Analysing the Performance of Web-services during Traffic Anomalies

Avneet Dhingra[1] **and Monika Sachdeva**[2],
[1] Research Scholar, I. K. Gujral Punjab Technical University, Kapurthala--144603, India
[2]Associate Professor, I.K. Gujral Punjab Technical University, Kapurthala-144603, India

Summary
Intentional or unintentional, service denial leads to substantial economic and reputation losses to the users and the web-service provider. Thus, it is essential to understand and quantify the impact of such anomalies on the victim, so that proper measures can be taken well in time. In this paper, important performance metrics distinguishing both transmission issues and application issues have been discussed and evaluated. The legitimate and attack traffic has been synthetically generated in hybrid testbed using open-source software tools. The experiment covers two scenarios, representing DDoS attacks and FEs, with varying attack strengths to analyze the impact of anomalies on the server and the network. It has been demonstrated that as the traffic surges, response time increases, and the performance of the target web-server degrades. The performance of the server and the network is measured using various network level, application level, and aggregate metrics, including throughput, average response time, number of legitimate active connections and percentage of failed transactions.

Keywords:
Network security, network anomaly, denial of service, flash event, performance metrics.

1. Introduction

With an exponential increase in the use of IoT, the strength of attacks, as well as Flash events (FE), has increased, giving a very valid reason to quantify and understand the impact of such heavily loaded anomalies. The need arises to generate realistic techniques to evaluate the performance and measure the impact of anomalies on the services of the web-server. In the event of an anomaly, the users get to grips with either a drastic slowdown of the services or a complete outage. The online encyclopaedia, Wikipedia, suffered a DDoS attack on September 6, 2019, that lasted for about three days [21]. The intermittent outages and performance degradation were faced by users in the Middle East, Europe, the United Kingdom, and the United States. Measuring the performance of the server under such anomalies can help understand the preventive measures that are required to be installed along with the type of potential defences. Thus, the importance of performance testing is realized in the situation when multiple users generate concurrent traffic creating a heavy load on the network. The network responding to anomalies needs to be tested repetitively with short-duration attack traffic to evaluate the overall performance of the server and the cost involved for installing the required security. The metric, thus calculated, provides the information related to the network in case of saturation.

This paper presents the exhaustive literature survey conducted to comprehend the concept of performance and quantifying the impact of anomalies on the web-services. Various performance metrics have been defined and evaluated using the synthetically generated traffic in DDoSTB hybrid testbed [18]. The results of the study have been presented as graphs to give the visual effect of traffic surges and realize their impact on performance. The insight into the performance metrics can assist in understanding the cause of performance degradation. The paper defines performance metrics quantifying the quality of service (QoS) of the web server during normal conditions and under the increased traffic load. The experimental setup and procedure to evaluate performance metrics of the designed network have been discussed.

The paper has been organized as follows. Section 2 gives an overview of what performance metrics are and its importance in the detection of anomalies. The related literature has been reviewed in section 3. Section 4 describes the model of an experimental network using realistic topology and software tools used to generate legitimate and attack traffic. Section 5 discusses the metrics selected for analysis, and the results obtained are presented as graphs for better understanding. The paper concludes in section 6.

2. Performance metrics

As each network has a different topology and varies in purpose, different factors define the performance for each environment. Performance metrics quantify the QoS provided by the server, during normal conditions, and under the increased traffic load. The performance of a network depends on the following factors:

- The rate at which the information can be or is transferred,
- The delay between the request sent, and the response received,
- Error in transmission.

These factors are quantified with generated metrics, keeping in view the user requirements for a particular

Manuscript received May 5, 2025
Manuscript revised May 20, 2025
https://doi.org/10.22937/IJCSNS.2025.25.5.12

application. Mirkovic et al [6] have explicated the response times of various kinds of applications and concluded that there is no particular set of metrics that can be considered as the benchmark for measurement of performance. The interactive applications are expected to respond in minimum time. For such applications, including VoIP, video streaming, chatting, and gaming applications, the delay is expected to be the minimum. The applications, such as email transfer, which are non-interactive, do not have any defined or tolerable limit for the delay. Also, there is no specific threshold or baseline to define the best performance. To efficiently examine the impact of anomalies in network traffic, metrics need to be accurate, quantitative, and versatile [5]. Ideally, this can be achieved using realistic networks, giving a holistic view of the network parameters. However, it is practically not possible to have such an expansive view due to certain economic, legal, and privacy issues. Thus, techniques like simulation or emulation are used to create a virtual network that can generate realistic synthetic traffic.

Table 1 discusses various network level, application level, and aggregate level metrics evaluated in this paper. As per [10], network level, transport level, and application-level metrics need to be assessed together to evaluate the overall performance of the network.

Table 1: Performance metrics

Level	Metric
Aggregate	Throughput, Goodput, Badput
Application	Response time Number of legitimate requests dropped Number of legitimate active connections Percentage of failed transactions Average serve rate/ Average request rate Legitimate packet drop probability
Network	Percentage of link utilization
Server	CPU-utilization

3. Literature Review

Researchers have studied the damage caused due to the poor performance of the victim server and suggested certain damage models. The models give an insight into the situation, the risks involved, and highlight the importance of analysing and evaluating the performance of the system. One such model proposed by [22], divides the financial damage into characteristics like disaster recovery, loss due to downtime, liabilities involved, and losing the customers. The model can be generalized to any traffic anomaly affecting the availability of the server (like FE) and hence degrading the performance.

