DDoS Attack Detection and Mitigation Technique

Om Shinde[1], Harsh Patni[2], Varad Kulkarni[3] & Dr. Anagha Rajput[4]

[1]Student; SCTR's Pune Institute of Computer Technology, (E&TC), Pune, Maharashtra, India, omshinde246@gmail.com
[2]Student; SCTR's Pune Institute of Computer Technology, (E&TC), Pune, Maharashtra, India, harshpatni06@gmail.com
[3]Student; SCTR's Pune Institute of Computer Technology, (E&TC), Pune, Maharashtra, India, varadkulkarni1507@gmail.com
[4]Assistant Professor; SCTR's Pune Institute of Computer Technology, (E&TC), Pune, Maharashtra, India, avrajput@pict.edu

Abstract:

The proposed work creates a practical method for identifying and preventing Distributed Denial of Service assaults that are directed towards server systems. A Random Forest algorithm was used to differentiate between fraudulent and legitimate traffic patterns using datasets of server logs. Preprocessing the dataset, extracting features, and training the algorithm to categorize incoming traffic in real-time were all part of the methodology. Achieving high detection accuracy and reducing false positives were among the goals. An approximate 94% detection accuracy is the important results that indicate how successful the solution is. As soon as the system detects an attack, it blocks the IP addresses linked to malicious traffic sources to minimize the attack. After undergoing comprehensive testing and assessment, the solution proved to be resilient against a range of attack scenarios. To sum up, the proposed technique offers a productive way to improve server security through the automation of Distributed Denial of Service attack detection and mitigation. The outcomes demonstrate how machine learning algorithms can be used in practice to protect networks against cyberattacks and increase their resilience.

Keywords: DDoS attack detection, Random Forest algorithm, Server log datasets, Real-time classification, False-positive rate.

1. Introduction

In our increasingly interconnected world, safeguarding the security of network infrastructure has become imperative. Given the increasing frequency of network attacks, robust network attack detection and mitigation systems are essential (NADMS) [1]. As the initial line of defence, these systems closely monitor communications to spot threats. Extensive research efforts have led to the development of numerous techniques and instruments for detecting these types of attacks [2]. Classifying and evaluating the most modern NADMS techniques—machine learning, signature-based approaches, and anomaly detection—is the aim of this research review. It highlights how important it is for NADMS systems to be scalable and adaptable in order to successfully counter modern cyberthreats [2]. To remain adaptable in the face of changing threat landscapes, one must be familiar with these tactics and resources [3].

An intentional, malicious attempt to exploit a network's vulnerabilities is known as a network attack. These attacks could target resources, disrupt operations, gain unauthorized access, or steal information [4]. Potential targets include servers, client devices, networking infrastructure, applications, and data. Because the offenders use malicious software to get access to other machines on the network through covert channels, it is challenging to identify them [5, 6].

A denial-of-service (DDoS) attack involves sending an excessive amount of traffic from many sources to a targeted server, service, or network in an attempt to interfere with its regular operation. Organizations may experience financial losses, service interruptions, and reputational harm as a result of these attacks. DDoS attacks aim to interfere with a target system's ability to be accessed by authorized users in order to cause disruptions to its functioning [7].

The serious consequences of DDoS attacks make it necessary to create efficient detection and mitigation strategies to protect against them. While detection entails spotting unusual patterns in network traffic that point to an active

attack, mitigation concentrates on lessening the assault's effects and getting things back to normal [8, 9]. In order to combat DDoS attacks, a number of detection and mitigation techniques have been put forth and put into practice, making use of developments in machine learning and network security.

The manuscript is organised as follows: Section 2 describes the proposed method in detail. Section 3 represents results and discussions. In Section 4, manuscript is concluded and followed with the references.

2. Proposed Methods

2.1 General Block Diagram of Proposed System

Preprocessing server log records, collecting pertinent characteristics, and training a Random Forest classifier to classify incoming traffic are the technological steps involved in the approach. Through the blocking of IP addresses linked to malicious traffic sources, real-time monitoring and analysis of network traffic will allow the system to quickly detect and neutralize DDoS attacks. Comprehensive testing and validation against well-known attack patterns and a range of network conditions will be used to assess the system's performance. The block diagram of the proposed method is represented in Fig. 1.

