JOETSR-International Journal of Emerging Trends in Scientific Research
Volume 4, Issue 1 | January-2026
e-ISSN: 2997-8718 Print ISSN: 2997-870X

Strengthening Cyber Resilience: Early Detection in Network Intrusion Prevention

Chris Joseph Levics N.M
Assistant Professor, Department CSE (cyber security), KCG College of technology, Chennai, Tamil Nadu.
Email: Chris.csecs@kcgcollege.com

Received: 02-11-2025; Revised: 23-11-2025; Accepted: 18-12-2025; Published: 12-01-2026

Abstract

Traditional Network Intrusion Detection Systems (NIDSs) based on pattern matching are inherently limited because they can only identify attacks that correspond to predefined signatures. To overcome this shortcoming, Machine Learning–based NIDS (ML-NIDS) have been introduced, with the goal of detecting anomalies by learning and analyzing protocol behavior. Nevertheless, ML-NIDS remain susceptible to previously unseen attacks, much like signature-based systems. This study examines how ML-NIDS learn and demonstrates that attacks operating outside the feature space represented in the training data can bypass detection. As a mitigation strategy, the study proposes early classification of active sessions before they deviate beyond the model's learned detection boundaries as an effective means of prevention.

Keywords: ***DecisionTreeclassifier, RandomForestclassifier, XGBoost classifier, AdaBoost classifier, ANN, CNN.***

1.Introduction

There is a myriad of digital defence mechanisms in use in the fast paced world of cybersecurity to defend networked systems. Intrusion Detection Systems (IDS) are some of them, and it is critical to note that they detect and act against unauthorized access or malicious acts in a network or system. IDS operate by observing and inspecting network traffic or system logs to identify some suspicious patterns or aboriginal behaviors that can indicate a security intrusion(1). These systems are necessary to provide timely alerts to system administrators in order to protect digital assets and eliminate cyber threats.

IDS are also important in the protection of networks against more complex and advanced attacks. Pattern-matching methods, whereby network traffic is compared with known attack signatures, are the main methods of detection of Network intrusions, conventional Network Intrusion Detection System (NIDS). Even though these systems are effective in the detection of the threats that have already been identified, they have a major limitation; they cannot detect any new attack or an unknown attack that does not match with their signature databases.

a) Objective of The Study

This project is primarily aimed at designing a robust Network Intrusion Detection System based on an early machine learning based classification system in order to tell the difference between normal network traffic and malicious attacks. The paper compares the results of various machine learning algorithms, which are Stochastic Gradient Descent (SGD), Decision Tree, Random Forest, XGBoost, AdaBoost, Artificial Neural Networks (ANN), and Convolutional Neural Networks (CNN), to see their effectiveness in precise fault and attack classification(2).

b) Scope Of The Study

This research aims at implementing the anomaly detection methods so as to detect new and sophisticated attacks that are not based on the familiar attack patterns. It discusses the issue of ensemble learning in order to take the synergistic advantages of multiple machine learning models in order to improve their detection. Moreover, the paper also includes the use of explainable AI (XAI) methods to enhance the transparency and interpretability of the system decision-making process. The use also covers real-time monitoring and visualization of network traffic to prevent threats and mitigate them. It is also believed that collaboration with cybersecurity experts will include domain knowledge and will make the system itself more effective.

c) Problem Statement

In order to secure the computer networks, there should be good intrusion detection mechanisms. Most of the current systems that are used in intrusion detection, however, have usability issues and rely on obsolete detection processes(3). This project will deal with these shortcomings by creating an easy-to-use network intrusion detection tool that will utilize the latest machine learning algorithms. These cutting-edge algorithms are the most significant technological advancement, as machines can learn based on data and make correct predictions. Through the use

of the current machine learning methodologies, the proposed system will correctly identify normal network operations and malicious operations and therefore improve the security posture of organizations significantly.

