Intelligent Honeypot-based IDS for Cyber Attack Detection using Generative Adversarial Networks

M Karthigha
karthighamohan@gmail.com

Sri Ramakrishna Engineering College
L Latha
Kumaraguru College of Technology
Sripriyan K
VIT Bhopal University

Article

Keywords: Honeypot, Machine Learning, Cyber Attack, Data Breach, Attack Prevention

Posted Date: June 10th, 2024

DOI: https://doi.org/10.21203/rs.3.rs-4370529/v1

Additional Declarations: No competing interests reported.

Abstract

Cyber Intrusion is the most threatening expression in the cyber world, and it is a dangerous crime that many corporations and individuals who are a part of the Cyber World are horrified of. It results from not only a financial loss but also includes personal data which is impacted as a flooded river when the data is exposed in a data breach. Cyber Intrusion Detection System refers to a technology utilized for recognizing and notifying any illicit entry to a network system or network device. It analyses network traffic logs files maintained by a device and reports or alerts when an outsider is trying to gain access to a network. A honeypot is a technology that acts as a catchy pot of honey for an attacker. When an attacker tries to catch the pot, the Honeypot system will alert the administrator and block it. These technologies can be combined with Artificial Intelligence to automate and improve the attack prediction rate so that attackers will be prevented. A network design is proposed for the correct placement of a honeypot in the network. Datasets extracted from implementing a honeypot and KDD-99 are used in this research experimentation. The datasets are trained using Generative Adversarial Networks, a deep learning-based generative model. From the results, it is inferred that the GAN model helps in achieving enhanced performance metrics in terms of accuracy and optimum training time in both datasets.

I. INTRODUCTION

The world is evolving with cutting-edge technology, and people around the world are interconnected with each other through the internet with the help of electronic devices and smart gadgets. There are about 5 billion active Internet users worldwide and a major threat to Internet users is cyberattacks [1–4]. Cyber-attacks may lead to any risk which includes financial losses, personal data leaks, business-related problems, corporate security, and mainly personal data security [5–8]. Data Breach sometimes lights up a few corporate secret crimes which are more commonly found these days. Though it lights up crimes, a data breach is a crime that involves one's or some personal details. In the realm of cybersecurity, a honeypot serves as a mechanism to discover, divert, or counter any unauthorized exploitation of information systems. A honeypot is essentially a decoy system that is designed to attract and trap potential attackers by imitating a vulnerable system or application [9–11]. The global average cost of a Data breach is about 4.35 USD. Verizon's 2022 Data Breach Investigation Report (DIBR) states that about 62% of incidents are of System Intrusion patterns involving threat actors compromising partners. 13% increase in Ransomware which is more than the combined past 5 years. Avoiding such attacks for an individual is impossible since targeted attacks are more common these days where the data of high-powered people is not safe. To avoid such attacks in a network, the use of new technologies is a must to get more secure.

Honeypot is one such type of cybersecurity technique that helps to curb cyber-attacks. The idea behind a honeypot is to give attackers a fake system to attack instead of the real one, allowing security researchers to observe the attacker's techniques and tactics without risking damage to real systems. When an attacker interacts with a honeypot, the honeypot captures information about the attack, including the attacker's IP address, methods, and tools used in the attack. Among the several types of honeypots are high-interaction and low-interaction honeypots, with the former being designed to provide a realistic environment for attackers to operate in, whereas low-interaction honeypots simulate only certain aspects of a system. Honeypots can be used as a proactive security measure to both identify and thwart attacks, along with being a research tool to study and understand attackers' behaviour and motives. However, the attackers are smarter as there are a few ways to bypass the honeypots [9]. The combination of Honeypot and Cyber Intrusion Detection using Artificial Intelligence (AI) to develop an intelligent honeypot can predict almost every input data frame in less time with a high prediction rate which can stop the attacker from accessing the data.

II. RELATED LITERATURE

Honeypots have been a topic of research for several years and have seen significant advancements in recent times, particularly with the integration of artificial intelligence (AI) techniques for enhanced detection and prevention capabilities. In this literature survey, various research on honeypots and AI-based honeypots to understand the current state of honeypot technology is explored and how AI can be leveraged to improve honeypot capabilities in which experts have done quite a few works. [12–14] provides a detailed examination of honeypots as a security mechanism for network-based attack prevention and detection. The authors describe the advantages, implementation, and use of honeypots as an effective information-gathering tool for network attacks. The study includes methodologies like roaming honeypots and backpropagation algorithms to identify and prevent source-address parodying DDoS attacks. In [1] the author discusses the importance of an Intrusion Detection System (IDS), which helps monitor network and system activities and detects malicious activities. The paper explores various types of IDS, such as network-based and host-based IDS, along with their components and functioning. The IDS tool can detect and prevent intrusion from attackers, making it a crucial security system for organizations in today's internet-driven world. [15] gives a Neural Network-based NIDS framework for real-time anomaly detection in modern-day network traffic, using the UNSW-NB15 dataset for training. The framework uses convex Logistic Regression cost functions, along with stochastic gradient descent and simulated annealing to fine-tune hyperparameters of the classifier. [16–17] discusses the use of data mining techniques in implementing intrusion detection systems (IDS) to protect computer networks from attacks. The study compares the performance of two decision tree algorithms, J48 and Random Forest, using attribute selection methods Information Gain and Rankers Algorithm

Ⅲ. OUR CONTRIBUTION

a. Developed an Intrusion Detection System (IDS) capable of effectively discriminating between normal network traffic and attack traffic of 23 different types in a real-time environment.

b. Designed a unique network around the honeypot that utilizes multiple layers of security to conceal all data on the hidden backside of the honeypot server. By doing so, it helps to attract more attackers and identify the attacks using honeypot bypass techniques.

c. We extracted 28 features from the real-time data collected in the honeypot, which were based on the KDD99 dataset, and using these features, the GAN model is implemented where discriminates between normal network traffic and attack traffic.

d. In order to process the real-time traffic, our system employs a Generative Adversarial Network (GAN) architecture consisting of two neural networks: a generator that learns to create synthetic data and a discriminator that learns to differentiate between real and synthetic data. The generator and discriminator are trained together in an adversarial manner until the generator produces synthetic data that is indistinguishable from real data.

e. Performance analysis is done in terms of the generator's gradient loss and the discriminator's accuracy. Experimental results show that the performance of GAN is better than traditional methods when comparing GAN with traditional machine learning methods.

