International Journal of Safety and Security Engineering
Vol. 15, No. 12, December, 2025, pp. 2649-2657
Journal homepage: http://iieta.org/journals/ijsse

Lightweight Threshold-Based Real-Time Distributed Denial of Service Attack Detection with WebSocket Alerts

Saleha Saudagar[1], Gayatri Jagnade[2], Mayura Shelke[1*], Sayali Belhe[3], Supriya Gorde[4], Amol Bhosle[1]

[1] School of Computing, MIT Art Design and Technology University, Pune 412201, India
[2] Department of Artificial Intelligence, Sardar Vallabhbhai National Institute of Technology, Surat 395007, India
[3] Department of Artificial Intelligence and Data Science, Vishwakarma Institute of Technology, Pune 411037, India
[4] Department of Computer Science and Engineering, Vishwakarma Institute of Technology, Pune 411037, India

Corresponding Author Email: mayura.shelke@gmail.com

https://doi.org/10.18280/ijsse.151220

Received: 9 October 2025
Revised: 9 December 2025
Accepted: 20 December 2025
Available online: 31 December 2025

Keywords:
cybersecurity, Distributed Denial of Service detection, Node.js, real-time monitoring, WebSocket

ABSTRACT

As digital platforms increasingly dominate day-to-day activities, ensuring robust and reliable security mechanisms has become a critical necessity. Among various cyber threats, Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks are widely studied, as they directly threaten service availability by overwhelming systems with excessive requests, making services inaccessible to legitimate users. Conventional DDoS detection techniques are often unsuitable for lightweight deployments because they rely heavily on expensive hardware or complex machine learning (ML) models. This work proposes a lightweight and scalable DDoS anomaly detection framework capable of identifying real-time anomalies from server log data. The system visualizes abnormal traffic patterns using WebSocket communication with a Node.js server. The web application, hosted on Google Cloud Storage (GCS), includes a real-time monitoring dashboard that is updated regularly. Experimental findings confirm the effectiveness of the proposed system in identifying high request rates with low latency, achieving 0.002 s latency and approximately 97.0% alert accuracy. The proposed solution is particularly suitable for small- to medium-sized online platforms, as it provides a cost-effective, scalable, and efficient real-time DDoS detection approach without requiring complex and intensive resource infrastructure.

1. INTRODUCTION

The world is digitalized, and the majority of everyday activities have been digitized, and as such, cyberspace has gained more significance to address the rising security demands. The users are seeking quicker and more dependable services and are ready to pay a premium. The digital age has made people more reliant on interconnected systems, and in this case, availability and integrity are essential to communication. With the intensive development of digital platforms, attackers have become more advanced. Regrettably, interconnectivity has resulted in more interconnectedness and consequently more cyber-attacks, which are posing dire challenges to any type of business [1].

The Distributed Denial of Service (DDoS) attack is one of the most disruptive types of attacks on the availability of services. These attacks intentionally drain system resources by flooding them with numerous requests from many different sources to the extent that the services are not available to the legitimate users [2]. This congestion denies the target capacity to serve valid requests and denies the target users service.

The websites that are dependent on the internet infrastructure, like the government and e-commerce websites, are the most vulnerable. With a large number of online platforms, DDoS attacks may affect nearly any organization that relies on the internet. Attackers usually create traffic flooding with the help of botnets, networks of infected devices run remotely by attackers [3]. The devices are usually spread within a variety of geographic locations, and it is hard to tell whether malicious traffic is being conducted or if the traffic is due to an actual user request; thus, it is hard to identify and stop it.

Newer versions of DDoS attacks have been developed in order to circumvent the older mechanisms of defense against them, especially volumetric and application-layer attacks. Volumetric attacks bombard the target with large volumes of traffic, and application-layer attacks take advantage of a given vulnerability in the application [4]. Such professional and adaptive strategies of attack necessitate constant observation and the timely inhibition. Although the methods of Artificial Intelligence (AI) and machine learning (ML) have demonstrated their usefulness in DDoS detection (via traffic analysis), they may be associated with an increased computational load [5-7]. Thus, the effective real-time monitoring strategies that could be used to identify anomalies with lower complexity are still needed.

The Proposed Contributions of work are as follows:

- Propose a lightweight DDoS anomaly detection framework for real-time protection of digital services.

- Overcome the limitations of traditional defenses against volumetric and application-layer attacks.
- Introduce a threshold-based traffic analysis approach that avoids complex AI/ML models and reduces computational overhead.
- Enable real-time monitoring and alert generation using limited system resources.
- Achieve scalable and efficient detection by focusing on critical traffic parameters.

2. REVIEW OF EXISTING DISTRIBUTED DENIAL OF SERVICE ANOMALY DETECTION STRATEGIES

DDoS attack detection is a crucial phenomenon as it maintains the integrity and availability of the system, especially when dealing with a sophisticated attacker. The significance of real-time DDoS cannot be underestimated. There are many researchers who develop strategies to identify and detect DoS anomalies.

2.1 Real-time Distributed Denial of Service attack detection

A real-time DDoS detection model was developed using ML-based classification algorithms, including Random Forest, XGBoost, and Support Vector Machine [8]. The CICDDoS2019 dataset was used to evaluate several classifiers and determine which are best suited for real-time deployment. The results emphasize that AI/ML-based methods are accurate, scalable, and effective for anomaly detection. However, implementing such systems in real-time scenarios makes the overall system more complex. A lightweight real-time DDoS detection system for Internet of Things (IoT) networks was presented in the study by Reed et al. [9]. It examines packet length and delta time; the system detects slow DDoS attacks with 98% accuracy. Further, the system is designed to reduce computational overhead and improve detection accuracy. An AI-based DDoS detection model enhances accuracy by combining stacked sparse denoising auto encoders with a firefly–black widow hybrid optimization algorithm [10]. The model is trained using the CICDDoS2019 dataset and performs better than existing techniques in terms of real-time detection speed and classification efficiency.

