2018 IEEE Symposium on Security and Privacy Workshops

Evaluating Insider Threat Detection Workflow Using Supervised and Unsupervised Learning

Duc C. Le
Faculty of Computer Science, Dalhousie University
Halifax, NS, Canada
Email: lcd@dal.ca

A. Nur Zincir-Heywood
Faculty of Computer Science, Dalhousie University
Halifax, NS, Canada
Email: zincir@cs.dal.ca

***Abstract*—Insider threat is a prominent cyber-security danger faced by organizations and companies. In this research, we study and evaluate an insider threat detection workflow using supervised and unsupervised learning algorithms. To this end, we study data exploration and analysis, anomaly detection and malicious behaviour classification on a publicly available data set. We evaluate several supervised and unsupervised learning algorithms - HMM, SOM, and DT - using this workflow.**

I. Introduction

Insider threat is a major problem for many companies across industries and government organizations. It refers to malicious activities, such as information system sabotage, intellectual property theft, fraud, disclosure of classified information, as well as unintentional threats introduced inadvertently by careless use of computing resources by authorized user.

Due to the fact that insiders are knowledgeable about an organizational structure and security procedures, as well as authorized to use the computer systems, insider threat is one of the most costly types of attacks and hardest to detect. According to the 2017 CyberSecurity Watch Survey, while insider threats only accounted for 13 percent of cybercrimes against US organizations, they are 29 percent being the most costly incidents [1]. Moreover, the report also indicated that while half of the surveyed organizations monitor user activities, only one-third have a way to interpret user's behaviour and intent. Given the complex contextual combinations of activities in large organizations, where insiders' activities usually account for only a minuscule portion of recorded activities on an organizational information system, one can imagine the challenges in detecting insider threats.

This paper mainly focuses on analyzing and evaluating a workflow using supervised and unsupervised learning algorithms for insider threat detection. To this end, experiments, from data preprocessing to machine learning model training, are conducted on a publicly available insider threat dataset provided by CERT [2]. On this dataset, we have employed Self Organizing Maps (SOM) and compared it against Hidden Markov Models (HMM) and C4.5 Decision Trees (DT) representing previous work from the literature [3], [4]. Our results show that SOM has good characteristics, from visualization to anomaly detection and classification, to provide data insights to the analyst for detecting insider threat. Additionally, we only require one SOM for the organization. This in return enables the insider threat detection workflow to scale better. The remainder of the paper is organized as follows. Section II summarizes the related work on insider threat detection. Section III discusses the methodology, whereas Section IV presents experiments and evaluation results. Finally, conclusions are drawn and future work are discussed in Section V.

II. Related Work

General literature reviews of insider threats and guidelines for preventing and identifying insider threats are presented in [5], [6]. In attempts to understand insiders' behaviours, some researchers approach the problem via psychological models and decision-making theories. In [7], Padayachee described the application of opportunity theories from the field of criminology in conceptualizing insider threats. Legg *et al* in [8] proposed a framework for modelling the insider threat problem based on behavioural and psychological observations. A reasoning structure based on the framework allows an analyst to build hypothesis trees describing potential insider threat from measurable states in different domains, such as human behaviours and organizational policies.

The amount of data acquired daily by an organization is enormous, making it essentially unsuitable for analyzing case by case or perhaps unmatchable by a set of predetermined frameworks. Hence, machine learning could find its application in the field for the ability to automatically learn from data and detect patterns characterizing malicious activities. One typical machine learning-based approach is modelling the normalcy in employee activities on a system as a baseline, and using that to detect the anomaly as the deviation from the baseline. Rashid *et al* in [3] applied Hidden Markov Model(s) to capture each user's normal weekly activity sequences and detect the deviation that may potentially indicate insider threats. In [9], Senator *et al* explored machine learning-based anomaly detection for detecting insider threats in the simulated corporate computer usage activities. They combined structural and semantic in-

DOI 10.1109/SPW.2018.00043

formation to detect malicious insider activities independently developed by red teams. They proposed a visual language for specifying elements, such as input data, features, algorithms and their connections, that are necessary for characterizing an anomaly. Gavai *et al* applied different machine learning-based methods on organizational activity data for anomaly and quitter detection, which possibly indicates underlying insider threats [4]. Goldberg *et al* presented a DARPA-supported anomaly detection system, PRODIGAL, that combines multiple machine learning-based anomaly detection techniques to support human analysts [10]. In general intrusion detection application, bio-inspired machine learning algorithms, such as artificial immune system, has been successfully applied [11]. More recently, in [12] Korczynski *et al* investigated the application of a bee-inspired method in a self-organizing, nonparametric distributed coordination framework for network intrusion early warning.

