Journal of Information Systems Engineering and Management
2025, 10(1s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Delay Tolerant Network Security: Enhanced Machine Learning Technique for Intrusion Detection System

Rajashri Chaudhari[1,*], Dr. Manoj Deshpande[2]

[1]Research scholar, ACPCE, Kharghar, Navi Mumbai, Maharashtra, India. c.rajashri2021@gmail.com, rajashri.chaudhari@outlook.com

[2]Professor and Dean, A. C. Patil College of Engineering, Navi Mumbai, India. mmdeshpande@acpce.ac.in

**Correspondence: rajashri.chaudhari@outlook.com*

ARTICLE INFO

Received: 30 Sept 2024

Revised: 27 Nov 2024

Accepted: 14 Dec 2024

ABSTRACT

Delay tolerant networks (DTNs) are intended for effective communication between nodes over huge distances and they are resourceful in extreme conditions. DTN stores and forwards messages when the participant node is in the range. Hence, there is no restriction on end-to-end connectivity and data is transferred over adaptable links between the nodes. Since the network functions on coordination among participant nodes, an untrusted node can affect the coordination. Thus, DTNs are vulnerable to different kinds of attacks influencing the performance of the network. The delay in the transfer of data and unstable connectivity of nodes depends on effective coordination while the possibility of misbehaviour by relay nodes increases network vulnerability to various types of network attacks like packet dropping attacks, flooded attacks, DoS attacks, gray hole attacks, and black hole attack disturbing the network connectivity. Denial of service attacks (DoS) is a major concern in DTN that adversely affects the network. Attacks on DTN can interrupt message delivery and degrade performance. The study proposes the detection of such attacks over DTN with an efficient machine learning (ML) algorithm. The delay tolerant network is a wireless network that transfers information among nodes and is monitored for malicious nodes using a pre-trained ML model. The voting technique is used to enhance the performance of detection. The network attacks are detected with significant accuracy and efficient secure communication is established in the network. Furthermore, the network simulator NS2 is employed to simulate the prevention of malicious attacks in the proposed system. This simulator offers a versatile and customizable environment for modelling various DTN scenarios and assessing the effectiveness of intrusion detection systems in a controlled setting. Our proposed model offers better performance than existing DTN security techniques.

Keywords: Flooding Attacks, Security, DDoS Attack, DTN, Delay Tolerant Network, NS2

INTRODUCTION

Modern devices are interconnected through wireless networks over massive distances. Despite the wide range of connectivity achieved, a large part of the world is however out of end-to-end connectivity. The availability of infrastructure and power is inadequate in most developing countries and lacks reliable end-to-end network connections. Hence, Delay tolerant networks (DTNs) are introduced [1] with significant connectivity in such areas regardless of end-to-end network connectivity.

In DTNs, node connections are unsteady and result in interrupted communication. The packet transmission is used for data transfer among the nodes in these networks. A store-and-forward [2] approach in DTN allows data storage in the buffer of a node. It is a low-cost solution that uses intermediate nodes for the safeguarding of the transferred data and then forwards it to the next node when network connectivity is available. The packet is either transmitted or stored when another node is found. Since DTNs differ from traditional techniques with a disconnected nature, the security parameters also vary including the different privacy solutions.

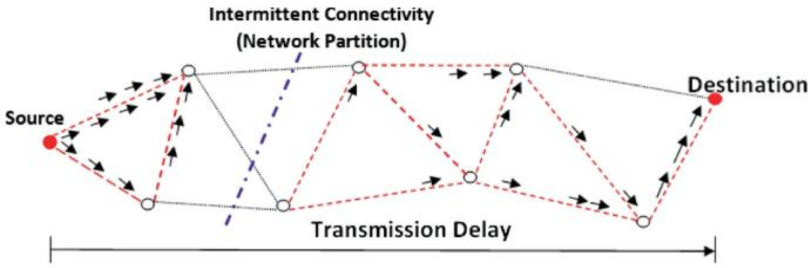


Figure 1: Data transfer in DTN [3]

DTNs comprise several types of entities and are not the same as ad-hoc networks. Hence, the resolutions in ad-hoc networks for security and obscurity are not applicable to DTNs. DTNs also suffer from network attacks like other networks. However, it has an adverse impact on data security and raises issues in reliable communication. Several challenges are introduced in network protection, including nodes from intruders, expensive techniques, and an unsteady network.

Network security is becoming an essential research topic with an increase in the use of the Internet. Traditional networks including wired as well as wireless connections are facing network security issues and several kinds of research focus on the efforts to protect the networks from attacks. Security issues have been recognized as new challenges with network evolution over the past few decades. DTN has a major challenge of passive attacks affecting the security of sensitive data. In these attacks, an intruder may extract some sensitive data just by monitoring the broadcast of a satellite. The intruders might eavesdrop on either intended or non–intended data [4]. The predefined metrics in DTN include delivery time (latency) and probability with previous encounter time and utility value. Also, the routing protocols in DTN are essential in the proposed method. For instance, MaxProp [5], Spray & Focus [6], and PRoPHET [7] are protocols used in DTN.

Flood attacks on DTNs result in an interruption in network traffic and result in a denial of service (DoS) attack. It occurs due to a large number of incomplete connection requests stopping network traffic of original connection requests from flooding a network or Service. It acquires the memory buffer of the host with incomplete connections and results in denial of service to make new connections [8]. It may cause degradation as well as system failure in the network. Hence, the study focuses on the detection of flood attacks during data transfer and monitoring packet transmission over the network. The performance of methods used to detect attacks is analyzed for better outcomes.

