU with 4 GB of RAM, 3 MB L2 cache, and Windows 7 operating system. Different trace files of DEFCON 8 and 11 dataset [23] and DARPA 2000 dataset [24] are used as well as fixed number of 1000 patterns of Snort [25] are used. The experiments involve recording the number of packets that are scanned by each algorithm in 300 seconds and the number of intrusion alert that are matched between the incoming packet and the Snort pattern in 300 seconds. Finally, the fourth experiment records the running time for all proposed algorithms.

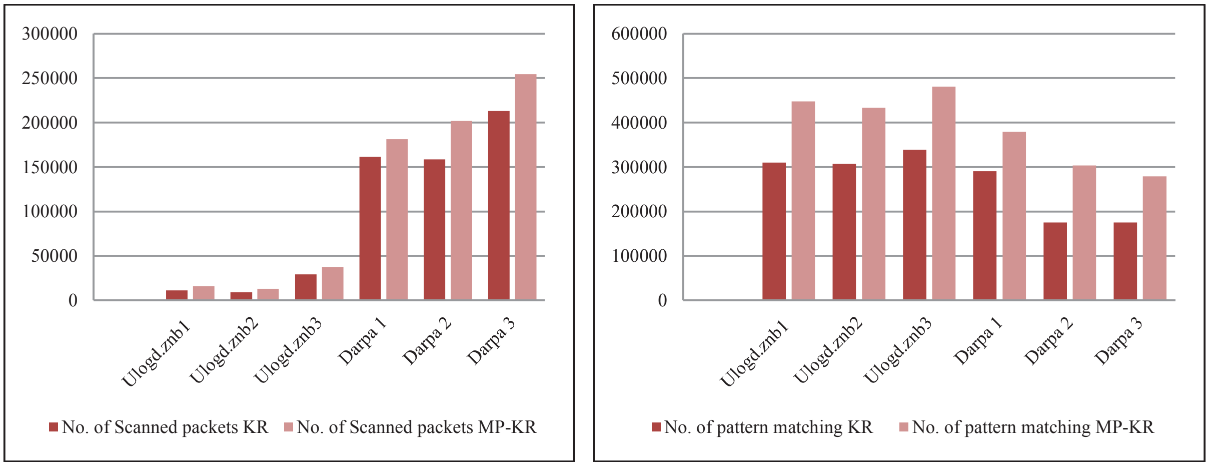
Comparison between MP-KR and KR

The first experiment is compared the performance of MP-KR and KR algorithms for 300 seconds as shown in Table 1.

TABLE 1. Performance of KR algorithm and MP-KR algorithm in 300 second

Trace file	File size MB	Total No. of packets	No. of Scanned packets KR	No. of Scanned packets MP-KR	Enhance ratio % MP-KR over KR	Number of pattern matching KR	Number of pattern matching MP-KR
Ulogd.znb1	133	402833	11100	15911	43.3	309850	447702
Ulogd.znb2	109	314775	9240	13223	43.1	307030	433131
Ulogd.znb3	205	639957	29243	37574	28.4	338582	481164
Darpa 1	382	1753377	161665	181177	12	290718	379168
Darpa 2	493	1807060	158779	201907	27.1	175075	303616
Darpa 3	527	1947815	213046	254322	19.3	175412	278781

Table 1 clearly shows that the performance of MP-KR algorithm is better than KR algorithm in terms of the number of scanned packets in 300 second as shown in Fig. 1(a) and the number of pattern matched that detected (Intrusion Alert) in 300 second as shown in Fig. 1(b). This result is caused by the multi-pattern matching quality of the MP-KR algorithm, which means it can search for a group of patterns in each comparison with the stream of payload content. By contrast, the KR algorithm, which is single-pattern matching, searches for only one pattern in each comparison. Therefore, the MP–KR algorithm outperforms the KR algorithm and accelerates searches for the required patterns. The MP–KR algorithm shows a better ratio of enhancement than the KR algorithm. However, the ratio of enhancement fluctuates in each trace file. This ratio depends on the number of packets that each algorithm can scan in each comparison for 300 seconds, which will be explained in the last section of this paper.