2.2 Time alert systems for Distributed Denial of Service anomaly detection

WebSocket alert mechanisms: Integrating real-time alert systems with monitoring techniques enhances responsiveness. The use of sliding time windows and single-directional filters enables immediate detection and response during an attack [11]. Wanjale et al. [12] examined the difficulties of utilizing WebSocket for real-time communication as well as the reasons why some applications favor long polling. The author tried to provide information on WebSocket vulnerabilities and optimization techniques and gave a substitute communication protocol that can improve DDoS defenses.

Table 1. Comparative literature review of DDoS detection techniques

Ref.	Technique Used	Key Features / Approach	Limitations	Remarks / Research Gap
[13]	DDoS defence mechanism is used with high level taxonomy	Classified prevention, detection, and response strategies for DDoS attacks	High-level taxonomy; lacks practical implementation for real-time systems	Identified the need for low-latency and adaptive detection frameworks
[14]	Traffic anomaly detection using flow-based metrics	Measured packet rates, byte count, and flow duration for anomaly detection	Computationally intensive feature extraction	Research work includes anomaly detection and lacks a lightweight design, which is more suitable for real-time deployment
[15]	Deep learning (CNN / Reinforcement learning) models for threat detection	Detected complex patterns in (real time IOT setup) network flows with high accuracy	The System relies on predefined models, which may require periodic updating to remain effective in a dynamic environment	Systems remain limited by high computational overhead and poor scalability when deployed in resource-constrained environments
[16]	Framework to mitigate DDoS vulnerabilities	Cloud-based system for network traffic monitoring	Its accuracy depends on the exact traffic and real-time response capability	The system provides a cloud-based framework for DDoS vulnerability prevention and detection, but limits when it comes to real-time monitoring
[17]	Statistical threshold-based DDoS detection method	Traffic threshold statistical analysis through mean and variance	Root cause analysis is not included	The paper focused on a statistical threshold DDoS detection method but lack in multi layered architecture, which limits its detection without root cause
[18]	Real-time WebSocket-based real-time monitoring	Persistent WebSocket-communication between client and server	The Implemented method face issue of scalability	It motivates the use of WebSocket at real-time based DDoS but lacks in scalability and can be extended with AI driven method.

Note: DDoS = Distributed Denial of Service; CNN = Convolutional Neural Network; IoT = Internet of Things; AI = Artificial Intelligence.

Real-time traffic visualization methods for early DDoS attack detection are explored through this platform [19]. Network operators and traffic patterns can be analyzed to identify anomalies. The mentioned method has demonstrated its improvements in incident response and in minimizing the intensity of anomalies. Real-time monitoring systems using Node.js and WebSocket technologies have already been proven to be effective in detecting DDoS anomalies [20]. Fast DDoS detection and mitigation is suggested, which is grounded on the observation of abnormal frequencies of web server visits. The system blocks suspicious traffic within seconds by noticing abnormal request rates in real-time and limiting the disruption of services and speed of response [21]. The practical-based strategies like rate limiting, event throttling, and connection control can safeguard WebSocket-based applications and hence are potential DDoS threat detection [22]. Additionally, an emerging security method for securing the cloud is mentioned, the result here demonstrates how hybridization improves time and efficiency [23]. Table 1 shows the comparative literature review analysis of DDoS attack detection techniques.

The development of lightweight DDoS attack detection methods for real-time monitoring systems is crucial nowadays to achieve network security. The literature review highlighted that existing DDoS detection systems have complex models like AI and ML or other technologies, and are detecting well, but demand high computing resources and are not suitable for small-scale infrastructure. On the other hand, lightweight DDoS anomaly detection methods are static and lack adaptability. The proposed DDoS anomaly detection method is light weight threshold based and real-time monitoring system that can bridge the gap and can enhance anomaly detection at small scale infrastructure.

3. PROPOSED DISTRIBUTED DENIAL OF SERVICE ANOMALY DETECTION FRAMEWORK

The proposed lightweight DDoS anomaly detection framework, along with its development algorithms and performance evaluation factors, is discussed in this section. A detailed background is provided to support the design and implementation of a real-time DDoS detection system. This section further describes the processes involved in traffic monitoring, anomaly identification, and real-time alert generation.

3.1 Threshold-based Distributed Denial of Service detection algorithm

Algorithm: Threshold-Based Distributed Denial of Service (DDoS) Detection

```
Input:
Window Size: T
Threshold: θ (Requests Per Second (RPS))
Incoming Request Stream: R
In the experimental evaluation, the window size T was fixed to 1 second, and the threshold θ was set to 5 RPS per source IP.

Output:
Alert for a potential DDoS attack

Algorithm Steps:
1. Initialize request_counter ← empty dictionary
2. For each incoming request r in R:
    a. Extract source_IP ← r.source_ip
    b. Extract timestamp ← r.timestamp
    c. Request_counter[source_IP] ++ for time window T
3. For loop -source_IP of request_counter:
    a. If request_counter[source_IP] > θ then:
        i. Trigger alert for source_IP
        ii. Perform response actions:
            - Option 1: Block source_IP
            - Option 2: Rate-limit source_IP
            - Option 3: Log source_IP activity
4. Reset request_counter after T seconds
5. Repeat steps 2–4
```

where, θ is the maximum number of allowable Requests Per Second (RPS) from an individual source Internet Protocol (IP) address within a fixed one-second monitoring window.