Machine learning-based approaches are effective solutions for network management and offer smarter security and optimization techniques. DoS attacks are crucial to detect for maintaining network security and captivating the essential measures regularly. The self-learning network can be obtained with machine-learning techniques in the DTN network for the accurate detection of DoS attacks. These models integrated into DTN reduce the impact of DoS attacks and prevent the networks efficiently. Furthermore, these models can be implemented to secure 5G networks in future studies.

DTNs are difficult to secure connections due to high instability in connections while the nodes move in and out of the network. Nodes can misbehave and cause problems for the network [9]. The attacks on DTNs cause a denial of service and restrict new connections. Thus, this study focuses on intrusion detection to reduce the impact of attacks like DDoS attacks (Distributed Denial of Service), flooded attacks and other types of attacks. It is difficult to pre-process the data with increasing network attacks. Thus, to recognize the attacks before they occur, Machine Learning can be used that enhance attack detection using a variety of algorithms. The ML algorithms are capable of analysing the data in DTNs for malicious nodes and different network attacks. The ML-based systems can detect possible flooding attacks in DTN and protect the networks from attacks.

LITERATURE WORK

According to recent research, cyber security is one of the crucial global issues with a continuous increase in digitalization. Thus, malware is a significant internet threat that rapidly spreads as a big threat to cyber security. Henceforth, network security includes neutralizing these threats as an important part of cyber security. In malware classification, several approaches are proposed with advanced techniques like machine learning. A study [10] proposes a machine-learning model including some malicious and self-generated benign PCAPs. The malware in

HTTP traffic is classified the HTTP traffic with a Random Forest (RF) algorithm as benign or malicious. The accuracy obtained is 90% in the machine learning-based classification model. It includes a sample dataset of Benign, Scareware, Ransomware and Adware types of malware to train the model. Results show greater accuracy and a small false positive rate.

Khan et al. [11] include an ensemble-based voting technique for intrusion detection that combines many classifiers. The voting selects predictions of classifiers for more accurate results. IoT-based Global Positioning System (GPS) and weather sensors are tested for binary and multi-class attacks with a higher accuracy of 96-97% as compared to an accuracy of 85-87% achieved by ML algorithms. While Alasmary et al. [12] proposed a majority voting technique ShieldRNN for DDoS attack detection over an IoT network. It includes 12 classifiers in the training of the model including LR, NB, ANN, RF and SVM classifiers. They used four different neuron sizes in ANN, three different numbers of trees in RF and three SVM (linear, RBF, Polynomial) classifiers. However, RF achieved the highest accuracy on the testing set.

The ensemble model uses a voting technique in [13], to combine IB1, RF, and SVM algorithms while XGBoost and majority voting are applied in [14], [15] for intrusion detection on ICSs. A voting approach in a robust model recognizes several types of attacks merging XGBoost, kNN, LightGBM, RF and DT algorithms. Furthermore, the VisDroid [16] model implemented by Bakour et al. uses a voting technique on ML classifiers such as AdaBoost, KNN, GBC, RF, DT, and Bagging for android malware detection. In [17], Fast Android Malware Detector is proposed (FAMD model) with the original feature set obtained by Dalvik opcode sequences and CatBoost classifier is used with reduced features in the malware detection process. The algorithm recognized as a Fast Correlation-Based Filter reduces feature dimensionality using symmetrical uncertainty and sends the features to CatBoost which achieves 97.40% of accuracy on the Drebin dataset.

Anomaly detection is enhanced with the Adaboost algorithm [18] combining the decision tree as the base weak learner and obtaining precision higher than 0.999 value. Significant detection is achieved with Adaboost against network attacks. Furthermore, ML techniques are compared to predict network attacks by Alekseeva et al. [19] using Linear Regression, Support Vector Machines (SVM) and Random Forest in addition to other techniques like Gradient Boosting, Bagging (Bootstrap Aggregation), Huber and Bayesian Regression etc. The results illustrate SVM and Gradient Boosting as faster training and the best prediction quality algorithms respectively. Also, Random forest and Bayesian regression are less efficient than Gradient Boosting, but their training time is also less. Moreover, a network intrusion detection system proposed with XGBoost [20] illustrates great performance on two representative datasets.

According to a literature survey conducted on several studies, proposed models are inadequate to detect some types of attacks due to a decrease in features. Also, the simulation of a few nodes and insufficient dataset information affect the reliability of the studies. Some studies use fewer evaluation parameters and miss important parameters. Studies are based on a few types of attacks and can detect only defined attacks in the network. The analysis of parameters like accuracy, precision, PDR, and E2E Delay is also needed to focus on enhancing the performance of detection systems. A longer training time and less PDR reduce the efficiency of attack detection. These studies are based on different algorithms that need to be studied for the best choice of an efficient algorithm. The rapid detection techniques proposed with reactive strategies show an increase in the cost of overhead though the time is reduced. Therefore, reducing overhead is another challenge in existing systems. The attack detection model needs secured strategies as well as connectivity between trusted nodes.

Thus, the study proposed the detection of attacks in DTN overcoming these issues as well as reduction of time and overhead. It considers DoS attacks in DTN networks using machine learning techniques to recognize and prevent attacks over the network.

METHODOLOGY

Machine learning technique is used to detect attacks on DTNs. The ML model is pre-trained with a dataset of intrusion examples to differentiate anomaly and normal connections. ML models are based on feature selection techniques for the recognition of attacks where selected features are extracted to analyze the data. Several machine learning techniques can be used for detecting flooded attacks with great performance. The proposed model uses different classifiers to enhance model accuracy. Figure 2 shows the complete architecture of the ML-based model for attack detection in DTNs.