▯. Proposed Methodology

AI approaches for intelligent honeypots can significantly enhance the capabilities in detecting, analysing and mitigating malicious activities. AI approaches with a change in the network setup can be used to predict the honeypot log data and whether any attacker is trying to gain access or not in an efficient way [2,6]. It is achieved by using the required data from the honeypot log files. It requires a few features which are extracted from the generated log report from the Honeypot [7]. The primary objective of this methodology is to achieve high accuracy while utilizing minimal processing power.

An AI algorithm's predictive accuracy is highly dependent on the dataset's quality. In this proposed methodology, the "KDDCUP99" dataset, which was provided by the Fifth International Conference on Knowledge Discovery and Data Mining, is utilized directly from the honeypot logs [3,4]. The dataset has labels and hence it is a supervised model, we test its accuracy with all suitable Machine Learning Algorithms.

A. Network Design

The network design is designed in such a way that all the data is the hidden backside of a honeypot server divided by firewalls. Multiple layered designs are required to attract more attackers and it helps in identifying attackers which use honeypot bypass techniques. Network design is depicted in Figure 1 using Cisco Packet Tracer.

i. The network design used is Hybrid architecture. We use it to customize the security features.

ii. The whole network is entered directly through the Internet router from the Internet Service Provider (ISP) or the outer network.

iii. All the traffic from the Internet Service Provider enters the Internet Router and the data packets are sent over Firewall 1 to filter the usual unwanted packets from the Internet or the outer network.

iv. Filtered packets then enter the Honey Hub. The Honey Hub is connected to a home gateway that is fully connected with more devices using the internet (IoT Devices), these may be physical devices or virtual devices running old or vulnerable operating systems. Old and vulnerable software versions for IoT devices are the 1st layer of trap for attackers. Since the software is vulnerable by nature, they are more likely to act as a natural honeypot.

v. The role of the Honey HUB is to transmit all packets to the IDS server since HUB broadcasts every packet detail to every device connected to it.

vi. The honeypot server, connected via optical fibre, receives data faster compared to standard twisted pair cables, such as Cat 5, Cat 6, Cat 7, and Cat 7E, due to the former's higher speed capability.

vii. IDS server analyses the network and uses a Machine Learning Algorithm that predicts the packet's nature and collects all required details for the prediction and sends the report to Firewall 2 via Honey Hub with an encrypted secured channel.

viii. Entry Router receives packets that are filtered by Firewall 2 as directed by the honeypot server. The packets move via a packet sniffer which collects all data for future study purposes since “Nothing in the internet is 100% safe”.

ix. Finally, the Switch0 and the Home Router or the regular network is connected and can be used normally.

B. Honeypot

The network design consists of two layers of security mechanisms as explained well in the network design. It is important to have multiple layers of security features to ensure more security. Hackers are always more intelligent and they get access if there is a minute defect in the code or even network design.

· The first layer of security has IoT devices with older software. Older software is more vulnerable than the latest one. IoT devices are more likely to be trapped by hackers to get into a network. If any hacker traps an IoT device, those IP will be blocked by a firewall.

· As the honeypot used is a high interaction honeypot, few ports were made open which mimics the open ports of software running inside it such as Microsoft update plugin’s port or database’s default port or TCP, UDP, TELNET, POP3, FTP, SMTP, etc.

The whole network is monitored and the monitored data is saved. The data is then used for the feature extraction process for machine learning purposes. The data is saved using network monitoring tools. The network sniffer or packet sniffer traps the whole data where we can find anything there.

C. Dataset

Two datasets are utilized for the investigation of intelligent honeypots. Datasets extracted from the implementing honeypot environment and KDD-99 datasets are used for this experimentation.

1. Feature Extraction From Honeypot

Wireshark is utilized to capture the network traffic inside the honeypot. This tool enables us to analyse the traffic data packets and extract the necessary information for further analysis. The captured data is then saved in a .pcap format file with 28 features. The extracted features include information about the packet content, TCP flags, and traffic statistics. The extractor uses the pandas library for data processing and can handle both the original and pre-processed versions of the KDD99 dataset and it also provides a convenient way to preprocess the KDD99 dataset for machine learning experiments. It starts by loading the raw data into a Pandas Data Frame and then performs some data cleaning operations, such as removing duplicates and renaming columns. Then, it applies label encoding to convert categorical features into numerical ones. It also drops some irrelevant features, such as the 'service' column. After preprocessing, the notebook uses the sklearn library's 'Standard Scaler' to normalize the data. Finally, it extracts 28 features from the pre-processed data, including both numerical and categorical features. The extracted features include the following:

- duration
- protocol type
- service
- flag
- src_bytes
- dst_bytes
- land
- wrong_fragment
- urgent
- count
- srv_count
- serror_rate
- srv_serror_rate
- rerror_rate

- srv_rerror_rate
- same_srv_rate
- diff_srv_rate
- srv_diff_host_rate
- dst_host_count
- dst_host_srv_count
- dst_host_same_srv_rate
- dst_host_diff_srv_rate
- dst_host_same_src_port_rate
- dst_host_srv_diff_host_rate
- dst_host_serror_rate
- dst_host_srv_serror_rate
- dst_host_rerror_rate
- dst_host_srv_rerror_rate

Through the honeypot , a total of 1043 data rows, each of which includes 28 features, were successfully extracted providing us with a rich source of information to analyse and draw insights from. Specifically, our dataset included 713 rows of normal data and 330 rows of attack data, providing us with a comprehensive view of the threats facing our systems and the methods used to compromise them. We utilized an offensive security operating system, such as Kali Linux, to simulate various attack methods within a virtual environment. Specifically, we targeted another network within the virtual environment to emulate the effects of different types of network attacks and recorded and extracted the dataset from the network traffic.

2. KDD99 Dataset

The dataset utilized for this study was released during the Third International Knowledge Discovery and Data Mining Tools Competition, which was held in tandem with KDD-99[3] The Fifth International Conference on Knowledge Discovery and Data Mining. Participants were tasked with constructing a network intrusion detector that could differentiate between "bad" connections (intrusions or attacks) and "good" normal connections by creating a predictive model. The dataset has 42 columns which are the main features that determine the prediction. The most important field mainly consists of attack type and type of connection are as follows [3]:

- ‘Normal’ for regular data.
- ‘dos’ type of attack for ‘Neptune, smurf, land, back, pod, teardrop’.
- ‘probe’ type of attack for the type ‘ipsweep, satan, port sweep or nmap’.
- ‘r2l’ type of attack for type ‘ftp_write, guess_passwd, imap, phf, spy, warezclient, warezmaster or multihop’,
- ‘u2r’ for ‘Perl, loadmodule, buffer_overflow or rootkit’

Within the database, there exists a standard set of data to undergo auditing, comprising an array of intrusions emulated within a military network environment, but the combination of honeypots with a 2 layered intrusion model using different ML Algorithms helps in identifying more efficiently and with low processing power. Figure 2 depicts the heatmap for all 42 features of the KDD99 dataset.