(a) (b)

FIGURE 1. Number of scanned packets in KR and MP-KR algorithms in 300 second (a) and number of pattern matched in KR and MP-KR algorithms in 300 second (b)

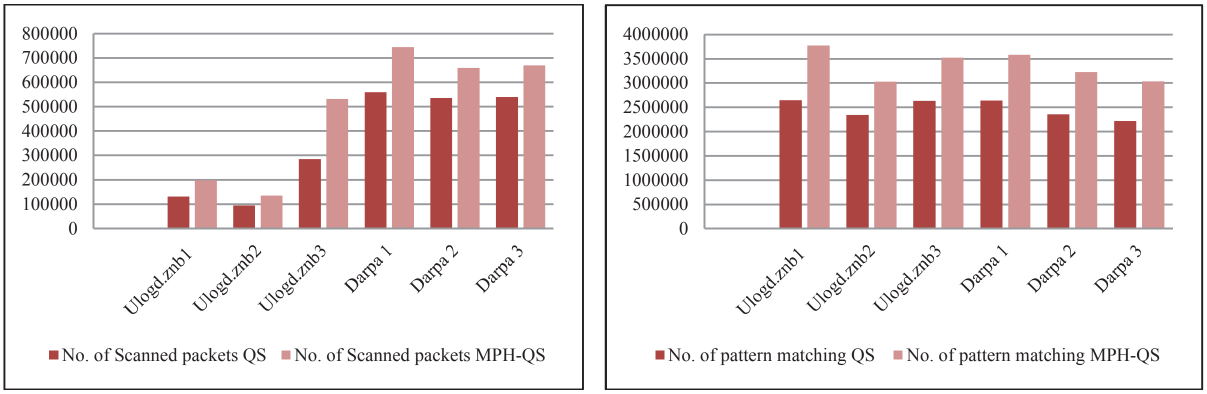
Comparison between MPH-QS and QS

The second experiment is compared the performance of MPH-QS and QS algorithms for 300 seconds as shown in Table 2.

TABLE 2. Performance of QS algorithm and MPH-QS algorithm for 300 second

Trace file	File size MB	Total No. of packets	No. of Scanned packets QS	No. of Scanned packets MPH-QS	Enhance ratio % MPH-QS over QS	Number of pattern matching QS	Number of pattern matching MPH-QS
Ulogd.znb1	133	402833	131189	197833	50.7	2646635	3773981
Ulogd.znb2	109	314775	94543	134761	42.5	2344504	3031956
Ulogd.znb3	205	639957	284569	531890	86.9	2635960	3523927
Darpa 1	382	1753377	559902	744039	32.8	2642182	3582564
Darpa 2	493	1807060	535460	658909	23	2354810	3223193
Darpa 3	527	1947815	539664	669240	24	2220805	3036431

Table 2 clearly shows that the performance of MPH-QS algorithm is better than QS algorithm in terms of the number of scanned packet in 300 second as shown in Fig. 2(a) and the number of pattern matched that is detected in 300 second as shown in Fig. 2(b). This result is due to the fact that MPH-QS algorithm works as a multi-pattern algorithm that can search for a group of patterns in each comparison. The MPH–QS algorithm also uses the hash function to reduce the character comparisons in each attempt, whereas the QS algorithm searches for one pattern only in each comparison between the pattern and the stream of payload data, and it does not use the hash function.



