A Stacked Ensemble Framework for Detecting Malicious Insiders

Abolaji B. Akanbi, Adewale O. Adebayo, Sunday A. Idowu, Ebunoluwa E. Okediran

ABSTRACT- One of the mainstream strategies identified for detecting Malicious Insider Threat (MIT) is building stacking ensemble Machine Learning (ML) models to reveal malevolent insider activities through anomalies in user activities. However, most anomalies found by these learning models were not malicious because MIT was treated as a single entity, whereas there are various forms of this threat with their own distinct signature. To address this deficiency, this study focused on designing a stacked ensemble framework for detecting malicious insider threat which utilizes a one scenario per algorithm strategy. A model that can be used to test the framework was proposed.

KEYWORDS- Ensemble Learning, Malicious Insider Threat, Machine Learning, Stacked Generalization.

I. INTRODUCTION

Insider threat have been considered to be very significant and likely to pose greater damage than external threats [1]. Insider attacks accounted for 34% of all data breaches in 2018 as addressed by IBM [3] and Verizon [2], and it was also the top threat in 2018 with 51% and cybercriminals as the second top threat with 44% [4]. Technical solutions to these threats do not suffice since this issue had been identified as a 'people issue' [5], including behavioral analysis is also required. These threats can be intentional (malicious) or unintentional [6].

To perform behavioral analysis, the machine learning based method is used. This methodology is utilized by collecting, aggregating, feature extracting, and parsing log files from various technical resources in an organization, following which learning algorithms (most commonly supervised and unsupervised) are used to evaluate the data and detect anomalies, which could reveal malevolent insider activities. Even with the most optimal and higher performing learning algorithms, most anomalies found by these algorithms are not malicious, thus leading to the creation of many false alarms in detection systems [7] [8]. This poses the problem of how to increase the predictive power of learning algorithms to reduce the rate of false positives in the detection of malevolent insider threat.

Malicious insider threats can take various forms such as insider Information Technology (IT) sabotage and insider fraud, among others. These forms/scenarios do not carry the same signature but, in most literature, malicious insider threat is usually treated as a single class problem. By treating it as a single class problem, the predictive power of the model is affected. This suggests that instead of training various algorithms based on classifying threats across a wide spectrum, log files activities can be broken down into various categories based on various malicious insider threat scenarios that can be fed into different algorithms to generate different models that will be combined to produce a single model, thereby giving the final model the ability to generalize accurately on all types of malicious insider threat, which will improve detection and directly reducing false alarms in systems that adopt the model.

For this to be achieved, Stacking Ensemble Learning (SEL) method, where two or more heterogenous algorithms' predictions are combined and used as an input to another algorithm called a meta-classifier or meta-regressor [8] [9] was adopted. Hence, this study designed a framework that utilizes Stacking Ensemble Learning (SEL) deploying one scenario per algorithm strategy. Base models generated from two or more heterogeneous base-learners that were trained with data streamlined to scenarios of malicious insider threat were combined as input to a generalizer to create more optimal model, in order to reduce false alarms.

II. CLOSELY RELATED WORKS.

Utilization of a single classification algorithm to detect malicious insider threat has the problem of having high false positives and false negatives. To minimize this problem, stronger algorithms such as Random Forest were chosen [13]. To determine the most optimal algorithm, comparative analysis of various learning algorithms used in the detection of this threat were performed to assess the

Manuscript received July 19, 2020.

Abolaji B. Akanbi, Department of Computer Science, Babcock University, Ogun State, Nigeria, (boljaeakanbi@gmail.com)

Adewale O. Adebayo, Department of Information Technology, Babcock University, Ogun State, Nigeria

Sunday A. Idowu, Department of Software Engineering, Babcock University, Ogun State, Nigeria

Ebunoluwa E. Okediran, Department of Computer Science, Babcock University, Ogun State, Nigeria

speed, accuracy, complexity and detection rate of such algorithms [14]. Based on the ‘No Free Lunch’ theorem that states that no one model works best for every problem, the comparative analysis done were not as impactful because various algorithms have the ability to perform better in certain circumstances. Even with the most optimal and higher performing models, most anomalies found were not malicious which led to the creation of many false alarms in systems that adopt such algorithms. To improve on this, models or systems which utilizes a single high performing algorithm with an authentication module applied, such as fingerprint or facial recognition were developed. This was also not as impactful on reducing false positives because the main problem is how to improve the predictive power of these detection algorithms.