D. Intrusion Detection System by Machine Learning

The common packets which are filtered by a firewall are almost safe from regular intrusions and regular web traffic. The firewall may be a hardware firewall or else a software-based firewall to filter the packets. All the data together come to the HUB and are broadcast to every device. A smart attacker can sense the use or any other mode of packet sniffing, and even the intruder can modify or destroy log files and reports generated by the Honeypot Server, but the use of Hub makes it look like a natural device used for connecting devices. The possibility of identifying IoT devices is very high because of the vulnerable software running in them. All the data is monitored by software or hardware in the Honeypot server which is shown in Figure 3

Honeypot Server is the main part that determines whether the packets are safe or not. It analyses every single packet in detail and gives predictions using AI algorithms [12]. Every machine learning model has its specific type of processing method. Every classifier has its working algorithms. As the desired output is a prediction of type “intruder” or “Normal” type, a proper machine learning model is to be selected for the required output. A set of ML classifiers are taken to test the dataset. The dataset has a label column which is suitable for supervised machine-learning algorithms. The utilized machine learning algorithms encompass Gaussian Naive Bayes, Decision Tree, Random Forest, Gradient Boosting, Extreme Gradient Boosting, Generative Adversarial Network, Linear Regression, Logistic Regression, Long Short-Term Memory, and Deep Belief Network.

E. Threat Detection

Various AI algorithms are utilized to determine whether the connection is good or it is an intruder. The data from the honeypot is analysed well by the AI techniques which have about 42 features used for prediction [12]. They are tested using Different machine learning models like Gaussian Naive Bayes [9], Decision Tree [10], Random Forest, Gradient Boosting, Extreme Gradient Boosting, Generative adversarial network, Linear Regression, Logistic Regression, Long Short-Term Memory, and Deep Belief Network as mentioned above. The one which is suitable for intrusion detection gives a high accuracy value and the high accuracy model is selected for the detection.

The application of Generative Adversarial Networks (GANs), a class of deep learning algorithms, involves generating new and synthetic data by pitting two neural networks Within a zero-sum game framework, the entities compete against each other. GAN architecture is shown in Figure 4 which has two main components: a generator and a discriminator. Taking a random noise vector as input, the generator maps it to an output that mimics the target data distribution, whereas the discriminator takes in both real samples from the target data distribution and fake samples generated by the generator and tries to distinguish between the two, they are trained in an adversarial manner, generator creating samples that are realistic to the discriminator, In an attempt to classify samples as real or fake, the discriminator scrutinizes them, while the generator refines its skill in generating realistic samples, and the discriminator enhances its ability to classify samples over time.

The GAN framework can be mathematically formulated as a game of minimum and maximum values between the generator and discriminator networks. The generator's objective function is to decrease the probability of the discriminator accurately classifying the generated data as fake, while the discriminator's objective function is to increase the probability of correctly classifying both the real and generated data. This can be represented as:

$$\min_{G} \max_{D} V(D,G)$$

$$V(D,G) = \mathbb{E}_{x} \sim pX[logD(x)] + \mathbb{E}_{z} \sim pZ[\log\left(1 - D\left(G(z)\right)\right)]$$

The network labelled as G is responsible for generating data samples that are intended to resemble the real data distribution, while the network labelled as D is responsible for distinguishing between the real data and the generated data samples, and x is a real data point drawn from the true data distribution pX, z is a random noise vector drawn from a prior distribution pZ, and G(z) is a generated data point obtained by applying the generator to the noise vector z. The first term in the objective function corresponds to the expected log probability that the discriminator correctly classifies a real data point, and the second term corresponds to the expected log probability that the discriminator incorrectly classifies a generated data point as real.

[illegible]. EXPERIMENTAL RESULTS

Tensor Flow and scikit-learn, the most popular AI frameworks, were employed for the experimental simulations, while Python was utilized as the programming language. The experiment evaluates the performance of the GAN model against the KDD-99 dataset and honeypot dataset. GAN's accuracy and training time is analysed and compared with various AI models.

A. Accuracy and training time on the kdd99 dataset

The choice of learning rate is crucial in training machine learning models as it can significantly impact their performance and convergence. From Figure 5, it appears that a learning rate of 0.01 achieved the highest accuracy of 98.10% on the KDD99 dataset, while a learning rate of 0.1 also performed well with an accuracy of 97.4%.

Table 1 shows the training progress of a GAN model against the KDD-99 dataset over multiple epochs with a learning rate of 0.01. The table includes information on the discriminator loss, generator loss, and accuracy at each epoch.

Table 1 Performance Evaluation of the GAN model on the KDD-99 dataset

Epoch	Discriminator Loss	Generator Loss	Accuracy
1	38.05141	0.689847	42.19
100	0.131214	1.496475	85.89
200	0.096201	1.761345	88.32
300	0.025205	3.041839	89.96
400	0.011418	3.800014	94.01
500	0.005777	4.490452	96.23
600	0.00349	4.992599	96.99
700	0.002289	5.401649	97.23
800	0.001711	5.725489	98.86
900	0.001114	6.107335	98.10
1000	0.000896	6.343067	98.10

Table 2 shows the performance evaluation of various AI models against the KDD-96 dataset. The Decision Tree algorithm shows a test accuracy of 91.62% and a training time of 10.85s. The Linear Regression algorithm shows a test accuracy of 92.75% and a training time of 5.99s. The Gaussian Naive Bayes algorithm shows a test accuracy of 93.21% and a training time of 3.74s. The Random Forest algorithm shows a test accuracy of 95.44% and a training time of 495.71s. The Gradient Boosting algorithm shows a test accuracy of 95.94% and a training time of 6372.18s. The Extreme Gradient Boosting algorithm shows a test accuracy of 96.93% and a training time of 443.02s. The Long Short-

Term Memory algorithm shows a test accuracy of 97.87% and a training time of 7966.02s. The Deep Belief Networks algorithm shows a test accuracy of 79.84% and a training time of 819.10s. The Generative Adversarial Networks algorithm shows a test accuracy of 98.10% and a training time of 130.02s. The Table 2 shows the results of all the classifiers and its training time with the testing accuracy of each.

