International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488|

||Volume 8, Issue 6, June 2020||

Intrusion Detection System using Deep Learning

Divyarani P. Babar, Dr. Pankaj Agarkar

P.G. Student, Department of Computer Engineering, Dr. D. Y. Patil School of Engg, Pune, India

Professor, Department of Computer Engineering, Dr. D. Y. Patil School of Engg, Pune, India

ABSTRACT: Intrusion can be malicious software or a program or a script which can be harmful to the user's computer. These malicious programs can perform a variety of functions, including stealing, encrypting or deleting sensitive data, altering or hijacking core computing functions and monitoring users' computer activity without their permission. There are various entry points for these programs and scripts in the user environment, but only one way to remove them is to find them and kick them out of the system which isn't an easy job as these small piece of script or code can be anywhere in the user system. In this research system implements a intrusion detection classification approach using deep learning based Recurrent Neural Network (RNN) technique, the system carried out static as well as hybrid classification using various intrusion and network anomaly dataset in large data environment

KEYWORDS : Machine learning, Security, Deep Learning, Entropy, Feature extraction, weight generation, Ph file systems

I.INTRODUCTION

Today, the use of technology and internet is growing at an exponential rate and so is the rate of malwares getting developed. Internet is used for online banking and online shopping and for many more things which has increased our dependence on it. Even though anti-virus companies have come up with technologies to block malwares from infiltrating systems, newer malwares are getting developed which can pass through those bars. This makes it inevitable to develop a technique which is automated to analyse malwares. Also, analysing huge amount of intrusion manually is not a solution and hence the need for an atomized method. There are in two methods to analyse malware, namely, Static and Dynamic. Static analysis consists of analysing the intrusion without executing it. Even though it’s faster it does not prove to be of much use if the intrusion is packed, encrypted. This brings the dynamic analysis in picture which requires the intrusion to be executed in a sandboxed environment. Even if dynamic analysis is apt for most obfuscated malwares, some malwares detect being executed in a sandbox environment and thus don’t reveal their malicious payload. It’s therefore not reliable to pick only one of these methods. A hybrid approach makes use of the advantage of both the methods. Although one issue in the hybrid approach is the size of the feature space which causes a limit on the scalability. Various machine learning and classification algorithms have been used for categorized the normal as well as abnormal classes. Distributed approach has used for parallel processing environment and haggle dataset has used for static as well as dynamic classification of system.

II.MOTIVATION OF THE PROJECT

- According to the challenge, one of the utmost troubles that antiintrusion faces today is the huge amount of data and _les which need to be assessed for potential malicious intent.
- For instance, Microsofts real-time detection anti-intrusion products exist on more than 160M computers worldwide and examine around 700M computers monthly. This generates millions of daily data points to be analyzed as potential malware.
- One of the major reasons for these high volumes of different files is polymorphism to exade detection, which means, malicious _les belonging to same intrusion family, with same forms of malicious behavior, are constantly modified or obfuscated using certain tactics, such thatthey look like many different files.
- To effectively analyze and classify such large amounts of _les, we need to be able to group them and identify their respective families.
- Because of the rise in intrusion threats each day, classifying the intrusion has become a dire need.

III.LITERATURE SURVEY

Yashu Liu et. al. [1] proposed a system A New Learning Approach to Intrusion Classification using Discriminative Feature Extraction. System carried out proposes a novel method to analyze intrusion visually and classify intrusion families. The method transforms intrusion binary files to gray-scale images. To obtain discriminative features, system presents a new learning framework which is formulated into a multi-layered model to characterize and analyze intrusion images using bag-of visual-words (BoVW). Starting from existing local descriptors (LBP or dense SIFT), system group them into blocks and build histograms. The extracted features are more flexible than global features (e.g. GIST) and more robust than local features. System evaluates this method on three datasets, which are all from the Windows platform.

Ming Fanet.al[2] author Android Intrusion Familial Classification and Representative Sample Selection via Frequent Sub graph Analysis. The novel approach that constructs frequent sub graphs to represent the common behaviors of intrusion samples that belong to the same family. Moreover, we propose and develop FalDroid, a novel system that automatically classifies Android intrusion and selects representative intrusion samples in accordance with graphs.

YAN Hanbing et.al [3]The system Pairwise rotation invariant co-occurrence local binary pattern (PRICoLBP) feature, and further extend it to incorporate the Term frequency-inverse document frequency (TFIDF) transform. Different from other static analysis techniques, our method not only achieves better linear reparability, but also appears to be more resilient to obfuscation. In addition, we evaluate PRICoLBPTFIDF comprehensively on three datasets from different perspectives, e.g., classification performance, classifier selection and performance against obfuscation.