In this paper, we aim to describe an inclusive process in which data from multiple sources in a corporate environment is processed, and machine learning algorithms are applied for detecting the anomaly in general and insider threat in particular. The approach attempts to construct a broad overview of user activities based on different data formats and learning techniques, from modelling, visualization to classification.

III. Methodology

The principal interest of this work is to assess the capability of a bio-inspired machine learning technique, namely Self-Orgnaizing Maps, for detecting insider threats in a corporate network. Fig. 1 illustrates the workflow of the proposed approach. The workflow is designed to be modular and easily expandable for a wide range of corporate environments, data acquisition conditions, as well as learning and analysis methods. Similarly, with extracted data in different formats, any learning algorithm can be easily employed for supporting human analysts in different environments.

A. Data preprocessing

The first step in the workflow is collecting and preprocessing data. Typically, the data collected from an organization can be grouped in two main categories: (i) users' actions and operational information, and (ii) organization's structure and users' information. Data from the first category comes from different logging systems such as network traffic capture, firewall logs and other sensors' records. These are sources of dynamic data, which are generated perpetually and need to be collected periodically, if not constantly for analysis and detection systems. The second category of data represents static data sources, which can be users' personal information, role in the organization, and so on. In many cases, this category also consists of more complex data, such as psychometric and behavioural models of users. In most organizations, Lightweight Directory Access Protocol (LDAP) is used to maintain and store this data.

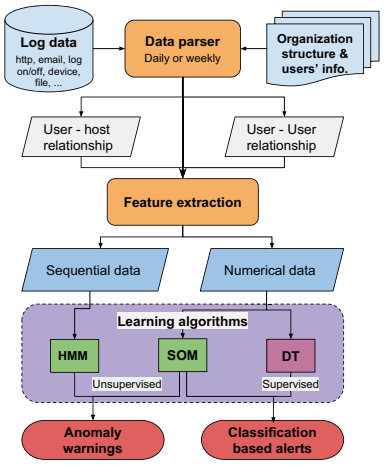


Figure 1. Workflow of the proposed system for insider threat/anomaly detection

Data from both categories can be parsed periodically, typically daily or weekly, depending on the organizational setup, amount of data and most importantly the timing requirements for detection systems. The data parser needs to fuse data correctly for each user or host, based on a set of properties, such as user ID, host ID, action ID, and time. This may very well represent a challenge in collecting data from multiple users/sources in large corporate environments. Further challenges can arise from the usage of encrypted [13], [14] or anonymized data [15] for designing and testing analytics systems.

To successfully derive meaningful information from multiple sources of data, another step may be required to obtain relationships in parsed data, such as user - user, and user - host relationships. Specifically, in the CERT dataset employed in this paper, only LDAP information is given for the second category of data. This leads to the analysis phase on parsed data to discover the assigned/authorized hosts, and the supervisor/subordinates of each user.

1) Feature extraction: Once data from different sources are aggregated, features are extracted for training and evaluation of machine learning algorithms. In this research, we extract numerical and sequential data. While numerical data, where each instance is represented by a fixed-length vector, is more common and widely applied in machine learning, sequential data with the inherent ordering structure may reveal interesting behaviours by considering each user's action in the relevant context.

Numerical data. Numerical features are exported to represent users' characteristics and activities during over a given time period. Two main categories of numerical features are user features and activity features. User features include each user's role, functional unit, department, psychomet-

ric scores, and employment status. The activity features are mostly extracted by counting the number of activities in each of the categories (log on/off, device connect/disconnect, file, email, http) over a given time period, such as `number of logins (after hours)`, `number of (external) emails`. On the other hand, the content of email, http, and file logs in the CERT dataset is synthesized from a set of randomly chosen words to represent a topic. This significantly reduces the effect of content analysis module in extracting meaningful features. Thus, they are not included in this research.

Sequential data. Sequential data summarizes the sequence of user's actions over a period of time. In the simplest form, the data sequence consists of an ordered list of actions taken by a user. For example, in the case of the CERT dataset, the sequential data feature set is {`log on`, `log off`, `device connect`, `disconnect`, `file`, `email`, `http`}. This results in variable length sequences of daily or weekly user actions. A more comprehensive set of features can also be extracted, with more information of each action, such as weekend login, or connect usb to a supervisor's machine. However, it is shown in [3] that the extended set of features does not improve the detection performance.