Table 2 Performance Evaluation of various AI models against KDD-99 dataset

Classifier	Test Accuracy	Training Time
Decision Tree	0.9162	10.846
Linear Regression	0.9275	5.988
Navie Bayes	0.9321	3.739
Random Forest	0.9544	495.712
Gradient Boost	0.9594	6372.185
XG Boost	0.9893	443.016
LSTM	0.9787	7966.016
Deep BN	0.7984	819.102
GAN	0.9810	130.016

It is inferred from Table 2 that GAN model produces highest performance with 98.10% accuracy when compared to other AI models. The training time of GAN is 130.016s which is low than most of the complex and efficient algorithm like Gradient boosting and LSTM. GANs, while also complex, might converge faster due to their architecture and training dynamics, making them computationally efficient. If GANs are well-suited for the specific problem and data distribution, they can yield better results in a shorter time compared to models that are not as well-matched.

B. Accuracy and training time on dataset extracted from honey pot

As with the previous dataset, the choice of learning rate is critical in training machine learning models. From Figure 6, it appears that a learning rate of 0.01 achieved the highest accuracy of 97.98% on the Honeypot dataset, while a learning rate of 0.05 also performed well with an accuracy of 96.01%.

Table 3 gives the accuracy and training time on dataset extracted from honeypot. The Decision Tree algorithm shows a test accuracy of 92.89% and a training time of 3.14s. The Linear Regression algorithm shows a test accuracy of 91.88% and a training time of 1.57s. The Gaussian Naive Bayes algorithm shows a test accuracy of 89.72% and a training time of 1.23s. The Random Forest algorithm shows a test accuracy of 94.34% and a training time of 120.29s. The Gradient Boosting algorithm shows a test accuracy of 96.11% and a training time of 800.58s. The Extreme Gradient Boosting algorithm shows a test accuracy of 96.55% and a training time of 590.16s. The Long Short-Term Memory algorithm shows a test accuracy of 96.09% and a training time of 7000.718s. The Deep Belief Networks algorithm shows a test accuracy of 69.26% and a training time of 120.21s. The Generative Adversarial Networks algorithm shows a test accuracy of 97.98% and a training time of 140.72s. The table 3 shows the results of all the classifiers and its training time with testing accuracy of each.

Table 3 Performance evaluation of various AI models against the honeypot dataset

Classifier	Test Accuracy	Training Time
Decision Tree	0.9289	3.146
Linear Regression	0.9188	10.578
Navie Bayes	0.8972	4.238
Random Forest	0.9434	120.299
Gradient Boost	0.9611	7800.581
XG Boost	0.9655	590.160
LSTM	0.9609	7000.718
Deep BN	0.6926	759.210
GAN	0.9798	140.712

It is inferred from Table 3 that GAN model produces highest performance with 97.98% accuracy when compared to other AI models. The training time of GAN is 140.016s which is low than most of the complex and efficient algorithm like Gradient boosting and LSTM.

Figure 7 and 8 presents the test accuracy and training time for different classifiers on two datasets - KDD-99 and Honeypot.

For the KDD-99 dataset, the best performing classifiers in terms of test accuracy are XG Boost (0.9701), followed by LSTM (0.9787) and GAN (0.9810). However, the training time for these classifiers is relatively high, with LSTM taking the most time (7966.016 seconds). The Navie Bayes classifier has the lowest training time (3.739 seconds), but its test accuracy is lower compared to the other classifiers.

For the Honeypot dataset, the best performing classifiers in terms of test accuracy are Gradient Boost (0.9611), followed by XG Boost (0.9655) and GAN (0.9798). The training time for these classifiers is relatively lower compared to the KDD-99 dataset, with the Navie Bayes classifier taking the lowest time (4.238 seconds). However, the Deep BN classifier has a very low-test accuracy (0.6926) and a relatively higher training time (759.210 seconds).

□. CONCLUSION AND FUTURE ENHANCEMENT

GAN model produces accuracy of 98.10 and 97.98 against KDD99 dataset and honeypot dataset respectively. It produces training time of 130.016 seconds and 140.712 against KDD99 dataset and honeypot dataset respectively. It is inferred from the comparison of the accuracy and training time of various Machine Learning models on both the KDD99 dataset and the dataset extracted from the honeypot that the GAN model performs better than other AI models in terms of accuracy and training time on both datasets.

In future work, we aim to implement the honey pot concept in real-time cyber intrusion detection using GAN to enhance the system's ability to identify and mitigate potential cyber threats. Our future work will focus on optimizing the performance of the honey pot in real-time cyber intrusion detection using GAN by incorporating more sophisticated techniques for generating and analyzing data. Future work will also include the development of a real-time monitoring system that will allow for the continuous monitoring and analysis of cyber threats using the honey pot and GAN-based intrusion detection system.

Implementation of Federated Learning: Federated learning is a distributed machine learning technique that allows multiple devices to collaborate on a model without sharing data. By implementing federated learning in our GAN-based system, we could potentially improve the scalability and privacy of our system, while also reducing the computational resources required to train the model.

Real-time Analysis of Large-scale Networks: In our current study, we focused on analyzing small-scale networks. In future work, we plan to investigate the performance of our GAN-based system in analyzing large-scale networks in real time. This would require developing new techniques to efficiently process and analyze the vast amounts of data generated by large-scale networks.

Declarations

Author contribution Karthigha M: Conceptualization, Methodology, Validation, Investigation, Resources, Data curation, Writing - original draft, Visualization. Dr. L Latha and Dr K Sripriyan: Resources, Data curation, Writing

Acknowledgements The authors thank the Management and Principal of Sri Ramakrishna Engineering College, Coimbatore, for supporting the activities of the research work.

Data availability Data sharing is not applicable to this article as no datasets were generated or analysed during the current study.

All data generated or analysed during this study are included in this published article

Declaration of competing interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Compliance with ethical standards

Conflict of interest The authors declare that have no conflict of interest.

Ethical approval The authors state that the research work was conducted according to ethical standards.