Fixed-Window Counter Approach (request_counter): This approach maintains a hash map that records the number of requests received from each source IP within a fixed time window T. Each incoming request increments the corresponding counter, and all counters are reset at the end of the window. While this method is simple and computationally efficient, it may suffer from boundary effects, where request bursts near window transitions can lead to inaccurate detection.

3.2 System overview: Architecture and components

The proposed DDoS Detection system has three primary components, which include DDoS attack detection, and the system's essential features are handled through Node.js backend. It takes in and records network traffic, responsible for determining the RPS of each IP address. In addition to this, it implements the technique for threshold-based anomaly detection and controls WebSocket connections to communicate with the monitoring dashboard in real-time.

The assessment of simulated attacks and legal traffic can be done on this platform. It replicates actual user traffic; a website hosted on Google Cloud Storage (GCS) is utilized. Real-time monitoring dashboard tool offers a spontaneous interface that alerts and visualizes the traffic. Chart.js, a JavaScript library used to make interactive charts and graphs, is used in its construction. With the help of WebSocket, the dashboard gets real-time updates from the Node.js backend. The Node.js backend calculates the per-IP request rate (RPS) in real-time, checks it against a predetermined per-IP threshold, and holds aggregated traffic statistics to be used in visualization. Resulting in a dynamic picture of the traffic patterns as they exist right now.

Real-time communication between the monitoring dashboard and the Node.js backend is enabled through the use of WebSocket, as shown in Figure 1. When the monitoring dashboard surpasses the predetermined RPS threshold, which is set to 5 in this case, the Node.js backend notifies the user of a possible DDoS attack by sending an alert over WebSocket.

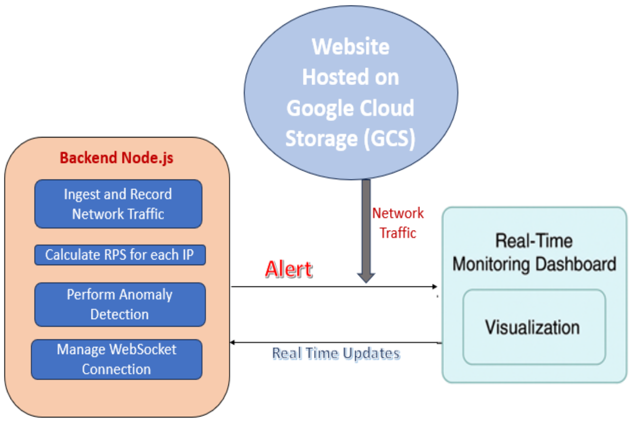
Website Hosted on Google Cloud Storage (GCS)
Backend Node.js
Ingest and Record Network Traffic
Calculate RPS for each IP
Perform Anomaly Detection
Manage WebSocket Connection
Network Traffic
Alert
Real Time Updates
Real-Time Monitoring Dashboard
Visualization

Figure 1. Real-time monitoring and Distributed Denial of Service (DDoS) attack detection system architecture

3.3 Experimental setup: Simulation and testing

To assess the efficiency and performance of the proposed DDoS detection, a controlled experimental setup was developed. This configuration enabled precise testing in both real-world traffic situations and simulated DDoS attack scenarios.

Traffic Generator: To mimic both malicious DDoS attack traffic and legitimate user traffic, a traffic generator was employed. As a result, testing scenarios might be regulated and replicated. Target Web Server: The DDoS detection algorithm was installed on a target web server. This server mimicked the protected system. Simulating Traffic: Legitimate Traffic: The common Hypertext Transfer Protocol (HTTP) request tool is used here in order to mimic a realistic online traffic pattern. DDoS attack Traffic: To create a simulated DDoS attack, a large number of requests from several simulated sources were sent in at once to the target server. The capacity of the system is assessed by creating multiple attack scenarios. The identification of different kinds of DDoS attacks can be assisted by these attack scenarios. Features that monitor traffic, logging, and alerting have been set up on the target server. The monitoring of important metrics like request rate, originating IP addresses, and system response time includes all inbound requests. The experimental setup carefully monitors the system's ability to detect real-time DDoS attacks. It aids in minimizing the detection of false positive attacks in typical traffic conditions.

Algorithm for DDoS Detection: Detailed Steps

The algorithm is based on a threshold-based approach and monitors RPS. The algorithm steps are as follows:

- **Log Requests:** The log of communication includes Uniform Resource Locator (URL), GET method, POST method information, and the source IP address. The proposed method systematically gathers complete data on every request sent by users to the targeted web server or simulated user in a testing environment.
- **RPS Determination:** In the proposed system, a Node.js backend architecture is used to continuously determine RPS. Request rate is calculated for all IP addresses of the source type within a time frame of one second. The number of requests received by each IP in the window is summed up to yield the RPS value per IP. The result graphs indicate aggregated values of RPS, which are the total traffic of all active IPs within the same window.
- **RPS Comparison with Threshold:** In this step, the recorded RPS of each IP address is compared to a predetermined threshold value of 5 RPS within the one-second monitoring window. The maximum allowable number of RPS from a single IP address under typical operating circumstances is represented by this barrier.
  - Normal Traffic: Traffic from an IP address is regarded as being within the normal range if its computed RPS is less than or equal to the threshold (in this case, $RPS \leq 5$). The system keeps tracking the traffic, and no alert is raised.
  - DDoS Attack Indication: A possible DDoS attack is indicated if an IP address's RPS is above the cutoff point ($RPS > 5$). A prominent feature of many DDoS attacks, in which attackers try to overload the server with traffic, is this abrupt spike in requests from a single source.
- **Trigger WebSocket Alert:** The Node.js backend instantly initiates a DDoS alert when it detects a possible DDoS attack (i.e., when $RPS > 5$). All WebSocket clients that are connected receive this notice; these are usually the dashboards that network administrators use for real-time monitoring. The alert message contains details about the attack's time, severity, and attacking IP address.
- **Visualize Data:** The WebSocket-updated real-time monitoring dashboard shows any warnings that have been generated and dynamically visualizes the incoming traffic data. Administrators may easily spot irregularities and comprehend the nature of the attacks by using Chart.js to generate graphs and charts that display traffic trends over time.

