A ring representation of data for immune-based intrusion detection systems

Nguyen Thanh Hai, Trinh Van Ha, Nguyen Van Truong

Abstract—**Many applications in real world use linear data structures, such as string or vector. The linear data type may omit the information at its edges, especially for flow data. In this paper, we present a ring representation technique for data. Our experiment results on flow-based Network data show that the new approach archives prominent classification rates.**

***Index Terms*—Ring data, linear data, classification, intrusion, immune system.**

I. INTRODUCTION

As we known, many applications use two types of linear data representation: string and real-valued vector. For both popular types, representations are linear structure of symbols or numbers. They may omit information at the edges (the begin and the end) of these structures and lead to reduce classification rates.

Our idea of new data presentaion originates from an earlier empirical implementation on binary ring-based strings. Using our ring-typed data representation shows that both detection rates and accuracy rate are higher than that of the linear ones, while false alarm rates are quite similar. So we use ring structures instead of linear ones for more exact classification.

In this paper, Artificial Immune System (AIS) [1], a multidisciplinary research area that combines the principles of immunology and computation, is used for experiments on the proposed representation.

AIS is inspired by the observation of the behaviors and the interaction of normal component of biological systems - the self -and abnormal ones - the nonself. Positive Selection Algorithm (PSA) is a popular model of AIS mainly designed for one-class learning problems such as anomaly detection.

The outline of a typical PSA contains two stages [1]. In the generation stage (Fig. 1), the detectors are generated by some random process and censored by trying to match given self samples taken from set S. collection of detectors (or detector set) is used to verify whether an incoming data instance is self or nonself.

Nguyen Thanh Hai, IT Center, Thai Nguyen University, Thai Nguyen City, Vietnam, Mobile No. +(84)-968550888

Trinh Van Ha, University of Information Technology and Communications, Thai Nguyen. Thai Nguyen City, Vietnam, Mobile No. +(84)-983454755

Nguyen Van Truong, Faculty of Mathematics, Thai Nguyen University of Education, Thai Nguyen City, Vietnam, Mobile No. +(84)-915016063

Those candidates that match are kept as detectors in set D and the rest are eliminated. In the detection stage (Fig. 2), the. If it matches any detector, it is claimed as self, otherwise it is nonself or an anomaly. This description is limited to some extent, but conveys the essential idea.

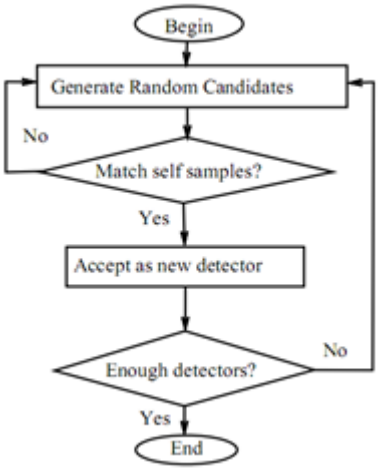

Fig. 1. Model of detector generation in PSA

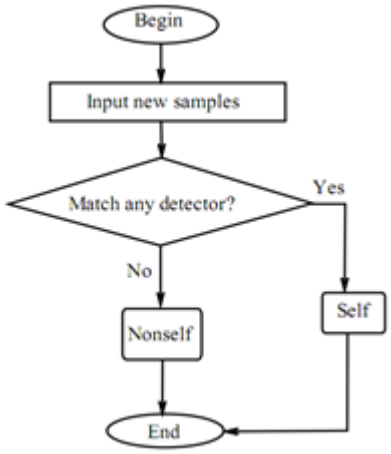

Fig. 2. Detection of new instances in PSA

II. BASIC TERMS AND DEFINITIONS

In PSAs, an essential component is the matching rule which determines the similarity between detectors and self samples (in the detector generation phase) and coming data instances (in the detection phase). Obviously, the matching rule is dependent on detector representation. In this paper, both self and nonself cells are represented as binary strings of fixed length. This representation is the most simple and popular representation of data in AISs, and other representations (such as real valued) could be reduced to binary [15, 16].

3.1 Strings

An alphabet Σ is nonempty and finite set of symbols. A string $s \in \Sigma^*$ is a sequence of symbols from Σ, and its length is denoted by $|s|$. A string is called empty string if its length equals 0. Given an index $i \in \{1, 2, \ldots, |s|\}$, then $s[i]$ is the

symbol at position i in s. Given two indices i and j, whenever $j \geq i$, then s[i . . . j] is the substring of s with length $j - i + 1$ that starts at position i and if $j < i$, then s[i . . . j] is the empty string.

