Deep Learning-based Hybrid Model for Efficient Anomaly Detection

Frances Osamor, Briana Wellman
Computer Science and Information Technology
University of the District of Columbia, Washington, USA

***Abstract*—It is common among security organizations to run processes system call trace data to predict its anomalous behavior, and it is still a dynamic study region. Learning-based algorithms can be employed to solve such problems since it is typical pattern recognition problem. With the advanced progress in operating systems, some datasets became outdated and irrelevant. System calls datasets such as Australian Defense Force Academy Linux Dataset (ADFA-LD) are amongst the current cohort containing labeled data of system call traces for normal and malicious processes on various applications. In this paper, we propose a hybrid deep learning-based anomaly detection system. To advance the detection accurateness and competence of anomaly detection systems, Convolution Neural Network (CNN) with Long Short Term Memory (LSTM) is employed. The raw sequence of system call trace is fed to the CNN network first, reducing the traces' dimension. This reduced trace vector is further fed to the LSTM network to learn the sequences of the system calls and produce the concluding detection outcome. Tensorflow-GPU was used to implement and train the hybrid model and evaluated on the ADFA-LD dataset. Experimental results showed that the proposed method had reduced training time with an enhanced anomaly detection rate. Therefore, this method lowers the false alarm rates.**

Keywords*—Anomaly detection; system call sequence; convolution neural network; long short term memory

I. Introduction

Intrusion detection is the procedure of recognizing malicious behaviors on the network [1]. There are two kinds of intrusion-based detection systems. They are network-based [2] and host-based [3] —network-based examines the traffic on the network and the set of protocols to govern the conceivable intrusions. Pattern matching procedures were used earlier by analysts [4]. Packet header of the data, string and port data are among the few matching features used. Such features improve the consistency and suitability. Particle swarm optimization, gray wolf algorithm, genetic algorithm are some of the current feature selection algorithms [5, 6]. The genetic algorithm runs to the issue of moderately outsized randomness; the gray wolf algorithm traps easily in the local optimum. Host-based intrusion detection system investigates the system maneuver data such as the log files and audits for anomalous pattern behavior. Learning-based methods are popular compare to the rule-based system with the increasing complexity of the environment [7]. Support Vector Machine, Decision Tree, and Random Forest are some of the traditional machine learning algorithms [8, 9, 10, and 11] applied for host-based intrusion detection systems. Hinton projected the idea of deep learning first in 2006 [12]. Deep Learning learns the complex patterns of the data in both supervised and unsupervised ways. Hence, deep learning based algorithms are widely used in natural language processing areas, image processing areas, etc. In recent years, scholars started smearing deep Learning based algorithms to solve intrusion detection problems. To reduce the dimensionality of features and extract meaningful patterns, autoencoders are widely used. It encodes the high-dimensional input data into lower dimension subspace. Convolution Neural Network extracts the significant features from the image data (Gray Scale or Color). Chawla et al. [13] applied stacked CNN and achieved an accuracy of 0.81. Diep et al. [14] focused on custom CNN with word embedding followed by Bi-LSTM and achieved a score of 0.96. In this work, we study the application of learning-based Convolution Neural Network (CNN) and Long Short Term Memory (LSTM) algorithm with optimization of hyperparameters to detect the anomalies [15] with reduced and efficient neural architecture and in optimized training time. The second focus is on the training time of the hybrid model. The training time of the proposed hybrid model is significantly reduced compared to the baseline models. The rest of the divisions are as follows. In Section 2, we discuss the related work on anomaly detection systems. The dataset is explained in Section 3. Next, we provide explanation of proposed framework in Section 4. Next, in Section 5 we explain the experimental results and finally we conclude in section 6 with future work discussion.