References

1. Mr. Mohit Tiwari, Raj Kumar, Akash Bharti, Jai Kishan (2017) Intrusion Detection System, International Journal of Technical Research and Applications e-ISSN: 2320–8163.
2. Gozde Karatas, Onder Demir, Ozgur Koray Sahingoz, "Deep Learning in Intrusion Detection Systems", *2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT)*, pp.113–116, 2018.
3. Nilesh Kunhare, Ritu Tiwari, "Study of the Attributes using Four Class Labels on KDD99 and NSL-KDD Datasets with Machine Learning Techniques", 2018 *8th International Conference on Communication Systems and Network Technologies (CSNT)*, pp.127–131, 2018.
4. Karthigha, M. and Latha, L. 'Clustered Ensemble Feature Selection with M-GRU Classification for Efficient Intrusion Detection System of Industrial Systems'. 1 Jan. 2023: 9109–9127.
5. Chih-Fong Tsai, Yu-Feng H. "Intrusion detection by machine learning: A review," Expert Systems with Applications 36,11994–12000, 2009.
6. Chia-Ying Lin, Wei-Yang Lin. "An Anomaly Intrusion Detection System Based on Intelligent User Recognition," Expert Systems with Applications Volume 36, Issue 10,, Pages 11994–12000, December 2009
7. Karthigha M, Latha L and Madhumathi R,"Feature Selection and Classification Models of Intrusion Detection Systems -A Review on Industrial Critical Infrastructure Perspective"
8. Basant Subba, "A Neural Network based NIDS framework for intrusion detection in contemporary network traffic", 2019 *IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, pp.1–6, 2019.
9. Mridula Sharma, Haytham Elmiligi, Fayez Gebali, Abhishek Verma, "Simulating Attacks for RPL and Generating Multi-class Dataset for Supervised Machine Learning", *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pp.0020–0026, 2019.
10. Sun N, Zhang J, Rimba P, Gao S, Zhang LY, Xiang Y, Data-driven cybersecurity incident prediction: A survey, IEEE Commun. Surv. Tutor. 21 (2) 1744–1772 (2019)
11. Akshat Divya, Anchit Bhushan, Nihal Anand, Rishabh Khemka, Sumithra Devi K.A.H (2020) HONEYPOT: Intrusion Detection System. International Journal of Education, Science, Technology, Engineering vol. 3, no. 1, pp. 13–18, June 2020.
12. Husak M, Bartos V, Sokol P, Gajdos A, Predictive methods in cyber defense: Current experience and research 43 challenges, Future Generation Computer Systems, Volume 115, 517–530 (2021)

13. Philip Wester, Fredrik Heiding, Robert Lagerström, "Anomaly-based Intrusion Detection using Tree Augmented Naive Bayes", *2021 IEEE 25th International Enterprise Distributed Object Computing Workshop (EDOCW)*, pp.112–121, 2021.
14. Ritu Bala, Ritu Nagpal, "Intrusion Detection Based on Decision Tree Using Key Attributes of Network Traffic", Applications of Artificial Intelligence and Machine Learning, vol.778, pp.583, 2021.
15. Bo-Xiang Wang, Jiann-Liang Chen (2022): An AI-Powered Network Threat Detection System Doi 10.1109/ACCESS.2022.3175886
16. Ruizhe Zhao, Yingxue Mu, Long Zou, Xiumei Wen, "A Hybrid Intrusion Detection System Based on Feature Selection and Weighted Stacking Classifier", IEEE Access, vol.10, pp.71414–71426, 2022.
17. Ying-Feng Hsu, Morito Matsuoka. "A Deep reinforcement Learning Approach for Anomaly Network Intrusion Detection System," IEEE International Conference, 978-1-9486-8-20, 2020
18. YAO Yu1, Yang Wei2, GAO Fu-xiang1, YU Ge1. "An Anomaly Intrusion Detection Approach Using Hybrid MLP/CNN Neural Network," Proceedings of the Sixth International Conference on Intelligent Systems Design and Applications (ISDA'06), 0-7695-2528-8/06, 2006

