Implementing Firewall to Mitigate YOYO Attack on Multi Master Cluster Nodes Using Fail2Ban

Muhammad Faraz Hyder[1], Muhammad Umer Farooq[2*], Mustafa Latif[3], Faizan Razi Khan[4], Abdul Hameed[5] , Noor Qayyum Khan[6] and M. Ahsan Siddiqui[7]

[2,4,5,6,7]*Department of Computer Science and Information Technology, NED University Of Engineering Technology, University Road, Karachi-75270, Pakistan*
[1,3]Department of Software Engineering Technology, NED University Of Engineering Technology, University Road, Karachi-75270, Pakistan

[*]Corresponding Author Email: umer@neduet.edu.pk

Summary
Web technology is evolving with the passage of time, from a single node server to high availability and then in the form of Kubernetes. In recent years, the research community have been trying to provide high availability in the form of multi master cluster with a solid election algorithm. This is helpful in increasing the resources in the form of pods inside the worker node. There are new impact of known DDoS attack, which is utilizing the resources at its peak, known as Yoyo attack. It is kind of burst attack that can utilize CPU and memory to its limit and provide legit visitors with a bad experience. In this research, we tried to mitigate the Yoyo attack by introducing a firewall at load-balancer level to prevent the attack from going to the cluster network.

Keywords:
DDoS attack, high availability, Kubernetes, multi-master, yoyo attack.

1. Introduction

The cloud computing, in the simplest terms, means storing and accessing data and applications over the Internet instead of a computer's hard drive. Cloud computing in daily life is enormous; Cloud computing is a scalable, cost-effective and the well distribution platform for providing services over the Internet. However, security issues exist, especially for businesses that keep moving their content across various cloud services. Maintaining a good cloud security infrastructure allows businesses to gain the proper benefits of cloud computing. Confidentiality, integrity, and availability, as well as non-repudiation, are all concerns with cloud security. In this paper, we will address avoiding insecurity in availability by protecting cluster; availability means accessing information in a certain time with the correct format.

Cluster is one of the most important component in Kubernetes. Clustering enables an application's functionalities to be faster or ensures that data is available for faster transmission. A Kubernetes cluster is a collection of nodes that are used to run containerized applications. It must have at least one cluster if organizations use Kubernetes for their application. Some companies employ a few huge clusters, while others have a lot of smaller clusters. Fig. 1 shows a chart distribution of cluster count and nodes per cluster in this section [1].

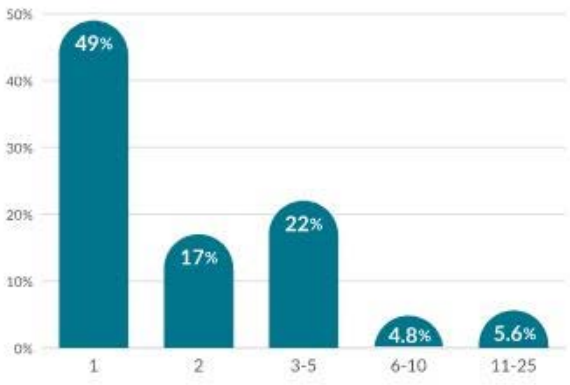

Fig. 1 Numbers of clusters [1]

However, a cluster is a key component of Kubernetes, but security challenges also remain. As it is known that a cluster is a key component for Kubernetes, the security of a cluster is very critical. One of the most well-known attack on cloud is called DDoS attack. DDoS attacks typically take the shape of long-term, high-volume traffic floods that gradually increase in volume, reach to peak and then come down slowly or quickly.

A new attack pattern has appeared in recent years. Burst attacks, also known as hit-and-run DDoS, apply a series of short bursts of high-volume attacks that occur at random

Manuscript received February 5, 2023
Manuscript revised February 20, 2023
https://doi.org/10.22937/IJCSNS.2023.23.2.13

intervals. A single short burst can last a few seconds, while a burst attack strategy can last hours or even days. The transfer rate of these attacks can reach hundreds of gigabits per second.