B. Learning algorithms

In this study, three algorithms - self-organizing map, hidden Markov model, and decision tree - are employed to learn and model the data to detect anomaly/insider threats. The aim is to evaluate both unsupervised and supervised learning algorithms for this purpose. While unsupervised learning algorithms are important for generating warnings of anomalous activities, supervised learning is more suitable for analyzing data with ground-truth. In many cases, the ground-truth comes from domain experts' knowledge embedded in the labels after analyzing the anomalous warnings.

1) Self-Organizing Map: SOM [16] is an unsupervised, competitive-learning based neural network based on how the human neural system works. The SOM produces a non-linear, ordered, low dimensional projection of data from multi-dimensional input space. The SOM consists of nodes that can act as decoders or detectors of their respective input space domains post training. The application of competitive learning and neighbourhood function in SOM provides a way to visualize high dimensional data in a 2-D space where topological properties are preserved.

As presented in Fig. 1, the SOM is trained and evaluated on the numerical features extracted in the previous step. Two different approaches for training the SOM are evaluated: (i) data representing all classes (normal and insider threat) is used to train the SOM, and (ii) only data representing normal user behaviours is used to train the SOM. When no label information for the training data is available, all data could be used to train the SOM. In this case, the SOM plays the role as a data clustering and visualization step to assist the human analyst. The first approach is also applicable when ground-truth for data from multiple classes (normal and insider threat) is available. In this case, ground-truth is used to label SOM nodes post training based on the best matching units (nodes on the map) for data in each class. Then, this labelled map can be used for mapping unseen test data.

On the other hand, when the ground truth for only one class, typically normal, is available for training the SOM, the second approach can be applied to model the data. In this case, the SOM acts as an anomaly detector post training, where new instances that are deviated from the profiled map of normal data are flagged for further inspection by the human analyst.

2) Hidden Markov model: HMM [17] is a statistical Markov model in which the states are hidden. Each hidden state emits a symbol in a set with probabilities before transitioning to a new state. This algorithm is particularly suited to model normal behaviours based on the extracted sequential data.

Basically, a HMM is trained to model each user's action sequence over a given time, in this work: weekly. Then for each of the user's new action sequence, the user's HMM is used to calculate the log probability of the sequence. The sequence is flagged as an anomaly for further analysis if the log probability value is larger than a threshold. If the action sequence is not flagged, or flag is cleared by an analyst, it is used in combination with the previous action sequence to train the user's HMM again. This approach is used for insider threat detection in [3]. In this work, we aim to evaluate the approach with a simpler feature set that is trained on only the most recent user data to be able to adapt to the shifts and drifts in the user's behaviours. For this purpose, we used the most recent two weeks of user data to train each HMM. We attempt to improve anomaly detection performance by introducing customized thresholds for different user groups.

Fig. 2 presents an example case demonstrating an anomaly detection system based on HMMs as introduced in [3]. As seen in the figure, there are four user HMM probability plots. Two of them are for normal users and two for malicious users (insider threats). HMM produces small log probabilities for normal user action sequences. Thus, a carefully selected threshold could help separating the anomalous data sequences from the normal ones. In the following, we will discuss how to select such thresholds.

3) Decision tree: DT is employed as the benchmark algorithm in this study for its popularity and interpretability. In particular, the decision tree is generated using C4.5 algorithm [18]. C4.5 is extended from the earlier ID3 algorithm developed by Ross Quinlan. C4.5 uses the concept of information entropy for creating an if-then rule at each tree node in order to build the tree. Labelled training data

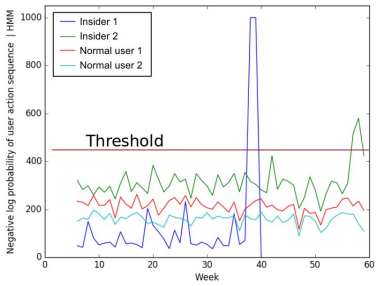


Figure 2. HMM log probability of four users' action sequences over time

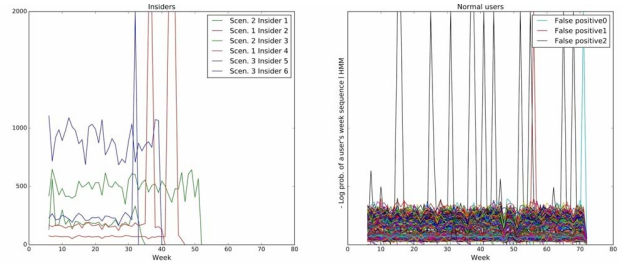


Figure 3. Insiders and normal users HMM log probs of week sequences.

for at least two classes is required. At each node of the tree, the data is split into subsets, where each contains only one or a few classes as the majority. The criteria is satisfied most effectively by choosing the feature and the split point that gives the highest normalized information gain. C4.5 algorithm then recurs on the subtree.