Vasudevan et al [17] recommended metrics based on the cost of losing customers and the cost of SLA violations, which is from the point of view of the network users. It draws attention to the possible financial impact a DDoS attack can have on the network-provider of the affected network. The MIDAS2007NET factor has been defined based on the fact that allocated bandwidth is proportional to the volumes of traffic on the network and which in turn are proportional to the related profits.

Gade et al [16] studied the impact of the DoS Land (Local Area Network Denial) attack to compare the memory and processor utilization during the attack. Chertov et al [15] concluded that simulated networks produced different results from the emulated ones. The difference is because of the assumptions taken for simulation. The authors suggest the use of testbeds for accurate results. The metrics computed to measure performance are the average goodput (computed using transfer size and time for completion of transfer), size of the congestion window (calculated by dividing the average of weighted congestion window by average of segment size), percentage of CPU utilization and packets sent (and received) per second.

Mirkovic et al [5,6,7,8] defined the various legacy metrics along with the applications where these metrics give accurate results. The metrics discussed are packet loss, throughput, request/response delay, transaction duration, and allocation of resources. Packets lost in the transit after an anomaly hits the network is termed as packet loss. In this case, a network could be either a direct hit network or nearby networks experiencing collateral damage. It measures the network congestion caused by flooding attacks. Throughput is the number of bytes transmitted or re-transmitted per time-interval. Goodput is the same as throughput with the difference that re-transmitted bytes are not counted. This metric, however, does not give accurate results for connections with few packets as the throughput, in this case, is already low.

Request/response delay metric cannot be applied to non-interactive applications and one-way traffic. Transaction duration captures the time required for exchanging a set of messages between a source and destination. The metric efficiently measures the services for the interactive applications (example-browsing internet) but fails to give results for one-way traffic. The allocation of resources is in proportion to a critical shared resource, like bandwidth. The ratio of resources allocated to legitimate traffic versus the attack traffic captures the service quality as viewed by the user. It measures the server load but fails to determine the collateral damage caused to the network.

The authors categorized the applications into five groups- interactive applications (web-based), media applications (audio-video streaming), online games, chat

applications, and non-interactive applications(email). Metrics vary across these applications and cannot be generalized. They proposed a few impact metrics keeping in mind the QoS requirements of different applications. One of them being a percentage of failed transactions (pft), which quantifies the quality of services experienced by the users. Another metric, DoS-hist, shows the resilience of an application to an attack with the help of histograms for pft.

Singh et al [10], has applied application layer metrics and network layer metrics on the trace generated using the NS-2 simulator. The author speculates that the network level and transport-level parameters are not sufficient by themselves to detect application-layer attacks. They have checked the performance of the network for various GET-HTTP DDoS attacks.

Sachdeva et al [13] evaluate the DDoS and FE impact on web services using DETER testbed. The metric throughput was evaluated as goodput and badput. Goodput is as defined by [5], and badput is the attack traffic over bottleneck link during a given time window. Authors have experimented with traffic for response time, percentage of request packets lost, legitimate packet survival ratio, percentage of failed transactions, and bottleneck bandwidth utilization (for goodput). Carrying forward the research, Sachdeva et al [14] have defined an additional metric giving the ratio of average serve rate and average request rate. This ratio is 1 for normal traffic and decreases as the strength of attack increases. These are evaluated using the NS-2 simulator.

Bhatia et al [20] recommended the performance evaluation of network using server-level metrics viz, system-level CPU utilization, user-level CPU utilization, CPU load, and real memory utilization. The authors have proposed a framework for generating realistic traffic for the anomalies under study, using minimal hardware. Apart from the legacy metrics, Behal et al [18] have also used sever load metrics – CPU utilization, memory utilization, and CPU load to test the performance of the server. The researcher has developed DDoSTB testbed to generate the required synthetic traffic for experimentation.

Bhandari et al [2] consolidate the metrics used for evaluating the performance of the defence framework and classifies them at a packet level, aggregate level, and application-level based on the level of the network they are used for. The author has also discussed the system parameters affected in the case of DDoS attacks and the different tools to measure the same.

Performance metrics have been classified as external metrics and internal metrics [2, 4, 5, 18]. The metrics that do not need privileged access to the system or its network are termed are external—for instance, attack strength or defence parameters. The internal metrics measure CPU load, CPU utilization, memory utilization, internal algorithms, and data structures. Another valid criterion for classification is the OSI layer involved in measuring the metric [4, 18]. The metric may measure the aggregate performance of networks such as throughput, or it may work on application level like transaction duration or at a packet level, such as a number of re-transmissions [2, 14].

4. Experimental Setup

The setup of an experiment and its procedure includes three components - network topology, legitimate traffic, and attack traffic [8]. The best option is to set up an experimental environment in a real-time operational network that handles live traffic. However, the drawback involved in a real-time environment is that it cannot be reconfigured or scaled as per the experiments' needs. Thus, other options of simulated environment or emulated environment can be taken into consideration. Out of the two, simulation and emulation, the former lacks realism, and traffic replay is slow. Thus, the latter is the preferred method for testing by the researchers [7, 10, 13, 18, 19]. Emulator, for instance - DETER, is the combination of real hardware plus simulator to achieve the desired topology of a network. It provides a more rational environment as compared to a theoretical model or simulation alone. Soft routers and soft network links are used with the real systems real applications. However, the emulator is scalable only to a certain extent. Identifying the compatible tool along with the data source and its deployment on the systems can, somehow, be a challenge.

In this section, the performance of the testbed network has been carefully examined using the three components discussed in [8]. Flooding attack is generated using the realistic topology on the available testbed with the help of available and compatible software tools.

