Chapter 8

MALICIOUS LOGIN DETECTION USING LONG SHORT-TERM MEMORY WITH AN ATTENTION MECHANISM

Yanna Wu, Fucheng Liu and Yu Wen

Abstract Advanced persistent threats routinely leverage lateral movements in networks to cause harm. In fact, lateral movements account for more than 80% of the time involved in attacks. Attackers typically use stolen credentials to make lateral movements. However, current detection methods are too coarse grained to detect lateral movements effectively because they focus on malicious users and hosts instead of abnormal log entries that indicate malicious logins.

This chapter proposes a malicious login detection method that focuses on attacks that steal credentials. The fine-grained method employs a temporal neural network embedding to learn host jumping representations. The learned host vectors and initialized attribute vectors in log entries are input to a long short-term memory with an attention mechanism for login feature extraction, which determines if logins are malicious. Experimental results demonstrate that the proposed method outperforms several baseline detection models.

Keywords: Malicious login detection, LSTM with attention mechanism

1. Introduction

Advanced persistent threats (APTs) have been the focus of considerable research [4, 8]. These threats manifest themselves as sustained and effective attacks against specific targets. A perpetrator typically compromises a host and adopts a hiding strategy to enter a sleep state. After the presence on the host is consolidated, information is collected about the targeted network. Lateral movements, the next crucial and time-consuming phase of the attack, involve attempts to progressively move to and control other machines in the network [24]. Advanced per-

sistent threat actors almost always perform lateral movements by stealing credentials [20]. Consequently, malicious login detection is vital to combating advanced persistent threats.

Some approaches for detecting lateral movements model logins in communication graphs that describe interactions between users and hosts [1, 4, 13]. However, these approaches primarily focus on partial properties of log entries (e.g., login relationships) to identify malicious users and hosts, which is a relatively coarse-grained detection strategy.

Other lateral movement detection techniques leverage machine learning to obtain better results [2, 18]. However, most techniques are limited by model interpretability. Additionally, some fine-grained detection methods (e.g., [5, 23]) do not consider interactions between hosts. These issues make it difficult to develop effective machine-learning-based classifiers that can detect lateral movements.

This chapter proposes a malicious login detection method that focuses on attacks that steal credentials. The method involves host representation learning and feature extraction from log files. Host representation learning engages a temporal neural network embedding model to learn the initial host vectors, enabling the translation of host login relationships to host vectors. The host vectors and initial expressions of other attributes are the inputs for log feature extraction. The proposed feature extraction model learns log entry vectors using long short-term memory (LSTM) coupled with an additional attention mechanism that enhances the extraction of information about important attributes. The log vectors are subsequently input to a multilayer perceptron (neural network with a full connection layer) that classifies them as malicious or benign.

The proposed malicious login detection method incorporates some novel features. Inspired by research in text classification, long short-term memory is employed to learn information about attributes and extract the meaning of a log file. Instead of merely detecting malicious hosts and users, each log entry is analyzed and classified as malicious or not, allowing for fine-grained detection. The attention mechanism emphasizes important attributes and strengthens the interpretability of the model. Additionally, since malicious logins occur between hosts, host interactions are considered using a temporal graph embedding to learn preferred host representations and integrate them in the log vectors. Experimental results demonstrate that the malicious login detection method has a false positive rate of just 0.002% and outperforms several state-of-the-art detection models.

2. Related Work

In recent years, considerable research has focused on advanced persistent threat detection, especially during the lateral movement phase. Siadati et al. [21] developed APT-Hunter, a visualization tool for exploring login data to discover patterns and detect malicious logins. They advocated the use of machine learning models to enhance feature extraction. Bai et al. [2] evaluated a number of supervised and unsupervised machine learning methods for detecting lateral movements, including Logistic Regression, Gaussian Naive Bayes, Decision Tree, Random Forest, LogitBoost, feed-forward neural networks and clustering methods. However, their results were modest because insufficient testing data was available.

Chen et al. [7] developed a network embedding approach that uses an autoencoder to detect lateral movements. However, the performance was limited by data imbalance – the available red team activity data was much less than normal activity data. Holt et al. [9] employed deep autoencoders to detect lateral movements. Three autoencoder models were developed that addressed the data imbalance problem to some extent, but the results were limited by the paucity of abnormal data. The three models achieved an average recall in excess of 92% and an average false positive rate less than 0.5%. Bohara et al. [4] focused on broader patterns of system interactions after attackers gain initial access to hosts. Their graph-based target system model yielded true positive and false positive rates of 88.7% and 14.1%, respectively.