II. Related Work

A system call is a call made by a program to the kernel for a service. Its system calls sequence can analyze the behavior of the process. Such traces are used to classify a process as normal or malicious in a host-based intrusion detection system. For such behavior classification of a process, numerous data representation methods can be found in the literature. Sequence n-gram model [16, 17] and pair-gram [18, 19] are among the few representations to extract the meaningful features from the trace of system call. Information retrieval and Natural language processing (NLP) techniques can be used by treating an individual system call as word and the trace of system call as the document. Vector space model and Boolean model-based document representation approach are used to extract the features from the traces. X. Wang et al. [20] extracted the feature using the Boolean model with n-gram approach and further applied the Support Vector Machine algorithm for classification purposes. Vector space model was used by K. Rieck et al. [21] with polynomial function for categorizing the traces of system call. As a distance metric, Y. Liao [22] used a

vector space model with a k-nearest neighbor classifier with cosine similarity. Nonetheless, such a classification system considers the system calls frequencies and not its sequence. System call trace datasets like DARPA [23] and University of New Mexico [24] were widely used by researchers for training the learning-based algorithms for analyzing the behavior of the processes. Nevertheless, with the advanced modernization of complex operating systems, these datasets are becoming irrelevant. ADFA datasets by G. Creech et al. [25, 26] are currently used to benchmark the evaluation of intrusion detection systems based on a system call. They used short sequences and a projected model for anomaly detection. It has an extensive pool of system call traces. They trained one-class SVM, Hidden Markov Model, and Extreme Learning Machine algorithms. With one-class SVM, the author achieved 80% accuracy and 90% with ELM [27]. It is a time-intensive task for learning a vocabulary of all sequences. Miao Xie et al. [28] uses principal component analysis [29] to reduce the dimensionality of features on the ADFA-LD dataset and further train k-means clustering and k-nearest neighbor algorithms. There result showed an additional endeavor, they achieved an accuracy of 70% by training one-class SVM. The results and experiments in the proposed work are mostly on the supervised classification [30, 31]. Thus, in this work, we implemented a hybrid learning-based approach.

III. ADFA-LD Dataset

Australian Defense Force Academy generated the ADFA-LD host-based intrusion detection dataset. Linux system calls are being recorded in this dataset. To communicate between the user and kernel mode, standard interfaces are provided by the Linux kernel. The programs in user-mode have very partial reference to hardware devices for accessing the resources of systems, read and write to the device, and new process creation. Every system call has a unique identifier number on the sequence trace. The host designed to characterize a recent Linux server which logs the traces of system calls during a specified period. Authentic programs functioned in normal behavior during such sampling period. The constituted Linux server comprises of file distribution, remote access, database, and web server type of functionality. Entirely patched Ubuntu 11.04 operating system with Linux kernel 2.6.38 was used. For access to services on the web, Apache 2.2.17 and PHP 5.3.5 were configured. Secure Shell Protocol, MySQL, and File Transfer Protocol with their default ports were permitted. As a collaborative web-based tool, TikiWiki 8.1 was installed. Subsequently, various cyberattacks listed in Table I are injected to generate the abnormal traces.

TABLE I. ADFA-LD Dataset

Trace Type	Count	Label
Hydra-FTP	162	Abnormal
Hydra-SSH	148	Abnormal
Webshell	118	Abnormal
Java Meterpreter	125	Abnormal
Meterpreter	75	Abnormal
Add user	91	Abnormal
Training	833	Normal
Validation	4373	Normal

IV. Proposed Framework

Fig. 1 depicts the overall framework. There are three phases. Dataset Collection, Data Engineering and Training of the Detection Algorithm with the final testing.

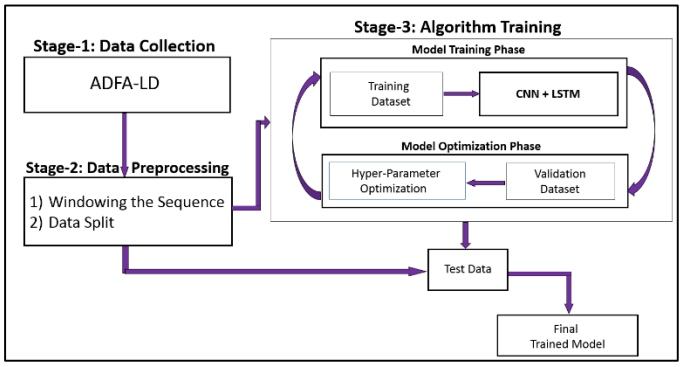

Fig. 1. Proposed Framework.