The entire framework of the proposed model is discussed below. It considers some of the exposed issues to detect attacks over the network. The attack detection and prevention system is proposed with a machine learning model and a competent algorithm is used to detect flood attacks.

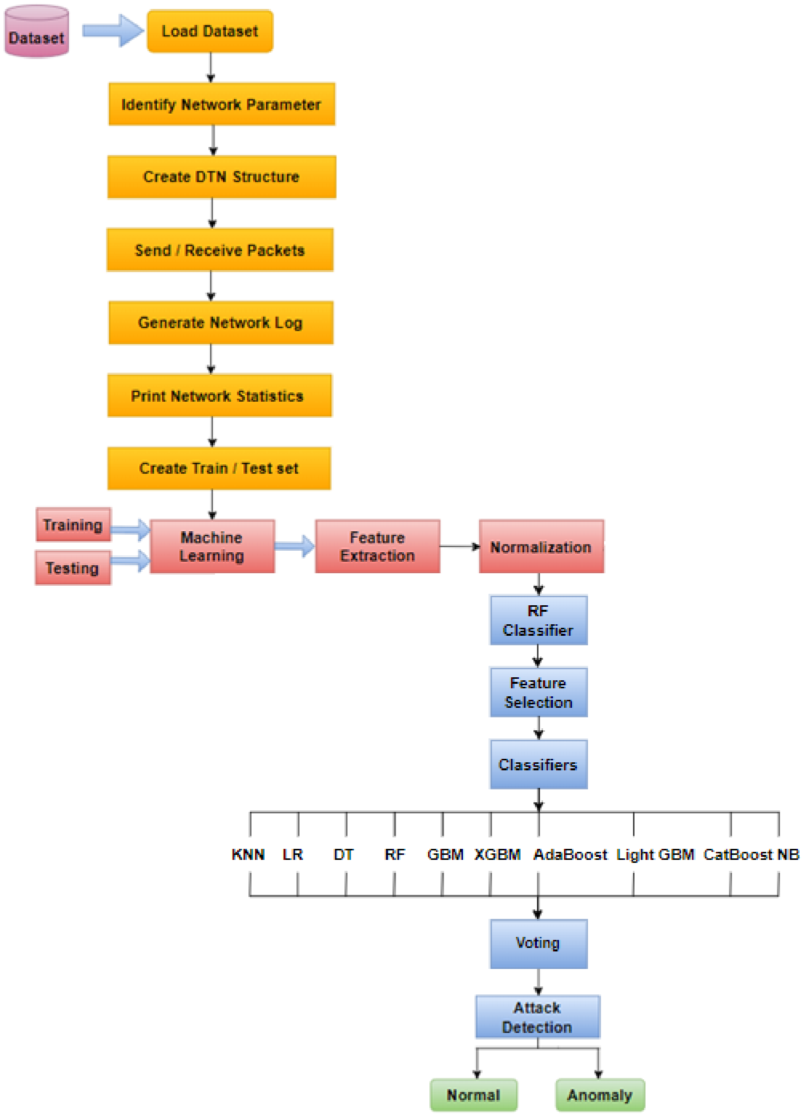


Figure 2: Architecture of DTN attack detection

A dataset loaded into the system identifies the parameters of attacks. Further, a DTN structure is created with a number of nodes. This network is allowed to transfer data packets from source to destination nodes over the created network. The network log is generated for data transfer occurrences and network statistics are allowed to print with details of the data packet.

It includes a number of packets and their size with source and destination IDs to transfer data. The time required to transfer data is calculated with loss rate and average system occupancy. Using the dataset, train and test datasets are created for the ML-based attack detection model. The process includes feature extraction and normalization of the dataset to introduce features of defined classes. Feature selection using an RF classifier reduces the features extracted from data and then KNN detects attack data classifying it into anomaly or normal class.

3.1 Dataset

The dataset is an important part of the ML model that should include a variety of intrusion data. The study used the Network Intrusion Detection (NID) [21] dataset available on Kaggle source to train and test models. The NID dataset counterfeit in a military network environment includes an extensive diversity of data including intrusion samples.

The US Air Force LAN was blasted with multiple attacks using raw TCP/IP dump data. For each TCP/IP connection, data flows from a source IP address to a target IP address for some time duration.

The dataset contains normal and attack data with 41 features including 3 qualitative and 38 quantitative features. The features used in this model are reduced by 2. Two class variables - normal and anomalous are used to label each connection for normal and attack-type data accurately with one specific attack type. NID dataset of size 5.29 MB occupies 60% of normal and 40% of anomalous attacks divided into train and test datasets.

3.2 Machine Learning

Machine learning models need to be trained for sample data related to flood attacks. The data is obtained from reliable datasets in DTN with a variety of intrusion examples. The attacks are introduced to machine learning models and processed with the algorithm. The dataset is divided into 60% and 40% of data for training and testing the model. An input to the model contains normal and intrusion data whereas the variety of data in the dataset enhances the model's accuracy. The training of the model extracts features from the data and selects desired features to identify anomaly input. Furthermore, the time window is analyzed to calculate time consumption. The system also aims to reduce time consumption for flood attacks and decrease memory consumption as well.

3.2.1 Feature Extraction

Feature extraction is a crucial part of the classifier model. A dataset consists of several features and the selected features are introduced to the training model to differentiate network attacks from normal connections. The study used a supervised ML approach where a set of examples train the classifier using data related to different features and variables. The training and testing datasets contain 25,192 and 22,544 samples, respectively, and each dataset includes 41 features for intrusion and normal data. Thus, to detect flood attacks, characteristics of normal network behaviour are differentiated from flood attacks. The proposed method includes data packets of the TCP/IP architecture to reduce the complexity of the model [22]. These features are analyzed to classify the input into two different classes – normal and anomaly class.