Malicious logins are key components of cyber attacks. As a result, much research has focused on malicious login detection, especially in internal networks. Some detection approaches analyze user activities. Powell [16] modeled the normal daily behavior of users in behavior graphs; alerts were raised when deviations in user behavior were detected. Kent et al. [13] employed bipartite authentication graphs to analyze user behavior in enterprise networks. Amrouche et al. [1] applied knowledge discovery techniques to detect malicious login events. Other detection methods focus on host activities. For example, Chen et al. [7] analyzed host communications whereas Bian et al. [3] examined host authentication logs.

Regardless of whether the methods mentioned above focused on user or host activities, they detected anomalies using partial attributes in login information. To address this limitation, researchers have analyzed login events using rule-based [20] and machine-learning-based approaches [5, 23]. However, malicious login detection has been hindered by the lack of model interpretability. Additionally, the methods do not

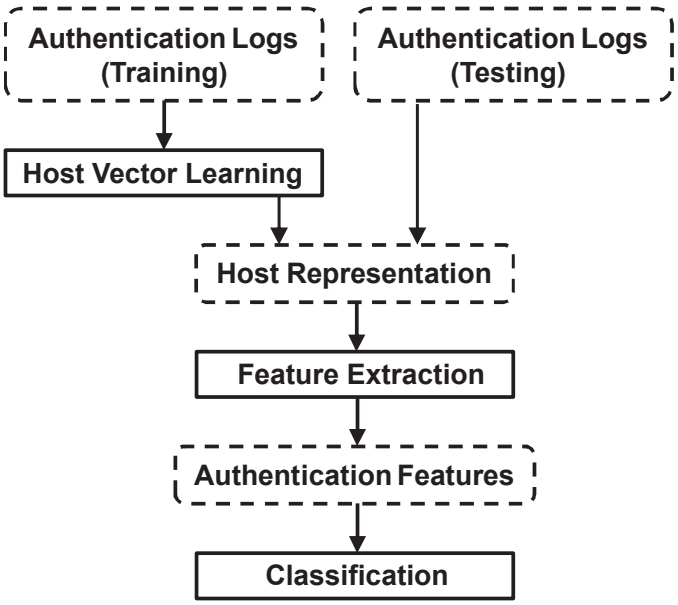

Figure 1. Proposed malicious login detection method.

focus on host interactions. In contrast, the proposed fine-grained malicious login detection method employs temporal neural network embedding to learn host jumping representations. The learned host vector and other initialized attribute vectors in log entries are then fed to a long short-term memory with an attention mechanism for login feature extraction, which helps determine whether or not logins are malicious.

3. Preliminaries

This section provides an overview of the malicious login detection method and its threat model.

3.1 Detection Method Overview

The proposed malicious login detection method focuses on jumps between hosts and login information. Inspired by the success of neural networks in text classification [6, 15, 25], the proposed method trains a classifier to distinguish between benign and malicious authentication log activities.

Figure 1 provides an overview of the proposed detection method. The method has three main phases, host vector learning, feature extraction

and classification. Host vector learning employs the Hawkes temporal network embedding (HTNE) model [26] to learn host characteristics in space and time from a host connection graph. Feature extraction employs a long short-term memory model with an attention mechanism to capture the semantics of logs. Specifically, a login entry is expressed as: "a source user employs a destination user's certificate to log into a target host from the source host." Drawing on research in text classification, log attributes are considered to be text that is input to a feature extraction process to produce feature vectors. The final classification phase employs a multilayer perceptron (fully-connected network) to classify login operations as benign or malicious.

3.2 Threat Model

The proposed method focuses on credential-based lateral movements. In this paradigm, attackers achieve lateral movements by establishing footholds with stolen login credentials.

The threat model assumes that login data can be collected in a timely manner and that the login data is not tampered with during transmission. Also, new login data is continually introduced to enable the model to re-learn the host initial vectors.

4. Proposed Method

This section describes the host vector learning approach, feature extraction model, attention mechanism and classification model optimization.

4.1 Host Vector Learning

A lateral movement involves a jump from one host to another. Lateral movements by a user are modeled in a jump graph in which nodes are hosts and jumps are edges. A jump between hosts is a jump in space and time. This is a typical problem in temporal network embedding learning, which means that the representation vectors of hosts can be learned.