2. Related Work

Over the past years, organizations embraced a broad variety of methods to safeguard and retain data and the main aim is to prevent the occurrence of internal and external threats on sensitive personal and organizational data(4). The real identification of attackers, when the opponents have adopted methods like masking of IP addresses and obfuscation of attack packets, is one of the biggest network security issues. Computer networks are composed of hardware and software each having various vulnerabilities and security risks. Attacks which are software-based are especially dangerous to the integrity of data and reliability of the system. Advanced system-level and programming-aware persons can scan log files to track and locate the system activities thus enhancing the security measures. Nevertheless, users who do not possess such technical skills might not see or diagnose any security intrusion, and their system is at risk of exploitation.

The method of network attack has numerous variations, and one of the most complicated issues on the matter of cybersecurity is the detection of insider threats. Users require the holistic security of all forms of malicious practices both internally based and externally based in the network(5). The primary aim of the Intrusion Detection Systems (IDS) is to identify external threats whereas Internal Intrusion Detection Systems (IIDS) aims at detecting malicious activities that are initiated by authorized internal users. The two methods are necessary towards a high level of network security(6).

Detection of malware remains a major issue because of the prevalence of sophisticated evasion methods used by the attackers such as polymorphism, obfuscation and zero-day exploits. The methods allow the malicious software to circumvent the old antivirus and antispyware programs that rely on signatures. To overcome this shortcoming, previous studies have suggested hybrid malware detection models to integrate signature based malware detection methods with genetic algorithms evolutionary methods. Under these structures, the two elements collaborate to identify new malware and automatically create new signatures thus increasing the performance of signature based detection systems(7).

Intrusion detection technologies are one of the most important mechanisms that can be used to reinforce the network defense as they allow systems to notice and react to an effective variety of cyberattacks. These technologies are quite effective in assisting system administrators in performing security operations and uphold information systems integrity. Pattern matching algorithms are the basis of much intrusion detection systems and feature heavily in modern intrusion detection systems. A number of researches have defined architectural systems of the IDS in terms of pattern matching and offer a thorough examination of the most important functional modules in the systems(8).

Intrusion Detection and Prevention Systems (IDPS) go further and provide an active capability of intrusion detection by preventing malicious activity. These systems are based on the known attack signatures and use pattern matching algorithm as the basis of detection. As the volume of traffic and the network bandwidth keeps growing, IDPS needs a high-performance pattern matching algorithm that can keep up with the pace of processing data. Despite the literature on the various pattern matching algorithms, it still has been a challenge to determine which algorithm is the most effective when it comes to application in IDPS. As such, various studies have been directed at measuring the performance of popular single-keyword matching algorithms of patterns with respect to their runtime performance with different number of patterns and different packet capture (pcap) file sizes.

3. Proposed Solution

There exists a large variety of machine learning algorithms applicable to predictive modeling that are used in the creation of robust network intrusion detection systems by means of early classification. The main algorithms that can be found in this field are Decision Trees, Random Forests, XGBoost, as well as AdaBoost. The paper proposes an Ensemble Voting technique as the technique of finding the most useful technique to network intrusion detection, emphasizing the fact that the choice of an algorithm is of great importance to the effectiveness and the quality of intrusion detection systems(9).

During the first step, the Random Forest classifier will be deployed to the dataset, and then each of the algorithms selected will be deployed and evaluated separately. After that, a Voting Ensemble technique then is used to merge the results of these models and the overall classification accuracy is calculated using the merged predictions.

 https://doi.org/10.5281/zenodo.18815817

4.Methodologies

4.1 Decision Tree

Decision Tree is a popular machine learning model, which can be used in classification and regression. It has an interpretable and intuitive structure, which reflects decision-making process in human beings. The model is depicted as a tree like model such that; the root node will be the first decision point, internal (decision) nodes will be the conditions depending on the values of feature and leaf nodes will be the ultimate predicted outcomes.

Decision trees are widely used in data mining and predictive analytics to draw decision rules and decision strategies using data. The decision tree algorithms are computationally cheap and interpretable yet the real-world datasets have high dimension feature spaces. The branches of the tree are split by a given feature, which allows one to easily identify the importance of the features and their relationships(10).

Building a decision tree entails the process of choosing the best features, splitting perquisites and terminating the growth of the tree. Despite the tendency of decision trees to become huge and deep, the use of pruning is common to simplify the model and decrease overfitting and thereby enhance the overall performance of generalization.