3.2.2 Feature Selection

The dataset is reduced using a feature selection approach and further classification technique is applied to distinguish the data according to features selected for each class such as normal data and attack data. Different classifiers efficiently perform in machine learning models. RF algorithm is popular for better performance in classifying large data and creating several decision trees while training the ML model. RF classifier reduces features extracted from data.

Table 1: List of features selected using the classifier

Sr. No.	Features	Description
1	src_bytes	number of data bytes from source to destination
2	diff_srv_rate	% of connections to different services
3	dst_bytes	number of data bytes from destination to source
4	service	network service on destination
5	same_srv_rate	% of connections to the same service
6	count	number of connections to the same host as the current connection in the past two seconds
7	dst_host_same_srv_rate	same_srv_rate for destination host
8	flag	normal or error status of the connection
9	dst_host_srv_count	srv_count for destination host
10	protocol_type	type of protocol – tcp/udp

The features are first extracted from the dataset and the RF classifier calculates the weight for each feature according to their importance. It helps to remove noise from the dataset and select relevant features enhancing the classification performance of the model. The figure shows a graphical representation of the feature importance obtained by using an RF classifier.

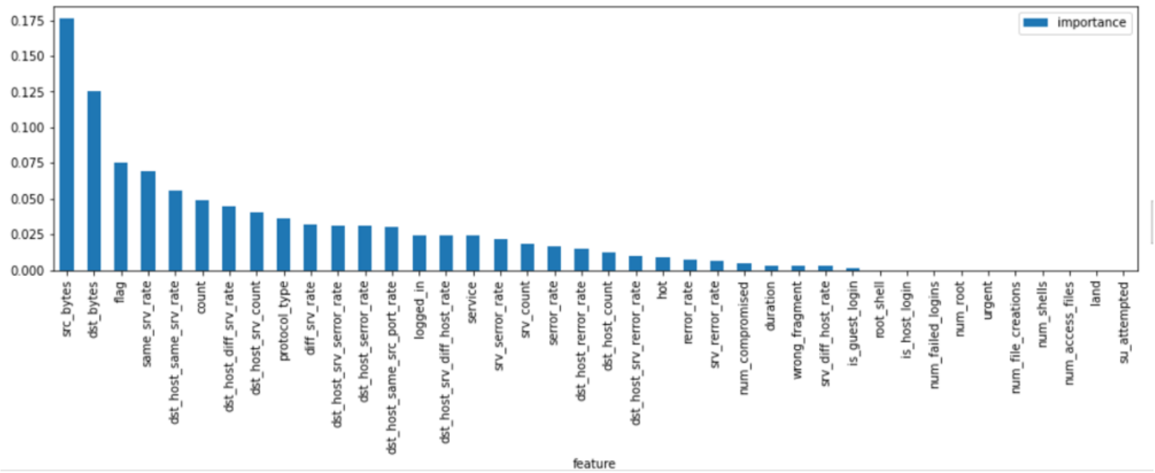


Figure 3: Train and test data feature classification with their importance

3.2.3 Attack detection

The attack detection is based on training data features. The KNN classifier is trained with all features in the dataset and tested for attack detection. As the dataset is divided into training and testing datasets, features are selected in the second stage to capture the most relevant and effective features. There are 40 features extracted from the dataset out of which 10 features are selected as shown in table 1.

RESULTS AND DISCUSSION

Packets are generated with details of source ID and type of protocol including time and size. The table shows these details for packets. These 10 packets are generated with TCP and UDP protocol defining packet ID, time in milliseconds and size in bytes for each packet.

Table 2: List of generated packets

Sr. No.	Packet ID	Type of protocols	Time (ms)	Size (bytes)
1	1	TCP	1.5	17.69
2	1	UDP	2.0	0.69
3	2	TCP	3.0	45.86
4	2	UDP	4.0	24.54
5	3	TCP	4.5	92.09
6	3	UDP	6.0	227.36
7	4	TCP	6.0	20.34
8	4	UDP	8.0	117.96
9	5	TCP	7.5	133.66
10	6	UDP	9.0	170.78

The generated packets are sent over the network and it is observed that out of 10 packets sent over the network only 4 packets with TCP protocol are received while 6 packets are dropped. Furthermore, the waiting time taken to reach the destination for each packet is also calculated in terms of the total time required to send all packets over the DTN.

Table 3: List of received packets

Packet id	Type of protocols	Time (ms)	Waiting time (ms)
1	TCP	1.5	4.0
2	TCP	3.0	6.5
3	TCP	4.5	9.0
4	TCP	6.0	11.5

Here the waiting time indicates the time taken for receiving the packets. Each packet is assigned a time to reach the destination and the overall time to receive all packets is evaluated. A number of packets sent over the network take time to reach the destination due to the unstable connectivity of nodes in the network.

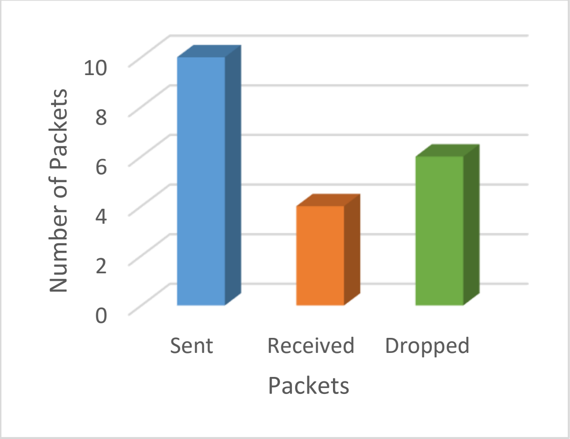


Figure 4: Packet transfer data

4.1 Voting Technique

Voting combines predictions of all algorithms to derive more accurate and precise outputs. Thus, the Voting technique is employed to evaluate the results and compare the results with individual algorithm output as shown in the table. Voting is recognized as a more accurate technique as compared to other ensemble techniques and system performance is enhanced with significant results of all algorithms. It combines the results of all algorithms and selects likely correct outputs. Figure shows voting classifier including four classifiers C1, C2, C3, C4 providing predictions P1, P2, P3 and P4 which are further processed by Voting algorithm to attain final prediction [11].