The proposed method uses the Hawkes temporal network embedding (HTNE) model [26] to create a temporal graph embedding node representation. The method learns the interactions between hosts and explores the time sequences of host interactions. It integrates the Hawkes process in network embedding to capture the impacts of previous neighbor nodes on the current neighbor node.

The login dataset used in this research records time in seconds. At this granularity, the amount of data is huge. Therefore, the times were

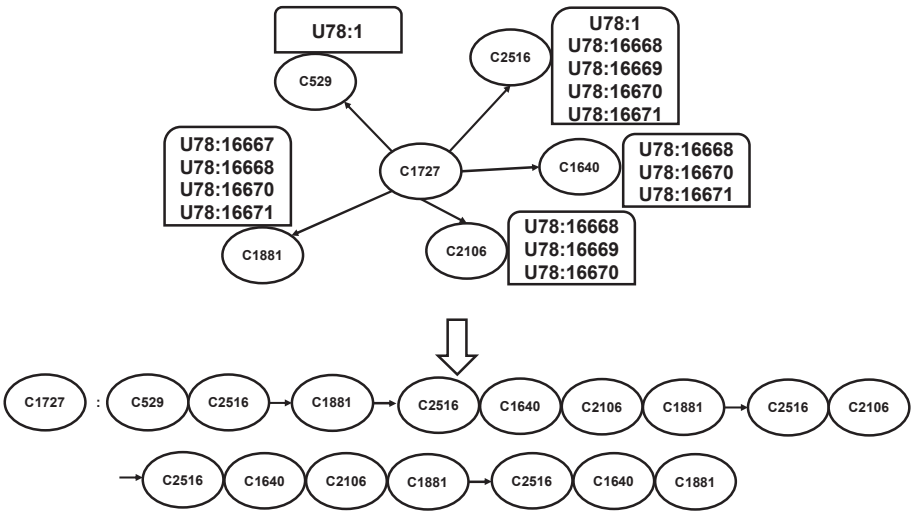

Figure 2. Host temporal network and neighborhood formation sequence.

converted to minutes and the temporal network was constructed in the order of minutes.

Figure 2 presents an example of a host temporal network and neighborhood formation sequence. The top portion of the figure shows a temporal network whose nodes are hosts. In addition to interactions between hosts, interaction time sequences exist for source users.

The bottom portion of Figure 2 shows neighborhood formation sequences initiated by source hosts. Each neighborhood formation sequence is a chronological sequence of the same user. After a sequence is created, it is modeled using the multi-dimensional Hawkes process and the model is trained as in [26] to obtain the vector corresponding to each node, specifically, the representation of each host.

4.2 Feature Extraction

Unlike previous work on malicious login detection that has focused on the macro-level entities (i.e., users or hosts), this research considers micro-level entities (i.e., log entries). This fine-grained approach requires the pure host-based representation to be modified. Each log entry has the form node-edge-node corresponding to (source host – login – target host). Specifically, the "login" information from the log entries has to be embedded in the edges.

Unfortunately, it is difficult to learn the representations of nodes and the expressions of edges in a temporal network graph. Additionally, handling the large number of edge combinations requires substantial computing resources. Therefore, a neural network model is used to acquire the login information.

A neural network offers three advantages when attempting to learn login information. First, a neural network can handle a large number of log entries generated in a complex computer network environment. Second, lateral movements have subtle (non-obvious) rules and features; a neural network can automatically extract features and adapt to a dynamic computer network environment. Third, malicious login detection methods are plagued by large numbers of false positives (i.e., benign logs are classified as malicious); the powerful feature extraction ability of a neural network enables the false positive rate to be reduced.

Combined with the characteristics of the dataset itself, the processed authentication log is essentially in a language with a grammar and each log entry is a sentence that conforms to the grammar. In fact, each log entry is translated to: "the source user logs into the target host using the credentials of the target user from the source host via some authentication type and login type." This is similar to the text classification problem in the field of natural language processing.

Research has shown that using a recurrent neural network for the feature extraction task is significantly better than using a convolutional neural network. However, a recurrent neural network has gradient disappearance and gradient explosion problems. Therefore, a long short-term memory neural network is employed by the proposed method. Figure 3 shows the long short-term memory based feature extraction model.