Yousefi Azar, Mahmood, et al. [4] System scheme to detect intrusion which system call Malytics. It is not dependent on any particular tool or operating system. It extracts static features of any given binary file to distinguish intrusion from benign. Mystics consists of three stages: feature extraction, similarity measurement and classification. The three phases are implemented by a neural network with two hidden layers and an output layer. system show feature extraction, which is performed by tf- sim hashing, is equivalent to the first layer of a particular neural network. system evaluate Malytics performance on both Android and Windows platforms. Malytics outperforms a wide range of learning-based techniques and also individual state-of-the-art models on both platforms. system also show Malytics is resilient and robust in addressing zero-day intrusion samples.

Haipeng Caiet. Al [5] Dynamic app classification technique, to complement existing approaches. By using a diverse set of dynamic features based on method calls and ICC Intents without involving permission, app resources, or system calls while fully handling reflection, Droid Cat achieves superior robustness than static approaches as system as dynamic approaches relying on system calls. The features system are distilled from a behavioral characterization study of benign versus malicious apps. Through three complementary evaluation studies with 34,343 apps from various sources and spanning the past nine years, system demonstrated the stability of Droid Cat in achieving high classification performance and superior accuracy compared to two state-of-the-art peer techniques that represent both static and dynamic approaches. Overall, Droid Cat achieved 97% F1-measure accuracy consistently for classifying apps evolving over the nine years, detecting or categorizing malware, 16% to 27% higher than any of the two baselines compared.

Fahad Alswaina et.al [6]. Presented an Android Intrusion Permission-Based Multi-Class Classification Using Extremely Randomized Trees. Through our research, we developed a reverse engineering framework (Rev Eng).Within RevEng, the applications' permissions were selected, and then fed into machine learning algorithms. Through our research, we created a reduced set of permissions by using extremely randomized trees that achieved high accuracy and a shorter execution time. Furthermore, system conducted two approaches based on the extracted information. Approach one used binary value representation of the permissions.

JINGJING GU, BINGLIN SUNet.al [7] authors Consortium Blockchain-based Intrusion Detection in Mobile Devices. Specifically, in view of different intrusion families in Android-based system, system perform feature modeling by utilizing statistical analysis method so as to extract intrusion family features, including software package feature, permission and application feature, and function call feature. Moreover, for reducing false-positive rate and improving the detecting ability of intrusion variants, system design a multi-feature detection method of Android-based system for detecting and classifying malware. In addition, system establish a fact-base of distributed Android malicious codes by blockchain technology.

PENGBIN FENG, JIANFENG MAet.al[8]as A Novel Dynamic Android Intrusion Detection System With Ensemble Learning. Effective dynamic analysis framework, called EnDroid, in the aim of implementing highly precise intrusion detection based on multiple types of dynamic behavior features. These features cover system-level behavior trace and common application-level malicious behaviors like personal information stealing, premium service subscription, malicious service communication. In addition, EnDroid adopts feature selection algorithm to remove noisy or irrelevant features and extracts critical behavior features. Extracting behavior features through runtime monitor, EnDroid is able to distinguish malicious from benign applications with ensemble learning algorithm.

Jin Liy, Lichao Sun et.al[9] SIGPID, a intrusion detection system based on permission usage analysis to cope with the rapid increase in the number of Android malware. Instead of extracting and analyzing all Android permissions, we develop 3-levels of pruning by mining the permission data to identify the most significant permissions that can be effective indistinguishing between benign and malicious apps. SIGPID then utilizes machine-learning based classification methods to classify different families of intrusion and benign apps. Our evaluation finds that only 22 permissions are significant. The results indicate that when Support Vector Machine (SVM) is used as the classifier, we can achieve over 90% of precision, recall, accuracy, and F-measure, which are about the same as those produced by the baseline approach while incurring the analysis times that are 4 to 32 times less than those of using all permissions.
JIANWEN FU, JINGFENG XUE et.al[10] A new visualization method for characterizing intrusion globally and locally to achieve fast and effective fine-grained classification. The take a new approach to visualize intrusion as RGB colored images and extract global features from the images. Gray Level Co-occurrence Matrix(GLCM) and color moments are selected to describe the global texture features and color features respectively, which produces low-dimensional feature data to reduce the complexity of training model. Moreover, a series of special byte sequences are extracted from code sections and data sections of intrusion and are processed into feature vectors by Sim hash as the local features. Finally, we merge the global features and local features to perform intrusion classification using RF(Random Forest), KNN(K-Nearest Neighbor) and SVM (Support Vector Machine).