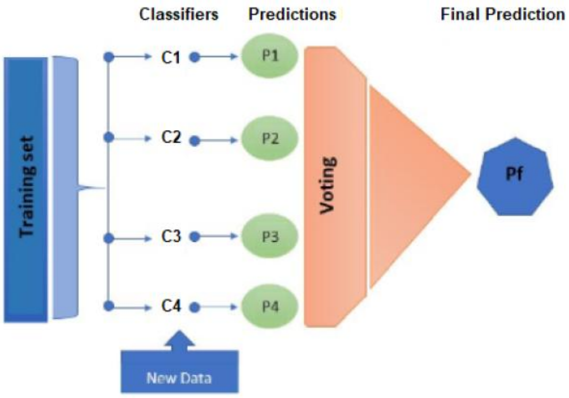


Figure 5: Voting technique using different classifiers for final prediction [11]

4.2 Performance of attack detection model

In the research, the voting algorithm combines the performance of 10 algorithms including KNN, DT, GBM, Adaboost, CatBoost, LR, RF, XGBM, Light GBM, and Naïve Baye Mode 1. The table shows precision, recall and F1-score for the model in the training and testing phase with train and test scores of the model.

Table 4: Performance of Voting Algorithm

Dataset	Precision	Recall	F1-score	Score	Model Accuracy
Training	0.99	1.00	1.00	0.998242	0.9957
Testing	1.00	1.00	1.00	0.995766	

Results show that the Voting technique gives more accurate results by combining the benefits of all included algorithms. After applying various algorithms, the model accuracy obtained by the voting technique performs well with an accuracy of 0.9957.

The proposed model works efficiently on the training datasets achieving the highest accuracy of 0. 9957 using the voting algorithm. As shown in Table, an excellent performance of the attack prediction model is obtained for defined parameters. Thus, the voting method attains state of art on the stated evaluation metrics. For extraction of more attack information, requires more features. An increase in the number of features also increases the number of inputs and computational complexity with higher processing time. Hence, the prediction time is reduced by extracting less number of features that help to reduce processing time. Since less number of inputs reduce complexity in processing and response faster, only the most relevant features are selected. In short, time is significantly decreased with better accuracy and precision value.

The graphical representation of these parameters is illustrated in the figure below.

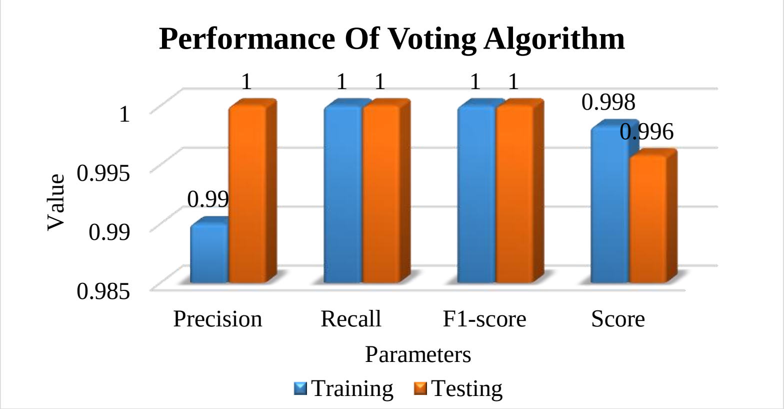


Figure 6: Training and Testing Performance of Voting Algorithms

The attacks detected by the model are further analyzed to evaluate the percentage of normal and anomaly attacks. It is found that most of the anomaly attacks detected by the model contribute to enhancing DTN data transfer efficiency. As depicted in the figure, the trained model detected 53.4% normal attacks and 46.6% anomalies across the network.

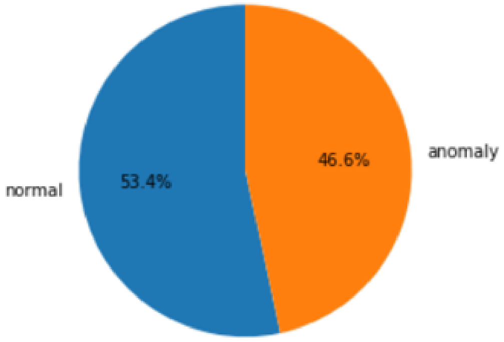


Figure 7: Classification of attack detection

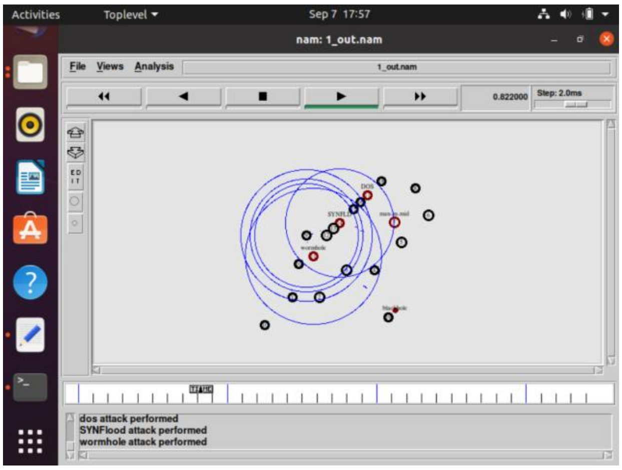
4.3 Simulation of Attack Detection on NS2

NS2 (Network Simulator 2) is a widely utilized open-source network simulation tool applicable in Delay-Tolerant Networks (DTNs) and attack detection. NS2 enables the creation of realistic DTN environment simulations, replicating typical DTN characteristics such as intermittent network connectivity, extended delays, and frequent disconnections. It introduces various types of malicious attacks into the simulated DTN environment, including intrusion attempts, data tampering, denial-of-service (DoS) attacks, and other security threats.