At this point, the feature extraction phase can be specified. Let $X = [x_1, x_2, ..., x_7]$ denote a log entry, where x_1 corresponds to source user, x_2 to destination user, x_3 to source host, x_4 to destination host, x_5 to authentication type, x_6 to login type and x_7 to authentication orientation (login/logoff).

The interactive host representation produced by the host vector learning phase and the other attribute representations are randomly initialized into a long short-term memory neural network for extracting the login characteristics. Next, each log attribute x_i is input to a two-layer long short-term memory neural network and the hidden vector h_i is produced for each attribute as follows:.

$$[h_1, h_2, ..., h_7] = \text{LSTM}([x_1, x_2, ..., x_7])$$

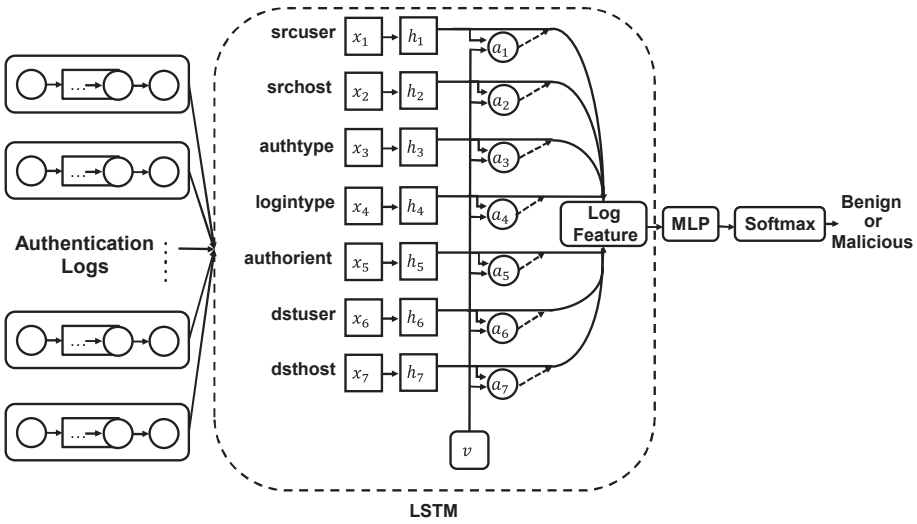

Figure 3. Feature extraction model.

4.3 Attention Mechanism

Not all attributes contribute equally to the meaning of a log entry. Therefore, an attention mechanism is incorporated to extract the attributes that are important to the meaning of a log entry and aggregate the representation of the informative attributes to create a log vector.

First, the hidden vector of each log attribute h_i obtained via feature extraction is passed to the activation function tanh to yield the attention score (importance) u_i of the log attribute:

$$u_i = v^T tanh(W_h \cdot h_i + W_b)$$

where W_h and W_b are the weight and bias parameters, respectively, and v is an artificially-defined parameter vector. These parameters are learned using the stochastic gradient descent technique.

The normalized attention score α_i of the log attribute is computed as:

$$\alpha_i = \frac{exp(u_i)}{\sum_{j=1}^{n} exp(u_j)}$$

where n is the number of attributes in a log.

Next, the log entry vector log_i corresponding to the log attribute is computed as:

$$log_i = \sum_{j=1}^{n} \alpha_j \cdot h_j$$

The log entry vector log_i is passed to a multilayer perceptron (MLP) with a full connection layer for supervised learning and classification, and the result is input to the softmax function to compute the probability p_{log_i} of the log entry:

$$p_{log_i} = \text{softmax}(\text{MLP}(log_i))$$

4.4 Classification Model Optimization

After obtaining the probability of a log entry p_{log_i}, the cross-entropy loss function is selected as the target function and used for training:

$$loss = -\sum_{i=1}^{N} log_i^{label} \cdot \log p_{log_i}$$

where log_i^{label} is the true label (benign or malicious) of the log entry, $\log p_{log_i}$ is the logarithm of the log entry probability p_{log_i} and N is the number of log entries.

Note that the *label* of each log entry in the training dataset is assigned to be benign or malicious during the training phase.

During the testing phase, two probability values corresponding to benign and malicious are obtained after processing by the multilayer perceptron and softmax function. Each log entry is classified as benign or malicious depending on which of the two has the higher probability value.

5. Experimental Evaluation

This section describes the experiments conducted to evaluate the performance of the proposed malicious login detection method along with the evaluation results.