IV.PROPOSED SYSTEM DESIGN

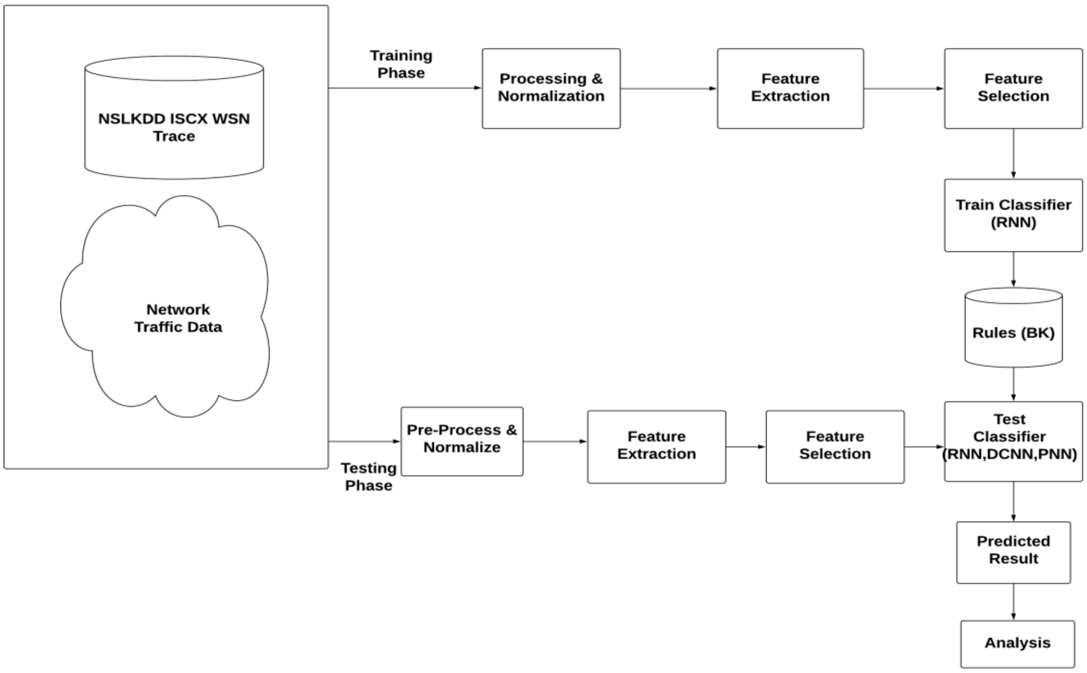

Figure 1 : Proposed system design

Training Phase:
Step 1: To generate the rules based on supervised learning algorithm we used synthetic dataset like KDDCup99, NSLKDD, ISCX and WSN Trace etc.
Step 2: Select features for each selected instances and execute the train classifier to generate the training rules.
Step 3: The result of training modules called as training rules or policies which has stored in repository those defined as Background Knowledge (BK).

System Testing Phase:

Step 1: System accumulate the network traffic data from network audit log data or NSLKDD

Step 2: Read each input packets from network environment and apply various machine learning as well as deep learning algorithm (RNN).

Step 3: RNN has apply to generate the runtime weight for each input packet and validate with the quality threshold.

Step 4: Classify the detected packet as master attack like DoS, PROBE, U2R, R2L, Network attacks etc), and finally also shows the subtype of attack for respective class.

Algorithms

Recurrent Neural Network

Input: Training set from PE files, network log or data packets, Attribute validation policies Background Knowledge (BK) policies, Threshold 'th'.

Output: Rule set as policies or signatures.

```
Step 1:a. Read values from PE file header fields to get Feature Vector.
          b. Read data from a network connection to append in Feature []
Step 2: Validate each attribute for the preprocessing phase
Step 3: Normalized irrelevant attribute, and get normalized set
        NormSet[] ← {Att[i……n]}
Step 4 : for each (Feature intoNormSet !=Null)
Step 5 :     calculate weight w= (Feature, Bk)
Step 6 :     if (w>=th)
             Ruleset.add ← {Feature,Label}
              End if
              End for
Step 7 : return Ruleset.
```

Testing Procedure

Input: Test Dataset which contains various test instances TestDBLits [], Train dataset which is build by training phase TrainDBLits[] , Threshold Th.

Output: HashMap <class_label, SimilarityWeight> all instances which weight violates the threshold score.

