Packet Drops and CPU Load in DDoS Attack Simulation and Mitigation through Dynamic Scaling and Rate Limiting

1st Najem Aldeen Abu Hamdah   
*Computer Engineering Program*  
*University West*Trollhättan, Sweden  
[naab0016@student.hv.se](mailto:naab0016@student.hv.se)

ٌ 2nd Mohamad Anas Mohamad Nour  
*Computer Engineering Program*  
*University West*Göteborg, Sweden  
[momo0022@student.hv.se](mailto:momo0022@student.hv.se)

*Abstract*— The effect of a Distributed Denial of Service (DDoS) attack on a Server, besides implementation of the mitigation methods by using auto scaling and rate limiting. The simulation was developed in Python language by using multiple libraries such as Simpy for the event driven modelling, Matplotlib to convert the events to graphs, Pygame for real time visualisation and Random to generate a random number of the clients and botnet requests. Simulation evaluates the server performance by some parameters such as packet drops and CPU load before- and under the attack. When the CPU load sensor exceeds over 80% of capacity, the auto scaling method and rate limiting should be activated to handle legitimate incoming traffic and stabilise the server. When the number of the request is massive the rate limiting try to restrict the number of requests, resulting in the graphs show that scaling the server and rate limiting reduces the CPU load and the number of dropped packets during the attack this will improve the system resilience and availability for the clients.

Keywords— DDoS, Auto Scaling, Rate Limiting, Simulation, Mitigation, CPU Load, Packet Drops.

# Introduction

All Services in the world seek to achieve the CIA triad in the systems which stand for Confidentiality, Integrity and Availability which represent information security and risk management. Distributed Denial of Service (DDoS) attacks are one of the worrying threads that affect an important part of the triad system which is the availability of the service [1].

The threat actor (Hacker) has a command and control server (C&C), that controls and communicates with multiple compromised or infected devices, called bots or zombies in the cyber world called Botnets are distributed over the world, when the threat actor activates an attack on a targeted server, therefore the botnet start spam requests to the target, it overwhelm the targeted server with a massive number of requests [2], and a DDoS attack depletes the targeted server resources to prevent the legitimate users to access the services. Traditional defence methods like firewalls and intrusion detection systems (IDS) are designed to block malicious traffic, but they fail when the service under a large scale of DDoS attacks.

The mitigation methods used as a defence to reduce the impact of DDoS attacks on the server is called dynamic server scaling and rate limiting, these methods adjust the server capacity and restrict the number of excessive requests based on the traffic load, therefore, it provides additional resources when the load on the server is high, to evaluate the impact of the DDoS attack and server scaling and rate limiting effectiveness.

Simulation is developed by using Simpy to discrete event simulation framework, Matplotlib to convert the events from the console output into graphs along with Pygame for visual representation.

The goal of the study is to demonstrate that server auto scaling and rate limiting, when merged with real time monitoring. Therefore, auto-scaling can mitigate the impact of the DDoS attack and the rate limiting restrict the excessive requests, which reduces the overloaded usage of the server resources to maintain the service availability to clients.

# Technique

## Overview of Simpy:

SimPy is a powerful Python based library used for process-based discrete-event simulation. SimPy is working in our project as a simulator for a queuing system where requests generated from both legitimate users and attackers, are processed by the server. All of these are important therefore Simpy is required to oversee the time and planning of such events [3].

This library is based on Python for modeling and simulating complex systems. Simpy is very useful for testing ideas before people implement them in the real world, but this environment can be controlled. Its tasks include: studying and analyzing systems through time and scheduling, and also providing tools for monitoring and analyzing results. It also contributes to allocating and managing resources in order to simulate the server scaling method, all of which leads to increasing the server capacity based on the CPU load.

There are two ways to simulate:

* Without Mitigation: This method is done when the server capacity is ten concurrent requests, but when a DDoS attack occurs, a large number of requests are sent in a short time in order to overwhelm the server and reduce its work or even stop it completely, which makes the server overloaded with the burdens, it may be forced here to increase the use of the CPU and lost packets. But when the requests exceed its upper limit, the server ignores them or deletes the excess ones spontaneously. All of this works to protect the system from its overload, which maintains the server's performance to the maximum, even if the attacks are severe.
* With Mitigation: This is done when a dynamic scaling mechanism is advanced, then the server capacity increases, and thus the CPU load is greater than the permissible limit. However, when necessary this strategy helps manage large traffic volumes by allocating additional resources. Also, the rate limitation works by the server to process the request rate through control mechanisms, as well as in the face of large loads and facing DDoS attacks.