Phase-1: Data Collection

We employed the ADFA-LD dataset which is explained in Section 3.

Phase-2: Data Engineering (Preprocessing)

The following tasks are performed:

- Windowing the sequence of the system calls as input and output and performing the encoding of the target category [32].
- Applying stratified sampling and divide the dataset into train, validate, and test.

Phase-3: Algorithm Training.

In this phase, the hybrid model is trained as discussed below:

A. Training of the Model

A hybrid deep learning-based Convolution Neural Network with long short term memory algorithm is trained in this phase.

1) Convolution neural network: It is a type of neural network, as depicted in Fig. 2, which is widely used for images dataset. It is used for image segmentation, classification, etc.

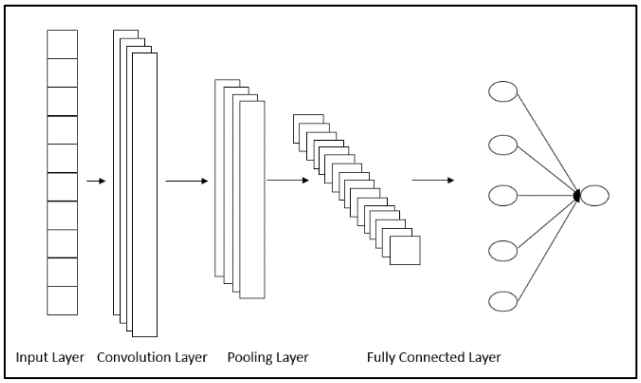

Fig. 2. Convolution Neural Network.

There are two layers in CNN architecture. They are:

a) Convolution layer: This layer applies convolution operation on the input image to extract the meaningful feature. Each pixel value of the image is multiplied by the corresponding filter value, and finally, all the values are added.

b) Pooling layer: This layer is used to reduce the dimensionality of the images. There are different types: Max-Pooling, Min-Pooling, Mean-Pooling, etc.

Rectifier Linear Unit (ReLu) activation function is applied to deal with the non-linearity of the data. Finally, a fully connected dense layer is employed to predict the outcome.

2) Long short term memory: LSTM has memory cells. Numerous gates are attached to the LSTM system to mitigate the vanishing gradient problem of the RNN [33]. These gates behave like a memory. The memory update occurs with the read of an input by the cell. It has the following four gates:

I gate: It enhances the updation of the new incoming memory.

O gate: It updates the new hidden state by selecting the new memory cell info.

F gate: It regulates the amount of old information to be discarded.

M gate: It creates new memory.

For a example dataset of U=(U_1, U_2, U_3, U_{N-1}, U_N) as input to network, it updates the above four gates values to learn the output variable. It is updated as follows:

$$(z_t, d_{t-1}, s_{t-1}) \rightarrow (d_t, s_t) \quad (1)$$

$$i_t = \sigma\,(k_{zi} z_t + k_{di} d_{t-1} + k_{si} a_{t-1} + j_i) \quad (2)$$

$$f_t = \sigma\,(k_{zf} z_t + k_{df} d_{t-1} + k_{af} a_{t-1} + j_f) \quad (3)$$

$$a_t = f_t * a_{t-1} + i_t * tanh(k_{za} z_t + k_{da} d_{t-1} + j_a) \quad (4)$$

$$y_t = \sigma\,(k_{zy} z_t + k_{dy} d_{t-1} + k_{ay} a_t + j_y) \quad (5)$$

$$d_t = y_t * tanh(a_t) \quad (6)$$

where j_i, j_f ,j_y ,j_a represents the bias units for I, F, O, and the memory cell, respectively. Next, k denotes the weight vector, and *a* denotes memory state with output of intermediate layer's as d as shown in Fig. 3.