A new kind of burst attack has been reported in recent years called the YoYo attack [2]. The YoYo attack targeted the cloud's auto-scaling mechanism for virtual machines and evokes the autoscaling process used by the cloud servers to control the workload.

The attack starts by sending a big wave of DDoS traffic to the target. Cloud autoscaler start adding more resources to cope with the workload. However, the reactive autoscaler does not immediately scale back down. Instead, it waits for a short period of time to ensure a lower traffic volume. After the autoscaling process has scaled back down, the attacker repeats the same approach. The attacker's primary goal is not just to damage your site and take it offline but also to cause major financial harm. The cyber attackers design a scheme to take a large amount of money from the user in exchange for the delivered resources. Many hackers employ the Yo-Yo scheme because of its great advantages.

YOYO Attack We follow the notation of the YoYo attack and implement it in the Kubernetes model. Consider a Kubernetes cluster that consists of autoscaling with identical service machines behind a load balancer. Requests arrive with an average rate of r requests per unit time, and the load balancer distributes them to Np Pods, which can be divided b/t Nn Nodes in the steady state. Let R be the number of Pods that can be in a single node. The number of Pods in a Node is determined by the Node machine type; a stronger machine can have a better quality of Pods.The YoYo attack consists of n cycles, and each cycle duration is T, comprised of an on-attack period, denoted as ton, and an off-attack period, denoted as toff. Thus T = ton+toff . We define the power of the attack because of the extra load on the cluster. Let k be the power of the attack and r the average request rate per second in a steady state. We assume that in the on-attack period, the attacker adds fake requests k.

The contribution of this document summarize as follow:

• Implement Fail2ban Firewall effectively to cater to Yoyo attack on the load balancer and provide an intelligent layer of security to a Kubernetes cluster.

• Benchmark testing of the implemented setup and attaining the maximum throughput with minimized server/application errors.

The rest of the document is prepared as follows. Section II explore related work. Section III is related to the model security, In Section IV, a methodology has been discussed. The experimental setup of this research has been discussed in section V. In section VI, results have been discussed. And finally, section VII conclusion has been discussed.

Table 1 Related Works

Name	***Approach***	***Result***
Yo-Yo Detection [2]	Using XG-Boost classifier Algorithm [2]	The XGBoost algorithm achieves the highest accuracy score on the testing dataset, with a score of 95 percent , whereas the CNN+LSTM, Logistic Regression, Decision Tree, and Random Forest algorithms earn accuracy scores of less than 90 percent [2].
Yo-Yo Mitigation [4]	A trust-based adversarial scanner delaying [4]	The 80 percent malicious ratio after the 1400 s of simulation time highlights that TASD can finally detect most of the adversarial users [4].
Mitigate Attack [5]	Clusterized Firewall [5]	Numerous simulations confirm analysis method, attack rate rises, the reaction time of the cloud firewall increases ,rate of packet arrival exceeds the rate of firewall service and response time accelerates [5].
EDoS Detection [6]	Entropy-Based Eco- nomic Denial of Sustainability Detection [6]	Entropy variation happening as malicious attack increases entropy value decreasing as well on graph [6].
Yo-Yo Detection in Container [7]	Entropy-Based Eco- nomic Denial of Sustainability Detection [6] Examine HAproxy Logs [7]	Collect IP addresses requesting at stage of scale up and at scale down from logs that HAproxy Produce. Both IP address array com- pare if one IP address occurrence from scale up is maximum but not present in scaled down array then IP address consider to attacker's IP. [7]