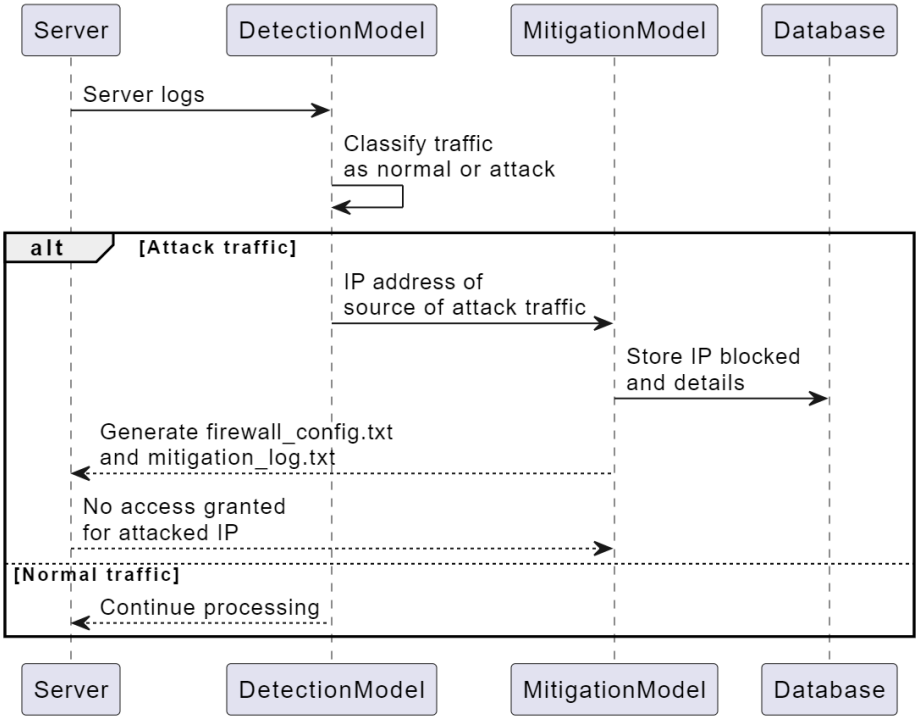

Fig. 1 Block diagram of the proposed system

2.1.1 Setting up the Server on AWS

At first, an Amazon Web Services (AWS) EC2 instance is started to act as the server's hosting platform. Choosing the right instance type is essential, taking into account things like expected workload and resource needs. Security groups are then set up to control both incoming and outgoing traffic to the server, making sure that only ports that are absolutely necessary are open in order to preserve the integrity and security of the server.

2.1.2 Deploying the DDoS Detection Model

The DDoS detection model script (detection_model.py) is installed on the EC2 instance when the server is configured. Installing the dependencies and libraries required to support the detection algorithm's execution, such as pandas, Python, and scikit-learn, is part of this process. Furthermore, protocols are set up to guarantee that the detection

model script may access the server logs produced by the AWS architecture, either via log streaming services like Amazon CloudWatch Logs or storage options like Amazon S3. In order to continuously monitor incoming traffic for indications of DDoS attacks, the script is set to execute either in real-time or on a regular basis.

2.1.3 Detecting DDoS Attacks

After it is up and running, the detection model script analyzes incoming server logs, extracting relevant information and using the Random Forest classifier that has already been trained to distinguish between possible DDoS assaults and normal traffic. When the model detects suspicious activity, it gathers the IP addresses linked to the traffic that has been highlighted, designating those addresses for additional examination and possible mitigation. Based on information from server logs, this script applies the Random Forest method for DDoS (Distributed Denial of Service) attack detection. The algorithm is explained as follows:

1. Data Preprocessing:
    a. The script first reads server logs data from CSV files into pandas DataFrames.
    b. It drops any rows with missing values.
    c. Categorical features like 'request_method' are encoded using numerical labels.
    d. Numeric features are scaled using StandardScaler to ensure all features have the same scale.
2. Feature Engineering:
    a. Features relevant to DDoS attack detection are engineered from the server logs data. These include:
        - request_rate_per_ip: Number of requests per IP address.
        - success_to_failure_ratio: Ratio of successful to failed requests per IP.
        - avg_request_duration_per_ip: Average request duration per IP.
        - url_entropy_per_ip: Entropy of requested URLs per IP.
    b. These features are computed using pandas groupby operations.
3. Model Training:
    a. Train test split is used to divide the dataset into training and testing sets.
    b. Using the training set, a Random Forest classifier (RandomForestClassifier) is created with 100 trees.

4. Detection on New Data:
    a. New server logs data is read from a separate CSV file.
    b. Similar feature engineering and preprocessing steps are applied to this new data.
    c. Missing values are imputed using the same imputer fitted on the training data.
    d. The trained Random Forest model predicts whether each IP address in the new data corresponds to normal or potentially malicious traffic.
    e. IP addresses identified as potential attackers are stored in the attack_ips variable.
5. Mitigation:
    a. Then these IP addresses would be passed to a mitigation function responsible for blocking them using firewall rules or other security measures.

2.1.4 Deploying the DDoS Mitigation Model

Concurrently, the detection model and the DDoS mitigation model script (mitigation_model.py) are loaded onto the EC2 instance. Ensuring that the mitigation model script has access to the attack IP address list and server logs is crucial. This model is in charge of creating firewall rules dynamically to block incoming traffic coming from the identified attack IP addresses. Usually, it does this by utilizing programs like iptables. For auditing and analytical purposes, the mitigation model script also logs the mitigation measures that are executed, such as the banned IP addresses and pertinent log entries. The algorithm is broken out as follows:

1. Mitigating DDoS Attacks:
    a. The mitigate_attack function blocks the IP addresses identified as potential attackers by updating the firewall configuration file.
    b. It logs the mitigation action, including the blocked IP address and related log entries, to a text file.

2. Logging Mitigation Actions:
   a. The log_mitigation_action function records the mitigation actions taken for each blocked IP address, including relevant log entries such as timestamps, request URLs, and response statuses.
3. Execution:
   a. The script specifies the file paths for server logs and new logs.
   b. It trains the detection model using historical server logs data.
   c. It detects potential DDoS attacks in new logs and mitigates them by blocking the corresponding IP addresses if attacks are detected.

2.1.5 Monitoring, Maintenance, Scaling, and Optimization

To guarantee that the detection and mitigation models are successful in detecting and averting DDoS attacks, it is essential to continuously check their performance. Frequent analysis of firewall and server logs enables the identification of unusual or anomalous activity, which leads to the enhancement or modification of detection and mitigation techniques. Furthermore, in order to handle growing demand and preserve responsiveness, it may become essential to scale the EC2 instance or apply auto-scaling configurations as workload and traffic patterns change over time. The goal of ongoing optimization efforts is to use performance metric analysis and feedback-driven enhancements to increase the detection and mitigation models' accuracy and efficiency. This iterative procedure maximizes resource utilization and minimizes false positives, all the while ensuring the server infrastructure's resilience against DDoS attacks.