This threshold-based technique offers a straightforward and efficient method for real-time detection of high-frequency DDoS attacks.

3.4 WebSocket-based alert system: Real-time communication

The Node.js server and the frontend monitoring dashboard can communicate in real time because the system uses WebSocket. The Node.js server and frontend monitoring are a crucial step, as it involves notification alerts to senders and communicating updates to admins and managers. Further, the WebSocket alert system's working is described as follows:

To communicate in real-time with all active clients or monitor dashboards, connect to the Node.js server through persistent, open WebSocket connections. WebSocket breaks the trend of one-way communication, unlike the traditional HTTP protocol, and enables two-way continuous communication. To perform instant data pushes, the server reacts immediately and informs all connected WebSocket clients whenever it logs a new request or when a DDoS alarm goes off. The users do not need to check in on a regular basis with server updates. The 'push' method used here reduces latency and lowers server load, making the overall detection method scalable and lightweight. Whenever the RPS value goes beyond the threshold limit, which has been set at five in this deployment, the dashboards get updates, and the server sends an attack alert to all connected dashboards; this phenomenon is completely dynamic, as the dashboard updates automatically to show an attack alert. This also includes unusual traffic patterns highlighted on the graph, displaying alert messages on the banner to notify the administrator and supplying the detailed attack information, such as attackers' source IP addresses and the time of occurrence.

3.5 Performance considerations: Optimization for real-time efficiency

When designing the proposed DDoS detection system, it's crucial to ensure accurate detection and performance. The system aims to operate with minimal latency, efficiently use resources, and expand to handle varying traffic loads. Important design choices and optimizations affect the performance goals mentioned below:

- **File-Based Logging:** Rather than relying on the conventional database system, the suggested system utilizes file-based logging for the storage of traffic data that arrives. A simple text file named traffic.log holds all traffic data. Since this removes the cost and latency associated with database connections, queries, and transactions, it offers significant performance gains in scenarios with high-frequency requests.

- **Lightweight Detection Logic:** In this case, the threshold-based RPS calculation is intentionally made lightweight and serves as the base for the detection core logic. The DDoS detection method does not involve any high-usage algorithms or complex calculations such as ML-based techniques. The approach provides a quick detection time and reduces the processing burden of the server.
- **WebSocket-Based Real-Time Updates:** It is the monitoring dashboard and server for monitoring dashboard communication. Manual monitoring and regular polling are no longer required, owing to WebSocket. It reduces latency and enhances responsiveness by decreasing network traffic and server load.
- **Efficient Front-End Visualization:** Due to its efficiency and inexpensive resource consumption, Chart.js is used for front-end visualization. The objective of Chart.js is to produce dynamic charts and graphs smoothly with the minimum memory required. This ensures that the real-time traffic graphs of the dashboard can be refreshed frequently without consuming much system power.

Such design choices can yield a system that can effectively handle high traffic volumes without sacrificing real-time detection features and yet offer an easy-to-use monitoring interface.

4. RESULT VISUALIZATION

By default, traffic volumes reported in this section are aggregated rates of requests across multiple clients running simultaneously, contrasting with the detection decision being based on per IP request rate thresholds.

A. Experimental Setup

Table 2. Experimental setup

Parameter	Configuration
Server Hardware	Intel i5 / 8 GB RAM
Operating System	Ubuntu 20.04 LTS
Node.js Version	v18.x
Web Server Framework	Express.js
Deployment Environment	Local VM / Cloud
Traffic Generator	Custom script / Apache JMeter
Number of Traffic Sources	1–50 concurrent clients
Request Rate	100–5000 requests/sec
Traffic Pattern	Constant + burst
Experiment Duration	5–10 minutes per run
Network Condition	Stable LAN, no packet loss
Monitoring Window Size	1 second
Per-IP Threshold	5 PRS
Attack Request Rate	10–100 PRS per IP

Note: RAM = Random Access Memory; LTS = Long Term Support; VM = Virtual Machine; LAN = Local Area Network; PRS = Requests Per Second; IP = Internet Protocol.

Table 2 shows the details of the server configuration, software environment, deployment setup, and traffic generation parameters used during the experimental execution. The ground truth labels were done according to the traffic generation process. Traffic created with high-rate automated request scripts above specified thresholds was identified as DDoS traffic, whereas low rate pattern of requests by users was identified as benign. This was because the generation of traffic was managed, so the actual classification of each request was available before being detected, thus making it possible to reliably assess it.

B. Result Analysis

By conducting tests on the system under normal traffic conditions, as well as under the conditions of a realistic DDoS attack, the proposed system was tested in terms of its performance, accuracy, and responsiveness in real-time. The actual data from the traffic log file was used to make all observations. It focused on the system based on the low-latency alert system, real-time, and the capacity to identify abnormal behavior.