2. Related Work

DDoS attack has gradually increased recent years so it is very critical to detect and mitigate this attack. There are many technique to detect and mitigate DDoS attack and related work have been proposed by researcher, some of the techniques which have been published in literature is listed in Table 1. In [2] d Ben David et al explored the effect of the Yo-Yo attack in the cloud auto-scaling system. XG-Boost algorithm approach also presented to detect Yo-Yo attack. Xu et al [4] proposed a TASD(Trust-based Adversarial Scanner Delaying) strategy embrace authorize users by assigning them a trust value. They also cheat the attacker by manipulating request response times. In this paper [5], liu et al. suggested a clustered firewall cloud framework. Extensive simulation confirms their analytical model, indicating that the response time of the cloud firewall grows as the attack rate rises. It is also concluded

that the cost of a firewall is acceptable in comparison to the expense of a cloud defender. In [6], Monge et al. introduced an entropy-based approach for detecting EDoS attacks in cloud environment. They observed per-client CPU times on the webserver by analyzing the entropy levels, which showed a decrement when malicious requests were initiated by the hacker node. In this paper [7], the authors develop a mathematical model based on queueing theory to simulate a low-rate DDoS attack scenario for container-based cloud environment, Collects IP addresses from HAproxy [8] logs to identify attacker's IP.

3. Background

The traditional architecture of Kubernetes is based on two parts, i.e. i) Control Plane ii) Worker Plane

i) Control Plane: It is the master node that controls the worker plane nodes. It has the heart of the Kubernetes cluster, i.e. etcd and Kube-schedular, which ensures where we can place the pod on the worker nodes[9]. Client tool such as kubectl uses kube-apiserver to take the request to the control plane, and it is also responsible for taking the request within the cluster.

ii) Worker Plane: All the pods are created at worker nodes by default obeying the taints and tolerance rules. Kubelet is basically the captain of the worker ship that takes care of all the operational aspects of the worker node with the help of kube-proxy, who is responsible for maintaining the node networking [10]. Fig. 2 shows the architecture of Kubernetes.

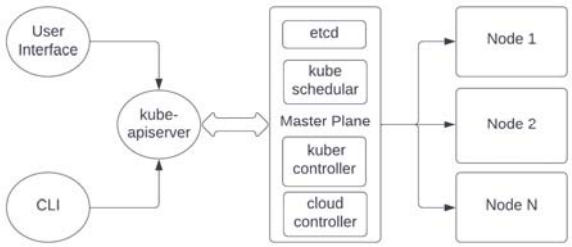

Fig. 2 Kubernetes Architecture.

3.1 Consensus Algorithm: Raft

It is the procedure in which each node in the network agrees upon the same state of the machine. It is usually used in Blockchain technology [11].

It is an easy algorithm to understand. It resolves the issue of consensus in fault tolerance. There are three roles involved in this procedure [12].

a) Leader

Considering the current setup, there are three master nodes. They will wait for a random time and initiate an election. It will cast a vote to itself and wait for others to vote. It will be elected as a leader if it gets majority votes. It will send a heart beat that the leader is alive.

b) Candidate

If follower nodes do not get the response within heart beat time. It will mark itself as a candidate node and starts the election after a random time.

c) Follower

These nodes follow the instruction provided by the leader node.

3.2 Role of External ETCD

In this experiment, we used multi mater nodes connected together through external ETCD. The configuration is defined inside the load balancer.

The technique is used in cloud security, which divides security strategies used in cloud-native systems into four tiers named "The 4Cs of Cloud-native Security" [3]. Kubernetes Security is based on the 4C's of cloud native security: Cloud, Cluster, Container, and Code, but we are going to discuss cluster security only.

3.3 Cluster Security

When discussing cluster security, the main focus is on Kubernetes because it is the most widely used container orchestration platform today; however, the security principles mentioned can also be applied to alternative solutions. There are three primary cluster elements that need to be considered.

1) Components of Cluster

This has to do with securing the components that make up your cluster (master node). When it comes to cluster security, things like managing API server access and blocking direct access to etcd, Kubernetes, and primary datastore, should first come to mind.

2) Cluster Services

: Make that your clusters have proper authentication and authorization, that communication is encrypted using Transport Layer Security (TLS) and protect sensitive information using secrets.