The concept of decision tree learning is based on information theory and especially on the concepts of entropy and information gain. Entropy is used to estimate the amount of uncertainty in the data and information gain is used to estimate the amount of uncertainty in the data reduced by splitting the data according to a specific feature. An increased information gain means a more informative feature and it leads the tree-building process to good splits.

4.2 Random Forest

Random Forest is an ensemble learning algorithm that is created with the aim of classifying and regressing with a high degree of accuracy and strength. It is based on the decision tree model but a forest of decision trees (also known as a forest) is created and their predictions are pooled to generate the final output.

Random Forest algorithm uses a bagging (bootstrap aggregating) technique according to which multiple subsets of training data are created by means of random sampling with replacement. An individual decision tree is trained using each subset. In the construction of trees, the algorithm chooses a random subset of features on every split, and this provides diversity across the trees and less correlation among them.

Random Forest can be used to solve the problem of overfitting that is often present with individual decision trees and enhance the overall accuracy of predictions. The last prediction is identified by majority in case of classification and averaging in case of regressions.

The important benefits of Random Forest are as follows:

- Increased predictive power than the use of single decision trees.
- Powerful data management of missing or noisy data.
- Reduced risk of overfitting
- Very limited hyperparameter tuning is required.

The basic building blocks in this method are the decision trees. Patterns are individually learnt by the different trees and the aggregate result is a more reliable and generalized prediction.

4.3 XGBoost Classifier

Extreme Gradient Boosting or XGBoost is a machine learning algorithm that is very efficient and scalable, and which is grounded on the gradient boosting framework. It is developed to provide maximum performance in regards to speed, accurateness, and resource consumption, which suit big and arranged datasets(11).

The technique of boosting is an ensemble learning method when several weak learners are trained one after another with each new model trying to correct the errors that the previous models are making. Compared to bagging that minimizes variance, boosting is very effective in minimizing bias as well as variance thus creating more powerful predictive models.

XGBoost is an extension of the classical gradient boosting, with additional optimization methods and system-level optimization. It builds a set of decision trees additively, whereby every tree is added to the final model. The model optimization is a minimization of a regularized objective function which trades model complexity and prediction accuracy.

The salient features of XGBoost are:

- Regularized learning that involves the use of penalty terms to deter overfitting and favor simpler and more generalizable models.
- Gradient tree boosting In which the trees are gradually added to achieve a sequence of minimum error prediction using gradient-based optimisation.
- Column subsampling, based on the idea of Random Forest which samples a random subset of features to minimize overfitting and fast training.
- Parallel processing is provided, which allows computation faster and better scaling(12).

These properties have led to its popularity in machine learning programs that need high-quality, high performance, and high reliability since quite a number of them adopt the XGBoost technology in their network intrusion detection systems.

4.4 Splitting Algorithms

Determining the optimal split of data at each node is one of the main problems in the decision tree learning. The Exact Greedy Algorithm is used to overcome this problem by searching all possible points of division on all features. Though the approach ensures the best splits, it is computationally expensive specifically when the features are continuous as the number of potential split points grows exponentially with the processing time.

To eliminate this weakness, an Approximate Algorithm is adopted, particularly when it is required to use datasets that are too large to effectively be placed in memory. The approximate approach does not involve splitting candidates based on the distributions of percentile values of feature values; it instead submits candidate splits based on the distribution of the potential points of splitting. The buckets are discrete and the two values that define these buckets are the points at which the continuous features are separated. Statistical data in each bucket are then summed and the best split is chosen on the basis of the summarized statistics. The strategy greatly lowers the computational complexity and still increases accuracy(13).

An important step in the approximate algorithm is generation of candidate split points. XGBoost does this effectively through a distributed weighted quantile sketch technique which is particularly suited to weighted data and large datasets.

In most real-life scenarios, the input data are normally sparse because of the existence of gaps in the dataset.

Another cause of sparsity is common in statistical data and feature engineering methods, including one-hot encoding, and frequent elements with zero values. Learning algorithms should be able to identify and handle such sparsity patterns.