4.1 Network Topology

For the experimentation, authors used the hybrid DDoSTB Testbed, consisting of real as well as emulated systems, developed by Behal et al [18, 11]. 45 physical systems have been deployed and grouped into 3 clusters of 15 computers each. Out of the 3 clusters, 2 clusters are specified for background traffic and 1 cluster for attack traffic. The systems are run on Windows XP and Ubuntu instances. To increase the nodes, the v-core emulator [18] is used for attack cluster that is, cluster 3. One v-core node implements 30 virtual nodes. This scales the network to (15 x 30) 450 attack nodes. Each node in cluster 1 and

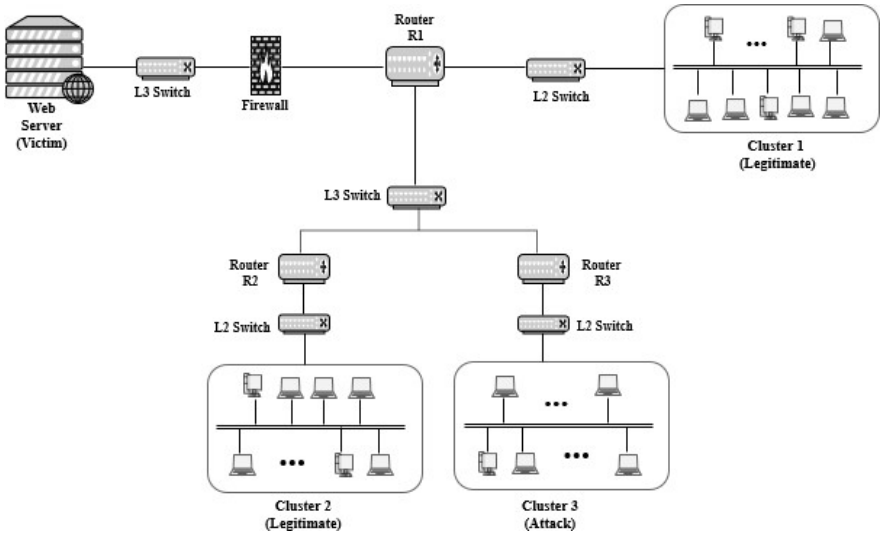

Fig. 1 Network topology for experimentation

cluster 2 is deployed with 30 instances of Apache JMeter to scale the number of legitimate nodes to a total of 900 nodes. For making identification of source easier, different pools of IP-address are assigned to the users in each cluster. The victim web server gives access to the resources to the users from both legitimate and attack clusters. Every request attempt is stored in the log file of the victim server. The topology deployed on DDoSTB is shown in Fig. 1, and the associated parameters have been summed up in Table 2. Generally, the typical link speeds are used to assign link bandwidths. In Fig. 1, the link between R1 and server acts as the bottleneck link. The bandwidth of this link is 100 Mbps, with a delay of 5ms. The rest of the links in the topology have a bandwidth of 1 Gbps. Topology for the experiment is finalized considering the principles of benchmarking, as suggested in [9]. An attempt has been made to keep it realistic and comparable to the topology of the internet.

Table 2: Topology parameters

Parameter	Value
Number of legitimate nodes	2 clusters times 15 nodes times 30 VMs = 900
Number of attack nodes	1 cluster times 15 nodes times 30 VMs = 450
Number of routers	3 (R1, R2, R3)
Number of switches	5 (2 L3 Switch, 3 L2 Switch)
Bandwidth from R1 to web Server (Bottleneck Link)	100 Mbps
Delay of a link from R1 to a web server	5 ms

4.2 Generating Traffic Traces

The network traffic is generally a blend of two protocols-Transmission Control Protocol (TCP) and User Datagram Protocol (UDP)- working at layer 4 of the OSI model. HTTP is an underlying generic protocol used by the world wide web to transfer data to and from the client and server. It operates in the application layer and relies on TCP in the transport layer for establishing a connection between the client and server. For the successful connection, TCP requires a valid IP-address. If an address is valid, it can establish a connection with the server. Each command is executed independently without the knowledge of the commands before and after it. GET and POST are two types of HTTP requests used for applications for which transmission time is not very critical but at the same time need reliability. These commands are not dependent on the commands that precede them or follow them. UDP is a connectionless protocol as it does not require any virtual connection to be established before any data transfer. It is used for fast, efficient transmission like games and VoIP. It enables process-to-process communication by sending messages, called datagrams. It is efficiently used for applications that are loss tolerating and require low latency.

In the absence of real-time datasets, the traffic is generated synthetically to evaluate the system [18, 19]. To achieve realistic trace, the client machines (nodes) interact with applications on a victim server with a non-spoofed and broad spectrum of source IP-addresses. Various open-source traffic generators are considered and studied to obtain a manageable mix of normal traffic and attack traffic for the experiment performed. Finally, for

Manuscript received May 5, 2025
Manuscript revised May 20, 2025
https://doi.org/10.22937/IJCSNS.2025.25.5.12

generating background traffic and attack traffic, Apache JMeter and D-ITG, respectively, were chosen. These tools have been discussed below.