(a) (b)

FIGURE 2. Number of scanned packets in QS and MPH-QS algorithms in 300 second (a) and number of pattern matched in QS and MPH-QS algorithms in 300 second (b)

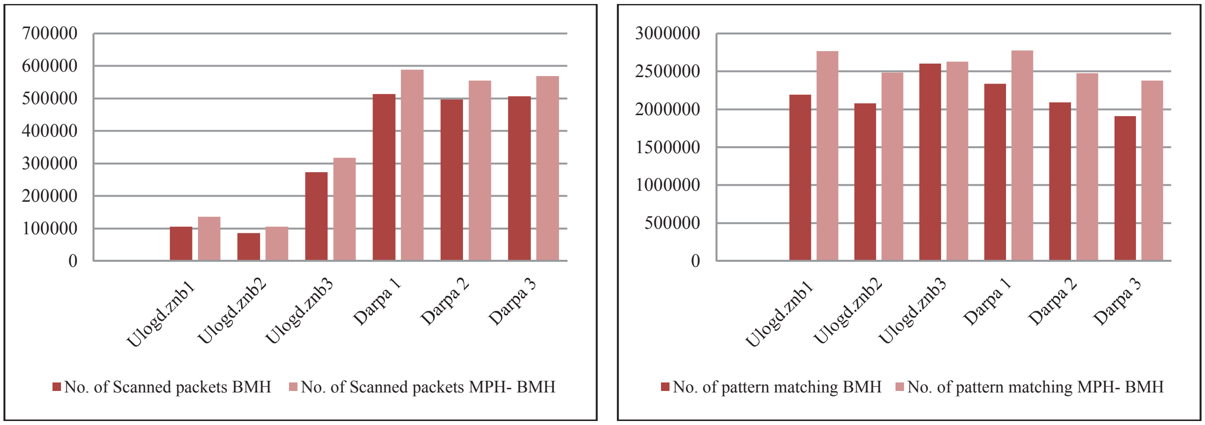
Comparison between MPH-BMH and BMH

The third experiment is compared the performance of MPH-BMH and BMH algorithms for 300 seconds as shown in Table 3.

TABLE 3. Performance of BMH algorithm and MPH-BMH algorithm for 300 second

Trace file	File size MB	Total No. of packets	No. of Scanned packets BMH	No. of Scanned packets MPH-BMH	Enhance ratio % MPH- BMH over BMH	Number of pattern matching BMH	Number of pattern matching MPH- BMH
Ulogd.znb1	133	402833	106021	135803	28	2192508	2766998
Ulogd.znb2	109	314775	85542	105902	23.8	2079642	2481861
Ulogd.znb3	205	639957	273565	316968	15.8	2601831	2626654
Darpa 1	382	1753377	513829	588687	14.5	2335954	2774700
Darpa 2	493	1807060	496753	554811	11.6	2089253	2476106
Darpa 3	527	1947815	506570	568189	12.1	1907735	2377373

Table 3 clearly shows that the performance of MPH-BMH algorithm is faster than BMH algorithm in terms of the number of scanned packets as shown in Fig. 3(a) as well as the number of pattern matched in 300 second as shown in Fig. 3(b). This result is caused by the multi-pattern function of the MPH–BMH algorithm, which means it can search for a group of patterns in each comparison by utilizing the hash function to reduce the number of character comparisons in each attempt.

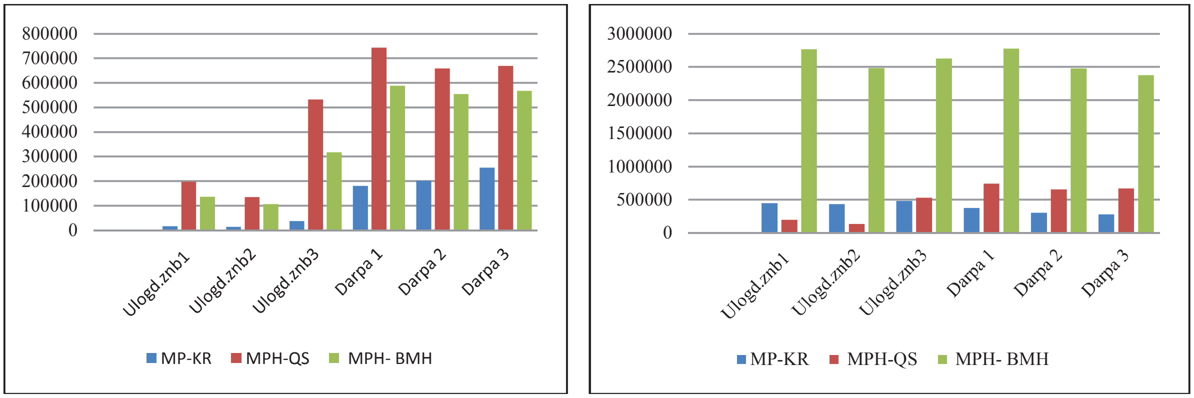


(a) (b)