1. The challenge that XGBoost will deal with is the ability to efficiently handle sparse data in a unified and robust manner.
2. XGBoost utilizes as many CPU cores as possible in the process of training, which makes it possible to construct multiple decision trees in parallel. The statistics are calculated based on features, which means that the best split points are identified faster.
3. The algorithm is aimed at optimizing the hardware efficiency by assigning internal buffers to every processing thread. The buffers are used to hold gradient statistics, which enhance the use of cache and general computational efficiency.
4. In cases of datasets that are beyond the memory capacity, XGBoost has out-of-core computation which can be applied to train data in smaller and manageable segments, thus training efficiently without memory restrictions.
5. XGBoost also can be distributed by using clusters of inter-connected machines and large-scale models can be trained on the cumulative computational power of a collection of systems.
6. To minimize the large time overhead in data sorting in tree learning, XGBoost sorts data into compressed blocks of columns, which thus minimizes sorting overhead and reduces training time.
7. XGBoost is particularly used in large-scale and time-intensive applications since it is always faster to run than the majority of other classic machine learning algorithms(14).
8. The algorithm is great at providing high predictive accuracy and strong results in various applications where it can process structured and tabular data to perform classification and regression.

Conclusion

The use of parallel and distributed computing abilities makes XGBoost a high-performance algorithm. Being developed with a close consideration of system level optimization and using advanced machine learning

 https://doi.org/10.5281/zenodo.18815817

algorithms, the framework does not only use as much computational resources as it can, but it is also scalable, portable, and highly predictive. It is a powerful architecture that is applicable in large-scale data processing and modeling tasks using advanced modeling applications in a wide range of areas.

4.5 AdaBoost

AdaBoost, also known as Adaptive Boosting, is a well-known ensemble learning algorithm, in which a powerful classifier is developed by combining multiple weak ones consecutively. In contrast with classical ensemble algorithms, which create a poly of full decision trees, on average AdaBoost uses simple models referred to as decision stumps, which are composed of a single split and two terminal nodes. The most important feature of AdaBoost is that it is based on adaptive learning, whereas false cases occurred during the previous rounds are boosted during the next round of training.

This sequential stressing makes sure that every new weak learner specializes in correcting the previous weak learners so that the average predictive performance of the ensemble improves, gradually. The sequence of generation of the weak learners is very important since each learner will be built on the cumulative information on the error made by the earlier models. Whereas the Random Forest algorithms produce trees with different depths randomly, AdaBoost purposely limits the complexity of its learners to take advantage of the fact that a large number of simple models can be used to obtain high accuracy(15).

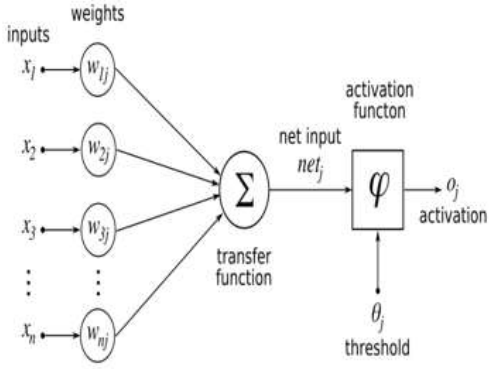
When used in classification problems and especially in binary classification, AdaBoost which was initially named AdaBoost.M1 has proved to be very useful in improving the performance of the models. The algorithm puts a default value of equal weights to all training instances, which are usually initialized as.

N refers to the overall sampling number. These weights are updated as the training advances to focus on those samples that are problematic to classify. This reweighting algorithm enables AdaBoost to convert weak learners into a strong ensemble that will be able to deal with difficult classification tasks.

Artificial Neural Network (ANN)

A basic implementation of the Artificial neural Network (ANN) is a key element of the contemporary computing systems, which are based on the concept of the biological neural networks of the human brain. ANNs are the basis of the artificial intelligence field and do especially well in tackling the problems that are difficult to resolve using the conventional statistical or rule methods. Learning and increasing performance with the availability of more data is one of their key features.