- Traffic Monitoring and Anomaly Detection: The system effectively logged all incoming requests and determined the RPS in real-time under normal traffic levels. It successfully detected possible anomalies with a predefined threshold of 5 RPS without producing false positives. No unusual spikes were seen, and traffic patterns generally stayed within predicted bounds. The live monitoring dashboard continuously displayed these real-time facts, giving users a precise and understandable picture of traffic flow and behavior over time.
- DDoS Attack Detection: The traffic scenario is created using tools/ Scripts and the proposed system successfully identifying anomaly in such traffic over a 5RPS threshold in the created simulation environment. The system is also validating the dependability of its threshold-based detection algorithm as it is successfully raising DDoS alerts in real-time. Further, the raised alerts were delivered to the dashboard promptly and with the least amount of delay, thanks to the usage of WebSocket. Real-time connectivity made it possible to see traffic spikes and response triggers right away.

These observations are supported and illustrated by the following graphs.

4.1 Real-time vs. delayed detection (latency comparison)

As illustrated in Figure 2, the bar graph contrasts the detection latency of the proposed threshold-based real-time approach with a representative delayed detection scenario observed in ML-based pipelines. The threshold-based approach demonstrates its suitability for real-time alerting by generating alerts with a latency of 0.002 s, whereas ML-based approaches typically incur higher detection latency due to feature extraction, model inference, and decision aggregation stages, resulting in delayed responses of ~0.5 s. While threshold-based detection enables faster response with lower computational overhead, static thresholds may be less adaptable to evolving attack patterns compared to learning-based models, which offer higher adaptability at the cost of increased resource consumption and latency.

Table 3 also tabulates the detection time and the accuracy of the system for different types of DDoS attacks. It is portraying high rates of detection of more than 96% of all types of attacks. Synchronized Flag (SYN) Flood is detecting attacks with a 0.092-second time, which best attack detection time of 0.145 seconds, was with Internet Control Message Protocol (ICMP). It implies high responsiveness and low latency in a wide range of patterns of DDoS. The comparative analysis of attacks is presented in Figure 3.

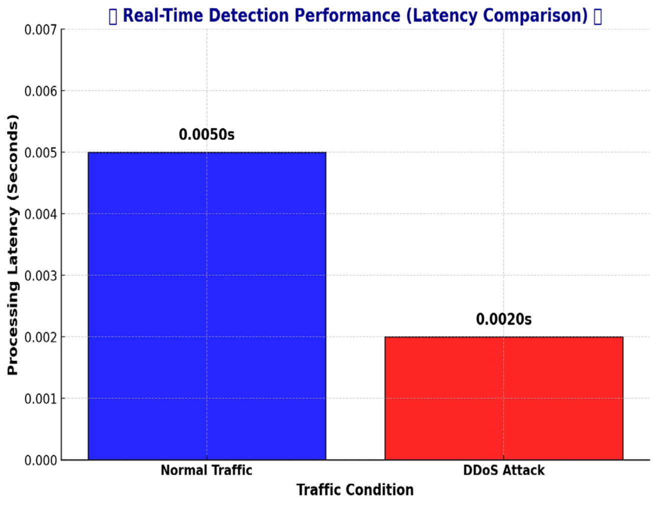

Figure 2. Real-time detection performance

Table 3. Performance metrics of the DDoS attack detection system

Attack Type	Detection Time (s)	Accuracy (%)	Precision (%)	Recall (%)
HTTP Flood	0.11	97.8	97	98
SYN Flood	0.092	98.2	98	98
UDP Flood	0.13	96.9	96	97
ICMP Flood	0.145	96.5	95	97

Note: DDoS = Distributed Denial of Service; HTTP = Hypertext Transfer Protocol; SYN = Synchronized Flag; UDP = User Datagram Protocol; ICMP = Internet Control Message Protocol.

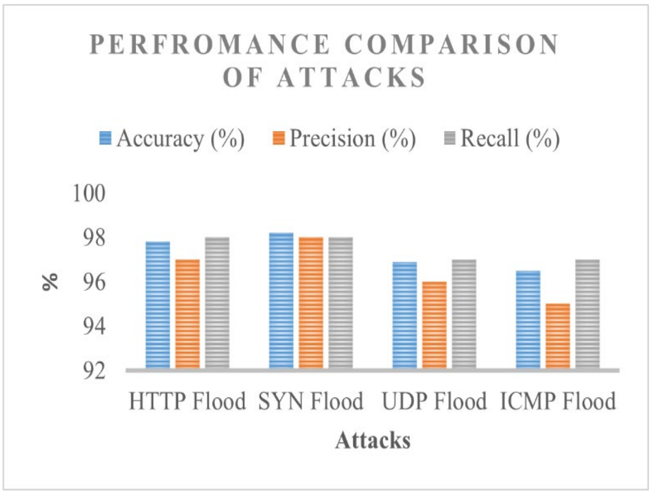

Figure 3. Performance comparison of attacks

The confusion matrices for the evaluated attack scenarios are illustrated in Figures 4(a)-(d), highlighting the classification performance for each attack type. The results demonstrate high true positive and true negative rates with minimal misclassification, confirming the effectiveness of the proposed model in accurately detecting different flooding attacks.