5.1 Dataset Description

The experiments drew on the Comprehensive, Multi-Source Cyber-Security Events Dataset from Los Alamos National Laboratory [12]. The dataset comprises five data elements with units of seconds: (i) Windows-based authentication events, (ii) process start and stop events, (iii) DNS lookups, (iv) network flow data and (v) well-defined red team events. The dataset, which is approximately 12 GB in compressed form, contains more than 1.6 billion events associated with 12,425 users and 17,684

Table 1. Dataset statistics.

Dataset	Total Logs	Malicious Logs
Training Dataset	3,67,6507	340
Testing Dataset	6,213,591	409

computers. The content includes authentication logs over 58 consecutive days. Although the data was collected in a real environment, some data is missing due to equipment errors or other causes [17].

In order to use valid data and avoid flawed data, only the authentication events and red team activities were employed for training and testing. The red team events, which corresponded to typical advanced persistent threat activity, were labeled as malicious. The red team events comprised 749 malicious logins involving 98 compromised users.

All the malicious logins involving the 98 compromised users occurred during the month of January. Therefore, the dataset used in the experiments included the logs for the entire month of January. Logs from January 1 through January 12 were used for training and the logs from January 13 through January 31 were used for testing.

Table 1 shows the compositions of the training and testing datasets. The 749 malicious logs in the training and testing datasets came from the red team file. Analysis of the datasets revealed that 203 of the 409 malicious logins in the testing dataset were not in the training dataset. The logs used in the experiments corresponded to <source user, source host, target host> triples.

Each authentication event in the datasets comprised nine attributes: time, source user, destination user, source computer, destination computer, authentication type, login type, authentication orientation and success/failure. The login success/failure attribute was eliminated because failed logins were ignored. Also, the time attribute was excluded during feature extraction to obtain a pure categorical feature space.

5.2 Experimental Setup

The computing system used in the experiments was an Nvidia Tesla V100 GPU wth three cores, each with 16 GB display memory. The model was constructed using `pytorch1.4.0` and the anomalous login detection method was programmed in `python3.6.8`.

The pre-processing vectors had attributes with 64 dimensions. During the training phase, it was determined that two-layer stacked LSTMs performed better than a one-layer LSTM and were not as good as three-layer stacked LSTMs. However, two-layer stacked LSTMs were selected

to save time and adjust the model quickly after adding new user data. The hyper-parameters used for training were dropout = 0.5, batch size = 128 and hidden size = 64. Preliminary experiments revealed that the learning rate was better when its initial value was set to one.

5.3 Evaluated Models

The malicious login detection method was compared against several baseline models. To demonstrate the utility of the attention mechanism two versions of the proposed method were employed. The proposed method, incorporating long short-term memory with an attention mechanism, is referred to as MLDLA. The second version, employing long short-term memory without the attention mechanism, is referred to as MLDL.

The following five baseline models were evaluated in the experiments:

- **Tiresias [19]:** Tiresias leverages recurrent neural networks to predict future events based on previous observations. It learns user login actions and predicts the next event using long short-term memory.

- **Ensemble [4]:** The Ensemble model relies on a graph of hosts and uses an ensemble of two anomaly detectors to identify compromised hosts. It employs three steps to detect attacks that rely on lateral movements: feature extraction, feature analysis and anomaly detection. The model uses network flow logs, and command and control and lateral movement traces to create a communication graph. The model is coarse grained, but it is a good unsupervised learning representative.

- **Bagging Machine Learning [11]:** Bagging ML is a supervised learning model designed to extract advanced composite features. It leverages three models, Random Forest, LogitBoost and Logistic Regression, in a majority voting configuration. The model uses authentication events and red team activities. It was applied to 21 users selected from 98 malicious users. The training dataset comprised the first 12.5 days of data totaling 199,090 events and the testing dataset comprised 25,900 events containing 37 batches.

- **Semi-Supervised Outlier Detection [10]:** The SSOD model employs an automatic semi-supervised outlier ensemble detector whose automatic feature is supported by unsupervised outlier ensemble theory. It has two phases, training dataset preparation and semi-supervised ensemble construction. It employed a random

Table 2. Model performance.