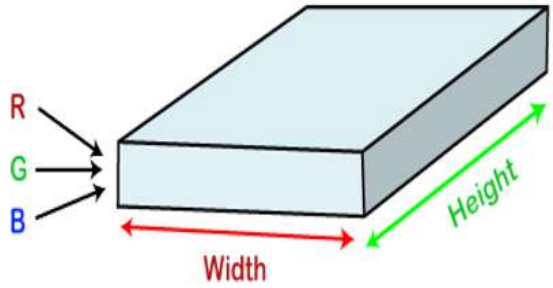
The training process at ANN level trains the ANN to detect patterns within the input data via a supervised learning process. The network produces an output and it is compared to the expected output. A learning algorithm referred to as backpropagation is used to minimize any difference between the predicted and actual values. This approach is reversing the mistake that occurred at the output layer down to the input layer modifying the connection weights. The ANN improves its internal parameters by means of repetition, making it less prone to errors in predicting and hence more accurate.



Convolutional Neil Network (CNN)

Convolutional Neural Networks (CNNs) are improved deep learning frameworks whose main application motor is image recognition and visual data processing tasks. Their applicability in detecting objects, classifying images, and segmenting them based on their semantics makes them very popular in different areas of application like object detection, image classification, and semantic segmentation because they automatically learn spatial and hierarchical features of raw input data.

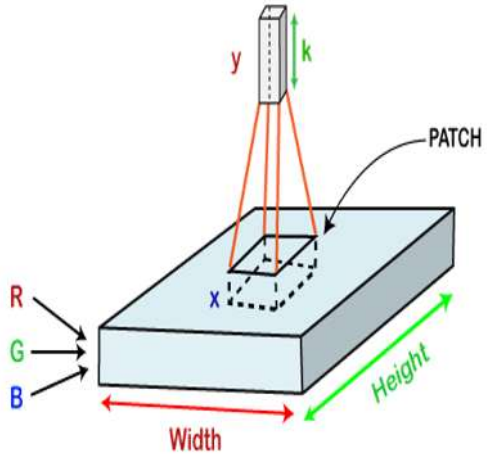
A CNN comprises of several interconnected layers, such as convolutional layers, pooling layers and fully connected layers. Convolutional layers are at the center of attention since they help to extract meaningful features of input images by applying learnable filters. The pooling layers also make feature maps fewer spatial dimensions to minimize the computation complexity but preserve the important information. The extracted features are then processed by fully connected layers to either do classification or prediction tasks.



The ability to use local connectivity and sharing of parameters is another important advantage of CNNs as it allows them to process large datasets efficiently while maintaining the spatial connectivity between the input data. Through training of hierarchical representations, on one hand, the simplest form (edges and textures) to the most complex form (object structures), CNNs have transformed computer vision and image analysis.

During the convolution process, a small neural network, which takes the shape of a filter, is swept on the input image. The result of this process is a series of feature maps that have varying dimensions to the original image and also include more channels other than the typical RGB channels. Convolution operation attains spatial patterns by learning weights related to local parts of the picture to enable the network to infer informative features.

- Convolutional layers consist of many learnable filters, which have small spatial dimensions in width and height, and depth equal to the input volume. In case of a color input, as an example, the filters are usually a combination of all three color channels.
- Take an input image of size 34x 34 x3.34×34×3. Filters can be used to convolute objects of size smaller than the filter. a×a×3 a×a×3, where a may take values such as 3, 5, or 7. These dimensions of the filters are made purposefully small as compared to the size of the input to capture the local spatial features.
- In the forward propagation process, every filter moves through the input volume in steps that are also known as stride. The possible values of the stride include 1, 2 or more, depending on the desired output resolution. The filter at every position calculates the dot product of the weights of the filter and the input region at that position and gives the values of the feature map.



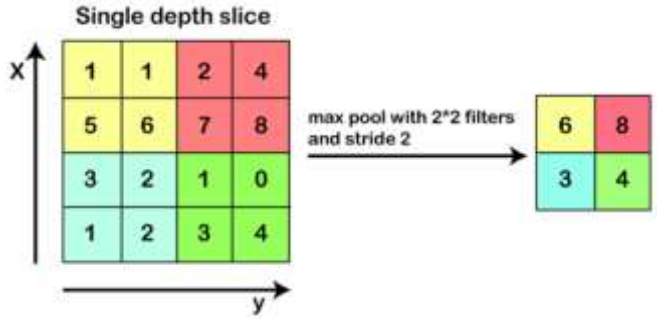
Convolutional Neural Network (CNN) working