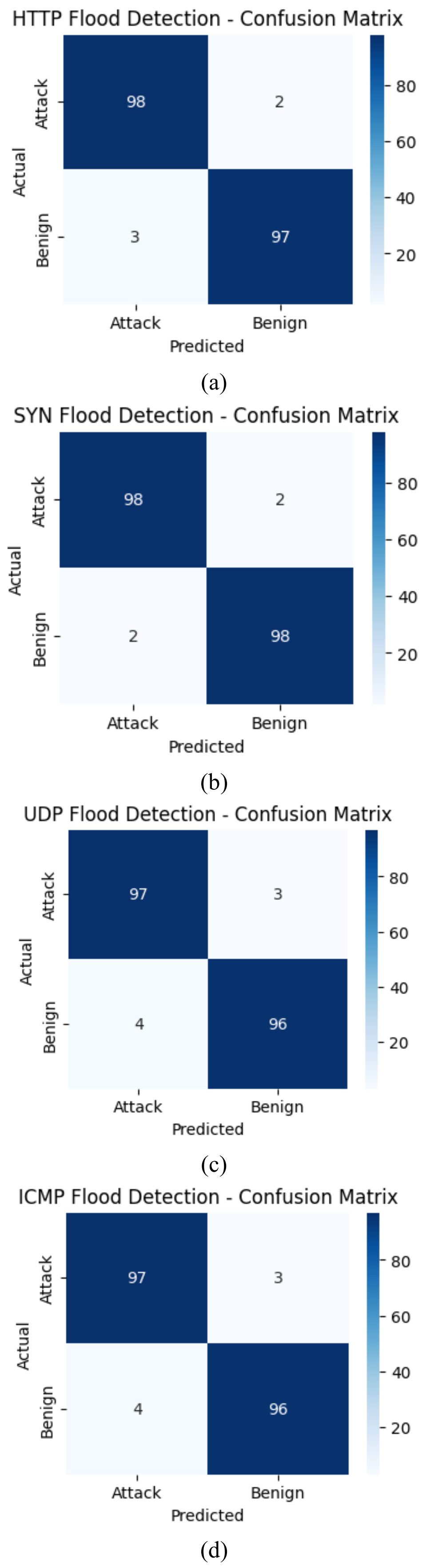

Figure 4. (a) Hypertext Transfer Protocol (HTTP) Flood detection – Confusion matrix; (b) Synchronized Flag (SYN) Flood detection – Confusion matrix; (c) User Datagram Protocol (UDP) Flood detection – Confusion matrix; (d) Internet Control Message Protocol (ICMP) Flood detection – Confusion matrix

4.2 Distributed Denial of Service detection threshold over time

The DDoS detection threshold is prominently displayed at 5 RPS on the time series graph, which displays the RPS over time. The rise in traffic indicates that the system was able to correctly initiate notifications when the RPS above this threshold was reached, as shown in Figure 5.

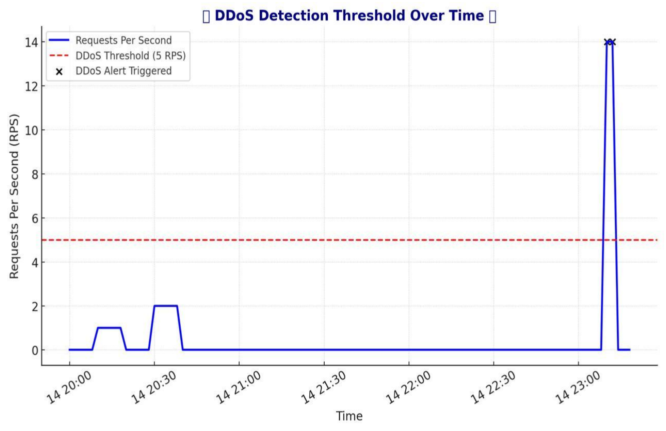

Figure 5. Distributed Denial of Service (DDoS) detection threshold

Table 4 displays sequential time intervals, their traffic volume, and the detection status of vulnerable events that may cause a DDoS attack. The detection status summary volume increases as time passes, and also if a DDoS attack alert is raised or confirmation of an attack is received.

Table 4. Distributed Denial of Service (DDoS) detection status over time intervals

Time Interval (s)	Traffic Volume (Requests/Sec.)	Detection Status
0–5	180	Normal
5–10	360	Alert Raised
10–15	410	DDoS Confirmed
15–20	200	Recovery

4.3 WebSocket notification speed

Figure 6 shows the WebSocket notification latency. It shows how a DDoS warning is triggered at a certain latency. The latency measure is at about 0.002 seconds and is notified. As per the observation, it is clear there was no discernible lag in the alarm in the broadcast of the detection event.

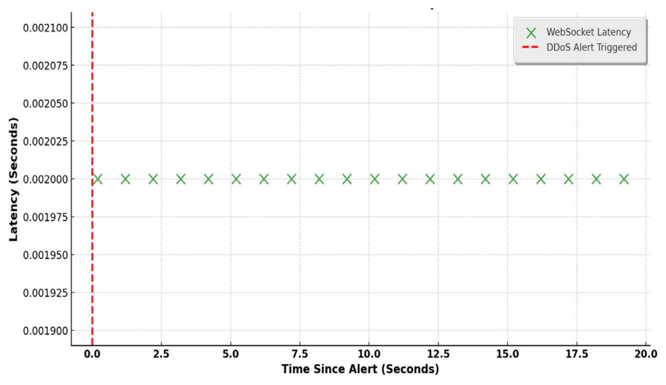

Figure 6. WebSocket notification latency

Table 5 presents the latency associated with each type of system notification during the DDoS detection lifecycle.

The threshold-based real-time detection model is simple, responsive, and can efficiently identify sudden traffic anomalies without the need for computational overhead. This makes the threshold model the best option for small-scale to medium-scale applications. The result shows that alert latency is extremely minimal, around 0.002 s alert latency, and maintains a high detection accuracy of over 96% for a variety of attack types. The attacks, such as HTTP, SYN, UDP, and ICMP Floods are analyzed in the proposed system. The constant RPS value is a reliable indicator of normal traffic fluctuations from malicious spikes. The WebSocket-driven updates ensure that alerts are transmitted and visualized immediately. The lightweight approach saves resources and provides immediate situational awareness. It is a superior option for small to medium-sized applications than heavier ML frameworks, which lead to rapid detection and low latency.