Figures

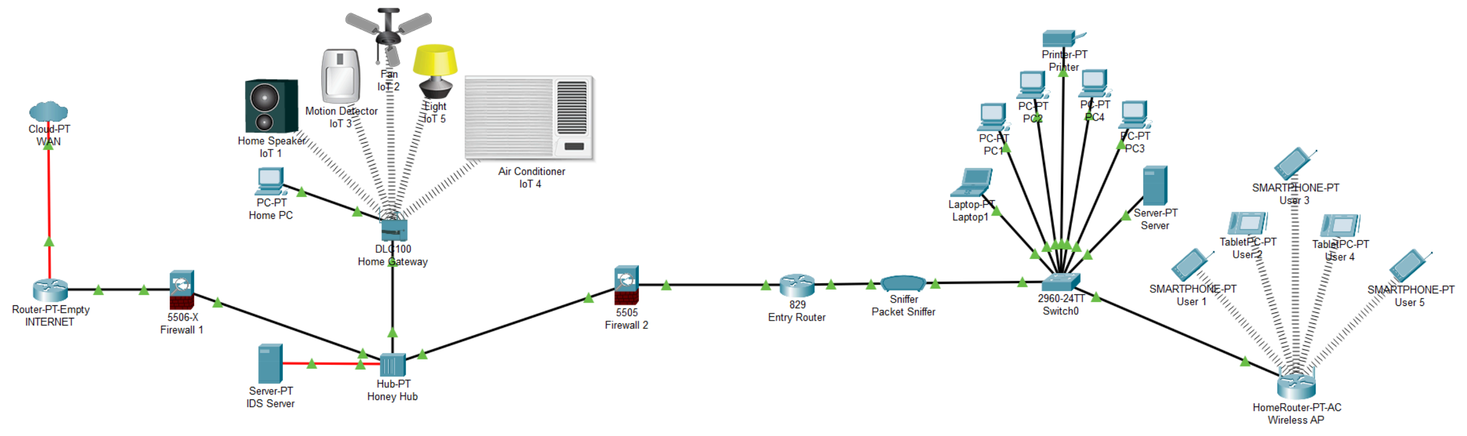

Figure 1

Network Design

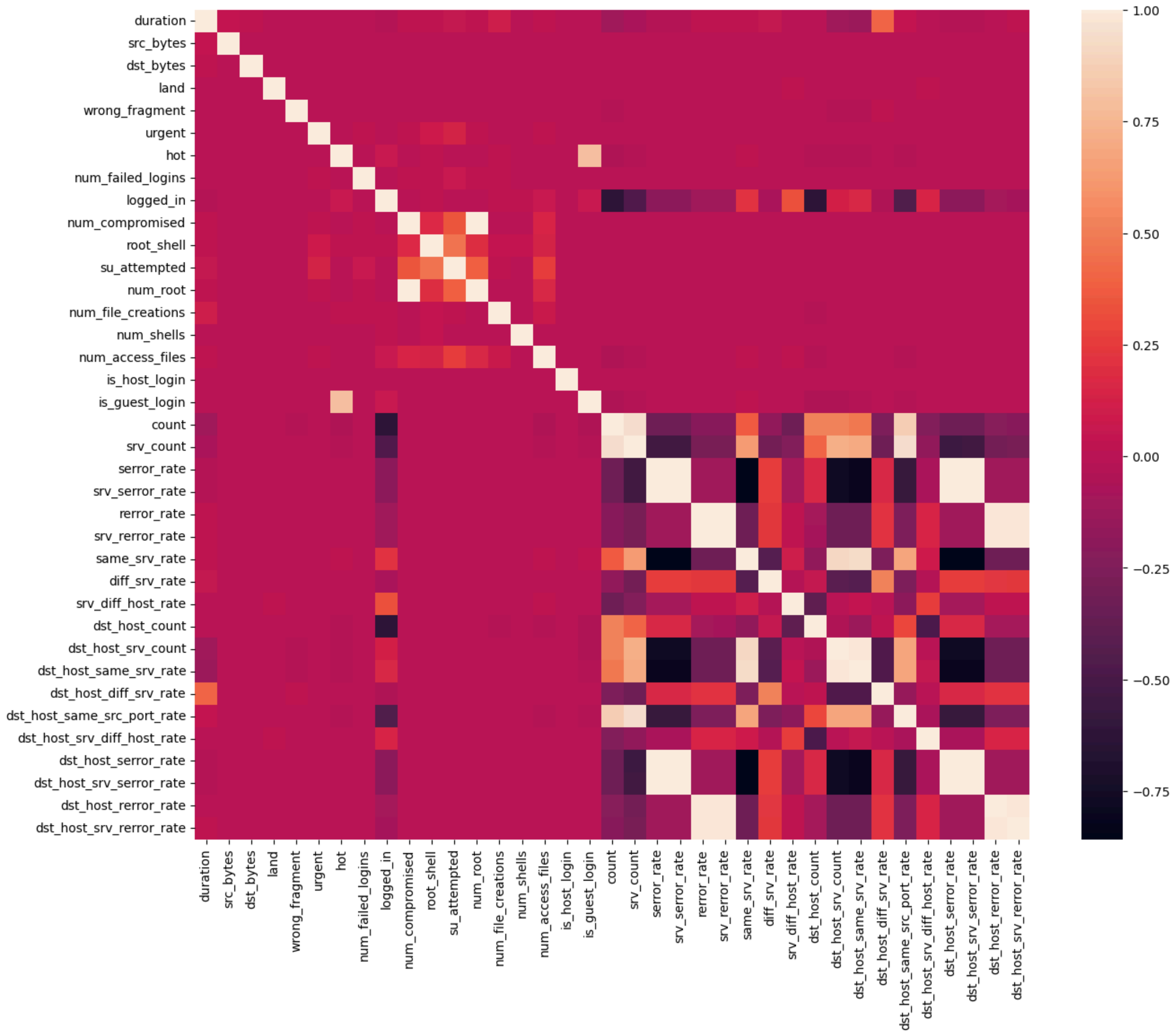

Figure 2

KDD99 Heatmap

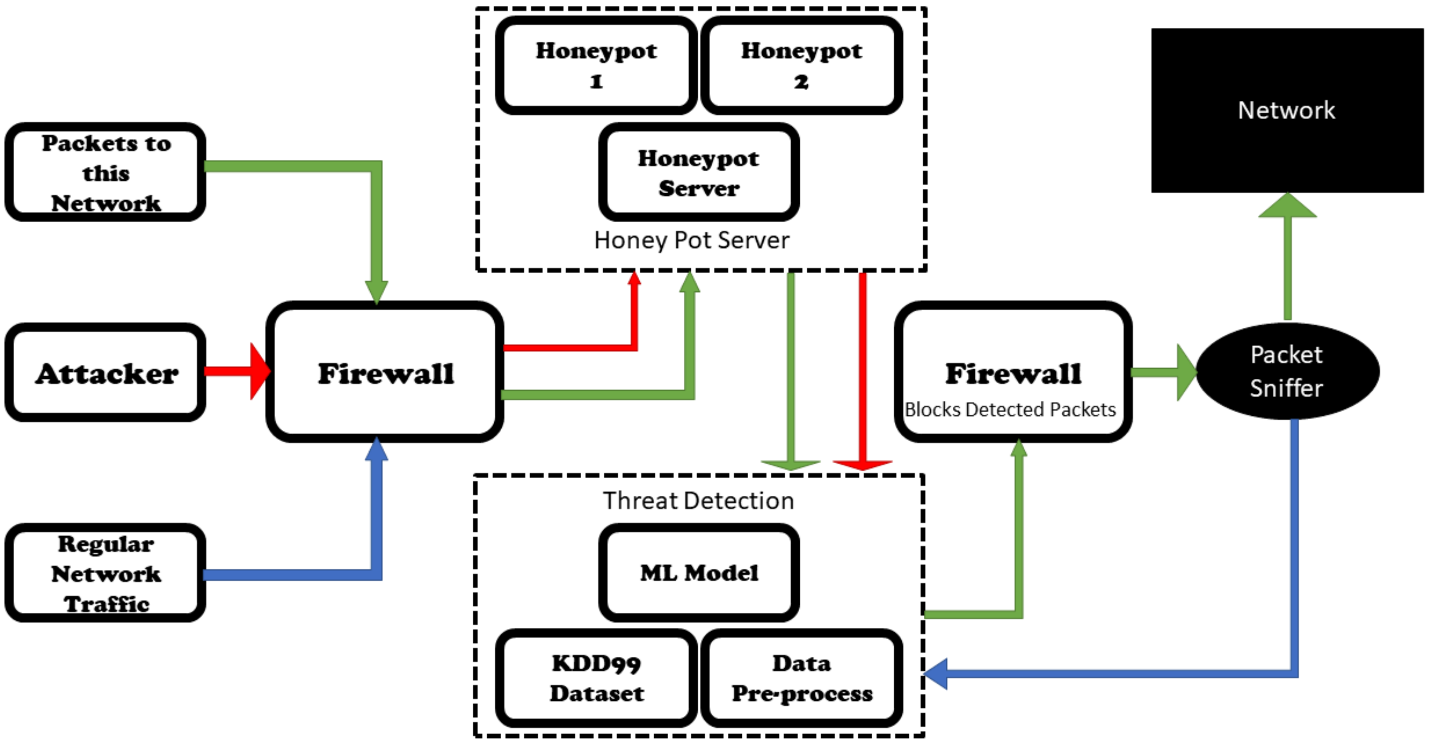

Figure 3

Honeypot architecture

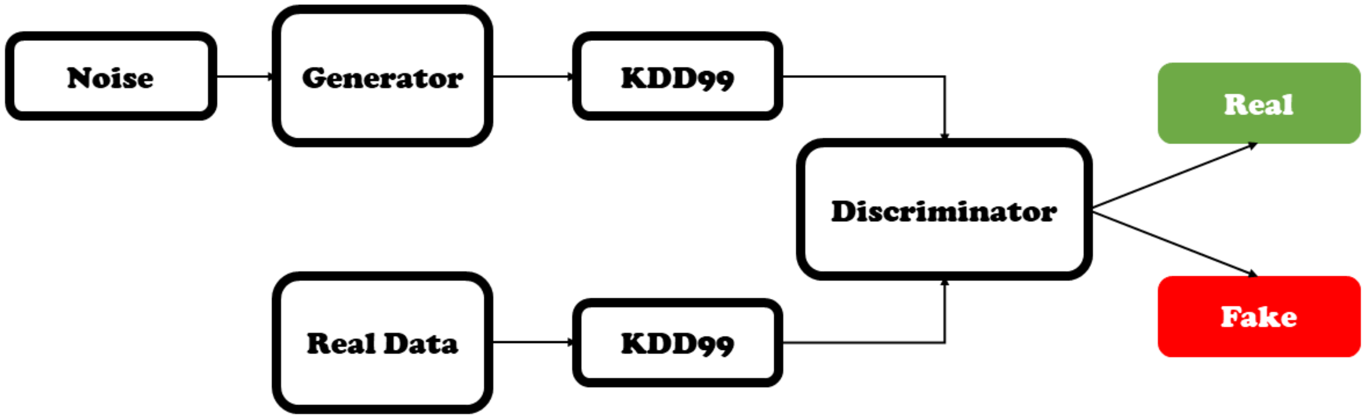