We will use ring structures instead of linear ones for more exact classification. A simple solution for this process is to concatenate each string with its fist k bits. Each new linear string is a ring representation of its original one. Fig. 3 shows a ring representation (b) and its original string (a) with k = 3.

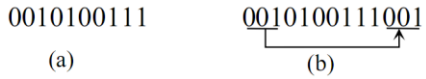

Fig. 3. A ring-based representation (b) of a string (a)

Given a set of strings $S \subset \Sigma^{\ell}$, a set $S_r \subset \Sigma^{\ell+r-1}$ includes ring representations of all strings in S by concatenate each string $s \in S$ with its fist $r - 1$ bits.

Note that we can easily apply the idea of ring strings for other data representations in AIS. One way to do this, for instance, is to create ring representations of other structures such as trees, automata, etc., from set S_r instead of S as usual. Our approaches can be implemented on any finite alphabet, but strings used in all examples are binary, $\Sigma = \{0, 1\}$, just for easy understanding.

2.2 R-chunk detectors

Given a set of strings $S \subset \Sigma^{\ell}$, a tuple (d, i) of a string $d \in \Sigma^{r}$, $r \leq \ell$, and an integer $i \in \{1, ..., \ell\}$ is called an r-chunk detector if there exists a $s \in S_r$ such that d matches s[i, . . . , i + r − 1]. We also use the notations: S_i = {(d, i), (d, i) is a positive r-chunk detector} is set of all positive r-chunk detectors at position i with respect to S_r, i = 1, . . . , ℓ.

Example 1. Let ℓ = 5 matching threshold r = 3. Suppose that we have the set of four strings S = {s1 = 00000; s2 = 10110; s3 = 10111; s4 = 11111}. S_r = {0000000; 1011010; 1011110; 1111111}. S1 = {(000,1); (101,1); (111,1)}, S2 = {(000,2); (011,2); (111,2)}, S3 = {(000,3); (110,3); (111,3)}, S4= {(000,4); (101,4); (111,4)}, S5 = {(000,5); (010,5); (110,5); (111,5)}.

Given a data set S of string with the same n features, length of binary representations of strings is calculated by equation $\ell = \sum_{i=1}^{n} k_i$ where k_i is the smallest integer such that $|D_i| < 2^{k_i}$, and D_i is value domain of feature i.

2.3 Proposed algorithms

Given ℓ, r, a normal dataset $N \subset \Sigma^{\ell}$, an abnormal dataset $A \subset \Sigma^{\ell}$. Algorithm 1 creates trees used in the algorithm 2. A new data instance $s \in \Sigma^{\ell}$ is detected as self or nonself by Algorithm 2.

Algorithm 1: Algorithm to generate trees

1: procedure TreesGeneration(N, r, T_A)

Input: A set of self strings $N \subseteq \Sigma^{\ell}$, a matching threshold $r \in \{1, \ldots, \ell\}$.

Output: A set T_A of $\ell - r + 1$ prefix trees presenting trees.

2: for i = 1, ..., ℓ do
3: Create an empty tree T_{Ni}
4: for all $s \in N_r$ do
5: for i = 1, ..., ℓ do
6: insert every s[i . . . i + r − 1] into T_{Ni}
7: for i = 1, ..., ℓ do
8: Create an empty tree T_{Ai}
9: for all $s \in A_r$ do
10: for i = 1, ..., ℓ do
11: insert every s[i . . . i + r − 1] into T_{Ai}

Algorithm 2: Algorithm PSA to detect if a new data instance $s \in \Sigma^{\ell}$ is self or nonself.

1: procedure PSA(N, A, s, r, T_N , T_A)

Input: A set of nonself strings $N \subseteq \Sigma^{\ell}$, a set of self strings $A \subseteq \Sigma^{\ell}$, an unlabeled string $s \in \Sigma^{\ell}$, a matching threshold $r \in \{1, 2, \ldots, \ell\}$.

Output: A set T_A of $\ell - r + 1$ prefix trees presenting nonself trees, a set T_N of $\ell - r + 1$ prefix trees presenting self trees, a label of s (self or nonself).

