**Related Works**

Other researchers have attempted this form of anti-DDoS scaling mechanism before. One method used was labelled as “Scale Inside-Out” where the researchers attempted to mitigate DDoS attacks by attempting to identify the strength availability of a server’s hardware undergoing a DDoS attack and determining how many requests the server can cut down to reduce resource contention. Compared to the goal of my method, this researcher’s method specifies that their method sacrifices a server’s own resources in order to establish and maintain its own mitigation scheme. This method tackles DDoS attacks by cutting as much of the incoming traffic as possible which would come at a cost at an honest user’s expense. On the other hand, my method is focused on fulfilling all requests but focusing on requests from assumed trusted users.

Other flexible methods include an emergency deployment of extra resources to assist in combatting the effects of a DDoS attack. One related research method looked into how cloud computing could take advantage of this opportunity by allowing idle hardware to be deployed as part of a server in an effort to temporarily inflate the scale of the server’s resource power to better handle a DDoS attack. This, unfortunately, requires having extra resources on demand and being able to quickly retailor server software and hardware to service the needs of a user, a situation which mainly Cloud Computing can take advantage of. Compared to my own, this method is designed specifically for handling DDoS attacks for Cloud-Based Networks where an owner already has extra resources available. This situation isn’t applicable for all users. On the other hand, my method is designed with the assumption that the resources being provided by the server are the only resources available. Their method is about increasing the power granted by a server while my method is focused on managing with the resources we already have.

A third solution has a similar goal of establishing control and maintaining a connection with already-existing users. This solution uses a high priority queue and a low priority queue where honest users are given higher priority while malicious users are given lower priority in order to allow high priority users to maintain basic levels of quality of service. This method involves analyzing packet data and calculating the suspicion based on the Harmonic Mean. Packets that exceed the mean are deemed suspicious and put in low priority queue.

This method is focused on identifying what packets and what source IP Addresses are malicious while my method is focused on identifying what sources are legitimate. My method is based on identifying packets based off their sources and prioritizing IP Addresses that were being serviced under the Least Recently Used (LRU) principle used in Operating Systems. This researcher emphasizes the challenge of identifying and reducing the chances of misidentifying users as adversaries. This method prefers false positives. My method, on the other hand, is focused on handling all messages and focusing on priority rather than what messages to discriminate against. My queue system has a goal of servicing all messages even if some requests won’t be handled in just time due to lower priority. While both of our methods and goals are similar, our solutions approach the problem from different angles.

G. Somani, M. S. Gaur, D. Sanghi, M. Conti and M. Rajarajan, "Scale Inside-Out: Rapid Mitigation of Cloud DDoS Attacks," in IEEE Transactions on Dependable and Secure Computing, vol. 15, no. 6, pp. 959-973, 1 Nov.-Dec. 2018, doi: 10.1109/TDSC.2017.2763160.

<https://ieeexplore.ieee.org/abstract/document/8068248>

A. Bremler-Barr, E. Brosh and M. Sides, "DDoS attack on cloud auto-scaling mechanisms," IEEE INFOCOM 2017 - IEEE Conference on Computer Communications, 2017, pp. 1-9, doi: 10.1109/INFOCOM.2017.8057010.

<https://ieeexplore.ieee.org/abstract/document/8057010>

**Introduction**

As our world becomes more interconnected, small businesses, companies, and even certain individuals may opt to provide online services to anyone around the world through the internet. As computer architecture becomes more faster, more accessible, and more powerful, adversarial groups start to have more options to disrupt such activities through DDoS attacks, leading to a virtual arms race between server owners and adversaries to see whose network processing power can outmatch the opponents. While more sophisticated methods of identifying and bypassing DDoS attacks have been theorized and tested, many solutions struggle to the exponentially increasing scale of botnets networks that can overtake even the networks of industrial giants of the industry.

This issue is caused in part because many of these solutions question how to identify and stop an attack rather than how to adapt to such a large-scale attack. Rather than finding out how to defeat a DDoS attack, this research instead aims to find a method to adapt to such an attack such that the core functionality of the defending network can remain in control. The goal of this experiment would determine the possibility whether it could be possible to use the scheduling disciplines of Operating Systems to manage the scale of network queries in such a way that a DDoS attack would not disrupt the server’s control of its network. This research paper will describe the software being used to simulate a network under normal circumstances and a network undergoing a DDoS attack along with the layout of the network itself. Then, the experiment will be conducted showing three experiment trials: One where the network is experiencing normal traffic flow, one where the network is experiencing a DDoS attack, and one where the server is experiencing a DDoS attack but using the described DDoS prevention measures implemented. Data will be gathered and recorded along with an explanation regarding the effectiveness or ineffectiveness of the experiment trials and perceived flaws of the experiment along with improvements that could be utilized to improve the reliability of the experiment.

**Hypothesis**

When handling a sudden influx of incoming requests, the challenge of handling a DDoS attack depends on the scale and resource power from the hardware of the server and the scale of the attack initiated by an attacker. The server faces a challenge of figuring out how to serve all possible users which becomes increasingly difficult the more complex these requests can become. If a server is unable to mitigate or identify a DDoS attack, my proposed solution would be a divide-and-conquer approach using a round-robin queueing system to stress handle a DDoS attack.

What makes this attack different from popular method of load-balancing is that this method would be using techniques from how Operating Systems handle large amounts of tasks for the CPU. Operating Systems use a preemptive scheduling solution to ensure that every tasks gets their share in the CPU on top of a priority system to make sure the most important tasks gets priority. Using this solution, we can use a scheduling algorithm for our servers so that if a server or router becomes unable to handle the large amounts of traffic, it will divide the work into groups and assign priority based on the types of requests being made. Slow, resource-intensive solutions could be given lower priority so that, while it would be inconvenient, these requests wouldn’t be troublesome for the server. Meanwhile, important tasks, such as security checks and network interfacing from an admin, would be given higher priority, allowing users with special privileges to remain in control of the situation. The system would work under a round robin system so that all users, regardless of priority, could still get a chance for server access rather than be blocked out due to a server experiencing overbearing capacity. While it may not stop a DDoS attack, it would allow the server’s security policy to remain in effect even under heavy stress.

**Problem Domain**

DDoS attacks can be considered one of the most widespread attacks in cybersecurity due to the attack being simple to execute and effective against establishments that are unwilling or unable to acquire the necessary hardware and/or software to resist such attacks. Attackers can simply use botnet services to launch a large-scale attack that, if effective, can either be used to halt all activity to that server or leave a server vulnerable to other potential threats due to having to sacrifice its processing power and attention against an overload of requests which could be difficult to discern from an ordinary user or a hijacked machine being told to continually spam requests to the server. One difficult challenge that service providers face is how to differentiate between a user and a malicious attacker as either party would be using the same requests and queries to the server for different purposes. In these scenarios, I would also like to describe the attack methods initiated by an attacker that masquerades their botnets as overloaded network traffic and how server owners are able to detect and reduce the impact of these DDoS attacks, if they are capable of doing so.