Figure 4

GAN Architecture

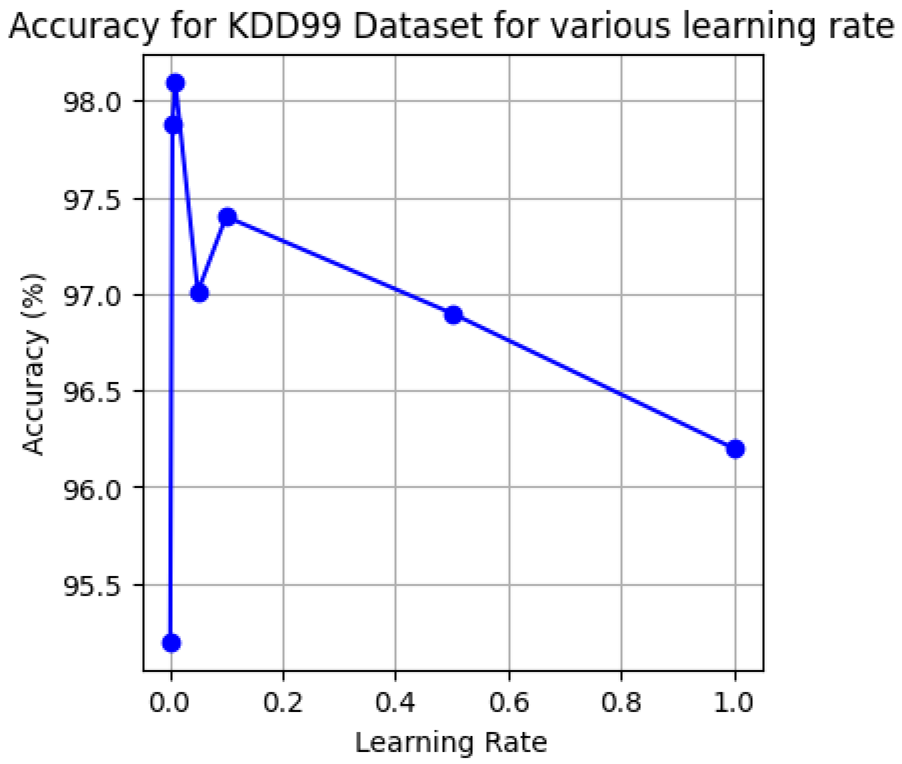

Figure 5

Accuracy for KDD99 dataset with various learning rate

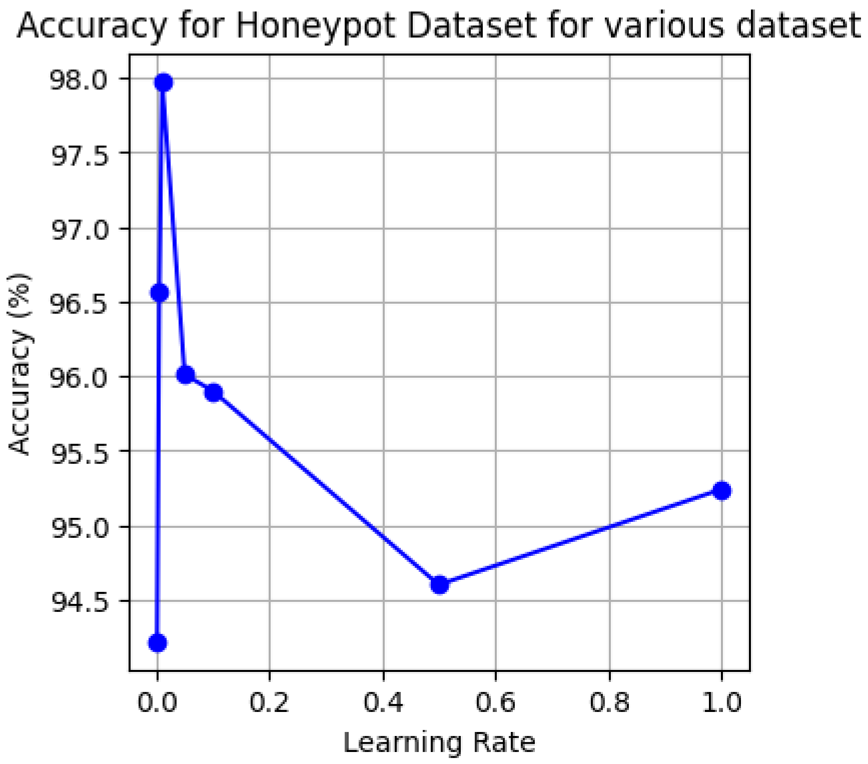

Figure 6

Accuracy of Honeypot dataset with various learning rate