Method	TPR	FPR	Accuracy	AUC	F1-Score
Tiresias	99.75%	66.24%	33.76%	5.3%	0.2%
Ensemble	91.15%	13.95%	86.05%	89.17%	–
Bagging ML	100%	0.19%	99.62%	–	65.87%
SSOD	100%	0.17%	–	–	–
Log2vec	100%	10%	–	91%	–
MLDL	98.28%	0.0024%	99.99%	99.99%	74.17%
MLDLA	98.78%	0.002%	99.99%	99.99%	76.08%

sample of 150,000 consecutive authentication events containing at least five malicious events in its evaluation.

- **Log2vec [14]:** Log2vec is a heterogeneous graph embedding based modularized method that proposes several rules to construct a graph from which the log character representation can be learned. It uses clustering to distinguish between malicious and benign log entries Log2vec considers process events, authentication events and red team activities. Fifty users from among the 98 malicious users were selected to construct the rule-based graph.

5.4 Evaluation Results

Table 2 shows the model performance results. Note that the accuracy of the Bagging ML approach is the balanced average accuracy reported in [11]. Although the true positive rate (TPR) of the proposed MLDLA method does not reach 100%, the method used all the malicious logs, which was more reliable than the other models that only used subsets of the malicious logs. The MLDLA method has the best false positive rate (FPR) of 0.002%. Specifically, only 127 out of 6,213,182 benign logs were misclassified as malicious. The MLDLA method also has the best F1-score of 76.08%. Comparison of the MLDLA and MLDL method results demonstrates the efficacy of the attention mechanism.

Figure 4 shows the true positive and false positive rates for the MLDLA (dark lines) and MLDL (light lines) methods. The true positive and false positive rates were computed for each training epoch. The MLDLA method outperforms the MLDL method for both metrics. Taking the averages of the ten epochs with the best performance, the MLDLA true positive rate is greater than 98% whereas the MLDL rate is 96%. In the best case, MLDLA misclassified 127 logs as malicious whereas MLDL misclassified 140 logs as malicious.

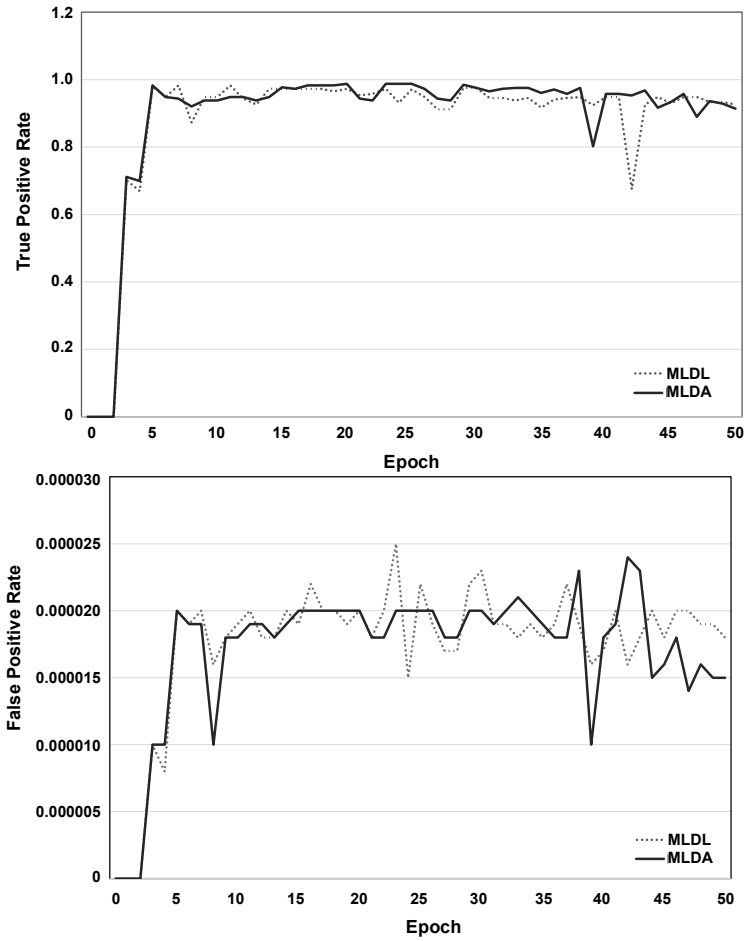

Figure 4. True positive and false positive rates for the MLDLA and MLDL methods.

In addition to improving detection performance, the attention mechanism enhances the interpretability of the model. Figure 5 shows six logs (rows), three benign and three malicious. The shades of the cells express the relative weights of the log attributes (darker shades correspond to greater weights). Clearly, the malicious and benign logs have different weights.