Apache JMeter is a pure Java open-source software designed to measure the performance of web services [24]. It is a multi-platform Java desktop application. It has a friendly and easy to use Graphical User Interface (GUI). The results can be visualized as a log file and using a chart, table, or tree. Multiple virtual users can be created to generate heavy load against the victim server. JMeter supports all basic protocols, such as HTTP and FTP. Therefore, JMeter has been the choice of researchers of late [11, 12]. For the experiment described in this paper, the same version of JMeter is loaded on all the systems in cluster 1 and cluster 2. The distributed testing in JMeter requires one master, number of slaves, and one target machine, as shown in Fig. 2. One of the nodes in cluster 1 is configured as a master. The GUI runs on master and controls the rest of the nodes in cluster 1 and 2. Slaves run JMeter-server and take commands from the GUI and send requests to the target system. In short, each cluster has 15 nodes, and 30 virtual machines (VMs are defined as users in JMeter) are added hierarchically in the topology. Thus, the number of users generating traffic is scaled up to 900. The ramp-up time is set to 10 seconds so that each VM sends 3 requests per second. Similarly, the master-slave model is configured in cluster 3 with JMeter to generate HTTP traffic as attack traffic. Users in each cluster are assigned IP-addresses from different pools to distinguish between legitimate and illegitimate users [23].

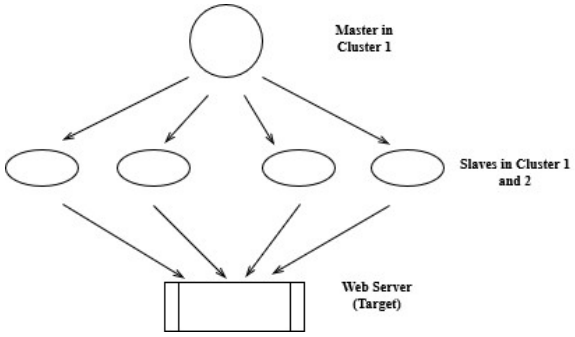

Fig. 2 Master- slave model implemented in the experiment

Distributed Internet Traffic Generator (D-ITG) is a software platform capable of producing traffic that accurately adheres to patterns defined by the inter-departure time (IDT) between packets and packet size stochastic processes [1]. It supports both linux and windows-based OS. It generates traffic having layer 3, layer 4, as well as layer 7 features. It emulates the sources of protocols like TCP, UDP, ICMP, DNS, Telnet, and VoIP [1, 3, 18]. The header fields like packet size, source, and destination IP-address, source, and destination port numbers can be customized as per the requirements. Due to its wide-ranging features, it has been widely used in research to generate synthetic traffic [17, 3] and hence is the choice for generating attack traffic for the experiment mentioned in this paper. Cluster 3, as shown in Fig. 1, contains 15 nodes and 30 VMs on each node. Thus, attack traffic is generated from 450 users in cluster 3.

5. Results and Discussions

The performance of the victim server has been analysed under two experimental setups. In the first setup, the UDP traffic (representing DDoS attack) is generated as attack traffic in cluster 3 using the D-ITG tool and mixed with the normal traffic from 60th second to 120th second of the experiment targeting the victim server. Traffic is studied with IDT of 50 pps, 100 pps, and 200 pps.

In the second setup, the HTTP traffic (representing FE) is generated in cluster 3 using Apache JMeter from 60th second to 120th second of the experiment. The attack traffic generated is mixed uniformly with the legitimate traffic generated by cluster 1 and 2 and targeted towards the victim server. Performance is observed for three instances- when attack traffic is 10%(approx.), 30%(approx.) and 75%(approx.) of total traffic. Set-ups 1 and 2 have been outlined in Table 3. The graphs obtained when the network was subjected to attack traffic, and FE traffic has been shown in Fig. 3 and 4, respectively. The performance metrics used for analysing the impact of an attack and FE are discussed below:

5.1 Throughput

It is the amount of data transferred (in packets, in bytes or bits) in the unit time interval. It gives insight to the congestion of the network. Goodput is computed as the number of bits per second of legitimate traffic received at the server, and badput is the number of attack bits received per second at the server. Higher the goodput, the higher is the efficiency of the system being tested. These can be expressed as

Throughput=Total traffic reaching server /Time-interval
Goodput=Legitimate traffic reaching server/ Time interval
Badput=Attack traffic reaching server / Time interval

Fig. 3(a) and 4(a) show that for normal traffic, goodput increases slowly in the beginning until it reaches the bandwidth limit where it stabilizes. The slow start can be attributed to the congestion control strategy used by TCP. The transmission rate is increased by the slow-start algorithm until either a loss is detected or the receiver

server's bottleneck link bandwidth is reached. In case loss occurs, the algorithm assumes that the network is congested and takes measures to reduce load. Otherwise, goodput increases exponentially until it reaches the bandwidth limit.

The traffic through the bottleneck link increases suddenly during the 60th second. This plummets the goodput value. As the attack strength increases, the value of goodput decreases. This is because as TCP senses packet loss, it decreases the transmission rate and hence decreasing the goodput. Badput, however, is proportional to the traffic rate irrespective of the type of traffic, as shown in Fig. 3(b) and 4(b). Hence, it concluded that as soon as the traffic increases, whether UDP or HTTP, the goodput of the server plummets to almost 7 Mbps and badput increases with an increase in attack strength.