IV. Experiment and Results

As discussed earlier, our goal is to asses the capabilities of supervised and unsupervised learning in detecting insider threat on the publicly available CERT dataset. The performances of learning algorithms are measured using insider threat detection rate (DR), false positive rate (FPR), and accuracy (A).

A. CERT insider threat dataset

The CERT insider threat dataset[1] is a publicly available dataset for research, development, and testing of insider threat mitigation approaches [2]. The dataset simulates an organization with 1000 to 4000 employees, and consists of users' computer activities (log on/off, email, web, file and thumb drive connects), as well as organizational structure and user information. As described in [2], a number of different model types including topic models, behaviours models, and psychometric models are employed for generating the data as close to what is seen in the real-world as possible. Furthermore, the insider threat data provided is synthesized in the same form and scope as the normal data. There are totally 5 insider threat scenarios, ranging from data leaking, intellectual property thief to IT sabotage. In this paper, release 4.2 of the dataset is employed for designing and evaluating insider threat detection approaches. According to the dataset description, release 4.2 contains a significantly greater amount of insider threat incidents than other releases, which makes it possible for testing the proposed detection systems against a more diverse set of scenarios.

The data processing step is described in III-A. In summary, 18 months of data of 1000 users in the organization is extracted in the two data formats either weekly or daily. This results in 67173 and 328342 data instances, respectively. Although the release 4.2 of the CERT dataset was meant to have higher malicious instance density, insider threat instances only account for 0.39% and 0.29% of the weekly and daily preprocessed data, respectively. For SOM and DT, only the data from week 20 to the end is used, as the first 20 weeks does not contain any insider threats. However, all 72 weeks are used for HMM to be able to compare it with the previous work [3].

[1] https://www.cert.org/insider-threat/tools/index.cfm

B. Evaluation results

In this section, we present the results of the algorithms - HMM, SOM and DT - on the CERT insider threat dataset. It is noteworthy that SOM and HMM have complementary characteristics for data exploring and analysis, such as data modelling and visualization, without using the ground truth information during training. However, while the HMM-based method requires one model for each week data per user of the organization [3], there is only one SOM required per organization.

1) Hidden Markov model results: The HMM is trained using BaumWelch algorithm. Due to the time limitation, only weekly users' action sequences are used for training. The number of hidden states in HMM is set to 5, 9, 15, or 25. A log probability threshold for generating anomaly flags is chosen for each number of HMM hidden states to give the best DR-FPR balance. In our experiments, we observed that the thresholds were typically settled around 400.

Fig. 3 shows samples of log probabilities produced by HMMs (with 9 hidden states) of insiders and normal users' action sequences over the course of the dataset. It is easy to see that different scenarios of insider threats exhibit different HMM log probabilities, and the probability pattern of the second scenario (insider soliciting employment from a competitor and stealing company's confidential data) is much harder to differentiate from the normal patterns. On the other hand, while most of the normal users' probability patterns fall under the anomaly threshold, many false positives are caused by new actions in a new week, which has not been modelled by the HMM during training.

Fig. 4 shows the training time, ROC curves, and AUC of HMMs with different numbers of states. It is obvious that HMM with 15 states provides the best balance between the

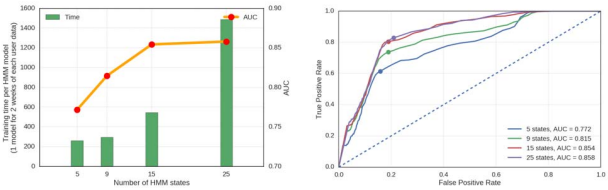


Figure 4. Training time, ROC, and AUC of HMMs with different numbers of states.