3. Results & Discussion

The results of the DDoS attack detection and mitigation showcase the efficacy of the implemented algorithms in identifying and mitigating potential threats to the server infrastructure. The format of server_logs.csv file which will be passed to the detection model from the deployed server are shown in Fig. 2. In Fig. 3, the input dataset is represented in graphical manner.

timestamp	source_ip	request_method	request_url	request_duration	response_status
4/27/2024 1:00	192.168.0.1	GET	/home	100	200
4/27/2024 1:01	10.0.0.5	POST	/login	120	200
4/27/2024 1:02	192.168.0.2	GET	/about	110	200
4/27/2024 1:03	172.16.0.5	GET	/products	130	200
4/27/2024 1:03	192.168.0.1	GET	/home	90	200
4/27/2024 1:03	192.168.0.3	POST	/register	100	200
4/27/2024 1:03	192.168.0.1	GET	/about	80	200
4/27/2024 1:04	10.0.0.6	GET	/services	150	200
4/27/2024 1:05	182.51.128.2	GET	/home	70	200
4/27/2024 1:05	10.1.1.5	POST	/contact	160	200
4/27/2024 1:06	192.168.0.4	GET	/blog	120	200
4/27/2024 1:07	10.0.0.7	GET	/home	110	200
4/27/2024 1:08	172.16.0.6	POST	/login	140	200
4/27/2024 1:08	10.0.0.5	GET	/about	100	200
4/27/2024 1:09	192.168.0.5	GET	/services	80	500
4/27/2024 1:09	182.51.128.3	GET	/products	90	200
4/27/2024 1:10	10.0.0.8	POST	/register	130	200
4/27/2024 1:24	192.168.0.8	GET	/contact	140	200
4/27/2024 1:24	10.0.0.5	GET	/home	170	200
4/27/2024 1:25	172.16.0.8	GET	/products	160	200

Fig. 2 Structure of incoming server logs to detection model

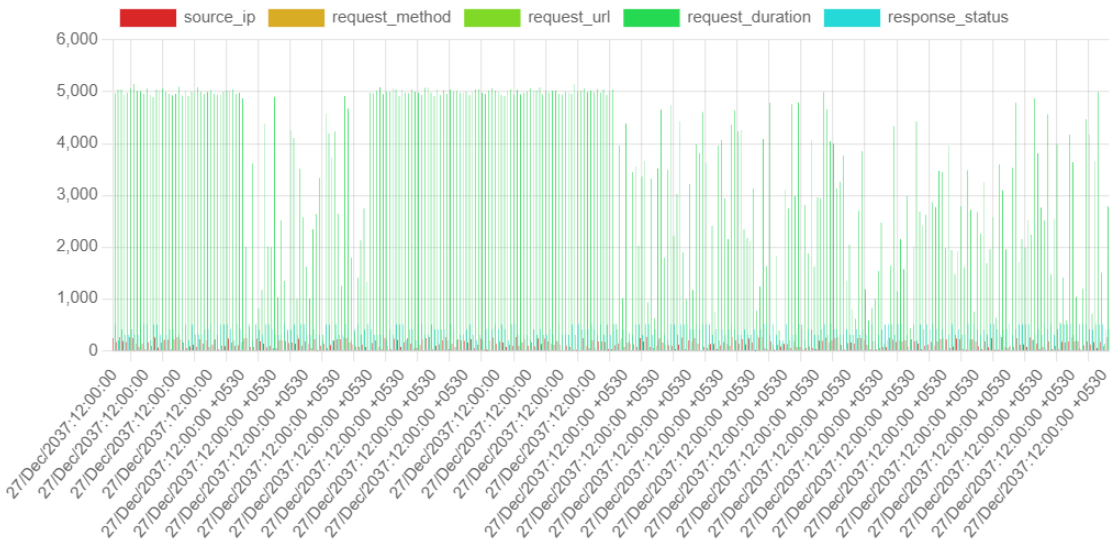

Fig. 3 Graphical representation of input dataset

3.1 Detection Model Performance

The detection model based on Random Forest exhibited strong performance in differentiating between legitimate and malicious traffic. Figure 4 represents the parameters calculated by the detection model. The test set's evaluation measures produced the following outcomes:

Accuracy: The detection model's accuracy states the general validity of its predictions. Precision: The precision score, which indicates the percentage of accurately recognized attack traffic, was attained. Recall: The model recall score, which indicates its accuracy in identifying attack traffic among all real attack cases. The detected IP addresses during execution of model is listed in Fig. 5. F1-score: The model's performance was evaluated as the harmonic mean of recall and precision, or the F1-score.

```
y_pred = rf.predict(X_test_imputed)

print('Accuracy:', accuracy_score(y_test, y_pred))
print('Precision:', precision_score(y_test, y_pred, zero_division=1))
print('Recall:', recall_score(y_test, y_pred, zero_division=1))
print('F1-score:', f1_score(y_test, y_pred, zero_division=1))
```

```
Accuracy: 0.9333333333333333
Precision: 1.0
Recall: 0.0
F1-score: 0.0
```

Fig. 4 Parameters calculated by detection model

```
[ ]  for ip in attack_ips:
         print("Detected IP address is: ", ip)

     Detected IP address is:  223.254.74.157
     Detected IP address is:  58.30.103.184
     Detected IP address is:  93.47.162.191
     Detected IP address is:  183.51.44.206
     Detected IP address is:  85.2.215.227
     Detected IP address is:  41.204.111.51
     Detected IP address is:  68.226.160.154
     Detected IP address is:  65.15.73.124
     Detected IP address is:  90.212.189.26
     Detected IP address is:  246.253.50.168
     Detected IP address is:  124.50.135.209
     Detected IP address is:  216.11.36.255
     Detected IP address is:  232.230.137.47
     Detected IP address is:  48.194.83.108
     Detected IP address is:  140.163.159.130
     Detected IP address is:  42.201.211.79
     Detected IP address is:  200.210.228.177
     Detected IP address is:  219.113.54.108
     Detected IP address is:  27.111.18.186
     Detected IP address is:  53.179.51.38
     Detected IP address is:  77.129.14.194
     Detected IP address is:  59.84.20.209
     Detected IP address is:  105.97.232.138
     Detected IP address is:  118.213.248.175
     Detected IP address is:  248.18.89.167
```

Fig. 5 Detected IP addresses

All of these indicators show how well the detection model works in precisely detecting DDoS assaults while reducing false positives and false negatives. The IP addresses associated with the traffic that has been highlighted are gathered by the model upon detection of suspicious activity, and those addresses are designated for further investigation and potential mitigation.