5.2 Average response time

It is defined as the time between the request being sent from the client and receiving the first response [14]. It is also known as average latency [10]. This attribute is directly proportional to the amount of congestion in the network—lower the value of response time, less the congestion, and vice versa. In [13], the authors suggest that in the case of HTTP transactions, it is important that the response time is less than 10 seconds to involve the users in the service and make the transaction successful. If

Table 3: Parameters of setup-1 and setup-2

Parameter	**Tool**	**Value**
Background/Legitimate traffic	JMeter	2 clusters x15 nodes x30 VMs = 900
Experiment time		180 seconds
Attack duration		60 seconds (from 60th sec to 120th sec)
Legitimate requests generated		Cluster 1: 15 x 30 x 3 = 1350 requests/sec Cluster 2: 15 x 30 x 3 = 1350 requests/sec Total = 2700 requests/sec
Packet Size		4096 bytes
Setup 1:		
Attack traffic	D-ITG	UDP
Attack type		Constant rate
Packet size		512 bytes
Attack generated		Cluster 3: At 50pps per user Attack traffic = 50 x 450 = 22500 pps = 92.16 Mbps At 100 pps per user Attack traffic = 100 x 450 = 45000 pps = 184.32 Mbps At 200 pps per user Attack traffic = 200 x 450 = 90000 pps = 368.64 Mbps
Setup 2:		
FE traffic	JMeter	HTTP
Traffic type		Uniformly distributed
Packet Size		500 bytes- 1000 bytes
Traffic generated		Cluster 3: For 10% FE traffic: VMs =10, 2 request/sec for each VM Total requests/sec = 15 x10 x 2 = 300 10% (approx.) of total traffic reaching server For 30% attack traffic: VMs =30, 6 request/sec for each VM Total requests/sec = 15 x 30 x 6 = 1200 30% (approx.) of total traffic reaching server For 75% attack traffic: VMs =50, 10 request/sec for each VM Total requests/sec = 15 x 50 x10 = 7500 75% (approx.) of total traffic reaching server

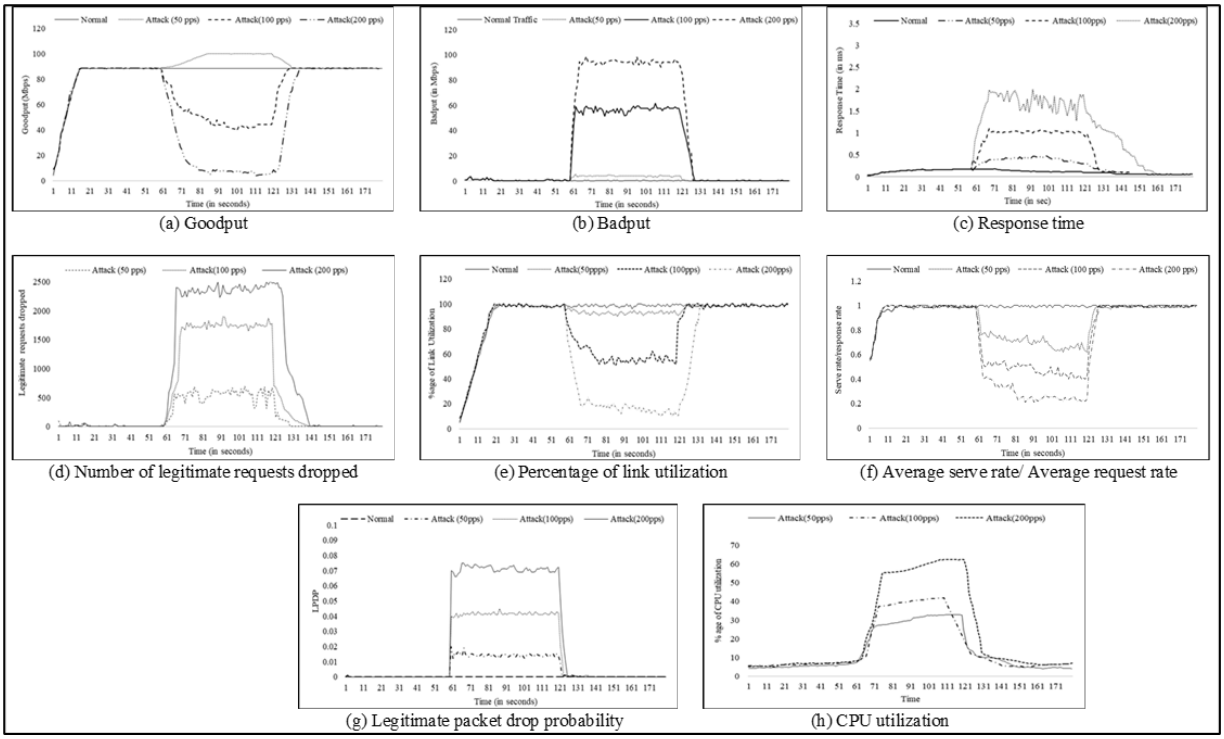

Fig. 3 Experimental setup 1- UDP traffic

the request crosses that limit, it is considered to be failed. Average response time can be computed as

$$Response\ time = \Sigma\ (T_c + T_d + T_s)/N$$

Where T_c is time for request sent from client to server, T_s is time for response sent from server to client, T_d is the time taken by the server to process the request, and N is the number of time intervals. Fig. 3(c) and 4(c) show an increase in response time with an increase in strength of attack in case of UDP traffic and HTTP traffic, respectively. UDP traffic has a slower response time as compared to HTTP traffic. The reason for the same is that in UDP, a connection is not formed between user and server, a datagram is just sent. Thus, UDP is faster than TCP, where the next packet is sent only once the acknowledgment is received for the previous one leading to the wait time and hence, an increase in the response time.

5.3 Number of legitimate requests dropped

The number of requests dropped, due to an attack, measures the amount of congestion in the bottleneck link. For experiment 1, as UDP traffic increases, the congestion of the bottleneck link leads to the dropping of a large number of requests to the server. Fig. 3(d) shows the scenario of experiment setup-1, where the number of legitimate requests dropped increases with an increase in the attack strength.