2: TreesGeneration(N, r, T_A)
3: $d_1 = d_2 = d_3 = 0$
4: Create a string s_r as ring representation of s
5: for i = 1, ..., ℓ do
6: $s_0 = s_r[i \ldots i + r - 1]$
7: if $s_0 \notin T_{Ni}$ then
8: $d_1 = d_1 + 1$
9: if $s_0 \notin T_{Ai}$ then
10: $d_2 = d_2 + 1$
11: if Leaf(s_0, T_{Ni}) < Leaf(s_0, T_{Ai}).t_1 then
12: $d_3 = d_3 + 1$
13: if $d_1 > t_2$ then output s is nonself
14: else if $d_2 > t_3$ then output s is self
15: else if $d_3.t_4 > \ell$ then output s is nonself
16: else output s is self

III. EXPERIMENTS

3.1 Datasets

In our experiments, we use two popular flow-based datasets: NetFlow [31] and TU [36]. The flow-based NetFlow is generated from packet-based DARPA dataset [38]. This dataset focuses only on flows to a specific port and an IP address which receives the most number of attacks. It contains all 129,571 traffics (including attacks) to and from victims. Each flow in the datasets has 10 fields: Source IP, Destination IP, Source Port, Destination Port, Packets, Octets, Start Time, End Time, Flags, and Proto.

Other labeled flow-based data set was captured by monitoring a honey-pot hosted in the University of Twente network, so we call it TU dataset. This dataset has three categories: malicious, unknown and side-effect. It has 14170132 flows which are mostly of malicious nature.

The NetFlow dataset is used for experiment 1 with features Packets, Octets, Duration, Scr port, Dst port, Flags and IP protocol. The UT dataset is used for experiment 2 with features Packets, Octets, Duration and Flags.

3.2 Performance

Table 1 illustrates the results of two experiments. PSA cannot train with unlabeled data, so we only use 10% labelled dataset, 5998 attack flows and 595 normal flows, for training phase in experiment 1. Meanwhile two other algorithms use 100% dataset for training, in which 90% unlabelled dataset used for S4VM. Results from the experiment strongly

confirm the efficiency of PSA in comparison with methods proposed in [35].

In experiment 2, we compare PSA performance with some other algorithms run on WEKA 3 with their default parameters. The detection rate of PSA is highest among the other algorithms, while accuracy is remarkable high. Some SVMs achieved admirable accuracies with 100%, but their FARs are very high with approximately or even 100%. Poor FARs of these algorithms means that they cannot verify the nature of benign traffic in the dataset, but PSA can.

Table 1. Comparison between PSA and other algorithms

Algorithms	ACC	DR	FAR
Experiment 1			
PSA (10% labelled data)	0.9781	0.9626	0.0157
S4VM (10% labelled data) [35]	0.9196	-	0.0384
EBP - based MLP [35]	0.9655	-	0.0315
Experiment 2			
PSA	0.9257	0.9974	0.1945
Naive Bayes	0.6832	0.9972	0.8668
SVM (linear)	0.7315	1.0000	0.7185
SVM (polynomial)	0.7143	0.8788	0.5613
SVM (RBF)	0.6263	1.0000	0.9998
SVM (sigmoid)	0.6263	1.0000	1.0000
Deep learning	0.8106	0.8201	*Nan*

Recent works focus on deep learning, so we produced another experiment using deep learning. In the experiment, we use 01 hidden layer, the number of unit on each layer is 100, activation function is Relu, optimization function is Adam, learning rate is 0.001, and number of epochs is 200. We can see results from the last row of table 1 that all ACC and DR is lower than those of our PSA.

IV. Conclusions

The major contribution of this study is to propose a ring representation instead of linear one for better performance in terms of both detection rate and accuracy rate.

To verify the effectiveness of the proposed approach, two different datasets are adopted to validate this approach. The results from four experiments indicate that the proposed approach can produce competitive and consistent classifying performance on real datasets. Moreover, results from experiment 2 with only 10% of training dataset confirm that PSA can detect anomalies in a small amount of labelled data.

The algorithm can be applied to the data set that has following characters: 1- Having strings with equal number of features, and 2 - Value domain of all features is discret. However, if the domain is continuous, such as real numbers, then a good quantification may be used before applying our algorithm.

In the future, we are planning to combine our algorithms with some machine learning methods to have better detection performance, reduce training time. Moreover, it would be interesting to further develop technique how to choose optimal parameters as well as to integrate them in new objective functions. We also would like to apply the method for other security problems with larger datasets.

To the best of our knowledge, there has not been any published attempt in using ring type of data instead of linear one to attain more exact classification.