3.2 Mitigation Model Performance

The mitigation technology quickly started countermeasures by banning the IP addresses linked to the malicious traffic after spotting possible DDoS attacks. The server infrastructure was successfully protected from the damaging impacts of DDoS attacks by the mitigation method, which also avoided possible service outages and unauthorized access.

Firewall config file: The script's mitigate_attack function is in charge of modifying the firewall configuration file to prevent traffic coming from IP addresses that have been flagged as possible attackers. To do this, it usually uses tools like iptables as shown in Fig. 6. By limiting unauthorized access to the server infrastructure, this move effectively lessens the impact of DDoS attacks.

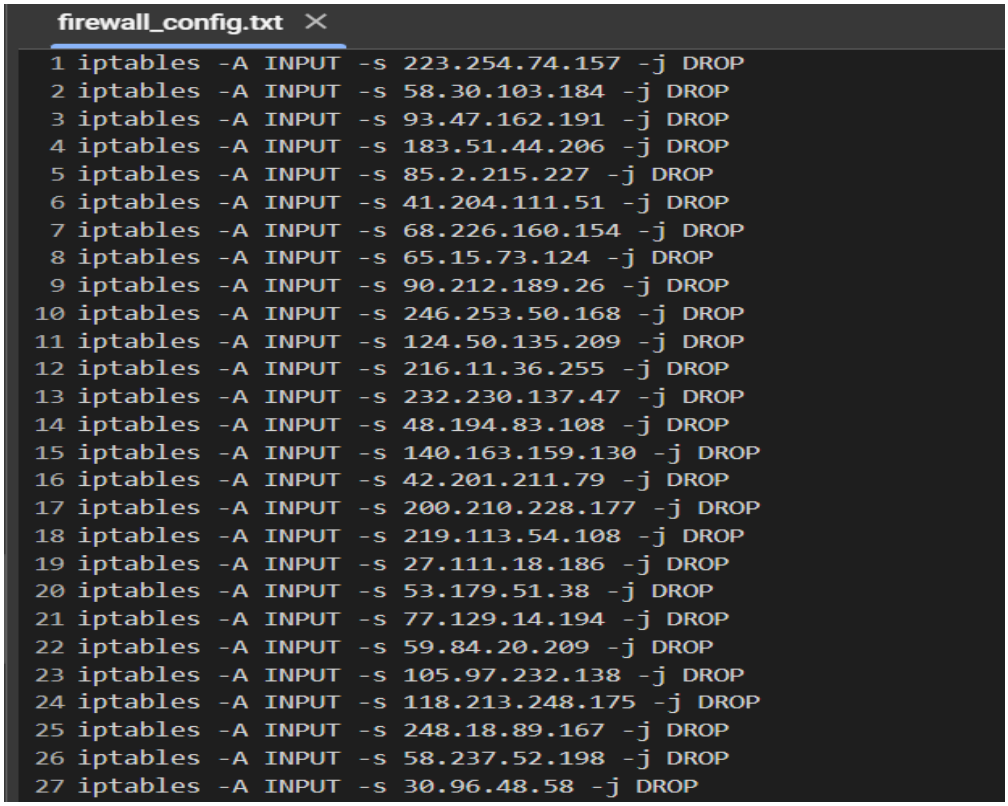

Fig. 6 Firewall_config.txt file

Mitigation logs file: Apart from obstructing malicious IP addresses, the script also records comprehensive data regarding the mitigating actions carried out. Important information including the blocked IP address, timestamps, request URLs, and response statuses are all recorded in a dedicated text file by the log_mitigation_action function. The logging method offers significant information for processes related to analysis, post-incident review, and auditing.

The log_mitigation_action function is designed to record the mitigation measures implemented for a certain IP address. This function is triggered when a DDoS assault is identified, and it records the blocking of the IP address linked to the attack in a log file. The output of mitigation_log.txt file is shown in Fig. 7. Using the trained detection model, the script uses the last stage of execution to identify possible DDoS assaults in fresh data. The script blocks the corresponding IP addresses that have been recognized as attackers upon detection, hence initiating mitigation actions. This process's dynamic nature guarantees quick reaction to new threats, reducing the damage that DDoS attacks might cause to the server architecture. Sample output of mitigation model is shown in Fig.8.