Step 1: For each read each test instances using below equation

$$testFeature(m) = \sum_{m=1}^{n} (.featureSet[A[i] \dots \dots . . A[n] \leftarrow \text{TestDBLits})$$

Step 2 : extract each feature as a hot vector or input neuron from $testFeature(m)$ using below equation.

$$\text{Extracted\_FeatureSetx}[t\ldots\ldots n] = \sum_{x=1}^{n}(t) \leftarrow testFeature\ (m)$$

Extracted_FeatureSetx[t] contains the feature vector of respective domain

Step 3: For each read each train instances using below equation

$$trainFeature(m) = \sum_{m=1}^{n} (.featureSet[A[i] \ldots \ldots . A[n] \leftarrow \text{TrainDBList})$$

Step 4 : extract each feature as a hot vector or input neuron from $testFeature(m)$ using below equation.

$$\text{Extracted\_FeatureSetx[t……n]} = \sum_{x=1}^{n}(\text{t}) \leftarrow testFeature\ (\text{m})$$

Extracted_FeatureSetx[t] contains the feature vector of respective domain.

Step 5 : Now map each test feature set to all respective training feature set

$$weight = calcSim\ (\text{FeatureSetx} \mid\mid \sum_{i=1}^{n} \text{FeatureSety[y]})$$

Step 6 : Return weight

V.RESULTS AND DISCUSSION

The proposed The current machine learning algorithm and the deep learning algorithms were used in two different ways by the Project. We have also introduced computational research in base system which can recommend algorithms with KDDCUP99 data set and power-contributing architecture incorporated with deep learning algorithms with custom network audit dataset. The program measured the consistency of the description and the time complexity in the same setting. Figure 2 above demonstrates the classification performance of data collection by KDDCUP using the density-based approach of the machine learning algorithm program Figure 3 used to classify and predict the precision of the proposed system using different methods like RNNalgorithm**.**

A. *EXISTING SYSTEM RESULTS*

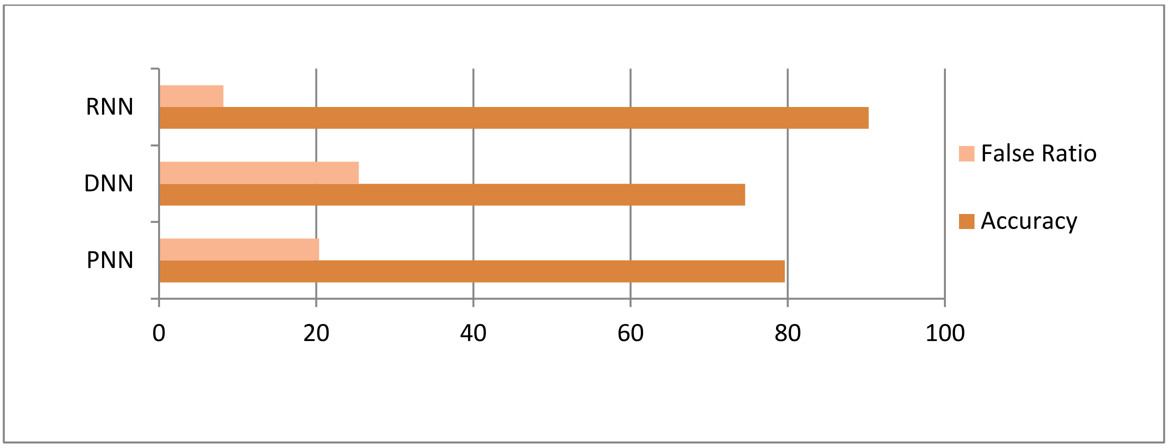

Figure 2: Detection accuracy for KDD : CUP99 dataset using machine learning

The above figure 2 Shows accuracy of kddCup 99 results classification, with five different classes. Average software output is around the algorithm for the machine learning 88.50% for all classes.