## Simulation Parameters

It is determined by specifying a second processing time for each request, and the normal request is two seconds, the DDoS rate is 0.03 seconds per request. Also, the request rate in the case of attacks before expansion, the server capacity is 10 requests as a permissible limit, and the CPU load exceeds 80% for the expansion stimulus. However, the increase in capacity is about five additional units to increase the expansion, and the simulation unit at the beginning of the DDoS attack is 150 seconds and the total simulation duration is 300 seconds.

But whenever the request limit exceeds the known limit, the server loses or forgets some requests, which is considered one of the disadvantages of this service. The server works to increase the capacity according to the needs of server.

## Visualisation

The simulation is visualised using Pygame (See Fig. 1), where normal clients represented by blue circles, Botnet (Zombies) are represented by yellow rectangles, dropped requests represented by red circles, server represented by green cylinder in the middle of the screen and the threat actor with can called as black hat hacker in the up top of the center.

A screen shot of a computer generated image

Description automatically generated

Fig. 1: Simulation by using Pygame.

The visual representation helps in understanding how traffic behaves under normal and attack conditions, as well as how the server responds to scaling.

# Results

The output of the simulation is presented in two important parameters such as CPU load for the server over a period of time, and packet dropped over a period of time, these metrics are compared in both scenarios the non-mitigated and the mitigated one.

## CPU Load over time:

* Without mitigation: The CPU load will be stable and the usage of the CPU server low under the first 150 seconds of the simulation, which will represent the normal state of the service. However, when the DDoS attack begins the server experiences effects and the CPU load starts to increase, it reaches approximately 100% because the server becomes overwhelmed by malicious requests, it uses the full capacity of the CPU load. This hinders the effective processing of all incoming requests to the server (see Fig. 2).
* With mitigation: Auto scaling is enabled while the server is under the DDoS attack the CPU load start to increase until it hits the point that the scaling trigger is set to (80%), it will dynamically turn on the scaling, therefore, it allows the server to go back to the stabilises status and allowing it to continue processing request without reaching the full capacity again، thus, the overall performance of the server can be maintained safely through the rate limiting mechanism and proven to be effective for the server operation. When a sudden increase in requests occurs, stability has been achieved by setting the permissible limit for processing requests (See Fig. 3).

A screen shot of a graph

Description automatically generated

Fig. 2: CPU Load without mitigation.

A screen shot of a graph

Description automatically generated

Fig. 3: CPU Load with mitigation.

## Dropped Packets Over Time:

There are two scenarios for the dropped packets over time without mitigation and with mitigation:

* Without mitigation: Before the DDoS attack the server runs normally and there is no packet loss but when the threat actor starts the attack the packet drops highly increased like linear under the attack time, which means the server won't respond or handle the requests from the clients because it’s overwhelmed (See Fig. 4).
* With mitigation: Before the DDoS attack the server will run normally but when the attack starts the server scaling and rate limiting will be enabled therefore the number of the dropped packets is drastically reduced, although at the beginning of the attack, there is an initial increase of the dropped packets, when the auto scaling turned on the number of the drop packets decreases, that means the scaling helps to maintain the availability for the clients, also, preventing redundant requests leads to an attempt to improve the use of resources through a rate-limiting mechanism. Instead of consuming all resources to process useless requests and distribute them well, it has worked to meet the requests of actual users (See Fig. 5).

A graph with a line going up

Description automatically generated

Fig. 4: Dropped packets without mitigation.

A graph with a line going up

Description automatically generated

Fig. 5: Dropped packets with mitigation.

# Discussion

The result from the simulation illustrates the disastrous effect of a DDoS on the server when it has no mitigation for the dynamic scaling techniques therefore the server becomes quickly overwhelmed by the massive number of spam requests coming from the botnet leading to the highly increased CPU load and a large number of the dropped packets. And maintaining the overall security of the server's performance through the rate-limiting mechanism and proving its effectiveness. Maintaining the stability of the server's operation by identifying multiple requests and processing them at a specific time, then a sudden increase in requests and setting the permissible limit during a specific time leads to slowing down or stopping the service. Not consuming large resources helps in achieving a balance between receiving and processing the request. That is true based on the previous studies that show how the DDoS attack can drain the resources of the server and make it unreachable for normal clients.