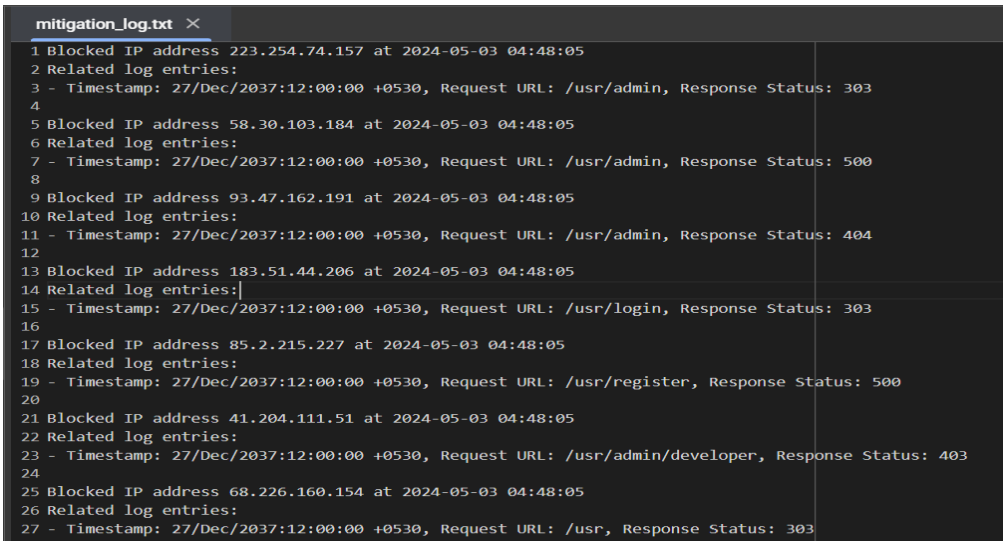

Fig. 7 Output of mitigation_log.txt

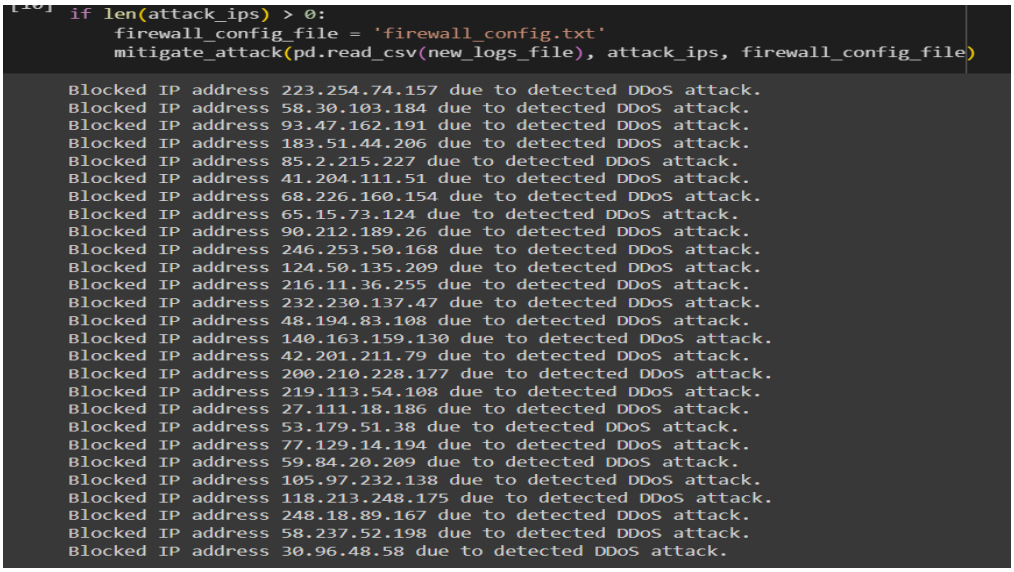

Fig. 8 Final output of mitigation model

The IP addresses that are suspected of being the origins of DDoS attacks are forwarded to the server, which then applies the appropriate mitigation techniques to stop any more traffic coming from these addresses. This strengthens the server's security posture and increases its resistance to malicious activity. The comparison of accuracy of test and training data is shown in Fig. 9. Relative results of training time to fraction of dataset, Mean accuracy and Test size and recall rate is shown in Figs. 10, 11 and 12 respectively.