Table 5. Notification latency during the Distributed Denial of Service (DDoS) detection lifecycle

Notification Type	Latency (Seconds)
Initial Alert	0.025
DDoS Confirmation	0.030
Status Update	0.020
Recovery Notice	0.022

5. COMPARISON WITH OTHER LIGHTWEIGHT DETECTION SYSTEMS

The proposed DDoS attack detection system is a lightweight and scalable system, as it does not rely on complex ML algorithms and heavy datasets for training and testing, and pattern analysis. This makes the system more straightforward, and it doesn't compromise performance. The system achieves a balanced approach as complex pattern matching algorithms always compromise efficiency, and lightweight systems lag in accuracy in detection, and also sometimes lack real-time visualization capacities. This balance is achieved by combining a WebSocket-based real-time updating method, effective file-based logging, and threshold-based detection. The proposed system ensures low resource usage and maintains high responsiveness.

The threshold-based approach that is used in the current system is a quick, simple, and resource-saving way to identify DDoS attacks, but it has its limitations. By highlighting these limitations in key areas during future development efforts, the system's scalability, resilience, and adaptability to cybersecurity threats can be significantly enhanced. Potential avenues for future development are identified in the following critical areas.

5.1 Adaptive detection algorithms: Intelligent thresholding

The existing system of a static threshold of 5 RPS is effective in many typical attack scenarios, but it might be somewhat constrained and not the most suitable choice in dynamic traffic or unpredictable behavior. A predefined and set threshold can either miss attacks or result in false positives when the system is under high traffic conditions. In settings where legitimate traffic changes, non-malicious traffic surges have the potential of raising false positive rates. In the future, adaptive ML techniques could be added to future improvements. By using ML methods that take scalability and complexity into account, it is possible to dynamically learn and adapt to the protected system's baseline traffic behavior. The ability of these algorithms to identify recurring patterns and track changes in acceptable user behavior is achieved by analyzing past network traffic data. The system can adjust the detection thresholds automatically by continuously monitoring and learning from previous traffic, which will increase its sensitivity to real anomalies while reducing the potential for false alarms. This adaptive strategy would maintain the system's lightweight nature by increasing the system's detection accuracy for both low-and-slow and high-volume (volumetric) DDoS attacks without adding a substantial computing expense.

5.2 Real-time traffic classification: Differentiating malicious intent

Request rate is the only sign of possible malicious behavior in the current system. As a result, it is unable to distinguish between criminal high-traffic events—like coordinated botnet attacks—and legal high-traffic events—like viral marketing campaigns, massive file downloads started by a large number of users, or abrupt increases in interest brought on by trending news. The system's capacity to distinguish between attack traffic and legitimate user behavior would be greatly improved by the addition of real-time traffic classification modules. This might entail looking at more than simply the request rate while analyzing traffic, like:

- Analyzing the order and trends of requests made during certain user sessions is known as session behavior. Unlike typical user browsing patterns, malicious sessions may display repetitive or odd request sequences.
- Examining the distribution of source IP addresses is known as source diversity. While normal traffic spikes are usually characterized by a wider dispersion of sources, an attack may be indicated by a highly concentrated volume of requests coming from a small number of IP addresses.
- Request Type and Frequency: Examining the different kinds of requests (such as GET and POST) and how frequently they are made for particular resources. Malicious activity may be indicated by unusual request types or unusually high frequency of requests for particular, resource-intensive endpoints.
- Payload examination (superficial): A lightweight system may not be able to handle the resource demands of deep packet inspection; anomalies suggestive of specific attack types may be found through superficial examination of request headers or payload sizes.

The system might considerably lower false positives and react more precisely and successfully to real DDoS attacks by integrating real-time information on the type and characteristics of the traffic.

6. CONCLUSIONS

This paper presented a lightweight and scalable real-time DDoS anomaly detection system designed for efficient traffic monitoring with minimal computational overhead. The proposed threshold-based approach enables early detection of flooding attacks while remaining suitable for deployment in resource-constrained environments. Experimental evaluation on multiple attack scenarios, including HTTP Flood and SYN Flood attacks, validates the effectiveness of the proposed system. The results demonstrate consistent real-time detection with a latency of 0.002 sec and an overall alert accuracy of ~97%, as reflected through confusion matrix analysis and comparative performance metrics. These findings highlight the system's ability to promptly identify abnormal request patterns across different attack types without reliance on complex learning pipelines. Future work will focus on incorporating adaptive thresholding to improve robustness against dynamic traffic variations and extending the framework with hybrid detection strategies that balance real-time responsiveness with adaptability to evolving attack behaviors.

REFERENCES

[1] Akhi, M., Eising, C., Dhirani, L.L. (2025). TCN-based DDoS detection and mitigation in 5G healthcare-IoT: A frequency monitoring and dynamic threshold approach. IEEE Access, 13: 12709-12733. https://doi.org/10.1109/ACCESS.2025.3531659

[2] Pebrianto, J., Suryani, V. (2025). Adaptive DDoS attack detection: Entropy-based model with dynamic threshold and suspicious IP reevaluation. IEEE Access, 13: 55858-55876. https://doi.org/10.1109/ACCESS.2025.3553144

[3] Salman Qasim, S., NSAIF, S.M. (2024). Advancements in time series-based detection systems for distributed denial-of-service (DDoS) attacks: A comprehensive review. Babylonian Journal of Networking, 2024: 9-17. https://doi.org/10.58496/BJN/2024/002

[4] Saudagar, S., Kulkarni, M., Giramkar, A., Godse, S., Gupta, S., Patil, G., Gunjal, S. (2023). Image encryption based on advanced encryption standard (AES). In 2023 International Conference for Advancement in Technology (ICONAT), Goa, India, pp. 1-4. https://doi.org/10.1109/ICONAT57137.2023.10080243