To solve this deficiency, ensemble methods were utilized. To this effect, a framework that uses the stacking ensemble method for detecting malicious insider threat using Principal Component Analysis PCA and Regression Analysis (RA) to detect variance in user behavior was proposed [9]. Another framework for detecting anomaly in users’ activities in an organization by using an ensemble of Negative Selection Algorithms to classify these activities into normal or malicious classes was also proposed [10]. The system was able to classify the input data with an accuracy of 89.00 and Area Under the Curve (AUC) value of 86.34%. Based on the result, there is an improvement in the detection rate, but it can still be further improved.

Treating malicious insider threat as a single class problem can lead to many forms of the threat with distinct signatures being undetected, necessitating the need for methods to train algorithms properly on all the forms to further reduce the rate of false positive and false negatives. Based on the experimental result of Junhong et al. [8], their proposed framework of an ensemble of anomaly detection algorithms was able to detect malicious activity in imbalanced data (more non-malicious and very less malicious) based on a leaker scenario.

III. METHODOLOGY

A. Model Design

In the existing system shown in fig 1, all essential logs are combined and passed through the data pre-processing stage before the combined logs are then used to train the selected algorithms. After the algorithms have been trained and tested, the decision maker will categorize users into normal or malicious category. By combining the logs and selecting the features on the combined logs, malicious insider threat is being treated as a single entity, which will lead to the system having the inability to detect all forms of malicious insider threat, therefore having the risk of some threats going undetected. To further improve on this work, the researcher designed a framework that categorizes the logs based on the signatures of the various types of malicious insider threat known, so as to give the system an improved ability to detect all forms of malicious threat, which will further reduce false alarms when adopted.

To properly test this idea, the stacking ensemble method was utilized. This method provides the possibility of different algorithms to be trained on different training data that has being streamlined and categorized based on various scenarios or forms of malicious insider threat to improve the detection of this threat. Succeeding sections present the necessary steps taken to properly implement the model.

B. Data Description and Preprocessing

The publicly available Computer Emergency Response Team (CERT) insider threat test dataset from the Software Engineering Institute of Carnegie Mellon University (CMU)’s insider threat repository was obtained for this research. This dataset consists of benign synthetic data gotten from malicious insider threats.

For data preprocessing, feature selection was done. Features related to each scenario was selected from each subset of the training dataset based on Elmrabit, Yang, and Yang [11] overview of various basic characteristics of malicious insider threats. Two main problems may occur when features are not selected properly: Curse of dimensionality and overfitting of model.

C. Stacking Ensemble Method

For stacking ensemble method, the chosen base-learners must be heterogeneous, the more diverse the classifiers the better the result. To model a stacker, the following was done:

- Data was split into ratio 80:20. 80% for training data and 20% for testing data.
- Bootstrap Sampling was used to create subset of the training data.
- For each training set, the features were finetuned to fit the scenario respective algorithms have been chosen to model.
- K-fold cross validation was used to tune the model.
- Result of the k-test fold for each model was stacked to form an input for the meta learner for training.
- The stacked model was then tested with the 20% testing dataset.
- The stacked model was evaluated by comparing the performance of the base-models with the performance of the stacked model

D. Pseudo Code for Stacking Ensemble Method

```
Input:
Data set D = {$(X_1, Y_1), (X_2, Y_2), (X_3, Y_3), …, (X_n , Y_n)$};
Subset the data D to training Dt and testing Ds
Tier-1 Level Learning Algorithms $C_1, C_2, C_3, ..., C_z$;
Tier-2 Level Learning Algorithms C*;
Process:
Training:
    Apply K-fold cross validation on Dt to give D-k and Dk for training and tuning the base-learners ($C_1, C_2, C_3, ..., C_z$) respectively
Train $C_1, C_2, C_3, ..., C_z$ on $D_{-k}$
Tune $C_1, C_2, C_3, ..., C_z$ on $D_k$
    Test C1, C2, C3, ..., Cz on Ds
    Build a new data set D* that contains yn in addition to
    the base-learners ‘outputs
    Train the meta-Classifier C* on the new data set D*
    Testing:
    Bring a new data set D**
    Test/Predict D** on the base-learners
    Join D** with the predicted classes
    Test/Predict D** on the meta-learner
    The final predicted output is the correct class for D**

End
```