5.4 Number of legitimate active connections

Clients who have successfully connected themselves to the server and started sending the data are considered active connections. In the case of TCP connections, the active connections complete the 3-way handshake. When the attack packets of type HTTP surge the traffic, a number of packets endure time-out and hence reduce the window-size to almost one. According to the slow-start algorithm, the network reduces the load on the server by dropping the requests. Thus, the number of active connections decreases with an increase in the traffic beyond the capacity of the bottleneck link. The number of connections can reach as low as 90%, as suggested in Fig. 4(d). It can be observed that a large number of legitimate clients are denied services as the strength of attack increases.

Manuscript received May 5, 2025
Manuscript revised May 20, 2025
https://doi.org/10.22937/IJCSNS.2025.25.5.12

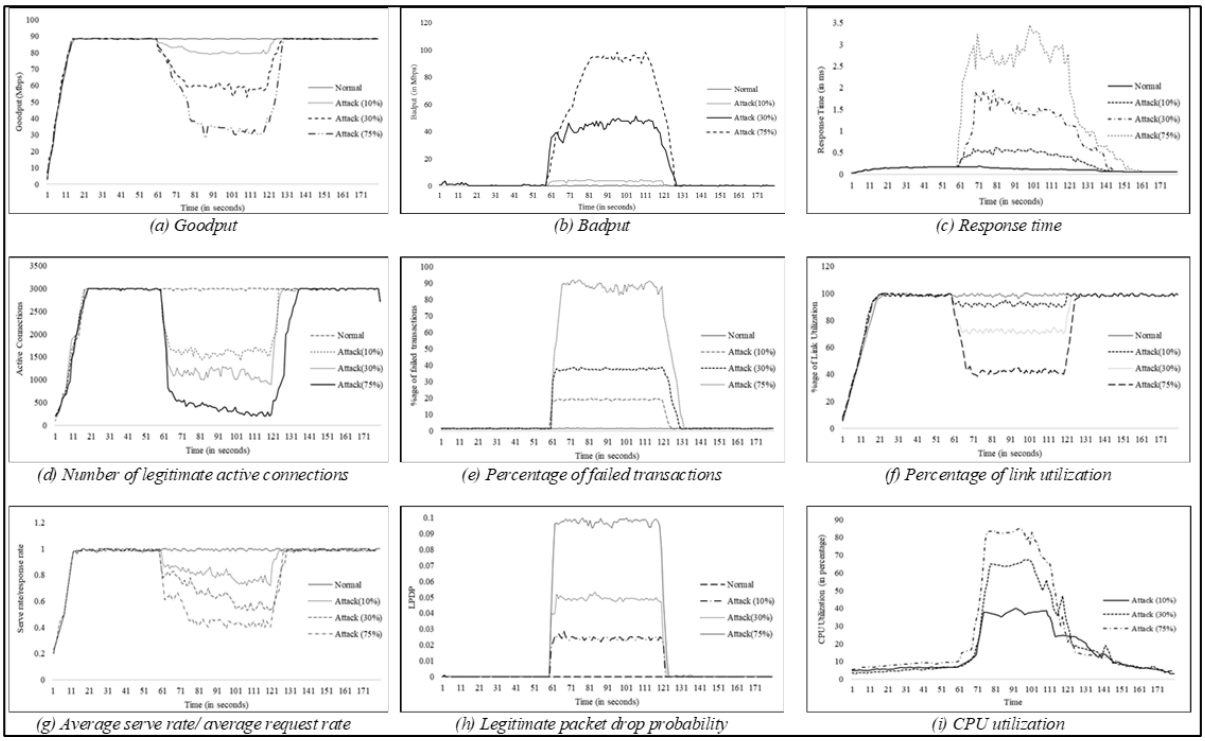

Fig. 4 Experimental setup 2- TCP traffic

5.5 Percentage of failed transactions

It gives an insight to a number of requests timed-out or not being served by the server for whatever reason. In case the user sends requests using HTTP, the transaction is complete only if the response is received within the defined time (3-way handshake). Fig. 4(e) shows that when traffic increases during the 60th second, the large number of transactions fail due to congested bottleneck link. This decreases the throughput and also increases the response time, increasing the percentage of transactions that fail. It is directly proportional to the attack strength and can be represented in the equation as

$$\%age\ of\ failed\ trans. = (T_{sent} - R_{recvd} / T_{sent}) \times 100$$

where R_{recvd} is the number of responses received, T_{sent} is the total transactions sent.

5.6 Percentage of link utilization

It is the percentage of bandwidth link used by legitimate requests. It can be computed as

$$\%age\ of\ link\ utilization = (BW_{used} / Total\ BW) \times 100$$

where BW_{used} is the bandwidth used. Fig. 3(e) and 4(f) show the link utilization (LU) in the case of UDP traffic and HTTP traffic, respectively. LU is 100% in case of normal conditions, but as soon as the traffic increases, LU reduces as a lesser number of legitimate requests reach the server. Legitimate traffic follows a congestion control protocol, so when the bandwidth gets clogged, the legitimate packets are dropped to decongest the bandwidth in case of FE (HTTP traffic).

5.7 Average serve rate / Average request rate

It is the ratio of the rate at which the server serves the requests to rate at which the request is generated. The value of 1 indicates that all the requests generated are being served. As the strength of the attack increases, the ratio decreases. Higher the attack strength, the lesser the ratio. Fig. 3(f) and 4(g) show the pattern the ratio follows when under load in case of UDP traffic and HTTP traffic, respectively.