- The design of the Convolutional Neural Network (CNN) is usually divided into three major parts. The first is the input layer which takes the raw pixel intensity values of an image. For example, an image of size The input layer is a 32x32x3-dimensional (3 color channel, red, green and blue) tensor(16).
- CNNs are based on the idea that they learn and identify patterns in pictures in a hierarchical feature extraction mechanism. The network consists of many layers but the most important layers are the

 https://doi.org/10.5281/zenodo.18815817

e-ISSN: 2997-8718 Print ISSN: 2997-870X

convolutional layers which run learnable filters to the input image to extract features of the image. These layers are based on local connectivity and sharing of the parameters to perform image data efficiently and maintain the spatial relationship. After convolution, an element-wise activation function like the Rectified Linear Unit (ReLU) is used. The ReLU function, defined as max(0,x) max(0, xt, and introduces non-linearity by clamping to zero all values under zero but preserves the spatial dimensions of the feature maps e.g. an output of size 32x32x12.



- The pooling layer applies the spatial reduction of the feature maps where in most cases the width and the height are downsampled but the depth remains the same. An example is that with the use of pooling, the feature map size can be reduced to 16 in terms of length and width, and 12 in terms of depth and so reducing the computational cost, and becoming more resistant to spatial variations.
- Lastly, the output of the previous layers is passed to the fully connected layers which compute class scores. These layers project the features maps into a one dimensional vector, and generate an output with the size of the number of target classes, which, through which the network can classify.

This is initiated by the entry of an image into the network whereby several feature detectors, called filters, are used in a convolutional layer to produce feature maps. This is then succeeded by the implementation of the Rectified Linear Unit (ReLU) activation function that, in turn, adds non-linearity and removes negative values, which, in turn, improves the representation of meaningful features in the image.

This is followed by the incorporation of a pooling layer which is used to down-sample the spatial dimension of the feature maps without distorting significant information. This action encourages the spatial invariance, it also reduces computational complexity and it also helps to reduce overfitting. These concatenated feature maps are then flattened into one one dimensional vector and fed into an artificial neural network by fully connected layers, and then the final result is a classification or prediction.

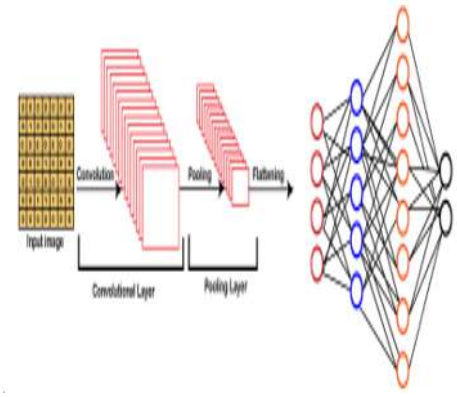
4.6 Convolutional Neural Network Construction

The CNNs introduce a basic layer that is called convolutional layer that greatly improves the ability of artificial intelligence and deep learning models to handle visual data. As opposed to traditional Artificial Neural networks (ANNs) which require one dimensional input vectors with limited feature representation, CNN is specifically made to accept three dimensional image data at once where spatial and hierarchical features are represented in a better manner.

The data that will be utilized in this research will be the images of cats and dogs, divided into training and testing sets. The training set of the CNN model consists of 4,000 cat images and 4,000 dog images, which are used to train the model. The model is then tested on a separate test set of 1,000 cat images and 1,000 dog images which were not shown to the model in training. The overall goal is to create a CNN that will be able to differentiate correctly between cat and dog pictures.