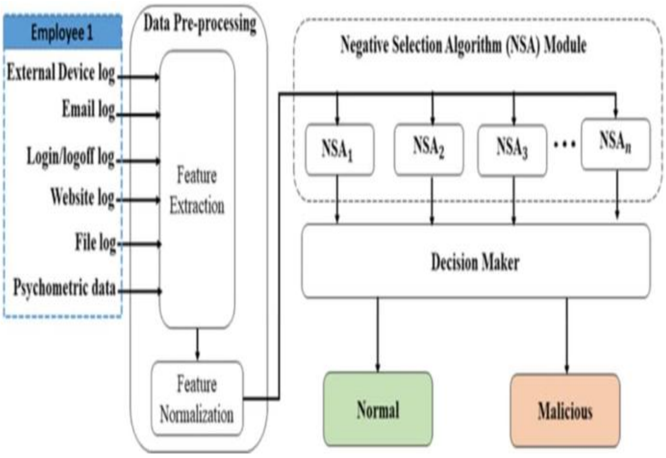
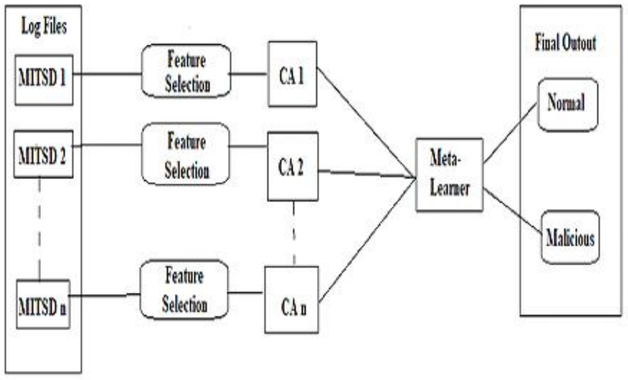


Fig 1: Existing Model [10]

IV. DESIGNED FRAMEWORK

Unlike the existing system shown in fig 1, where all the log files are combined, the designed framework takes into account the different signature of various malicious insider threat scenario to break down the log files and feed them into different algorithms. The models generated are combined by a meta-classifier to generate a model, which can detect all forms of malicious threat, thereby leading to a reduction in insider false alarms in systems that adopt the model. Fig 2 depicts the designed framework.



MITSD: Malicious Insider Threat Scenario Data
CA: Classification Algorithm

Fig 2: Designed Framework

V. PROPOSED MODEL

To test the designed framework, a stack ensemble model was proposed. Based on the proposed model presented in fig 3, the base-learners and the stacker were trained on the same data. In order to tune the “Level 1" classifiers, k-fold validation was used to select some set of hyperparameters. Then the base-learners was fit on the train fold and predict on the test fold. Predictions on each of the test fold became the new fold for the stacker. Then, training and testing set was created using the new folds for the stacker

In simple details, the first phase of the ensemble is to create ‘Level 1’, which is by training the base classifiers. The second step is feeding the outputs of the “Level 1” to train the meta-classifier in “Level 2”. Three classification learning algorithms, K-Nearest Neighbor, Support-Random Forest and Artificial Neural Network, will be used on the dataset D while XGBoost Classifier which has been established as a suitable algorithm for predicting rare events is the meta-classifier that was used to make prediction based on the combination of the output of these algorithms.

These algorithms were selected because they have been used by various researchers to detect malicious insider threat and they have been established as high performing detection classification algorithms. For stacking ensemble, the base-learners must be heterogeneous, the more diverse the classifiers the better the result. These chosen classifiers are from different classification families and provides different models.

To evaluate the model, the performance of the base-learners and the meta-learner was compared based on the chosen evaluation metrics.

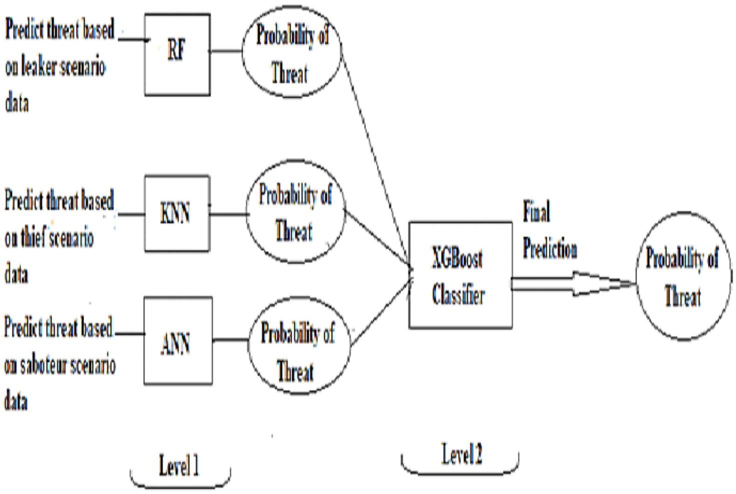


Fig 3: Proposed Model

VI. EVALUATION METRICS

A.. *Accuracy*

This is a measure of the correctly classified (correctly selected; true positive and correctly rejected; true negative) classes compared to the total classified sample size and it is calculated as can be seen in the below equation:

Accuracy = True Positive + True Negative
True Positive + True Negative + False Positive +False Negative

B. Sensitivity

This is the ability of the algorithm to detect the positive classes correctly. In this case, the ability of the machine learning algorithms to detect the malicious threat correctly. This represented by the equation below:

Sensitivity = True Positive
True Positive + False Negative

C. Specificity

This is the ability of the algorithm to detect the negative classes correctly. In this case, the ability of the machine learning classification algorithms to detect the non-malicious threat correctly. This represented by the equation below:

Specificity = True Negative
True Negative + False Positive

D. Area under the Curve (AUC)

AUC is a representation plot of the true positive rate vs the false positive rate. This curve shows the sensitivity and specificity tradeoff.