FIGURE 3. Number of scanned packets in BMH and MPH- BMH algorithms in 300 second (a) and number of pattern matched in BMH and MPH- BMH algorithms in 300 second (b)

Comparison between MP-KR, MPH-QS, MPH-BMH

The previous experiments clearly show that the MPH–QS algorithm is the best among the proposed algorithms, in terms of the number of scanned packets in 300 second as shown in Fig. 4(a) as well as the number of pattern matched that detected in 300 second as shown in Fig. 4(b). The MP–KR algorithm is the slowest algorithm.



(a) (b)

FIGURE 4. Number of scanned packets in MP-KR, MPH-QS and MPH- BMH algorithms in 300 second (a) and number of pattern matched in MP-KR, MPH-QS and MPH- BMH algorithms in 300 second (b)

The structure of QS as well as its enhanced version MPH-QS are enabled them to work very fast with small and medium pattern length. Besides, MPH-QS uses only bad character table and calculate the hash function for all patterns only for one time as well as it performs the rehash for the new window text after each attempt. This can speed up the performance of MPH-QS compared to other algorithms.

The experiments show that each proposed algorithm is faster than its corresponding original algorithm in terms of the number of scanned packets and matched patterns that are detected in 300 seconds. A comparison between the algorithms, MP-KR and MPH-QS show a good ratio of enhancement over their corresponding algorithms KR and QS respectively and MPH–BMH is the worst as shown in Fig. 5. This ratio depends on the capability of the proposed algorithm and its corresponding algorithm to scan the packets in 300 seconds as well as the content of each trace file.

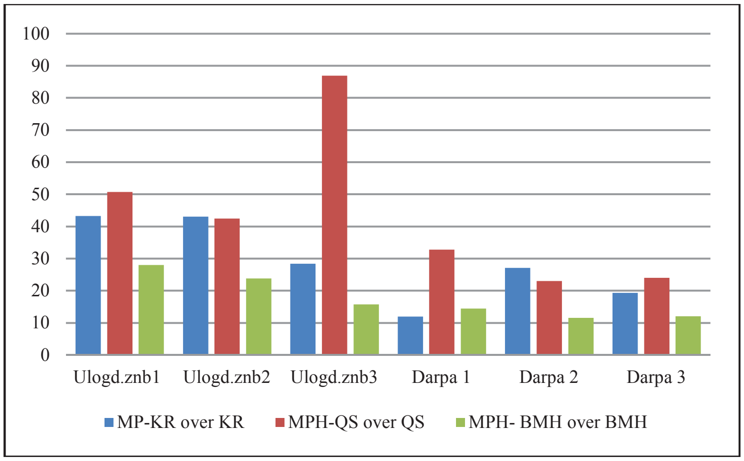


FIGURE 5. Ratio of enhancement in 300 second

The fourth experiment records the running time of the MP-KR, MPH-QS and MPH-BMH algorithms by using a fixed number of 1000 patterns and different number of trace files as shown in Table 4.

TABLE 4. Running time in seconds for MP-KR, MPH-QS and MPH-BMH algorithms

Trace file	File size MB	MP-KR	MPH-QS	MPH-BMH
Red 2.5	11.9	49.3	6.3	9.6
Orange 3.2	80.6	972.2	130.2	165.2
Ulogd.znb2	109	1352.3	173.9	232.5
Ulogd.znb3	205	3268.3	417.6	541.1
Darpa 4	358	6858.7	918.9	1159.2
Darpa 3	527	9634	1260	1650.9

Table 4 shows that the runtime increases when the trace file size increases in all algorithms. MPH–QS is the best algorithm because of its shorter runtime. It is followed by the MPH-MBH algorithm as shown in Fig. 6.