Implementation is done on a Jupyter Notebook environment and the form of implementation is the same as one in the past with Artificial Neural Networks. The difference between the preprocessing steps however is significantly different because the data is a collection of images, not tabular. There are two primary preprocessing steps to the workflow, which include training dataset preparation and test dataset preprocessing. This happens by first importing the necessary libraries and then preprocessing the data, designing the CNN architecture, training the model and then making individual predictions. Although the general pipeline is similar to those of an ANN-based algorithm, the preprocessing methods, as well as the model architecture, are tailored to be effective in processing an image-based data.

 https://doi.org/10.5281/zenodo.18815817



5.Conclusion

This paper successfully created and trained a Convolutional Neural Network that was used to classify cat and dog images. Through the ability of the convolutional layer to learn spatial and hierarchical features directly by using image data, the model breaks the constraint of the old neural networks that used flattened input vectors. The complexity of the methodological procedure of preprocessing image data, developing a relevant CNN architecture, and testing the model on unknown test images proves the efficiency of CNN in image classification tasks. These findings demonstrate the effectiveness of CNNs in the acquisition of meaningful visual patterns and indicate that it is applicable to real-world computer vision problems with a high level of accuracy and reliability.

Acknowledgement: Nil

Conflicts of interest

The authors have no conflicts of interest to declare

References

1. Borkar A, Donode A, Kumari A. A survey on intrusion detection system (IDS) and internal intrusion detection and protection system (IIDPS). Proceedings of the International Conference on Inventive Computing and Informatics (ICICI). 2017;1(1):949–953.
2. Zhou Z, Zhongwen C, Tiecheng Z, Xiaohui G. The study on network intrusion detection system of Snort. Proceedings of the International Conference on Networking and Digital Society. 2010;1(1):194–196.
3. Zolkipli MF, Jantan A. A framework for malware detection using combination technique and signature generation. Proceedings of the 2nd International Conference on Computer Research and Development. 2010;1(1):196–199.
4. Zhang H. Design of intrusion detection system based on a new pattern matching algorithm. Proceedings of the International Conference on Computer Engineering and Technology. 2009;1(1):545–548.
5. Gupta V, Singh M, Bhalla VK. Pattern matching algorithms for intrusion detection and prevention system: A comparative analysis. Proceedings of the International Conference on Advances in Computing, Communications and Informatics (ICACCI). 2014;1(1):50–54.
6. Halimaa A, Sundarakantham K. Machine learning based intrusion detection system. Proceedings of the 3rd International Conference on Trends in Electronics and Informatics (ICOEI). 2019;1(1):916–920.
7. Phadke A, Kulkarni M, Bhawalkar P, Bhattad R. A review of machine learning methodologies for network intrusion detection. Proceedings of the 3rd International Conference on Computing Methodologies and Communication (ICCMC). 2019;1(1):272–275.
8. Bondan L, Marotta MA, Kist M, Faganello LR, Both CB, Rochol J, Granville LZ. Kitsune: A management system for cognitive radio networks based on spectrum sensing. Proceedings of the IEEE Network Operations and Management Symposium (NOMS). 2014;1(1):1–9.
9. Gaddam R, Nandhini M. Analysis of various Snort-based techniques to detect and prevent intrusions in networks with code refactoring in Kali Linux environment. Proceedings of the International Conference on Inventive Communication and Computational Technologies (ICICCT). 2017;1(1):10–15.
10. Wang W, Sheng Y, Wang J, Zeng X, Ye X, Huang Y, Zhu M. HAST-IDS: Learning hierarchical spatial–temporal features using deep neural networks to improve intrusion detection. IEEE Access. 2018;6(1):1792–1806.
11. Anderson JP. Computer security threat monitoring and surveillance. Technical Report, James P. Anderson Co. 1980;98(17):1–45.

12. Denning DE. An intrusion detection model. IEEE Transactions on Software Engineering. 1987;13(2):222–232.
13. Wu SX, Banzhaf W. The use of computational intelligence in intrusion detection systems: A review. Applied Soft Computing. 2010;10(1):1–35.
14. Lazarevic A, Ertoz L, Kumar V, Ozgur A, Srivastava J. A comparative study of anomaly detection schemes in network intrusion detection. Proceedings of the SIAM Conference on Data Mining. 2003;1(1):1–12.
15. Kumar S, Spafford EH. A pattern matching model for misuse intrusion detection. Proceedings of the 17th National Computer Security Conference. 1994;1(1):11–21.
16. Chebrolu S, Abraham A, Thomas JP. Feature deduction and ensemble design of intrusion detection systems. Computers & Security. 2005;24(4):295–307.