VII. EXPECTED OUTCOME

Using the publicly available CERT insider threat test dataset acquired from the Software Engineering Institute of CMU’s insider threat repository, the base-learner classifiers such as K-Nearest Neighbor, Artificial Neural Network and Random Forest will be trained on a bootstrapped sampled of each scenario dataset with k-fold cross validation. The prediction of these classifiers will then be combined as an input for the meta-classifier, XGBoost classifier, to produce a single more accurate predictive model to improve accuracy of malicious insider threat detection. The stacked model will be created and tested on RapidMiner version 9.3.

The evaluation of the model will be done by comparing the predictive performance of the non-stacked models with the new stacked model based on the following evaluation metrics: Accuracy, Sensitivity, Specificity, Area under the Curve (AUC). Then, the findings form the evaluation will help to deduce whether the model generated by the meta-classifier will be able to detect malicious insider threat correctly and greatly reduce false alarms in insider threat detection systems when adopted.

VIII. CONCLUSION AND RECOMMENDATION

This study fills the gap existing literatures by proposing a stacked ensemble model based on different scenarios of malicious insider threat that will be able to accurately classify insiders in order to reduce false alarm. Not treating malicious insider threat as a single problem, and focusing on the different signatures of each scenario will greatly improve accuracy and reduce false alarms. This will benefit many researchers in insider threat mitigation community and in a long run can be adopted by security companies or IT departments of organization to improve their security systems

REFERENCES

[1] P. A. Legg, "Visualizing the insider threat: Challenges and tools for identifying malicious user activity," in Proceedings of the 2015 IEEE Symposium on Visualization for Cyber Security, Chicago, IL, USA, 2015.

[2] Verizon, "2019 Data Breach Investigations Report," Verizon, United States of America, 2019.

[3] IBM, "IBM X-Force Threat Intelligence Index," 2018. [Online]. Available: https://www.ibm.com/security/data-breach/threat-intelligence

[4] Thales Security, " Thales Data Threat Report," 2018. [Online]. Available: http://go.thalesesecurity.com/rs/480-LWA-970/image/2018-data-threatreport-global-edition-ar.pdf

[5] A. E. Abdallah and I. A. Gheyas, "Detection and prediction of insider threats to cyber security: a systematic literature review and meta-analysis," Big Data Analytics, 2016.

[6] S. E. Adewumi, C. K. Ayo and T. O. Oladimeji, "Review on Insider Threat Techniques," in Journal of Physics: Confernce Series, 2019.

[7] S. J. Berdal, A holistic approach to insider threat detection, Doctoral thesis, University of Oslo, 2018.

[8] K. Haedong, K. Junhong, P. Minsik, K. Pilsung and C. Suhyoun, "Insider Threat Detection Based on User Behavior Modelling and Anomaly Detection Algorithms," Journal of Applied Sciences, pp. 1-5, 2019.

[9] A. Kondaveeti, "Insider Threat Detection: Detecting variance in user behavior using an ensemble approach," 2017. [Online]. Available: https://content.pivotal.io/blog/insider-threat-detection-detecting-variance-in-user-behavior-using-an-ensemble-approach.

[10] O. Igbe and T. Saadawi, "Insider Threat Detection using an Artificial Immune System Algorithm," IEEE, pp. 10-19, 2018.

[11] N. Elmrabit, S.-H. Yang and L. Yang, "Insider Threats in Information Security," in 21st International Conference on Automation and Computing (ICAC), 2015.

[12] A. N. Erekat, An Ensemble Learning Approach for Surgery Cancellation Prediction for Efficient Operating Room Planning, New York: ProQuest LLC, 2017.

[13] E. B. M. Bashier, M. B. Khan and M. Mohammed, "Machine Learning: Algorithms and application," in Machine Learning: Algorithms and application, Boca Raton, CRC Press, 2016, pp. 2-16.

[14] W. Li, W. Meng and L. F. Kwok, "Enhancing collaborative intrustion detection networks against inside attacks using supervised intrusion sensitive-based trust management model," Network and Computer Applications, pp. 135-145, 2017.