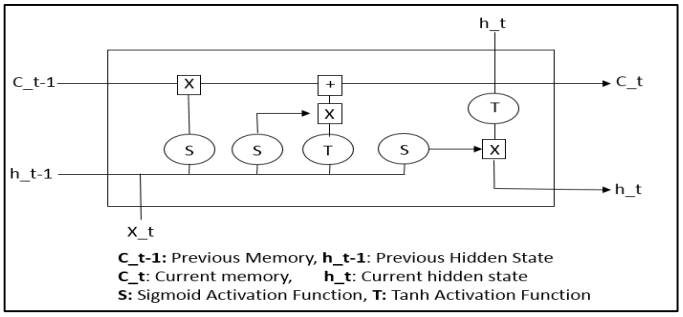

Fig. 3. Long Short Term Memory.

We developed and trained the hybrid CNN [34] with LSTM model architecture shown in Fig. 4. To optimize the hyper-parameters value, we use the K-fold technique with K being 10 [35]. The proposed architecture has a custom ConvLSTM2D layer. This layer takes the raw system call sequences of normal operation as input. The convolution layer applies the convolution operation followed by the pooling technique to extract the meaningful features from the sequences. We use a kernel size of (1,2) with 64 filters. Next, the extracted sequence is fed to the LSTM layer, which learns the patterns of the normal behavior of the sequences. TimeDistributed layer is applied to pass the hidden output at every time step. Finally, the output is flattened and passed to a fully connected dense layer, predicting the final result.

The following metric is used to evaluate the model.

Mean Squared Error (MSE): It finds the square deviations between predicted and actual value.

$$\text{MSE} = \frac{1}{N}\sum(Y - \overline{Y})^2 \quad (7)$$

N is the total data points, Y is the actual ground-truth label, and $\overline{Y}$ is the predicted label.

B. Optimization of the Model

We use the validation part of the dataset for optimizing the hyper-parameter [36, 37] for the hybrid model. Parameters that are tuned are as follows:

- Number of Epochs: Total amount of time, data is passed to the model.
- Batch-size: Total count of data sequence given to the model to calculate the loss and update the weight. After the fine-tune optimization, test data is used to evaluate the model.

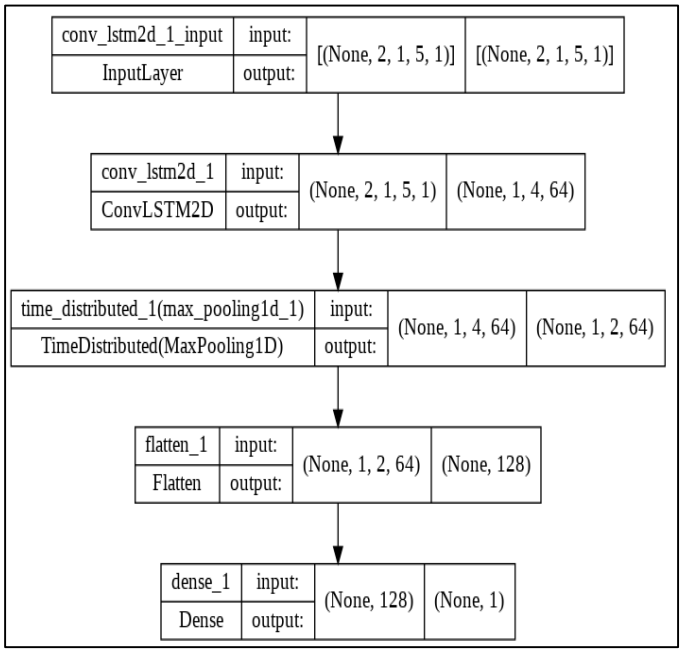

Fig. 4. Model Architecture.

V. Experimental Results

The hybrid model was trained with activation function namely Rectifier Linear Unit. The optimal accuracy rate is achieved with a batch-size of 128. While training the deep neural network, the size of the batch is one of the critical hyper-parameter to be tuned. Stochastic Batch, Batch, and Mini-Batch are the three variants of batch sizes. The model waits till the end of the processing to update the weights in batch variant. Next, model updates the weights after every input sequence in stochastic batch approach. Model updates the weights after every batch in minibatch approach. We train the hybrid model with mini-batch of {16, 32, 64, 128, and 256}. At each batch value, the accuracy value is shown in Fig. 5. The highest accuracy of 0.961 is achieved with a batch value of 128. Fig. 6 and 7 depicts the accuracy and loss at each epoch. After 100th Epoch, the accuracy and loss didn't change much. With the proposed hybrid approach, we achieved 96% accuracy rate and 0.04% loss. The comparison with baseline is given in Table II.