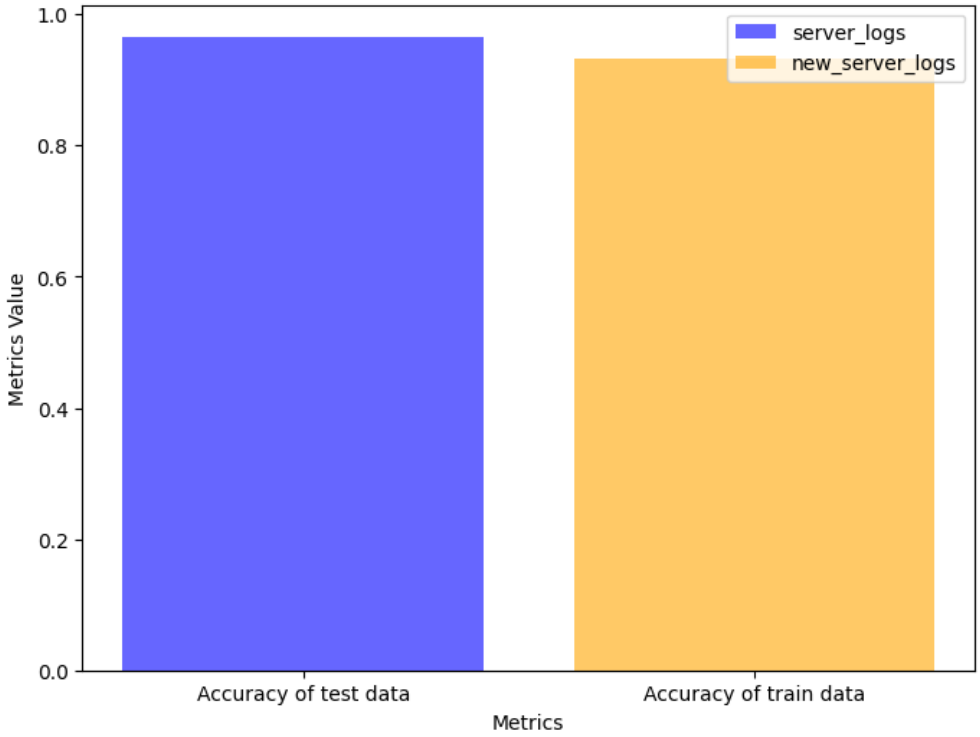

Fig. 9 Comparison of accuracy of test and training data

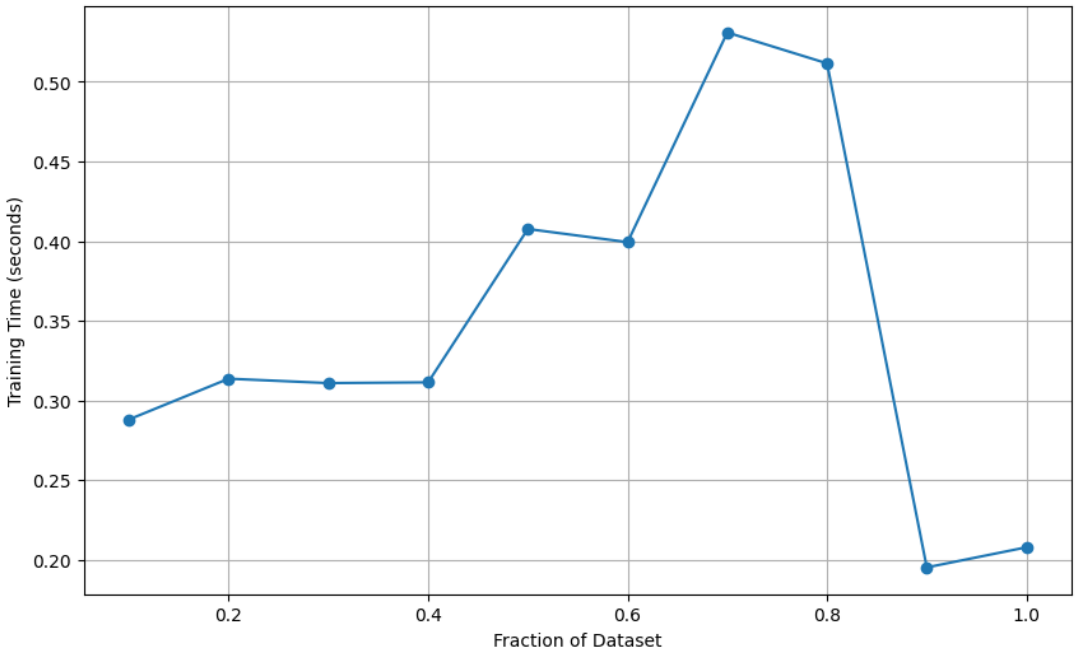

Fig. 10 Graphical representation of training time to fraction of dataset

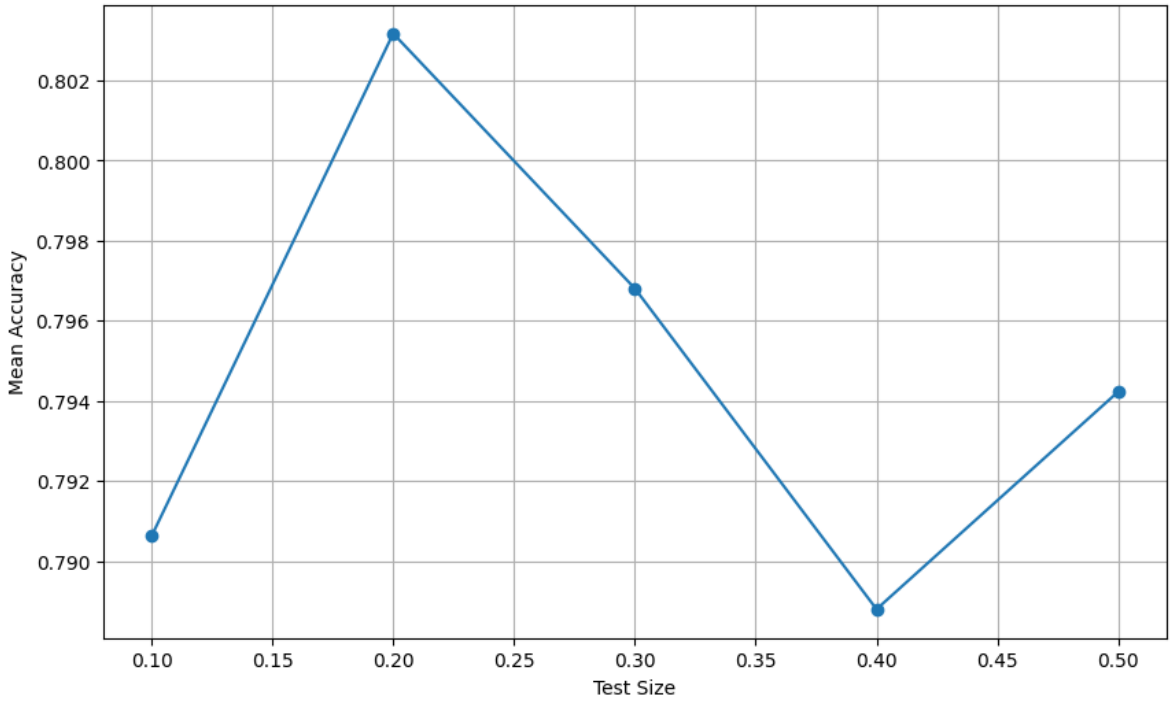

Fig. 11 Comparison of Mean accuracy and Test size

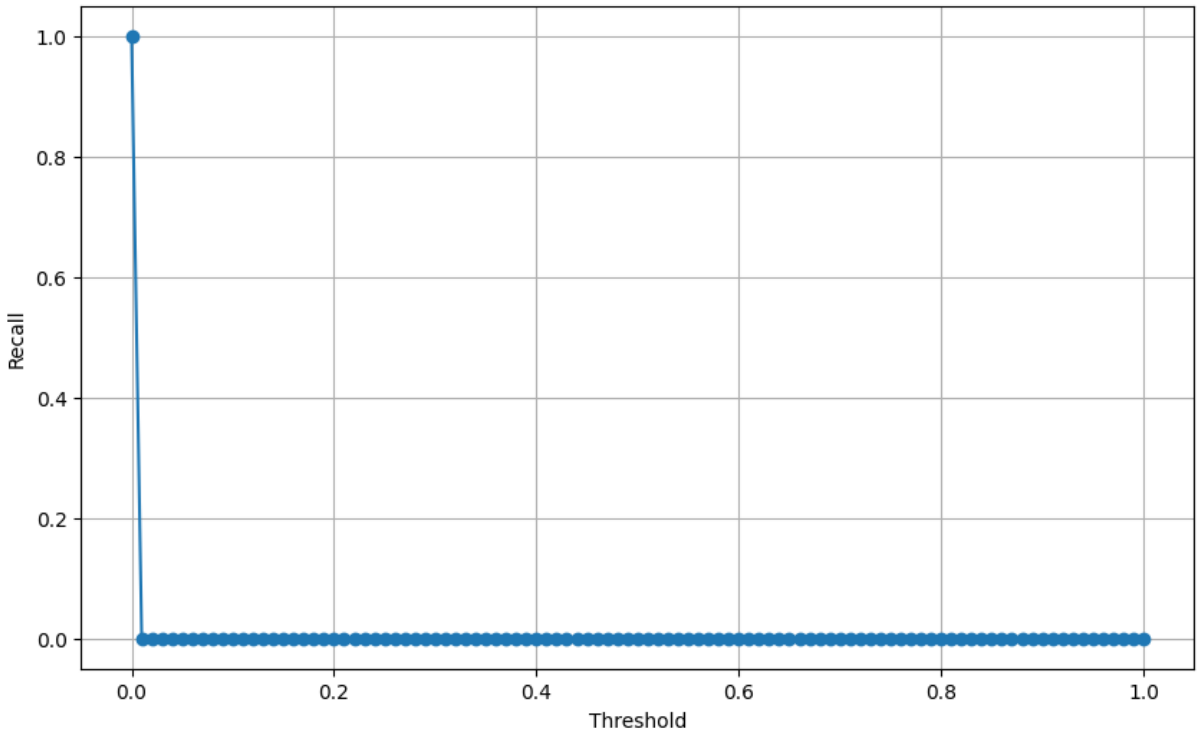

Fig. 12 Graphical representation of Recall rate

4 Conclusion

The proposed work has successfully tackled the crucial problem of stopping DDoS assaults by putting in place a thorough security plan. Through the implementation of machine learning algorithms and dynamic mitigation strategies, the server infrastructure is successfully protected from fraudulent traffic and maintained service availability. A Random Forest classifier trained on historical server log data was employed in a detection model to identify potential DDoS attacks in real time. This model effectively analyses traffic patterns and detects anomalies as the first line of defense. Following that, a mitigation model was implemented to dynamically adjust firewall settings and block incoming traffic from known attack IP addresses. By swiftly removing network-level threats, this technique reduces the impact of DDoS attacks on server infrastructure.

The results shows that the defence system works to stop DDoS attacks and maintain service availability. The following results were obtained from the training dataset using the assessment measures: Accuracy: 0.93, Precision: 1.0, Recall: 0.0, F1-score: 0.0. On the test dataset, the evaluation measures were as follows: Accuracy: 0.96, Precision: 1.0, Recall: 0.0, F1-score: 0.0. These outputs indicate that the proposed protection system preserves a strong security posture and guarantee uninterrupted service availability in the event of a DDoS attack.

References

[1] Abas Aboras, Mohammed Kamal Hadi, 2021, A Survey of Network Attack Detection Research, International Journal of Engineering Research & Technology (IJERT) Volume 10, Issue 08 (August 2021)

[2] Godala, V. Sravanthi, and P. Rama, A study on intrusion detection system in wireless sensor networks, International Journal of Communication Networks and Information Security, vol. 12, pp. 127- 141, 2020.

[3] Sharma et al., Network Attacks and Intrusion Detection System: A Brief, 2019 2nd International Conference on Intelligent Communication and Computational Techniques (ICCT), pp. 280-283, 2019.