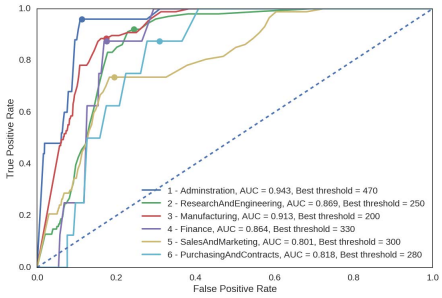


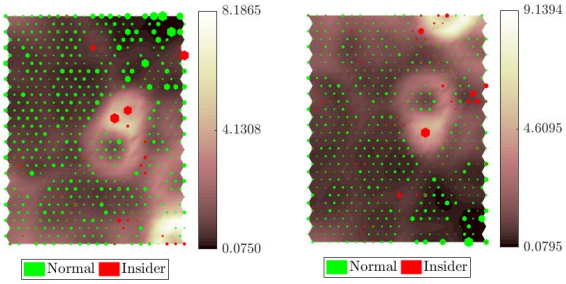
Figure 5. ROCs for different units with adjusted thresholds.

training time and the detection performance. It achieves 80% DR at the cost of 19% FPR.

Further investigation on HMMs of users from different functional units in the organization shows that adjusting the thresholds for each unit (department) may improve the performance. Using this intuition, a threshold adjustment scheme is introduced to find the best threshold for each unit, as presented in Figure 5. This helps improve the performance of HMM with 15 states, where DR increases from 80% to 86% at the expense of only 1% increase (19% to 20%) in the FPR.

The acquired results from HMM in this study are comparable with those reported in [3], without the need of a complicated training model. This implies that training HMMs on only most recent two weeks data seems to be enough to model user's behaviours. Moreover, this may allow the model to adapt better to the shift and drift in user's actions over time. On the other hand, the method [3] has several drawbacks. Firstly, a new HMM is required for each new weekly action sequence for each user in the organization. This process is time-consuming to train and evaluate over sequences of thousands of actions per user per week. Secondly, sequential data structure for training HMM is incapable of carrying details representing user's actions that are valuable for anomaly detection, such as irregular log in time, downloading files from unauthorized machines, etc.

2) Self-organizing map results: The SOM based approach is implemented in Matlab SOM Toolbox [19]. More details regarding the implementation and parameters can be found in [13]. Results from the SOM experiments on the data is



(a) Approach (i) (b) Approach (ii)

Figure 6. SOM hit maps of normal and insider data

presented in Table I. The hit maps of testing data on the SOMs trained using the first and second training approaches (section III-B1) are shown in Fig. 6. In the figure, the SOM nodes are the red and green hexagons, and the size each hexagon denotes the proportion of data that best matched the node.

While both of the SOM training approaches produce similar SOMs, the SOM node labelling processes is different, where labels from both classes are used in the first approach, and only normal data label is used in the second approach. This explains the difference in the performance metrics given in Table I.

On the other hand, one advantage of the SOM is that it provides a topographically preserved visualization of the training data. Hence, even when the label information is not available, the system analyst has a tool to support inspecting the data by groups, as regions projected in SOM map, and to identify what portion of data needs to be inspected in more detail. As shown in Fig. 6, the insiders' data instances (red hexagons) are concentrated in the lighter regions of the SOM map. The background color represents the distance between adjacent SOM nodes, where lighter the background color is, more different the adjacent SOM nodes are. This visually shows that the insider threat data exhibits different characteristics from the normal data. Hence, by inspecting the data mapping to the lighter region first, the analyst may be able to figure out the anomaly in the data and identify the the suspicious behaviours.

3) Decision tree results: The C4.5 results are obtained using Weka implementation. In the case of randomly sampled training data, 10-fold cross-validation results are presented in Table I. These results show that the decision tree achieves a high accuracy by concentrating on the normal behaviour, but is not able to learn the patterns to distinguish the minority class, i.e. the insider threat behaviours.

However, the performance of C4.5 from randomly sampled training data is significantly better. This confirms the observation that there is shifting and drifting in users' behaviours over time, and the traditional supervised learning model may not adapt well to the changes in data. This

Table I
SOM AND DT EVALUATION RESULTS

		Train on first 50%			Randomly sampled			Train Time
		DR	*FPR*	*A*	*DR*	*FPR*	*A*	
Weekly data								
SOM	*(i)*	77.88	7.77	92.16	79.75	9.19	90.74	93
	(ii)	15.04	11.31	88.33	25.32	7.50	92.05	83
C4.5		47.79	0.12	99.62	73.10	0.09	99.73	2.77
Daily data								
SOM	*(i)*	65.66	18.69	81.26	70.60	19.37	80.59	3522
	(ii)	34.64	84.76	84.60	24.02	8.73	90.98	3465
C4.5		42.77	0.02	99.87	82.51	0.01	99.93	89

suggests the application of adaptive learning algorithms for insider threat classification. It is also noteworthy that in real-world applications, a detection system must be trained on obtained data for classification of future observations. Hence, the results from training the algorithms on the first 50% of data better reflect real-world in this case.