5.8 Legitimate Packet Drop Probability

A packet-level metric that compares the number of legitimate packets dropped with the total number of

Manuscript received May 5, 2025
Manuscript revised May 20, 2025
https://doi.org/10.22937/IJCSNS.2025.25.5.12

legitimate packets in the network. During normal traffic conditions, a number of dropped legitimate packets is negligible, making the ratio value to be zero. With an increase in traffic, the number of dropped legitimate packets increase, thus, increasing the ratio. Fig. 3(g) and 4(h) show the response of the server for packet drop in the case of DDoS and FE, respectively. As can be seen, the performance of the server degrades with an increase in traffic, whether UDP or HTTP.

5.9 CPU utilization

It is the server level metric that quantifies the utilization of CPU of the victim server. As the traffic increases in case of an anomaly, the total utilization of CPU increases though variations are seen for different applications. In the case of UDP attack traffic as in experiment 1, CPU utilization increases with an increase in strength of the attack, as shown in Fig. 3(h). If the attack is layer 7 attack (HTTP), the level of CPU utilization is more as compared to scenario-1, as shown in Fig. 4(i). This is because the request made to some running application has to be processed by a victim server. This leads to higher CPU utilization as compared to the UDP attack, where the requests are just dropped.

6. Conclusions

To understand the impact of traffic anomalies on the system, performance needs to be quantified using efficient and accurate metrics. The paper attempts to define some of the metrics which can be used to compute the performance of the system under surveillance. It evaluates the performance metrics for the network, assuming that the traffic consists mainly of TCP, HTTP, and UDP protocols. The impact metrics have been quantified using throughput, response time, number of active connections, percentage of failed transactions, percentage of link utilization, serve rate/response rate, and legitimate packet drop probability. The experiment was done under two setups using a hybrid testbed. The first setup creates the scenario of a DDoS attack, and the second setup creates the FE effect. Each traffic is generated with varying strengths and has been analysed for the system with a realistic topology using the testbed. The analysis of the impact indicated that the performance of the system degraded with the surge in traffic due to anomalies. As the number of requests increases, the link bandwidth choked, CPU utilization increased, the number of legitimate active connections decreased, legitimate requests reaching the server decreased, thus, increasing the response time. An increase in response time degraded the services. However, it is worth mentioning that there is a correlation between the performance and the hardware (server speed, HDD, the available RAM) used at the server. The response time gets affected by the increase in throughput as well as the number of intermediate points through which the user and server are connected. The user's experience is affected by a change in response time, especially the commercial websites where performance degrades with an increase in response time. As per the literature review done, each network environment uses different metrics for evaluating the performance. There is no standardized methodology or metrics that can be generalized for all types of applications. Also, the metric chosen should be able to reveal the users' scenario when service is denied due to the attack. Such a metric should compare the values with the baseline values defined during normal conditions. There is a need to form a shared repository of the metrics being used for various applications to facilitate future work to generalize the same. Future work is focussed on explicating these metrics with the baseline model and also defining the detection metrics.

Acknowledgments

The authors would like to acknowledge I. K. Gujral Punjab Technical University, Kapurthala, India, for the great support and assistance rendered to carry out this research work.

References

[1] A. Avallone, S. Guadagno, D. Emma, A. Pescape, G. Ventre, "D-ITG- distributed internet traffic generator," Proc. First Intl. Conf. on the Quantitative Evaluation of Systems (QEST '04), IEEE, Computer Society, 2004.

[2] A. Bhandari, A.L. Sangal, K. Kumar, "Performance metrics for defense framework against distributed denial of service attacks," Intl. J. of Netw. Security, vol.6, April 2014.

[3] A. Botta, A. Dainotti, A. Pescapè, "A tool for the generation of realistic network workload for emerging networking scenarios," Comp. Netw. vol.56, no.15, 2012, pp.3531-3547.

[4] C. Bannwart, "Predicting the impact of denial of service attacks," Master thesis submitted to ETH, Zurich, Semantics scholar, 2012.

[5] J. Mirkovic, A. Hussain, S. Fahmy, P. Reiher, R. K. Thomas, "Accurately measuring the denial of service in simulation and testbed experiments," IEEE Trans. on Dependable and Secure Comput. vol.6, no. 2, 2009, pp. 81-95.

[6] J. Mirkovic, A. Hussain, B. Wilson, S. Fahmy, P. Reiher et al, "Towards user-centric metrics for denial-of-service measurement," Proc. 2007 workshop on experimental computer science, San Diego, California, 2007.

[7] J. Mirkovic, S. Fahmy, P. Reiher, R. K. Thomas, "How to test DoS defenses," Proc. 2009 Cybersecurity Applications and Technology Conference for Homeland Security, March 2009, pp. 103–117. DOI: https://doi.org/10.1109/CATCH.2009.23.