Aknowledgement

This research is based upon work supported in part by Thai Nguyen University for university's research; code number DH2017-TN01-03.

References

[1] H. Yang, T. Li, X. Hu, F. Wang, Y. Zou, "A survey of artificial immune system based intrusion detection", The Scientific World Journal, 2014.
[2] J. N. Stephen Northcutt, "Network Intrusion Detection", New Riders, 2003.
[3] B. Li, J. Springer, G. Bebis, M. Hadi Gunes, "Review: A survey of network flow applications, Journal of Network and Computer Application", vol. 36, no. 2, pp. 567–581, 2013.
[4] S. X. Wu, W. Banzhaf, "The use of computational intelligence in intrusion detection systems: A review", Applied Soft Computing, vol. 10, no. 1, pp. 1–35, 2010.
[5] D. Dasgupta, F. Gonzalez, "An immunity-based technique to characterize intrusions in computer networks", IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, pp. 281–291, 2002.
[6] K. B. Sim, D. W. Lee, "Modeling of Positive Selection for the Development of a Computer Immune System and a Self-Recognition Algorithm", International Journal of Control, Automation, and Systems, vol. 1, no. 4, pp. 453–458, 2003.
[7] X. Hang, H. Dai, "Applying both positive and negative selection to supervised learning for anomaly detection", in: Conference on Genetic and Evolutionary Computation (GECCO), pp. 345–352, 2005.
[8] L. X. Peng, Y. F. Chen, "Positive selection-inspired anomaly detection model with artificial immune", in: International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), pp. 56–59, 2014.
[9] Z. Fuyong, Q. Deyu, "Run-time malware detection based on positive selection", Journal in Computer Virology, vol. 7, no. 4, pp. 267–277, 2011.
[10] Y. Tan, "Anti-Spam Techniques Based on Artificial Immune System", CRC Press, 2016.
[11] Z. Fuyong, Q. Deyu, "A positive selection algorithm for classification", Journal Computational Information Systems, vol. 7, pp. 207–215, 2012.
[12] P. H. Pisani, A. C. Lorena, A. C. Carvalho, "Adaptive positive selection for keystroke dynamics", J. Intell. Robotics Syst., vol. 80, no. 1, pp. 277–293, 2015.
[13] J. Vykopal, "Flow-based Brute-force Attack Detection in Large and High-speed Networks", Dissertations, Masaryk University, 2013.
[14] A. Sperotto, G. Schaffrath, R. Sadre, C. Morariu, A. Pras, B. Stiller, "An Overview of IP Flow-Based Intrusion Detection", IEEE Communications Surveys Tutorials, vol. 12, no. 3, pp. 343–356, 2010.
[15] F. Gonzalez, D. Dasgupta, J. Gomez, "The effect of binary matching rules in negative selection", in: Genetic and Evolutionary Computation Conference (GECCO), pp. 195–206, 2003.
[16] Z. Ji, D. Dasgupta, "Revisiting negative selection algorithms", Evolutionary Computation, vol. 15, no. 2, pp. 223–251, 2007.
[17] C. Guo, Y. J. Zhou, Y. Ping, S. S. Luo, Y. P. Lai, Z. K. Zhang, "Efficient intrusion detection using representative instances", Computers & Security, Part B, vol. 39, pp. 255 – 267, 2013.
[18] D. Y. Yeung, Y. Ding, "Host-based intrusion detection using dynamic and static behavioral models", Pattern Recognition, vol. 36, no. 1, pp. 229–243, 2003.
[19] C. A. Mart´ınez, G. I. Echeverri, A. G. C. Sanz, "Malware detection based on cloud computing integrating intrusion ontology representation", in: IEEE Latin-American Conference on Communications, pp. 1–6, 2010.
[20] P. Winter, E. Hermann, M. Zeilinger, "Inductive intrusion detection in flow-based network data using one-class support vector machines", in: International Conference on New Technologies, Mobility and Security (NTMS), pp. 1–5, 2011.
[21] V. D. Kotov, V. Vasilyev, "Immune model based approach for network intrusion detection", in: International Conference on Security of Information and Networks, pp. 233–237, 2010.
[22] T. S. Sobh, W. M. Mostafa, "A cooperative immunological approach for detecting network anomaly", Applied Soft Computing, vol. 11, no. 1, pp. 1275 –1283, 2011.
[23] Zeng, Jie and Liu, Xiaojie and Li, Tao and Li, Guiyang and Li, Haibo and Zeng, Jinquan, "A novel intrusion detection approach learned from the change of antibody concentration in biological immune response", Applied Intelligence, vol. 35, no. 1, pp. 41–62, 2011.
[24] S. B. Inadyuti Dutt, I. Maitra, "Intrusion detection system using artificial immune system", International Journal of Computer Applications, vol. 144, no. 12, pp. 19–22, 2016.