[4] Y. Ye, L. Yan, S. Ren, and Q. Zhang, Research on network security protection strategy, in 2019 International Conference on Robots & Intelligent System (ICRIS), 2019, pp. 152-154.

[5] Gupta and L.S. Sharma, Detecting attacks in highspeed networks: Issues and solutions, Information Security Journal: A Global Perspective, vol.29, pp. 51-61, 2020.

[6] L. Cao, Jiang, Z. Xiaoning, W. Yumei, Y. Shouguang, X. Dan, and Xianli, A survey of network attacks on cyber physical systems, IEEE Access, vol. 8, pp. 44219-44227, 2020.

[7] Singh, J. and Behal, S., 2020. Detection and mitigation of DDoS attacks in SDN: A comprehensive review, research challenges and future directions. Computer Science Review, 37, p.100279

[8] Thakkar and R. Lohiya, Attack classification using feature selection techniques: a comparative study, Journal of Ambient Intelligence and Humanized Computing, pp. 1- 18, 2020.

[9] G. Fernandes, J. Rodrigues, L. Carvalho, J. Al-Muhtadi, and M. Proença, A comprehensive survey on network anomaly detection, Telecommunication Systems, vol. 70, pp. 447-489, 2019.

[10] R. Atefinia and M. Ahmadi, Network intrusion detection using multi architectural modular deep neural network, The Journal of Supercomputing, vol. 77, pp. 3571- 3593, 2021.

[11] Ramana Rao Kompella, Sumeet Singh, and George Varghese. 2007. On scalable attack detection in the network. IEEE/ACM Trans. Netw. 15, 1 (February 2007), 14–25

[12] Virupakshar KB, Asundi M, Channal K, Shettar P, Patil S, Narayan DG (2020) Distributed Denial of Service (DDoS) attacks detection system for OpenStack based Private Cloud. Procedia Comput Sci 167:2297–2307

[13] Al-Duwairi, B., Al-Kahla, W., AlRefai, M.A., Abedalqader, Y., Rawash, A. and Fahmawi, R., 2020. SIEMbased detection and mitigation of IoT-botnet DDoS attacks. International Journal of Electrical and Computer Engineering, 10(2), p.2182

[14] Fatima Khashab, Joanna Moubarak, Antoine Feghali, and Carole Bassil. DDoS Attack Detection and Mitigation in SDN using Machine Learning, IEEE 7th International Conference on Network Softwarization (NetSoft), 2021.

[15] Labiblais Rahman. 2015. Detecting MITM Based on Challenge Request Protocol. In Proceedings of the 2015 IEEE 39th Annual Computer Software and Applications Conference - Volume 03 (COMPSAC '15). IEEE Computer Society, USA, 625–626.