[8] J. Mirkovic, S. Wei, A. Hussain, B. Wilson, R. Thomas et al, "DDoS benchmarks and experimenter's workbench for the DETER testbed," 3rd International Conference on Testbeds and Research Infrastructure for the Development of Networks and Communities, Lake Buena Vista, FL, 2007, pp.1-7, DOI: https://doi.org/10.1109/TRIDENTCOM.2007.4444680.
[9] J.T.J. Midgley, "The linux HTTP benchmarking HOWTO," Available at: http://www.xenoclast.org/doc/benchmark/HTTPbenchmarking-HOWTO/node3.html, 2001.
[10] K. Singh, K. K. Saluja, P. Singh, "Impact analysis of application layer DDoS attacks on web services: a simulation study," Intl. J. of Intl. Engg. Informatics, vol.5, no. 1, 2017, pp.80-100, DOI: https://doi.org/10.1504/IJIEI.2017.10003432.
[11] K. Singh, P. Singh, K. Kumar, "User behavior analytics-based classification of application layer HTTP-GET flood attacks. J. Netw. Comput. Appl. Vol. 112, C (June 2018), 97–114. DOI:https://doi.org/10.1016/j.jnca.2018.03.030K.
[12] M. A. Saleh, A. A. Manaf, "A novel protective framework for defeating HTTP-based denial of service and distributed denial of service attacks," Sci. World Journal, 2015, DOI:10.1155/2015/238230, Article ID e238230.
[13] M. Sachdeva, G. Singh, K. Kumar, "An emulation based impact analysis of DDoS attacks on web services during flash events," Proc. 2nd International Conference on Computer and Communication Technology (ICCCT-2011), pp 479–484, September 2011. DOI: https://doi.org/10.1109/ICCCT.2011.6075134.
[14] M. Sachdeva, K. Kumar, G. Singh, K. Singh, "Performance analysis of web service under DDoS attacks," Proc. IEEE Intl. Advance Comput. Conf., IEEE, March 2009, pp. 1002–1007, DOI: https://doi.org/10.1109/IADCC.2009.4809152
[15] R. Chertov, S. Fahmy, N. B. Shroff, "Emulation versus simulation: A case study TCP-targeted denial of service attacks," Proc. 2nd International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities, 2006, TRIDENTCOM 2006, pp. 316–325, DOI: https://doi.org/10.1109/TRIDNT.2006.1649164.
[16] R. S. R. Gade, H. Vellalacheruvu, S. Kumar, "Performance of windows XP, windows vista and Apple's leopard computers under a Denial of Service Attack," Proc. Fourth International Conference on Digital Society, IEEE, February 2010, pp. 188–191, DOI:https://doi.org 10.1109/ICDS.2010.39.
[17] R. Vasudevan, Z. M. Mao, O. Spatscheck, J. Van der Merwe, "MIDAS: An impact scale for DDoS attacks," Proc. 15th IEEE Workshop on Local and Metropolitan Area Networks, Princeton, NJ, 2007, pp. 200-205. DOI: https://doi.org/10.1109/LANMAN.2007.4295999.
[18] S. Behal, K. Kumar, "Measuring the impact of DDoS attacks on web services - A Real-time experimentation," Intl. J. of Comp. Sci. Info. Security (IJCSIS), vol. 14, no. 9, 2016.
[19] S. Bhatia, D. Schmidt, G. Mohay, A. Tickle, "A framework for generating realistic traffic for Distributed Denial-of-Service attacks and flash events," Comp. and Sec. vol.40, 2014, pp. 95-107.
[20] S. Bhatia, "Ensemble-based model for DDoS attack detection and flash event separation," Proc. Future Technologies Conference 2016, San Francisco, United States, 2016, pp. 958-967, DOI: https://doi.org/10.1109/FTC.2016.7821720.
[21] S. Newman, "DDoS attack on Wikipedia site," Available at: https://www.corero.com/blog/934-ddos-attack-on-wikipedia-site-smacks-of-hacktivism.html.
[22] T. Dubendorfer, A. Wagner, B. Plattner, "An economic damage model for large-scale Internet attacks," Proc. 13th IEEE Intl. Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, IEEE Comput. Soc, 2004, pp.223–228, DOI:https://doi.org/10.1109/ENABL.2004.11.
[23] Cisco Prime Network User Guide, 4.2.3, https://www.cisco.com/c/en/us/td/docs/net_mgmt/prime/network/4-2-/user/guide/CiscoPrimeNetwork423UserGuide/ip-pool.pdf.
[24] Apache JMeter, Available online: https://jmeter.apache.org/

Avneet Dhingra received the BTech Degree (Honours) in Computer Science & Engineering from Dr. B. R. Ambedkar National Institute of Technology (NIT), Jalandhar, Punjab (India), in 1997 and ME Degree (Honours) in Software Engineering from Thapar University, Patiala, Punjab (India), in 2002. She has worked as a lecturer at Thapar University, Patiala, Punjab, from 1998 to 2002. She has experience working with an MNC as a Project Manager for a few years. Presently, she is pursuing a Ph.D. in network security under the Department of Computer Science & Engineering, Inder Kumar Gujral Punjab Technical University, Kapurthala, Punjab (India). Her primary research interests are in the areas of software engineering, algorithm designing, web services, network security, particularly the detection of anomalies in network traffic and discriminating them.

Monika Sachdeva received the BTech Degree in Computer Science & Engineering from the National Institute of Technology NIT, Jalandhar, Punjab (India), in 1997, and MS Degree in Software Systems from Birla Institute of Technology and Science BITS, Pilani, Rajasthan (India), in 2002. She received her Ph.D. Degree in Computer Science & Engineering from Guru Nanak Dev University, Amritsar, Punjab (India), in 2012. She is currently an Associate Professor in the Department of Computer Science & Engineering, Punjab Institute of Technology (PIT), Inder Kumar Gujral Punjab Technical University, Kapurthala, Punjab (India). Her primary research interests are in the areas of Network Security and Wireless Sensor Networks with particular emphasis on the timely detection of Distributed Denial of Service attacks in the Wireless Networks and various related network layer attacks in Wireless Sensor Networks. She is also interested in the applications of sensor networks for enhancement in the lifetime of the network, Web Services and Design and Analysis of Algorithms. She has published more than 70 scientific papers in the field of Computer Science & Engineering.