3) Cluster Networking

This has to do with the right allocation of ports to let containers, pods, and services to communicate with one another. It is necessary to deploy the Kubernetes networking paradigm securely utilizing a Container Network Interface (CNI) that allows users to control pod traffic[13-14].

4. Methodology

Fail2Ban[15] is a third-party application that checks logs in the background.Fail2ban works by dynamically changing firewall rules to block addresses that have attempted to log in unsuccessfully a particular number of times. It may be set up to send you an e-mail notice or to utilize IP Tables to automatically ban an offending IP address if too many attempts are made in a given amount of time. To set up Fail2Ban, it is required to generate a few files that tell the Fail2Ban what to look for in your logs and what to do if it finds one. In our methodology, we use Fail2ban to filter out IP addresses hitting on ports 80 and 443, i.e. HTTP and HTTPS. Fig. 3 shows the flowchart of the proposed scheme. Fig. 4 shows the block diagram of the proposed scheme.

Filters can be placed in a folder in the Fail2Ban by default setup. These filters include strings in them that can be searched against your logs. You can use as many filters as you wish to search your logs for various forms of suspicious traffic. The jail configuration file, the second file, applies the filters to rules like how often an error is allowed to occur and what action should be taken if that threshold is violated. Algorithm 1 shows the technique of detection and mitigation of the YOYO attack.

4.1. Threat Model

The threat model we are following is that our attack is outside the Kubernetes network and could be located anywhere in the world.

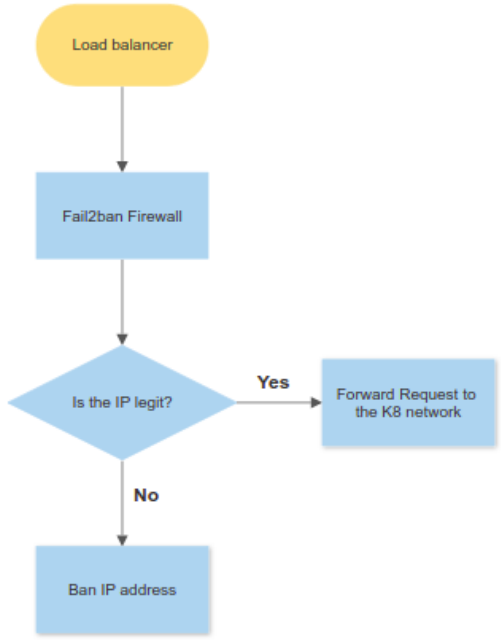

Fig. 3 Flowchart of the proposed scheme.

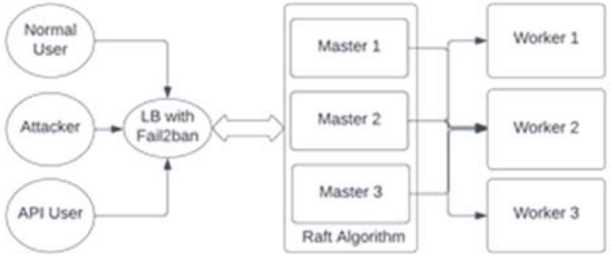

Fig. 4 Block diagram of proposed scheme.

Algorithm 1

```
Procedure ()
initialize traffic to zero
while (loadbalancer is running)
input the traffic
if )the received IP address hits 5 times in last 10 seconds at fail2ban firewall)
add IP Address in Fail2ban jail
else
pass request to the cluster load balancer is down
```

Table 2: Configuration Details

No	*Name*	*Value*
1	Infrastructure	Google Compute Engine
2	Type	Standard VMs
3	Operating System	Ubuntu 18.04.6 LTS
4	Kernel Version	5.4.0-1062-gcp
5	Number of Nodes	7
6	Load Balancer	Nginx
7	Master Nodes	3
8	Worker Nodes	3
9	VPC	10.128.0.0/16
10	Node Networking	Flannel
11	Container Runtime Engine	Docker
12	Kubernetes Version	1.23.2

5. Experiment Setup

5.1 Implementation of External ETCD

We have setup a cluster of 3 master nodes. The role of ETCD is to maintain stability by providing one node as the leader and the other two as followers. Table 2 shows the configuration details of the experimental setup. Fig 5-8 indicates the implementation details of ETCD, HPA Fail2ban configurations

Fig. 5 External ETCD working.