[25] K. S. Desale, R. Ade, “Genetic algorithm based feature selection approach for effective intrusion detection system”, in: International Conference on Computer Communication and Informatics, pp. 1–6, 2015.
[26] R. Bronte, H. Shahriar, H. M. Haddad, “A signature-based intrusion detection system for web applications based on genetic algorithm”, in: International Conference on Security of Information and Networks, pp.32–39, 2016.
[27] H. B. M. Mehmod, Tahirand Rais, “Ant Colony Optimization and Feature Selection for Intrusion Detection”, Springer International Publishing, pp. 305–312, 2016.
[28] X. Yang, Z. Hui, Intrusion detection alarm filtering technology based on ant colony clustering algorithm”, in: “International Conference on Intelligent Systems Design and Engineering Applications, pp. 470–473, 2015.
[29] M. Zolotukhin, T. Hmlinen, T. Kokkonen, J. Siltanen, “Online detection of anomalous network flows with soft clustering”, in: International Conference on New Technologies, Mobility and Security, pp. 1–5, 2015.
[30] Y. Sawaya, A. Kubota, Y. Miyake, “Detection of attackers in services using anomalous host behavior based on traffic flow statistics”, in: International Symposium on Applications and the Internet, pp. 353–359, 2011.
[31] Q. A. Tran, F. Jiang, J. Hu, “A real-time netFlow-based intrusion detection system with improved BBNN and high-frequency field programmable gate arrays”, in: IEEE International Conference on Trust, Security and Privacy in Computing and Communications, pp. 201–208, 2012.
[32] M. Sheikhan, Z. Jadidi, “Flow-based anomaly detection in high-speed links using modified GSA-optimized neural network”, Neural Computing and Applications, vol. 24, no. 3, pp. 599–611, 2014.
[33] Z. Jadidi, V. Muthukkumarasamy, E. Sithirasenan, “Metaheuristic algorithms based flow anomaly detector”, in: Asia-Pacific Conference on Communications, pp. 717–722, 2013.
[34] M. J. Chapple, T. E. Wright, R. M. Winding, “Flow anomaly detection in firewalled networks, in: Securecomm and Workshops”, pp. 1–6, 2006.
[35] Z. Jadidi, V. Muthukkumarasamy, E. Sithirasenan, K. Singh, “Flow-based anomaly detection using semisupervised learning”, in: International Conference on Signal Processing and Communication Systems, pp. 1–5, 2015.
[36] A. Sperotto, R. Sadre, F. Vliet, A. Pras, “A labeled data set for flow-based intrusion detection”, in: IEEE International Workshop on IP Operations and Management, pp. 39–50, 2009.
[37] R. Hofstede, A. Pras, A. Sperotto and G. D. Rodosek, “Flow-Based Compromise Detection: Lessons Learned”, IEEE Security & Privacy, vol. 16, no. 1, pp. 82-89, 2018.
[38] DARPA Dataset, https://www.ll.mit.edu/r-d/datasets, [accessed 21-May- 2019]

BIOGRAPHY

Nguyen Thanh Hai is an expert in the IT Center at Thai Nguyen University. He finished his master course on Computer science at Thai Nguyen University in 2012 He has taught a wide variety of courses for UG students and guided several projects. He has published several papers in National Journals. His research interests are network architecture and network services.

Trinh Van Ha is a lecturer in the Faculty of Information Technology - University of Information Technology and Communications, Thai Nguyen. He finished his master course on Computer science at Thai Nguyen University in 2008. He has taught a wide variety of courses for UG students and guided several projects. He has published several papers in National Journals. His research interests are network architecture and computer security.

Nguyen Van Truong is a lecturer in the Faculty of Mathematics at Thai Nguyen University of Education. He is currently a PhD student at Institute of Information Technology, Vietnamese Academy of Science and Technology. He has taught a wide variety of courses for UG students and guided several projects. He has published several papers in National Journals & International Conferences. His research interests are embedded systems and artificial immune systems.