B. *PROPOSED RESULT*

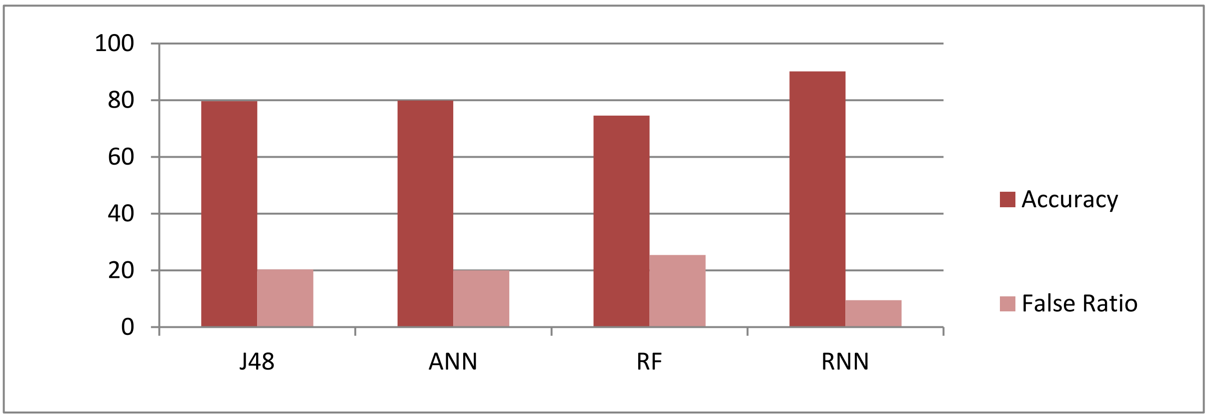

Figure 3 : Detection accuracy various network dataset using deep learning (RNN)

The above figure 3 Shows average efficiency of identification in various databases, of (n) different classes. The system's mean performance with the machine learning algorithm is around 95% for all (n) classes.

VI.CONCLUSIONS

In this work, proposed a deep learning based RNN-IDS method to proposed effective IDs system. We utilized the synthetic based intrusion dataset - NSL-KDD to evaluate anomaly detection accuracy. In future, we plan to implement an IDS using deep learning technique on cloud environment. Additionally, We Evaluate and compare different deep learning technique, namely. RNN, DNN, CNN and PNN on NSL-KDD dataset to detect intrusions in the network. The system basically works like machine learning as well as reinforcement algorithm to evaluate the unknown instances during the data testing. The effective rule system provides better classification and detection accuracy for classes. Various datasets has used for experiment analysis to evaluate the algorithm performance with multiple test and conclude we get result on satisfactory level. After upon completion of this analysis, we can conclude that it is possible to use different techniques for detection, some soft computing and some approaches to classification to detect various attacks. Some system has worked with the application of various rules to identify baseline signature anomalies. For training and testing purposes, the KDD cup data set was used. The device essentially shows the highest detection accuracy for attacks, but none of them focuses on inconsistent detection or misuse of attacks detection.

REFERENCES

[1] Liu Y, Lai YK, Wang Z, Yan H. A New Learning Approach to Intrusion Classification using Discriminative Feature Extraction. IEEE Access. 2019 Jan 11.

[2] Fan M, Liu J, Luo X, Chen K, Tian Z, Zheng Q, Liu T. Android Intrusion Familial Classification and Representative Sample Selection via Frequent Subgraph Analysis. IEEE Transactions on Information Forensics and Security. 2018 Feb.

[3] Yan H, Zhou H, Zhang H. Automatic Intrusion Classification via PRICoLBP. Chinese Journal of Electronics. 2018 Jul 1;27(4):852-9.

[4] Yousefi-Azar M, Hamey L, Varadharajanz V, Cheng S. Malytics: A Intrusion Detection Scheme. arXiv preprint arXiv:1803.03465. 2018 Mar 9.

[5] Cai H, Meng N, Ryder B, Yao D. DroidCat: Effective Android Intrusion Detection and Categorization via App-Level Profiling. IEEE Transactions on Information Forensics and Security. 2018 Nov 1.

[6] Alswaina F, Elleithy K. Android Intrusion Permission-Based Multi-Class Classification Using Extremely Randomized Trees. IEEE Access. 2018;6:76217-27.

[7] Gu J, Sun B, Du X, Wang J, Zhuang Y, Wang Z. Consortium Blockchain-Based Intrusion Detection in Mobile Devices. IEEE Access. 2018;6:12118-28.

[8] Feng P, Ma J, Sun C, Xu X, Ma Y. A Novel Dynamic Android Intrusion Detection System With Ensemble Learning. IEEE Access. 2018 Jun 6.

[9] Li J, Sun L, Yan Q, Li Z, Srisa-an W, Ye H. Signi?cant Permission Identi?cation for Machine Learning Based Android Intrusion Detection. IEEE Transactions on Industrial Informatics. 2018 Jan 12.

[10] Fu J, Xue J, Wang Y, Liu Z, Shan C. Intrusion Visualization for Fine-Grained Classification. IEEE ACCESS. 2018 Jan 1;6:14510-23.