5.2 Implementation of HPA

The role of Horizontal Pods Autoscaling is significant, it ensures that a sufficient number of replicas are available in the cluster network to serve the traffic.

Fig. 6 Role of HPA.

The logic is based on the usage of resources such as CPU and Memory. In our experiment we configured CPU to 50 and Memory to 80 percent respectively.

5.3 Implementation Of Firewall

The biggest challenge that we face in Kubernetes is its security. For that purpose, we used fail2ban as a firewall on loadbalancer. Nginx as load balancer has all three IP on port 6443. Implementing a firewall on the load balancer will help us reducing the impact of an attack on the actual cluster network.

On the same load balancer, we created a jail in Fail2ban as nginx-http-auth. In this jail, we have the configuration to monitor the traffic on log file /var/log/nginx/access.log on the find time value of 10s with a maximum try of 5 on ports 80,443 and 6443 to block the traffic.

Fig. 7 Fail2ban Jail.

This will provide a decision based security on the Kubernetes blocking the suspicious hits within the time span of 10 secs. Assuming that no legit IP will hit the Kubernetes cluster more than 5 times in 10 seconds.

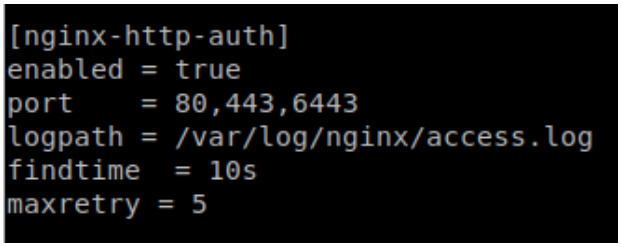

Fig. 8 Jail Configuration.

We used k6.io for benchmarking. K6.io is a cloud service that is open source providing scenario based load testing for developers and QA. In this experiment, we will try the efficiency of the K8 with and without implementing the firewall to prove our idea.

Note: In this experiment, N is the number of virtual users and T is the time duration for the test. It will try to keep the users from 0 to N within first one minute hold it for the at least T-2 minutes. In the last minute of the test, it will decrease virtual users gradually.

6. Results and Analysis

Scenario 1 Normal Traffic Without Firewall: Testing our setup using 20 virtual users for 5 minutes 30 seconds duration.

Fig. 9 Scenario 1 Normal Traffic Without Firewall.

In Fig. 9, the P95 response time is around 31ms means that 95 percent of the requests took around 31ms or less. There is no HTTP failure occurred it means all requests were served perfectly fine with no downtime.

Scenario 1 Normal Increased Traffic Without Firewall: Testing our setup using 100 virtual users for 5 minutes 30 seconds duration.

Fig. 10 Scenario 1 Normal Increased Traffic Without Firewall.

In Fig. 10, the P95 response time is less than the Test1 without changing any other parameter. It shows the stability of the cluster.

Scenario 2 Yoyo Attack With Firewall: Testing our setup using 20 virtual users for 5 minutes 30 seconds duration.

Fig. 11 Scenario 2 Yoyo Attack with Firewall.

Fig. 11 shows that the firewall allowed five attempts to the certain IP within the span of 10 seconds and then started blocking all the following requests. Four hundred seventy-seven were the total requests, 472 were blocked, and only five were allowed.

Scenario 2 Yoyo Attack [16] with Increased Intensity with Firewall: Testing our setup using 100 virtual users for 5 minutes 30 seconds duration.