The study involves the implementation and testing of robust and reliable security protocols designed to safeguard DTNs from attacks. Data collected from NS2 simulations is used to analyze network behavior and assess the impact of malicious attacks on DTNs. This data aids in fine-tuning security mechanisms and the development of more resilient DTN architectures.

4.3.1 NS2 Results

To simulate an Intrusion Detection System in NS2, it is essential to define the network topology, nodes, and other simulation parameters. Features are specified (refer to Table 1) and the number of nodes is set to 20 in order to incorporate a routing protocol into NS2.



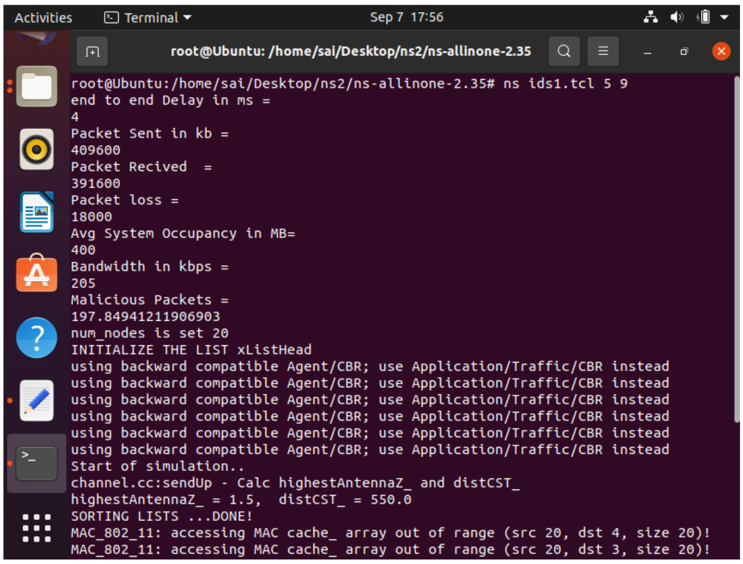
```
dos attack performed
SYNFlood attack performed
wormhole attack performed
```

Figure 8: Simulation of nodes in NS2 environment

Defining the network topology with these 20 nodes involves specifying their locations, and connectivity parameters, as well as assigning different colors and shapes to each node. These nodes move within a rectangular area at random speeds and pause for varying durations. Subsequently, a routing protocol is selected and configured to meet simulation requirements, encompassing routing algorithms and packet forwarding strategies. Additionally, the study implements intrusion detection mechanisms to monitor network traffic for suspicious activities.

After configuring attack scenarios, such as packet drops, data tampering, or intrusion attempts, within the simulated environment, the NS2 simulation is executed. The routing protocol governs data forwarding among the 20 nodes, while the intrusion detection system monitors and responds to potential attacks. This allows for the analysis of the network's performance and security under various conditions.

In the test case, nodes 5-9 are selected for sending packets, and the simulator observes packets received and identifies malicious packets. Implementing attacks on nodes 5 to 9 generates malicious traffic or behavior on these nodes. The intrusion detection system in the scenario examines network traffic and flags packets or nodes as malicious based on defined criteria. After running the simulation, the generated packets are sent over the network, and packet loss and the accuracy of received packets are evaluated using the simulation results.



```
root@Ubuntu:/home/sai/Desktop/ns2/ns-allinone-2.35# ns ids1.tcl 5 9
end to end Delay in ms =
4
Packet Sent in kb =
409600
Packet Recived  =
391600
Packet loss =
18000
Avg System Occupancy in MB=
400
Bandwidth in kbps =
205
Malicious Packets =
197.84941211906903
num_nodes is set 20
INITIALIZE THE LIST xListHead
using backward compatible Agent/CBR; use Application/Traffic/CBR instead
using backward compatible Agent/CBR; use Application/Traffic/CBR instead
using backward compatible Agent/CBR; use Application/Traffic/CBR instead
using backward compatible Agent/CBR; use Application/Traffic/CBR instead
using backward compatible Agent/CBR; use Application/Traffic/CBR instead
using backward compatible Agent/CBR; use Application/Traffic/CBR instead
Start of simulation..
channel.cc:sendUp - Calc highestAntennaZ_ and distCST_
highestAntennaZ_ = 1.5,  distCST_ = 550.0
SORTING LISTS ...DONE!
MAC_802_11: accessing MAC cache_ array out of range (src 20, dst 4, size 20)!
MAC_802_11: accessing MAC cache_ array out of range (src 20, dst 3, size 20)!
```

Figure 9: Simulation Results for 5 to 9 nodes

Table 5: Simulation results

Sr. No.	Parameter	Value
1	Number of Nodes Generated	20
2	Selected Nodes (Sending)	5 to 9
3	Packets Sent (KB)	409600
4	Packets Received (KB)	391600
5	Packet Loss (KB)	18000
6	Average System Occupancy (MB)	400
7	Bandwidth (kbps)	205
8	Malicious Packets (KB)	197.85
9	End-to-end delay (mS)	4