[5] Ouhssini, M., Afdel, K., Akouhar, M., Agherrabi, E., Abarda, A. (2024). Advancements in detecting, preventing, and mitigating DDoS attacks in cloud environments: A comprehensive systematic review of state-of-the-art approaches. Egyptian Informatics Journal, 27: 100517. https://doi.org/10.1016/j.eij.2024.100517

[6] Abiramasundari, S., Ramaswamy, V. (2025). Distributed denial-of-service (DDOS) attack detection using supervised machine learning algorithms. Scientific Reports, 15: 13098. https://doi.org/10.1038/s41598-024-84879-y

[7] Saudagar, S., Ranawat, R. (2023). Attack classification and detection for misbehaving vehicles using ML/DL. International Journal on Recent and Innovation Trends in Computing and Communication, 11(8s): 491-496. https://doi.org/10.17762/IJRITCC.V11I8S.7230

[8] Mohsin, M.A., Hamad, A.H. (2022). Performance evaluation of SDN DDoS attack detection and mitigation based random forest and K-nearest neighbors machine learning algorithms. Revue d'Intelligence Artificielle, 36(2): 233-240. https://doi.org/10.18280/ria.360207

[9] Reed, A., Dooley, L., Mostefaoui, S.K. (2024). The

guardian node slow DoS detection model for real-time application in IoT networks. Sensors, 24(17): 5581. https://doi.org/10.3390/s24175581

[10] Batchu, R.K., Bikku, T., Thota, S., Seetha, H., Ayoade, A.A. (2024). A novel optimization-driven deep learning framework for the detection of DDoS attacks. Scientific Reports, 14: 28024. https://doi.org/10.1038/s41598-024-77554-9

[11] Franco, M., von der Assen, J., Boillat, L., Killer, C., et al. (2021). Poster: DDoSGrid: A platform for the post-mortem analysis and visualization of DDoS attacks. In 2021 IFIP Networking Conference (IFIP Networking), Espoo and Helsinki, Finland, pp. 1-3. https://doi.org/10.23919/IFIPNetworking52078.2021.9472850

[12] Wanjale, K., Shah, S.A., Kharole, C., Kakad, S., Yadav, I., Wakade, R. (2025). Detecting DDoS attacks in real-time with minimal infrastructure: A Node. js approach. In 2025 International Conference on Future Technologies (ICFT), Sangli, India, pp. 1-8. https://doi.org/10.1109/ICFT66708.2025.11336760

[13] Zargar, S.T., Joshi, J., Tipper, D. (2013). A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks. IEEE Communications Surveys & Tutorials, 15(4): 2046-2069. https://doi.org/10.1109/SURV.2013.031413.00127

[14] Bhuyan, M.H., Bhattacharyya, D.K., Kalita, J.K. (2014). Network anomaly detection: Methods, systems and tools. IEEE Communications Surveys & Tutorials, 16(1): 303-336. https://doi.org/10.1109/SURV.2013.052213.00046

[15] Joshi, M., Tiwari, A., Dhabliya, D., Lavate, S.H., Ajani, S.N., Gandhi, Y. (2025). Building AI-driven frameworks for real-time threat detection and mitigation in IoT networks. In 2025 International Conference on Emerging Smart Computing and Informatics (ESCI), Pune, India, pp. 1-6. https://doi.org/10.1109/ESCI63694.2025.10988310

[16] Saravanan, A., Bama, S.S., Kadry, S., Ramasamy, L.K. (2019). A new framework to alleviate DDoS vulnerabilities in cloud computing. International Journal of Electrical & Computer Engineering, 9(5): 4163-4175. https://doi.org/10.11591/ijece.v9i5.pp4163-4175

[17] Kumavat, K.S., Gomes, J. (2025). Multi-layer DDoS attacks detection with mitigation in IoT-enabled sensor network. Cybersecurity, 8(1): 72. https://doi.org/10.1186/s42400-025-00378-1

[18] Ma, K., Abraham, A., Yang, B., Sun, R. (2016). Intelligent web data management of WebSocket-based real-time monitoring. In Intelligent Web Data Management: Software Architectures and Emerging Technologies, pp. 105-124. https://doi.org/10.1007/978-3-319-30192-1_6

[19] Adedeji, K.B., Abu-Mahfouz, A.M., Kurien, A.M. (2023). DDoS attack and detection methods in internet-enabled networks: Concept, research perspectives, and challenges. Journal of Sensor and Actuator Networks, 12(4): 51. https://doi.org/10.3390/jsan12040051

[20] Saudagar, S., Ranawat, R. (2024). An amalgamated novel IDS model for misbehaviour detection using VeReMiNet. Computer Standards & Interfaces, 88: 103783. https://doi.org/10.1016/j.csi.2023.103783

[21] Tran, T.M., Nguyen, K.V. (2019). Fast detection and mitigation to DDoS web attack based on access frequency. In 2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF), Danang, Vietnam, pp. 1-6. https://doi.org/10.1109/RIVF.2019.8713762

[22] Babbar, H., Rani, S., Driss, M. (2024). Effective DDoS attack detection in software-defined vehicular networks using statistical flow analysis and machine learning. PLoS One, 19(12): e0314695. https://doi.org/10.1371/journal.pone.0314695

[23] Saudagar, S., Mahatme, S., Jadhav, A., Shelke, M. (2025). Securing cloud based transmission through integration of cryptographic method. In 2025 9th International Conference on Computing, Communication, Control and Automation (ICCCBEA), Pune, India, pp. 1-5. https://doi.org/10.1109/ICCUBEA65967.2025.11283899