Nevertheless, server scaling allows the server to able to respond dynamically when the traffic is increasing, stabilizing the CPU load and it will try to reduce the number of dropped packets. That supports the earlier studies which suggest an effective method related to server scaling to mitigate the impact of the DDoS attack on server performance [4].

During auto scaling and rate limiting it will boost server resilience when the DDoS attack happens, The server still encounters a temporary overload before the sensor of the mitigation detect the attack and turn the auto scaling on and rate limiting will try to restract the exeed number of the requests that coming from the Botnets. Therefore the research explains the machine learning and server scaling techniques to discover and reduce the impact of the DDoS attack on the services.

# Future Trends

The future is clear and it will be fulfilled with different methods to support the mitigation against the DDoS attacks on services one of the important methods to use Artificial intelligence (AI), The AI can help the services and the systems to monitor the traffic then analyse it, when it detects any anomaly in the traffic it will respond to cyber threats in real time to defend the service against the DDoS attack [5]. The second important method is the machine learning algorithm it plays a critical role in identifying between normal traffic and anomaly traffic then it will help with analysing the traffic and make the system ready and responding before the attack happens to trigger the auto scaling method faster than the normal this will guarantee to reduce the number of dropped packets [6], [7].

The combination of the mitigation methods like auto scaling, network anomaly and blacklisting help to improve the defence of the system against the DDoS attack.

# Conclusion

This study shows that dynamic scaling plays an important role in defending against DDoS attacks. Our simulation both without- and with the mitigation explained at the scaling improves the system performance by reducing the usage of CPU load and preserving the level of service to make the normal user achieve the service. The presence of mechanisms to control the rate of requests that the server processes, specifically to deal with excessive loads to deal with DDoS. This process transforms the system from a static system into an adaptive dynamic one that makes the system respond to a large number of requests at any moment.

Dynamic scaling will become indispensable in the future of cybersecurity architecture and future research could merge this approach with machine learning to improve the system quality and predict traffic movement then it will enable auto scaling to improve the flexibility of the system when the DDoS attack happens. Therefore, we should discover more defences that can predict any cyber attack on the services to protect the CIA triad in the services.

# References

[1] A. Bakr, A. A. Abd El-Aziz, and H. A. Hefny "A Survey on Mitigation Techniques against DDoS Attacks on Cloud Computing Architecture" Int. J. Adv. Sci. Technol., vol. 28, no. 12, pp. 187-200, 2019. [Online]. Available: <https://www.researchgate.net/publication/336923078_A_Survey_on_Mitigation_Techniques_Against_DDoS_Attacks_on_Cloud_Computing_Architecture>

[2] Cloudflare, "What is a DDoS attack?" Accessed: Oct. 29, 2024. [Online]. Available: <https://www.cloudflare.com/learning/ddos/what-is-a-ddos-attack/>

[3] Team SimPy, “SimPy Discrete event simulation for Python” Accessed: Oct. 29, 2024. [Online]. Available: <https://simpy.readthedocs.io/en/latest/>

[4] Y. Al-Dhuraibi, F. Paraiso, N. Djarallah, and P. Merle, "Elasticity in Cloud Computing: State of the Art and Research Challenges," IEEE Transactions on Services Computing, vol. 11, no. 2, pp. 430-447, Mar.-Apr. 2018. [online]. Available: <https://ieeexplore.ieee.org/document/7937885>

[5] Sophos, "What is AI in cybersecurity?" Accessed: Oct. 29, 2024. [Online]. Available: <https://www.sophos.com/en-us/cybersecurity-explained/ai-in-cybersecurity>

[6] S. Singh, "ML Beginner's Guide to DDoS Attack Detection Model" Labellerr Blog, Feb. 28, 2024. [Online]. Available: <https://www.labellerr.com/blog/ddos-attack-detection/>

[7] A. I. Hassan, E. A. El Reheem, and S. K. Guirguis, "An entropy and machine learning based approach for DDoS attacks detection in software defined networks" Scientific Reports, vol. 14, no. 18159, Aug. 2024. [Online]. Available: <https://www.researchgate.net/publication/382910350_An_entropy_and_machine_learning_based_approach_for_DDoS_attacks_detection_in_software_defined_network>

[8] GitHub repositories “Python Codes” Okt, 2024. [online]: <https://github.com/Discover1998/DDoS_Project_DTA400.git>