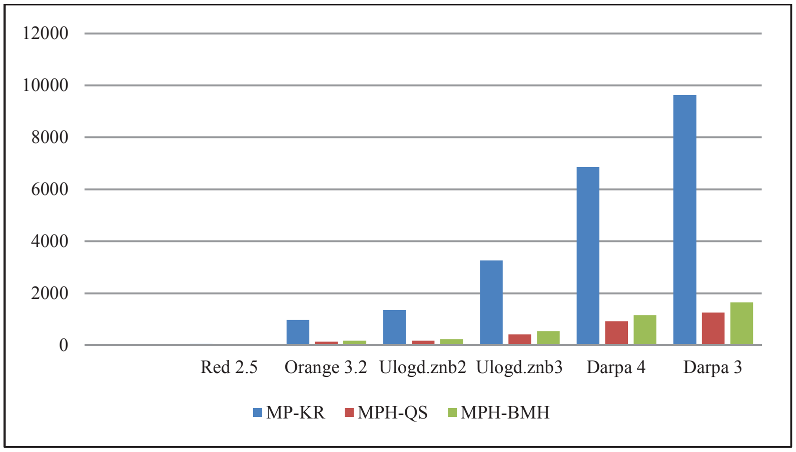


FIGURE 6. Running time in seconds

Factors that Influences the Performance of String Matching in IDS

The previous experiments showed a ratio of results fluctuation in all trace files. The performances of string matching algorithms are affected by many factors. This section explains these factors.

- The structure of the algorithms where, each algorithm has a different pre-processing phase as well as a different searching phase that consumed more time in one algorithm compared to other algorithm. For example, the pre-pressing phase of MP-KR is faster than that of MPH-QS and MPH-BMH. By contrast, the searching phase of MPH-QS and MPH-BMH are very fast compared to MP-KR searching phase. Noted that MP-KR, MPH-QS and MPH-BMH algorithms have different method of comparisons and different tables that used to shift the patterns after each comparison.
- Each string matching algorithm is affected by the length of the payload data and the number of patterns. MP–KR performs the shifting for the pattern group by one location only after each comparison. Thus, if the length of the window text is long, then reaching the end of the text window takes a long time. This analysis also applies to the KR algorithm. MPH–BMH and MPH–QS perform the shifting to the next comparison based on the value in the bad character table and, consequently, perform fewer character comparisons.
- The structure of datasets that are used in IDS is completely different from that of other datasets that are used in other fields, such as DAN, Protein, and English datasets. In these datasets, the string matching algorithm directly reads the text from the dataset, and then it starts to search for the required patterns. These datasets are not encrypted and do not contain signature attacks that intend to slow down string matching performance. By contrast, in IDS, an extra task consumes extra time before the search for patterns commences. The IDS first reads the network packets and extracts the payload data. Then, it concatenates a group of payload data to form a long text. Finally, the IDS starts to search for the required patterns [26].
- In IDS the number of scanned packets in each algorithm and the ratio of enhancement fluctuate in each trace file because the content of each packet payload in each trace file differs from the content in the other packet payload. Some packet payloads contain a small amount of data, but these packets are counted as one packet from the total number of packets in each trace file. Therefore, the actual data in the payload by which the string matching algorithm reads and performs the matching will vary from one trace file to another [27].
- Finally, some packets hold encrypted data for security purposes. They are used by governments or by a third party or in some cases by attackers to conduct intrusion activities. Consequently, the string matching algorithms are unable to read the content of the payload data, which are encrypted, and the string matching skips or does not recognize them [28].

CONCLUSION

The IDS depends on the string matching algorithms to perform the detection of the network threats. But the traditional single-pattern matching algorithms cannot work properly in high-speed networks. The moving to the multi-pattern matching can accelerate the performance of IDS. In this paper, we performed a comparison between three multi-pattern matching algorithms for IDS. The experiments show that MPH–QS algorithm is the best in term of the number of pattern matched, followed by MPH–BMH. The MP–KR algorithm is the slowest algorithm. There are many factors that affect the performance of the string matching algorithms in IDS such as the structure of algorithm and dataset, the size of data, the security status of the packets and the content of data.