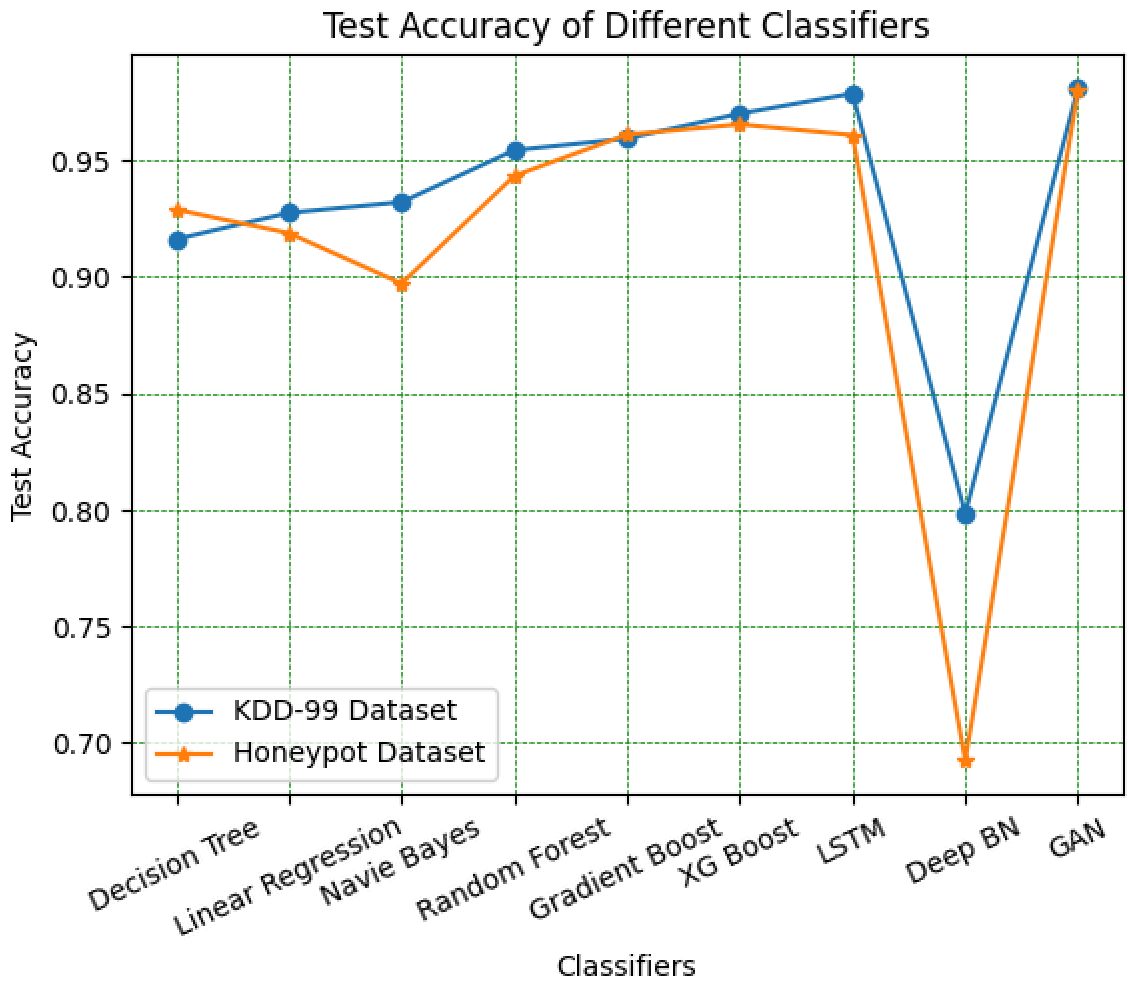

Figure 7

Testing Accuracy of various DL/ML models on KDD-99 and honeypot dataset

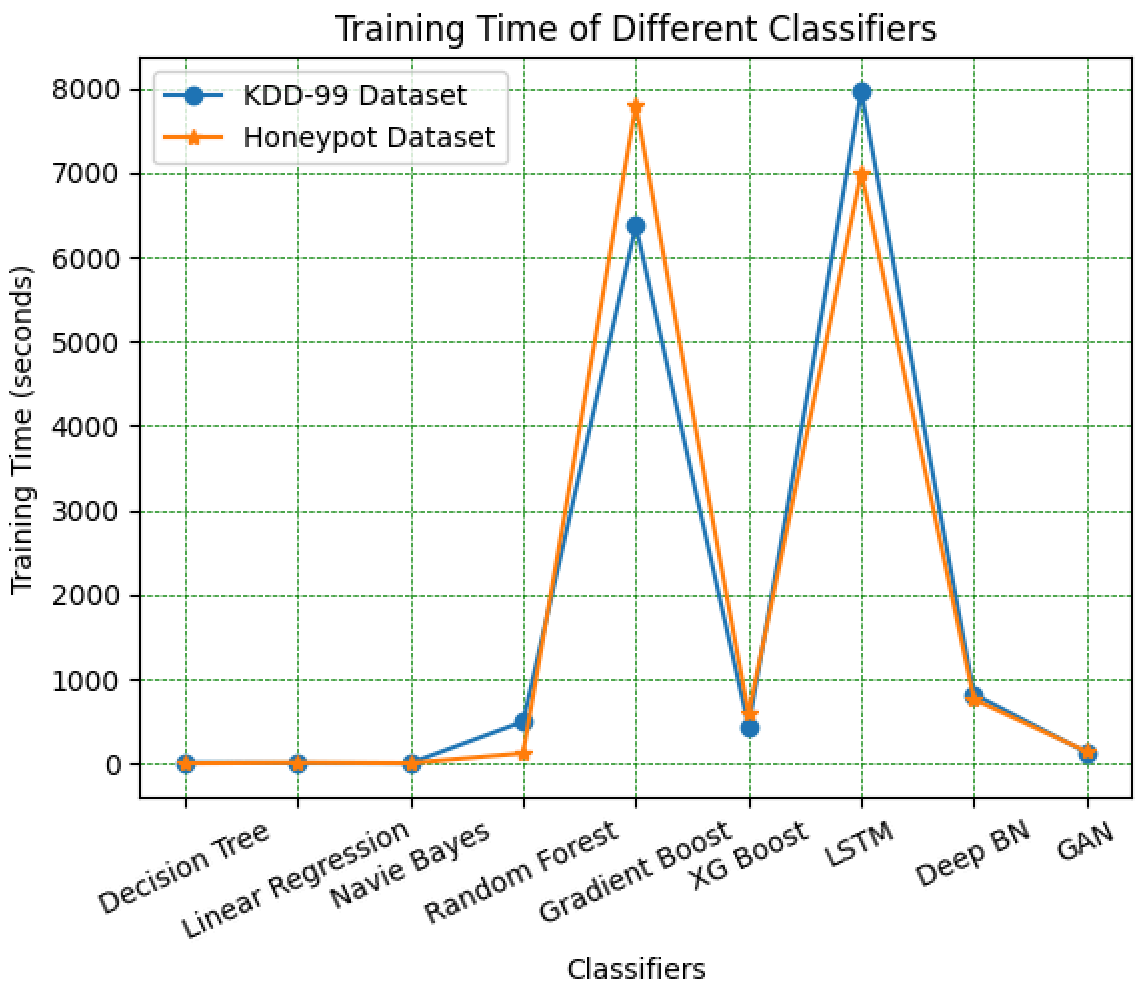

Figure 8

Training Time (in seconds) of various DL/ML models on KDD-99 and honeypot dataset