5.5 Optimization and Learning Rate

The stochastic gradient descent and Adam optimizers were used in the experiments. The best results were obtained with the stochastic gradient

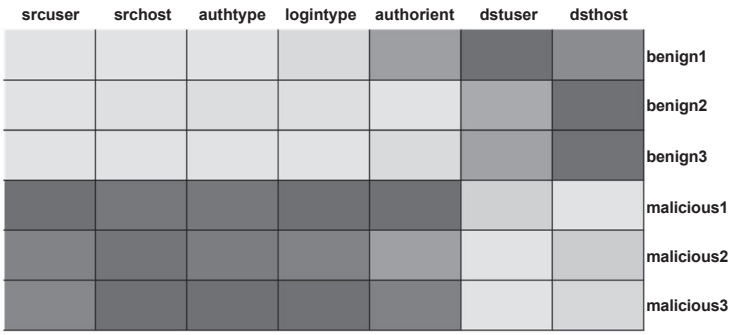

Figure 5. Attribute attention in logs.

descent optimizer. For this optimizer, setting the initial learning rate to one achieved better results. When the learning rate was set to 0.01, all the logs in the testing dataset were classified as benign (i.e., 409 malicious logs were classified as benign). Although it is rare to set the initial learning rate to one, the experiment demonstrated that it was effective for the dataset.

The Adam optimizer essentially employs the RMSprop algorithm with a momentum term. Setting the initial learning rate to one resulted in all the logs being classified as benign. Setting the initial learning rate to 0.001 yielded a false positive rate of 0.0014%. However, the true positive rate was only 76%.

6. Conclusions

Advanced persistent threats routinely leverage lateral movements in networks to cause harm. Lateral movements typically involve malicious logins using stolen credentials. However, current detection methods that focus on malicious users and/or hosts are too coarse grained to detect lateral movements effectively. In contrast, the fine-grained method presented in this chapter focuses on abnormal log entries. It employs a temporal neural network embedding to learn host jumping representations. The learned host vector and initialized attribute vectors in log entries are input to a long short-term memory with an attention mechanism for login feature extraction, which help determine if logins are malicious. Experimental results demonstrate that the proposed method has a true positive rate in excess of 98% and a false positive rate of 0.002%. It also outperforms several baseline detection models, especially with regard to the false positive rate and F1-score.

Future research will evaluate the performance of the proposed method on other datasets. Additionally, it will attempt to engage unsupervised learning to account for the paucity of labeled malicious data.

References

[1] F. Amrouche, S. Lagraa, G. Kaiafas and R. State, Graph-based malicious login events investigation, *Proceedings of the IFIP/IEEE Symposium on Integrated Network and Service Management*, pp. 63–66, 2019.

[2] T. Bai, H. Bian, A. Daya, M. Salahuddin, N. Limam and R. Boutaba, A machine learning approach for RDP-based lateral movement detection, *Proceedings of the Forty-Fourth IEEE Conference on Local Computer Networks*, pp. 242–245, 2019.

[3] H. Bian, T. Bai, M. Salahuddin, N. Limam, A. Daya and R. Boutaba, Host in danger? Detecting network intrusions from authentication logs, *Proceedings of the Fifteenth International Conference on Network and Service Management*, 2019.

[4] A. Bohara, M. Noureddine, A. Fawaz and W. Sanders, An unsupervised multi-detector approach for identifying malicious lateral movement, *Proceedings of the Thirty-Sixth IEEE Symposium on Reliable Distributed Systems*, pp. 224–233, 2017.

[5] A. Brown, A. Tuor, B. Hutchinson and N. Nichols, Recurrent Neural Network Attention Mechanisms for Interpretable System Log Anomaly Detection, arXiv: 1803.04967 (`arxiv.org/abs/1803.04967`), 2018.

[6] H. Chen, M. Sun, C. Tu, Y. Lin and Z. Liu, Neural sentiment classification with user and product attention, *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1650–1659, 2016.

[7] M. Chen, Y. Yao, J. Liu, B. Jiang, L. Su and Z. Lu, A novel approach for identifying lateral movement attacks based on network embedding, *Proceedings of the IEEE International Conference on Parallel and Distributed Processing with Applications, Ubiquitous Computing and Communications, Big Data and Cloud Computing, Social Computing and Networking, and Sustainable Computing and Communications*, pp. 708–715, 2018.