ACKNOWLEDGMENTS

Research reported here is pursued under the Exploratory Research Grant Scheme (ERGS) by Ministry of Higher Education (Malaysia) and Universiti Sains Malaysia for ”Parallel Exact String Matching Algorithm For Intrusion Detection System” [203/PKOMP/6730074].

REFERENCES

1. W. Kim, O-R. Jeong, C. Kim and J. So. Information Systems. 36, 675 (2011).
2. K. Huang, D. Zhang and Z. Qin. Computer & Communications. 33, 1785 (2010).
3. H. Liu. Advanced Materials Research. 129, 1410 (2010).

4. T-F. Sheu, N-F. Huang, and H-P. Lee. Dependable and Secure Computing. 7, 175 (2010).
5. A. A. Hasan, N. Abdul Rashid, M. Abu-Hashem and A. A. Abdulrazzaq. Global Journal on Technology. 4, 511 (2013).
6. A. A. Hasan, N. Abdul Rashid, M. Abu-Hashem and A. A. Abdulrazzaq. Lecture Notes on Information Theory. 1, 69 (2013).
7. J. Yuan, J. Zheng and S. Ding, in: *Third International Symposium on Intelligent Information Technology and Security Informatics* (IEEE, Jinggangshan, 2010), pp. 599-603.
8. C. Khancome and V. Boonjing, in: *Ninth International Conference on Information Technology: New Generations* (IEEE, Las Vegas, NV, 2012), pp. 195-200.
9. A.M. Alshahrani and M.I. Khalil, in: *World Congress on Computer and Information Technology* (IEEE, Sousse, 2013), pp. 1-4.
10. R.S. Boyer and J.S. Moore. Comm. ACM. 20, 762 (1977).
11. R.N. Horspool. Software-Practice and Experience. 10, 501 (1980).
12. A.V. Aho and M.J. Corasick. Comm. ACM. 18, 333 (1975).
13. S. Wu and U. Manber, Tech. Rep. No. TR94-17, 1994 (University of Arizona).
14. S. Antonatos, K.G. Anagnostakis, E.P. Markatos and M. Polychronakis, in: *Proceedings of the International Symposium on Applications and the Internet* (IEEE, 2004), pp. 208-215.
15. M. Fisk and G. Varghese, Tech. Rep. No. CS2001-0670 (updated version), 2002 (University of California).
16. M. Norton, http://pdf.aminer.org/000/309/890/optimizing_pattern_matching.pdf.
17. M. M. Zhang, Y. Sun and J.Z. Wang. Appl. Math. 7, 755 (2013).
18. C. Charras and T. Lecroq, *Handbook of Exact String Matching Algorithms*, (King’s College Publications, 2004).
19. D. M. Sunday. Comm. ACM. 33, 132 (1990).
20. R. M. Karp and M.O. Rabin. IBM Journal of Research and Development. 31, 249 (1987).
21. M. Góngora-Blandó and M. Vargas-Lombardo. Journal of Information Security. 3, 314 (2012).
22. Z. Qiang, in: *International Conference on Intelligent Computing and Intelligent Systems* (IEEE, Xiamen, 2010), pp. 124-127.
23. DEFCON Capture−the−Flag Game Traces, http://cctf.shmoo.com.
24. DARPA, http://www.ll.mit.edu/mission/communications/cyber/CSTcorpora/ideval/data/2000data.html
25. Snort, http://www.snort.org.
26. J. o. Nehinbe. Information Security and Digital Forensics. 41,111 (2010).
27. N. Subramanian and S. Rao, in: *First International Conference on Computational Intelligence, Communication Systems and Networks* (IEEE, Indore, 2009), pp. 296-301.
28. Z. M. Fadlullah, T. Taleb, A.V Vasilakos, M. Guizani and N. Kato. IEEE/ACM Transactions. 18, 1234 (2010).