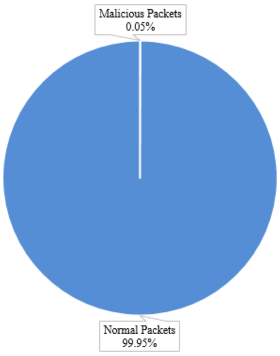


Figure 10: Performance of attack prevention

4.3.2 Test cases

To validate the simulation results and assess the effectiveness of IDS algorithms in controlled scenarios, the incorporation of test cases is crucial within the simulation environment. The table below presents various test cases involving distinct node selections. These test cases assess the accuracy of packets transmitted across the network using randomly selected nodes. In each test case, attacks are introduced to the system and subsequently detected and prevented during node communication within the network.

Table 6: Test cases for sending packets

Sr. No.	Test Case	Nodes Selected	Packets Sent	Packets Received	Packet Loss	Accuracy %
1	TC1	5 to 9	409600	391600	18000	95.59
2	TC2	3 to 7	350000	332500	17500	95.00
3	TC3	8 to 12	420000	400000	20000	95.24
4	TC4	10 to 14	380000	363000	17000	94.74
5	TC5	15 to 19	450000	427500	22500	94.83

In each test, the accuracy is consistently high, indicating the superior performance of the proposed system. The accuracy ranges from 94.74% to 95.59%. The highest accuracy achieved within the NS2 simulation environment is 95.59%. The figure provides a graphical representation of the accuracy obtained in various test cases.

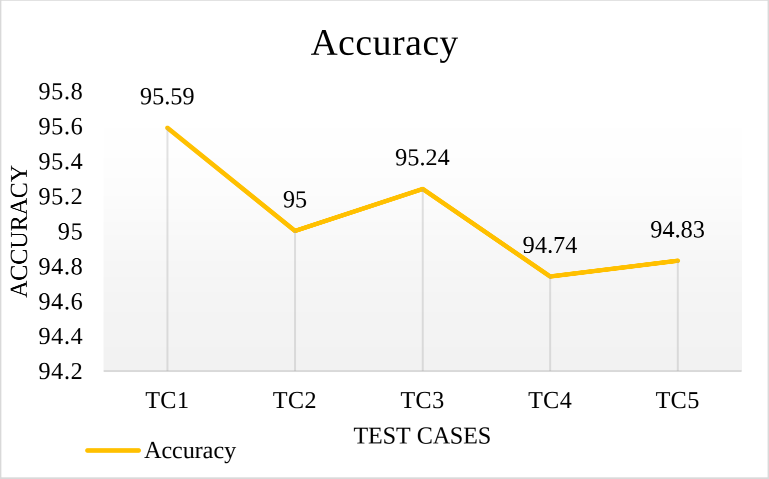


Figure 11: accuracy of various test cases

CONCLUSION

With increasingly sophisticated attacks on DTN, different methodologies are promoted to deal with attacks. Due to easily accessible techniques in DTN, malicious practices are difficult to detect in a large network. However, new attack techniques in the network are challenging to detect various types of attacks. Machine learning has the capability to handle complex data processing in the detection of such attacks and provides efficient accuracy in recognizing anomaly attacks from normal data transfer. However, ML classifiers play an important role in classifying the anomaly and normal activities over the network. The proposed model focuses on flood attack detection and uses 10 different classifiers to detect attacks over the network precisely. The model used a voting technique to enhance the accuracy and attain 99.57 per cent accuracy in attack detection over DTN and practically addressed attack reduction within less processing time. The results obtained illustrate the successful detection model with significant performance of the voting algorithm using a potential dataset of Network Intrusion Detection. However, the Voting technique improves the attack detection by selecting each output from each algorithm making it a more appropriate model.

Additionally, the network simulator used in DTN attack detection with the routing protocol validates the system's performance. NS2 serves as a valuable tool for DTN attack detection by providing a platform to model, simulate, and evaluate various aspects of DTNs, including their vulnerability to attacks and the effectiveness of countermeasures such as intrusion detection systems. It also facilitates controlled experiments and offers insights for enhancing the security of DTNs.

The system achieves a higher accuracy in attack prevention with a lower average system occupancy of 400 MB and a bandwidth of 205 kbps. Moreover, the accuracy of packet reception is considerably high with a negligible number of malicious packets (197.85 kb) received. As the system efficiently prevents attacks, it successfully receives 99.95% of normal packets. Thus, the proposed system demonstrates significant performance in attack detection.

5.1 Future Work

The proposed model is significant for detecting different intrusion attacks over the DTN network. In future work, the study will include a study and test the model for different datasets of various types of intrusion attacks. Furthermore, the study will focus on developing an algorithm which can detect and prevent DOS attacks in DTN. It is intended for generating node movement over the DTN network. The study will implement the proposed system to detect various attacks over the DTNs and the security threats are to be reduced significantly. It can be further improved to the higher performance of the system and can deal with a multitude of other attacks in future research.

Acknowledgments: There is no funding for this research project.

Conflicts of Interest: The authors declare no conflict of interest.

REFERENCES

[1] S. Perumal, V. Raman, G. N. Samy, B. Shanmugam, K. Kisenasamy, and S. Ponnan, "Comprehensive literature review on delay tolerant network (DTN) framework for improving the efficiency of internet connection in rural regions of Malaysia," *Int. J. Syst. Assur. Eng. Manag.*, vol. 13, pp. 764–777, Mar. 2022.