[8] I. Ghafir, M. Hammoudeh, V. Prenosil, L. Han, R. Hegarty, K. Rabie and F. Aparicio-Navarro, Detection of advanced persistent threat using machine learning correlation analysis, *Future Generation Computer Systems*, vol. 89, pp. 349–359, 2018.

[9] R. Holt, S. Aubrey, A. DeVille, W. Haight, T. Gary and Q. Wang, Deep autoencoder neural networks for detecting lateral movement in computer networks, *Proceedings of the International Conference on Artificial Intelligence*, pp. 277–283, 2019.

[10] G. Kaiafas, C. Hammerschmidt, S. Lagraa and R. State, Auto semi-supervised outlier detection for malicious authentication events, in *Machine Learning and Knowledge Discovery in Databases*, P. Cellier and K. Driessens (Eds.), Springer, Cham, Switzerland, pp. 176–190, 2020.

[11] G. Kaiafas, G. Varisteas, S. Lagraa, R. State, C. Nguyen, T. Ries and M. Ourdane, Detecting malicious authentication events trustfully, *Proceedings of the IEEE/IFIP Network Operations and Management Symposium*, 2018.

[12] A. Kent, Comprehensive, Multi-Source Cyber-Security Events, Los Alamos National Laboratory, Los Alamos, New Mexico (`csr.lanl.gov/data/cyber1`), 2015.

[13] A. Kent, L. Liebrock and J. Neil, Authentication graphs: Analyzing user behavior within an enterprise network, *Computers and Security*, vol. 48, pp. 150–166, 2015.

[14] F. Liu, Y. Wen, D. Zhang, X. Jiang, X. Xing and D. Meng, Log2vec: A heterogeneous graph embedding based approach for detecting cyber threats within an enterprise, *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pp. 1777–1794, 2019.

[15] P. Liu, X. Qiu and X. Huang, Recurrent Neural Network for Text Classification with Multi-Task Learning, arXiv: 1605.05101 (`arxiv.org/abs/1605.05101`), 2016.

[16] B. Powell, Detecting malicious logins as graph anomalies, *Journal of Information Security and Applications*, vol. 54, article no. 102557, 2019.

[17] M. Pritom, C. Li, B. Chu and X. Niu, A study on log analysis approaches using the Sandia dataset, *Proceedings of the Twenty-Sixth International Conference on Computer Communications and Networks*, 2017.

[18] T. Schindler, Anomaly Detection in Log Data Using Graph Databases and Machine Learning to Defend Advanced Persistent Threats, arXiv: 1802.00259 (`arxiv.org/abs/1802.00259`), 2018.

[19] Y. Shen, E. Mariconti, P. Vervier and G. Stringhini, Tiresias: Predicting security events through deep learning, *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pp. 592–605, 2018.

[20] H. Siadati and N. Memon, Detecting structurally-anomalous logins within enterprise networks, *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pp. 1273–1284, 2017.

[21] H. Siadati, B. Saket and N. Memon, Detecting malicious logins in enterprise networks using visualization, *Proceedings of the IEEE Symposium on Visualization for Cyber Security*, 2016.

[22] G. Tang, M. Muller, A. Rios and R. Sennrich, Why Self-Attention? A Targeted Evaluation of Neural Machine Translation Architectures, arXiv: 1808.08946 (`arxiv.org/abs/1808.08946`), 2018.

[23] A. Tuor, R. Baerwolf, N. Knowles, B. Hutchinson, N. Nichols and R. Jasper, Recurrent Neural Network Language Models for Open Vocabulary Event-Level Cyber Anomaly Detection, arXiv: 1712.00557 (`arxiv.org/abs/1712.00557`), 201.

[24] L. Yang, P. Li, Y. Zhang, X. Yang, Y. Xiang and W. Zhou, Effective repair strategy against advanced persistent threat: A differential game approach, *IEEE Transactions on Information Forensics and Security*, vol. 14(7), pp. 1713–1728, 2019.

[25] Z. Yang, D. Yang, C. Dyer, X. He and E. Hovy, Hierarchical attention networks for document classification, *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1480–1489, 2017.

[26] Y. Zuo, G. Liu, H. Lin, J. Guo, X. Hu and J. Wu, Embedding temporal network via neighborhood formation, *Proceedings of the Twenty-Fourth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2857–2866, 2018.