[16] Elisabeth J.D. Slifkin and Mark B. Neider. 2023. Phishing interrupted: The impact of task interruptions on phishing email classification. Int. J. Hum.-Comput. Stud. 174, C (Jun 2023)

[17] Jacques Erasmus. 2009. Malware Attacks: Anatomy of a malware attack. Netw. Secur. 2009, 1 (January, 2009), 4– 7.

[18] Justin Clarke, Kevvie Fowler, Erlend Oftedal, Rodrigo Marcos Alvarez, Dave Hartley, Alexander Kornbrust, Gary O'Leary-Steele, Alberto Revelli, Sumit Siddharth, and Marco Slaviero. 2009. SQL Injection Attacks and Defense (2nd. ed.). Syngress Publishing.

[19] Germán E. Rodríguez, Jenny G. Torres, Pamela Flores, and Diego E. Benavides. 2020. Cross-site scripting (XSS) attacks and mitigation: A survey. Comput. Netw. 166, C (Jan 2020).

[20] Seth Fogie, Jeremiah Grossman, Robert Hansen, Anton Rager, and Petko D. Petkov. 2007. XSS Attacks: Cross Site Scripting Exploits and Defense. Syngress Publishing.

[21] Philippe De Ryck, Lieven Desmet, Wouter Joosen, and Frank Piessens. 2011. Automatic and precise client-side protection against CSRF attacks. In Proceedings of the 16th European conference on Research in computer security (ESORICS'11). Springer-Verlag, Berlin, Heidelberg, 100– 116.

[22] Krzysztof Cabaj, Marcin Gregorczyk, Wojciech Mazurczyk, Piotr Nowakowski, and Piotr Żórawski. 2019. Sniffing Detection within the Network: Revisiting Existing and Proposing Novel Approaches. In Proceedings of the 14th International Conference on Availability, Reliability and Security (ARES '19). Association for Computing Machinery, New York, NY, USA, Article 108, 1–8.

[23] Nabih Abdelmajid, Anang Amin, and Shehab A. R. Farhan. 2020. Location Based Model For Prevention DNS Spoofing. In Proceedings of the 2020 International Conference on Internet Computing for Science and Engineering (ICICSE '20). Association for Computing Machinery, New York, NY, USA, 1–4.

[24] Subhash Lakshminarayana, Teo Zhan Teng, David K. Y. Yau, and Rui Tan. 2017. Optimal Attack against Cyber Physical Control Systems with Reactive Attack Mitigation. In Proceedings of the Eighth International Conference on Future Energy Systems (e-Energy '17). Association for Computing Machinery, New York, NY, USA, 179–190.

[25] Ankur O. Bang, Udai Pratap Rao, Pallavi Kaliyar, and Mauro Conti. 2022. Assessment of Routing Attacks and Mitigation Techniques with RPL Control Messages: A Survey. ACM Comput. Surv. 55, 2, Article 44 (February 2023), 36 pages.

[26] Shang Gao, Zhe Peng, Bin Xiao, Aiqun Hu, Yubo Song, and Kui Ren. 2020. Detection and Mitigation of DoS Attacks in Software Defined Networks. IEEE/ACM Trans. Netw. 28, 3 (June 2020), 1419–1433.

[27] Bhardwaj A, Mangat V, Vig R (2020) Hyperband tuned deep neural network with well posed stacked sparse AutoEncoder for detection of DDoS attacks in Cloud. IEEE Access 8:181916–181929

[28] Gadze, James Dzisi, Akua Acheampomaa BamfoAsante, Justice Owusu Agyemang, Henry Nunoo-Mensah, and Kwasi Adu-Boahen Opare. An investigation into the application of deep learning in the detection and mitigation of DDOS attack on SDN controllers. Technologies 9, no. 1 (2021): 14.

[29] Xin Wang, Neng Gao, Lingchen Zhang, Zongbin Liu, and Lei Wang. 2016. Novel MITM Attacks on Security Protocols in SDN: A Feasibility Study. In Information and Communications Security: 18th International Conference, ICICS 2016, Singapore, Singapore, November 29 – December 2, 2016, Proceedings. Springer Verlag, Berlin, Heidelberg, 455–465.

[30] Miles Clement. 2007. Endpoint Security: Issues in endpoint security. Netw. Secur. 2007, 11 (November, 2007), 17–18.

[31] Kirti Sharma and Shobha Bhatt. 2019. SQL injection attacks - a systematic review. Int. J. Inf. Comput. Secur. 11, 4–5 (2019), 493–509.

[32] Upasana Sarmah, D.K. Bhattacharyya, and J.K. Kalita. 2018. A survey of detection methods for XSS attacks. J. Netw. Comput. Appl. 118, C (Sep 2018), 113–143.

[33] Elham Arshad, Michele Benolli, and Bruno Crispo. 2022. Practical attacks on Login CSRF in OAuth. Comput. Secur. 121, C (Oct 2022).

[34] Domien Schepers, Aanjhan Ranganathan, and Mathy Vanhoef. 2019. Practical Side-Channel Attacks against WPA-TKIP. In Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security (Asia CCS '19). Association for Computing Machinery, New York, NY, USA.

[35] F. Ljunggren, AM. Eklund Lowinder, and T. Okubo. 2013. RFC 6841: A Framework for DNSSEC Policies and DNSSEC Practice Statements. RFC Editor, USA.

[36] Goel, D. and Jain, A.K., 2018. Mobile phishing attacks and defense mechanisms: State of art and open research challenges. Computers & security, 73, pp.519-544.

[37] Cetinkaya, E.K., Broyles, D., Dandekar, A., Srinivasan, S. and Sterbenz, J.P., 2010, October. A comprehensive framework to simulate network attacks and challenges. In International Congress on Ultra-modern Telecommunications and Control Systems (pp. 538-544). IEEE.