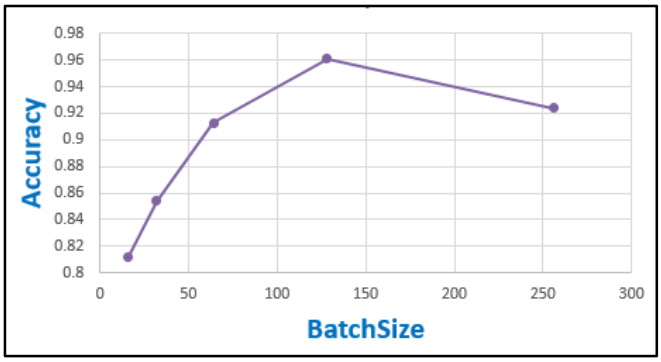

Fig. 5. Accuracy for Individual Batch.

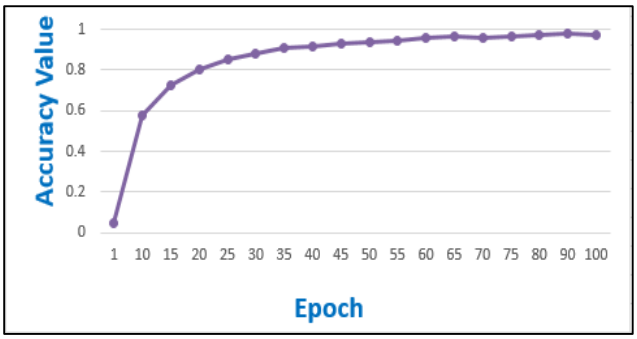

Fig. 6. Accuracy for Individual Epoch.

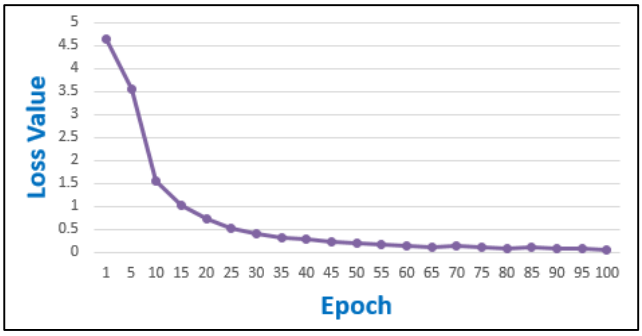

Fig. 7. Loss for Individual Batch.

TABLE II. Comparison with Baseline

Model	Accuracy
CNN	0.81
LSTM	0.88
CNN+LSTM	**0.96**

VI. Conclusion

Intrusion-based detection algorithms exert on the postulate that normal events differ from abnormal events. Anomaly detection algorithms learn a program's behavior during its normal operation. Process behavior is defined by the occurrence of the system call in a particular sequence. We proposed a hybrid deep learning-based CNN with an LSTM model to detect the anomaly in the sequence of system calls. CNN was used to extract the meaningful features, and LSTM was used to learn the patterns of the sequence from the reduced features. The model is trained with the normal process behavior and tested against the normal and malware-infected process. We use the ADFA-LD dataset to test our proposed hybrid model. We achieved an accuracy rate of 96% with a reduced time.

This work can be extended by applying the AutoEncoder neural network to reduce the dimensionality of the data further. Furthermore, numerous natural language processing-based algorithms can be trained, such as Bi-Directional LSTM and Transformers for comparative analysis.

References

[1] Tidjon, L. N., Frappier, M., & Mammar, A. (2019). Intrusion detection systems: A cross-domain overview. *IEEE Communications Surveys & Tutorials*, *21*(4), 3639-3681.

[2] Hamed, T., Dara, R., & Kremer, S. C. (2018). Network intrusion detection system based on recursive feature addition and bigram technique. *computers & security*, *73*, 137-155.