[2] P. Godha, S. Jadon, A. Patle, I. Gupta, B. Sharma, and A. Kumar Singh, "Architecture, an efficient routing, applications, and challenges in delay tolerant network," in *2019 International Conference on Intelligent Computing and Control Systems, ICCS 2019*, May 2019, pp. 824–829. doi: 10.1109/ICCS45141.2019.9065315.

[3] K. K. Ahmed, M. H. Omar, and S. Hassan, "A Comprehensive Survey on Delay Tolerant Networks," *4th Int. Conf. Internet Appl. Protoc. Serv.*, pp. 1–6, 2015.

[4] N. Singh, A. Dumka, and R. Sharma, "Comparative Analysis of Various Techniques of DDoS Attacks for Detection & Prevention and Their Impact in MANET," in *In Performance Management of Integrated Systems and its Applications in Software Engineering, Springer, Singapore.*, 2020, pp. 151–162.

[5] J. Burgess, B. Gallagher, D. Jensen, B. L.- Infocom, and U. 2006, "MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks.," *Infocom*, vol. 6, 2006.

[6] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility," in *Proceedings - Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops 2007*, 2007, pp. 79–85.

[7] K. Henmi and A. Koyama, "Hybrid Type DTN Routing Protocol Considering Storage Capacity," in *Lecture Notes on Data Engineering and Communications Technologies, Springer International Publishing*, Springer Science and Business Media Deutschland GmbH, 2020, pp. 491–502. doi: 10.1007/978-3-030-39746-3_50.

[8] I. ALODAT, "Monitor Potential Attack Locations in a Specific Area within DTN Network," *Comput. Inf. Sci.*, vol. 14, no. 2, p. 42, 2021, doi: 10.5539/cis.v14n2p42.

[9] W. Khalid, N. Ahmed, M. Khalid, A. Ud Din, A. Khan, and M. Arshad, "FRID: Flood attack mitigation using resources efficient intrusion detection techniques in delay tolerant networks," *IEEE Access*, vol. 7, pp. 83740–83760, 2019.

[10]A. P. Singh and M. Singh, "Classification of Malware in HTTPs Traffic Using Machine Learning Approach," *El-Cezeri J. Sci. Eng.*, vol. 9, no. 2, pp. 644–655, 2022.

[11]M. A. Khan *et al.*, "Voting Classifier-Based Intrusion Detection for IoT Networks," Springer, Singapore, 2022, pp. 313–328. doi: 10.1007/978-981-16-5559-3_26.

[12]F. Alasmary, S. Alraddadi, S. Al-Ahmadi, and J. Al-Muhtadi, "ShieldRNN: A Distributed Flow-Based DDoS Detection Solution for IoT Using Sequence Majority Voting," *IEEE Access*, vol. 10, pp. 88263–88275, 2022.

[13]J. Vavra and M. Hromada, "Anomaly detection system based on classifier fusion in ICS environment," in *Proceedings - 2017 International Conference on Soft Computing, Intelligent System and Information Technology: Building Intelligence Through IOT and Big Data, ICSIIT 2017*, 2017, pp. 32–38. doi: 10.1109/ICSIIT.2017.35.

[14]D. Upadhyay, J. Manero, M. Zaman, and S. Sampalli, "Gradient Boosting Feature Selection with Machine Learning Classifiers for Intrusion Detection on Power Grids," *IEEE Trans. Netw. Serv. Manag.*, vol. 18, no. 1, pp. 1104–1116, 2021.

[15]D. Upadhyay, J. Manero, M. Zaman, and S. Sampalli, "Intrusion Detection in SCADA Based Power Grids: Recursive Feature Elimination Model with Majority Vote Ensemble Algorithm," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 3, pp. 2559–2574, 2021.

[16]K. Bakour and H. M. Ünver, "VisDroid: Android malware classification based on local and global image features, bag of visual words and machine learning techniques," *Neural Comput. Appl.*, vol. 33, no. 8, pp. 3133–3153, Apr. 2021.

[17]H. Bai, N. Xie, X. Di, and Q. Ye, "FAMD: A fast multifeature android malware detection framework, design, and implementation," *IEEE Access*, vol. 8, pp. 194729–194740, 2020, doi: 10.1109/ACCESS.2020.3033026.

[18]W. Guo, Z. Luo, H. Chen, F. Hang, J. Zhang, and H. Al Bayatti, "AdaBoost Algorithm in Trustworthy Network for Anomaly Intrusion Detection," *Appl. Math. Nonlinear Sci.*, 2022, doi: 10.2478/amns.2022.2.0171.

[19]D. Alekseeva, N. Stepanov, A. Veprev, A. Sharapova, E. S. Lohan, and A. Ometov, "Comparison of Machine Learning Techniques Applied to Traffic Prediction of Real Wireless Network," *IEEE Access*, vol. 9, pp. 159495–159514, 2021.

[20]A. Gouveia and M. Correia, "Network Intrusion Detection with XGBoost," in *Recent Advances in Security, Privacy, and Trust for Internet of Things (IoT) and Cyber-Physical Systems (CPS)*, 2020, pp. 137–166. doi: 10.1201/9780429270567-6.

[21]D. J. Marchette, "Network Intrusion Detection," in *Handbook of Computational Statistics*, 2012, pp. 1139–1165. doi: 10.1007/978-3-642-21551-3_38.

[22]D. Bhavana, K. Kishore Kumar, V. Chilakala, H. G. Chithirala, and T. R. Meka, "A comparison of various machine learning algorithms in designing an intrusion detection system," *Int. J. Sci. Technol. Res.*, vol. 8, no. 12, pp. 2407–2413, 2019.