Fig. 12 Scenario 2 Yoyo Attack with Increased Intensity with Firewall.

In Fig 12, increasing the number of virtual users will impact the same, blocking all requests except those allowed by the firewall before taking the pre-defined decision.

Scenario 2 Yoyo Attack with Intense Traffic from Different Regions With Firewall: Testing our setup using 100 virtual users for 5 minutes 30 seconds duration.

Fig. 13 Scenario 2 Yoyo Attack with Intense Traffic from Different Regions with Firewall.

In Fig. 13, we initiated DDoS from 5 different locations, ensuring that all hit the cluster network. 2418 out of 2448 were blocked (98.77 percent) that is defining the efficiency of our firewall.

7. Conclusion

In this paper, we proposed the implementation of a traditional firewall technology Fail2ban, on a multi master Kubernetes cluster network, ensuring the mitigation of the Yoyo attack. The proposed approach not only prevents our network from malicious requests but also saves our infrastructure resources from dumped requests.

References

[1] Sysdig,The fifth annual Sysdig Cloud-Native Security and Usage Re- port.2022
[2] Ben David et al. Kubernetes Autoscaling: YoYo Attack Vulnerability and Mitigation,2021
[3] Kubernetes, The 4C's of Cloud Native security.[online].Available:https://kubernetes.io/docs/concepts/security/overview/the-4c-s-of-cloud-native-security
[4] Xiaoqiong Xu et al,Towards Yo-Yo attack mitigation in cloud auto-scaling mechanism.2019,

[online]Avaiable:https://www.sciencedirect.com/science/article/pii/S2352864819301440
[5] Liu et al, A clusterized firewall framework for cloud computing. In Pro- ceedings of the 2014 IEEE International Conference on Communications (ICC), Sydney, Australia, 10–14 June 2014; pp. 3788–3793.
[6] Monge et al, Entropy-Based Economic Denial of Sustain- ability Detection. Entropy 2017, 19, 649.[online] Available: https://doi.org/10.3390/e19120649
[7] Viktor Danielsen, Detecting Yo-Yo DoS attack in acontainer-based environment,2021,[online] Available:https://hdl.handle.net/11250/2774518
[8] Zeebaree, Subhi Rafeeq, Karwan Jacksi, and Rizgar R. Zebari. "Impact analysis of SYN flood DDoS attack on HAProxy and NLB cluster-based web servers." Indones. J. Electr. Eng. Comput. Sci 19.1 (2020): 510-517.
[9] Carrión, Carmen. "Kubernetes scheduling: Taxonomy, ongoing issues and challenges." ACM Computing Surveys (CSUR) (2022).
[10] Mondal, Subrota Kumar, et al. "Kubernetes in IT administration and serverless computing: An empirical study and research challenges." The Journal of Supercomputing 78.2 (2022): 2937-2987.
[11] Netto, Hylson, et al. "Incorporating the Raft consensus protocol in containers managed by Kubernetes: An evaluation." International Journal of Parallel, Emergent and Distributed Systems 35.4 (2020): 433-453.
[12] Xiong, Huanliang, et al. "Research on Progress of Blockchain Consensus Algorithm: A Review on Recent Progress of Blockchain Consensus Algorithms." Future Internet 14.2 (2022): 47.
[13] Minna, Francesco, et al. "Understanding the security implications of kubernetes networking." IEEE Security & Privacy 19.05 (2021): 46-56.
[14] Shamim, Shazibul Islam. "Mitigating security attacks in kubernetes manifests for security best practices violation." Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. 2021.
[15] Ylli, Enkli, Igli Tafa, and Flavia Marku. "Examining the Server's Load Average When Trying to Attack the Firewall." Proceedings of Sixth International Congress on Information and Communication Technology. Springer, Singapore, 2022.
[16] Mostamand Kashi, Meraj. Implementation of an approach to mitigate Yo-Yo attack in cloud auto-scaling mechanism. MS thesis. OsloMet-storbyuniversitetet, 2022.