[3] Marteau, P. F. (2018). Sequence covering for efficient host-based intrusion detection. *IEEE Transactions on Information Forensics and Security*, *14*(4), 994-1006.

[4] Kim, H., Hong, H., Kim, H. S., & Kang, S. (2009). A memory-efficient parallel string matching for intrusion detection systems. *IEEE communications letters*, *13*(12), 1004-1006.

[5] Wei, B., Zhang, W., Xia, X., Zhang, Y., Yu, F., & Zhu, Z. (2019). Efficient feature selection algorithm based on particle swarm optimization with learning memory. *IEEE Access*, *7*, 166066-166078.

[6] Ghamisi, P., & Benediktsson, J. A. (2014). Feature selection based on hybridization of genetic algorithm and particle swarm optimization. *IEEE Geoscience and remote sensing letters*, *12*(2), 309-313.

[7] Porras, P. A., & Kemmerer, R. A. (1992, January). Penetration state transition analysis: A rule-based intrusion detection approach. In *Proceedings Eighth Annual Computer Security Application Conference* (pp. 220-221). IEEE Computer Society.

[8] Soni, J., Prabakar, N., & Upadhyay, H. (2019). Feature extraction through deepwalk on weighted graph. In *Proceedings of the 15th international conference on data science (ICDATA'19), Las Vegas, NV*.

[9] Soni, J., & Prabakar, N. (2018). Effective machine learning approach to detect groups of fake reviewers. In *Proceedings of the 14th international conference on data science (ICDATA'18), Las Vegas, NV* (pp. 3-9).

[10] Soni, J., Prabakar, N., & Kim, J. H. (2017, May). Prediction of component failures of telepresence robot with temporal data. In *30th Florida conference on recent advances in robotics*.

[11] Thejas, G. S., Soni, J., Chandna, K., Iyengar, S. S., Sunitha, N. R., & Prabakar, N. (2019, April). Learning-based model to fight against fake like clicks on instagram posts. In *2019 SoutheastCon* (pp. 1-8). IEEE.

[12] Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, *18*(7), 1527-1554.

[13] Chawla, A., Lee, B., Fallon, S., & Jacob, P. (2018, September). Host based intrusion detection system with combined CNN/RNN model. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases (pp. 149-158). Springer, Cham.

[14] Diep, N. N., Thuy, N. T. T., & Duy, P. H. (2018). Combination Of Multi-Channel Cnn And Bilstm For Host-Based Intrusion Detection. Southeast Asian Journal Of Sciences, 6(2), 147-159.

[15] Soni, J., Prabakar, N., & Upadhyay, H. (2019, December). Behavioral Analysis of System Call Sequences Using LSTM Seq-Seq, Cosine Similarity and Jaccard Similarity for Real-Time Anomaly Detection. In *2019 International Conference on Computational Science and Computational Intelligence (CSCI)* (pp. 214-219). IEEE.

[16] Forrest, S., Hofmeyr, S. A., Somayaji, A., & Longstaff, T. A. (1996, May). A sense of self for unix processes. In *Proceedings 1996 IEEE Symposium on Security and Privacy* (pp. 120-128). IEEE.

[17] Hofmeyr, S. A., Forrest, S., & Somayaji, A. (1998). Intrusion detection using sequences of system calls. *Journal of computer security*, *6*(3), 151-180.

[18] Hubballi, N., Biswas, S., & Nandi, S. (2011, January). Sequencegram: n-gram modeling of system calls for program based anomaly detection. In *2011 Third International Conference on Communication Systems and Networks (COMSNETS 2011)* (pp. 1-10). IEEE.

[19] Hubballi, N. (2012, January). Pairgram: Modeling frequency information of lookahead pairs for system call based anomaly detection. In *2012 Fourth International Conference on Communication Systems and Networks (COMSNETS 2012)* (pp. 1-10). IEEE.

[20] Wang, X., Yu, W., Champion, A., Fu, X., & Xuan, D. (2007, September). Detecting worms via mining dynamic program execution. In *2007 Third International Conference on Security and Privacy in Communications Networks and the Workshops-SecureComm 2007* (pp. 412-421). IEEE.