Finally, since weekly data covers more behavioural information than the daily data, the results for all learning algorithms are generally better on the weekly than on the daily data. This shows the trade-off between detection results and the ability to detect malicious behaviours quickly. Hence, we suggest taking into account the trade-off when proposing novel insider detection approaches.

V. CONCLUSION

This paper presents the necessary steps, from data processing to pattern learning for employing both supervised and unsupervised learning algorithms in insider threat detection. Different learning algorithms showed promising results as well as unique characteristics for further studies. Based on our experimental results we observe that SOMs seem to provide best of both worlds in terms of DR, FPR and supporting the human analysts via visualization of the data.

Future work on the topic will focus on the use of sequential data in the numerical format. This potentially can combine the advantages of sequence-based learning methods and classification algorithms. Another notable research direction is to apply streaming data classification/exploration techniques for insider threat detection/discovery.

REFERENCES

[1] CSO, U.S. Secret Service, CERT Division of SRI-CMU, ForcePoint, "The 2017 U.S. State of Cybercrime Survey," IDG, Tech. Rep., 2017.

[2] J. Glasser and B. Lindauer, "Bridging the gap: A pragmatic approach to generating insider threat data," *2013 IEEE Security and Privacy Workshops*, pp. 98–104, 2013.

[3] T. Rashid, I. Agrafiotis, and J. R. Nurse, "A new take on detecting insider threats," in *International Workshop on Managing Insider Security Threats*, 2016, pp. 47–56.

[4] G. Gavai *et al.*, "Supervised and unsupervised methods to detect insider threat from enterprise social and online activity data," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, vol. 6, no. 4, pp. 47–63, December 2015.

[5] The CERT Insider Threat Center, "Common sense guide to mitigating insider threats, fifth edition," CERT, SRI, Carnegie Mellon University, Tech. Rep. CMU/SEI-2015-TR-010, 2016.

[6] National Cybersecurity and Communications Integration Center, "Combating the Insider Threat," The US Department of Homeland Security, Tech. Rep., 2014.

[7] K. Padayachee, "An assessment of opportunity-reducing techniques in information security: An insider threat perspective," *Decision Support Systems*, vol. 92, pp. 47–56, 2016.

[8] P. Legg *et al.*, "Towards a conceptual model and reasoning structure for insider threat detection," *JoWUA*, vol. 4, no. 4, pp. 20–37, 2013.

[9] T. E. Senator *et al.*, "Detecting insider threats in a real corporate database of computer usage activity," in *19th ACM SIGKDD*, 2013.

[10] H. G. Goldberg *et al.*, "Explaining and aggregating anomalies to detect insider threats," in *Annual Hawaii International Conference on System Sciences*, 2016, pp. 2739–2748.

[11] P. K. Harmer *et al.*, "An artificial immune system architecture for computer security applications," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 3, pp. 252–280, 2002.

[12] M. Korczynski *et al.*, "Hive oversight for network intrusion early warning using diamond: a bee-inspired method for fully distributed cyber defense," *IEEE Communications Magazine*, vol. 54, no. 6, pp. 60–67, June 2016.

[13] D. C. Le, A. N. Zincir-Heywood, and M. I. Heywood, "Data analytics on network traffic flows for botnet behaviour detection," in *2016 IEEE Symposium Series on Computational Intelligence, SSCI 2016*, 2017.

[14] F. Haddadi and A. N. Zincir-Heywood, "Benchmarking the effect of flow exporters and protocol filters on botnet traffic classification," *IEEE Systems Journal*, vol. 10, pp. 1390 – 1401, 2016.

[15] K. Shahbar and A. N. Zincir-Heywood, "Packet Momentum for Identificationof Anonymity Networks," *Journal of Cyber Security and Mobility*, vol. 6, pp. 27–56, 2017.

[16] T. Kohonen, *Self-Organizing Maps*, ser. Springer Series in Information Sciences, 2001, vol. 30.

[17] Z. Ghahramani, "Hidden markov models," in *An Introduction to Hidden Markov Models and Bayesian Networks*. World Scientific Publishing Co., Inc., 2002, pp. 9–42.

[18] J. R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., 1993.

[19] T. Kohonen, *MATLAB Implementations and Applications of the Self-Organizing Map*. Unigrafia Oy, Helsinki, Finland, 2014.