[21] Rieck, K., Holz, T., Willems, C., Düssel, P., & Laskov, P. (2008, July). Learning and classification of malware behavior. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment* (pp. 108-125). Springer, Berlin, Heidelberg.

[22] Liao, Y., & Vemuri, V. R. (2002). Using text categorization techniques for intrusion detection. In *11th USENIX Security Symposium (USENIX Security 02)*.

[23] DARPA Intrusion Detection Dataset. http://www.ll.mit.edu/ideval/data/

[24] Forrest, S. (2010). University of New Mexico (UNM) Intrusion Detection Dataset. http://www.cs.unm.edu/~immsec/systemcalls.htm

[25] Creech, G. and Hu, J. (2013) Generation of a New IDS Test Dataset: Time to Retire the KDD Collection. Wireless Communications and Networking Conference (WCNC 2013), Shanghai, 7-10 April 2013, 4487-4492.

[26] Creech, G. (2014) Developing a High-Accuracy Cross Platform Host-Based Intrusion Detection System Capable of Reliably Detecting Zero-Day Attacks. Ph.D. Dissertation, University of New South Wales, Sydney.

[27] Creech, G. and Hu, J. (2014) A Semantic Approach to Host-Based Intrusion Detection Systems Using Contiguous and Discontiguous System Call Patterns. IEEE Transactions on Computers, 63, 807-819.

[28] Xie, M., Hu, J. and Slay, J. (2014) Evaluating Host-Based Anomaly Detection Systems: Application of the One-Class SVM Algorithm to ADFA-LD. Proceedings of the 11th IEEE International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2014), Xiamen, 19-21 August 2014, 978-982.

[29] Soni, J., Peddoju, S. K., Prabakar, N., & Upadhyay, H. (2021). Comparative Analysis of LSTM, One-Class SVM, and PCA to Monitor Real-Time Malware Threats Using System Call Sequences and Virtual Machine Introspection. In *International Conference on Communication, Computing and Electronics Systems* (pp. 113-127). Springer, Singapore.

[30] Xie, M. and Hu J. (2013) Evaluating Host-Based Anomaly Detection Systems: A Preliminary Analysis of ADFA-LD. Proceedings of the 6th IEEE International Congress on Image and Signal Processing (CISP 2013), Hangzhou, 16-18 December 2013, 1711-1716.

[31] Xie, M., Hu, J., Yu, X. and Chang, E. (2014) Evaluating Host-Based Anomaly Detection Systems: Application of the Frequency-Based Algorithms to ADFA-LD. Proceedings of 8th International Conference on Network and System Security (NSS 2014), Lecture Notes in Computer Science, 8792, 542-549.

[32] Soni, J., Prabakar, N., & Upadhyay, H. (2019). Comparative Analysis of LSTM Sequence-Sequence and Auto Encoder for real-time anomaly detection using system call sequences.

[33] Medsker, L. R., & Jain, L. C. (2001). Recurrent neural networks. *Design and Applications*, *5*, 64-67.

[34] Albawi, S., Mohammed, T. A., & Al-Zawi, S. (2017, August). Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)* (pp. 1-6). IEEE.

[35] Rathore, P., Soni, J., Prabakar, N., Palaniswami, M., & Santi, P. (2021). Identifying groups of fake reviewers using a semisupervised approach. *IEEE Transactions on Computational Social Systems*, *8*(6), 1369-1378.

[36] Thejas, G. S., Soni, J., Boroojeni, K. G., Iyengar, S. S., Srivastava, K., Badrinath, P., & Upadhyay, H. (2019, December). A multi-time-scale time series analysis for click fraud forecasting using binary labeled imbalanced dataset. In *2019 4th International Conference on Computational Systems and Information Technology for Sustainable Solution (CSITSS)* (Vol. 4, pp. 1-8). IEEE.

[37] Soni, J., Prabakar, N., & Upadhyay, H. (2020). Visualizing High-Dimensional Data Using t-Distributed Stochastic Neighbor Embedding Algorithm. In *Principles of Data Science* (pp. 189-206). Springer, Cham.