1. Introduction

Overview Work:

- **Target:**
  - The goal of your project is to implement a WordPress School Management System and simulate DoS (Denial of Service) attacks on the system, then defend against these attacks. The system will include user roles like students, teachers, and admins, with specific access controls and security features.
- **Difficulty:**
  - The primary challenge is in configuring and securing the WordPress system, ensuring that it is robust enough to withstand DoS attacks. Additionally, managing user roles, server configuration, and ensuring the system remains performant under load presents a technical challenge.
- **Techniques Needed:**
  - **WordPress Installation & Configuration:** Setting up Apache, MySQL, and PHP for WordPress.
  - **Network Security:** Implementing firewalls and DoS prevention techniques.
  - **Attack Simulation & Defense:** Using tools like **LOIC** or custom scripts to simulate DoS attacks and evaluate defenses.
  - **System Monitoring:** Observing system behavior under attack using tools like **htop**, **netstat**, and logging systems.
- **Results:**
  - Successfully simulating DoS attacks on the WordPress School Management System and evaluating the effectiveness of security measures.
  - Ensuring the system can recover from attacks, demonstrating the firewall's and server's response capabilities.

2. Design

Paperwork on Your Whole System:

- **System Components:**
  - **WordPress:** A content management system used to manage student, teacher, and administrative data.
  - **Web Server (Apache):** To serve WordPress content.
  - **Database Server (MySQL):** To store user data, course data, and other information.
  - **Firewall:** To prevent malicious access and defend against attacks like DoS.
  - **SSH Server:** To allow secure administration by sysadmins.
- **System Flow:**
  - Users interact with the WordPress front-end.
  - Admins and sysadmins have specific access to administrative pages and server configurations.
  - Firewall inspects and filters incoming traffic.
  - Sysadmins access the system remotely via SSH to perform maintenance.

User-case Diagram for System Features Description:

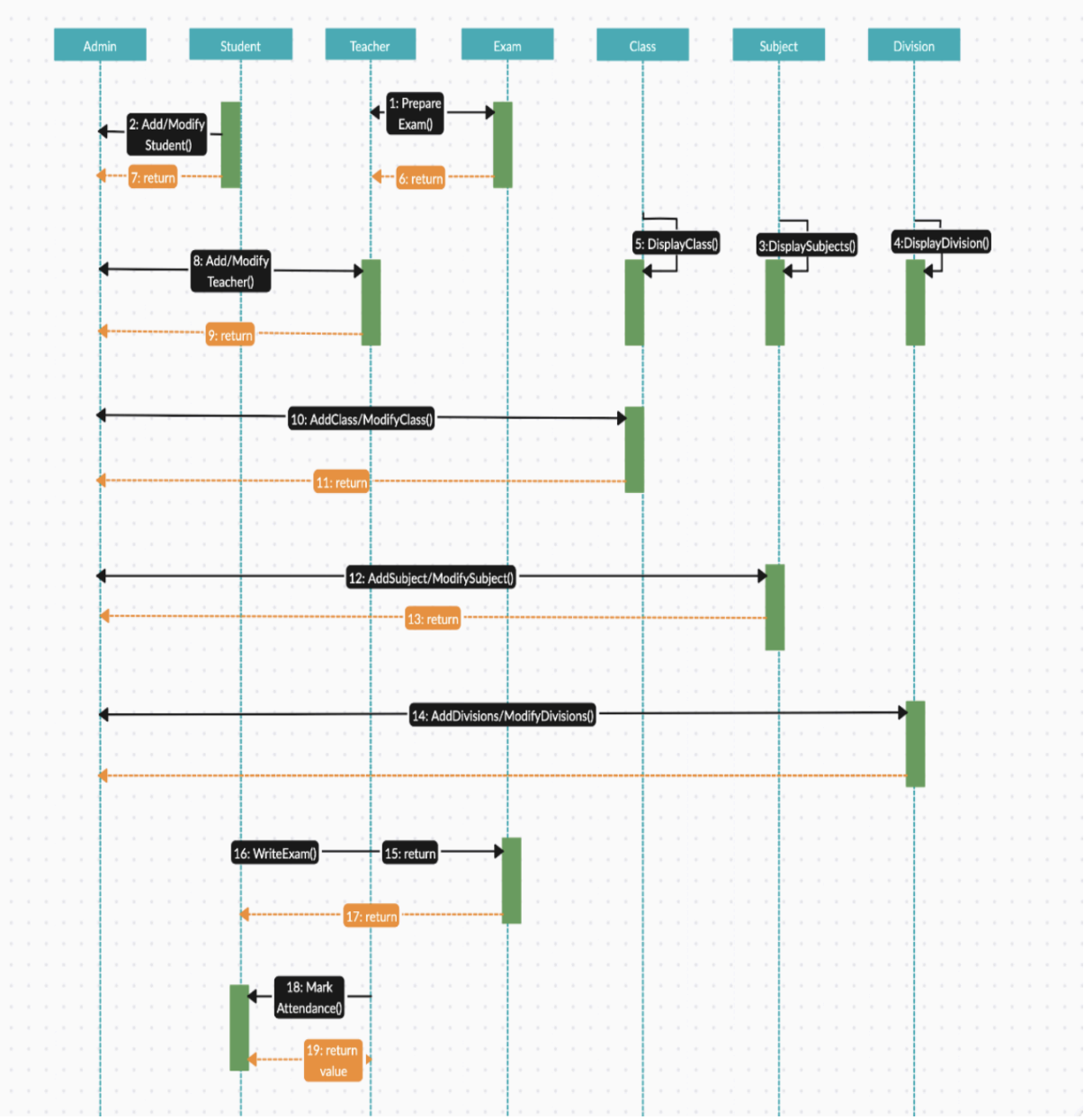
- **Users:** Define user roles such as **Admin**, **Sysadmin**, **Teacher**, and **Student**.
- **Actions:** Describe key actions each user can perform, such as login, adding/removing students or courses, viewing grades, etc.

Architecture Diagram for System Description:

Sequence Diagram for Action Flow Description:

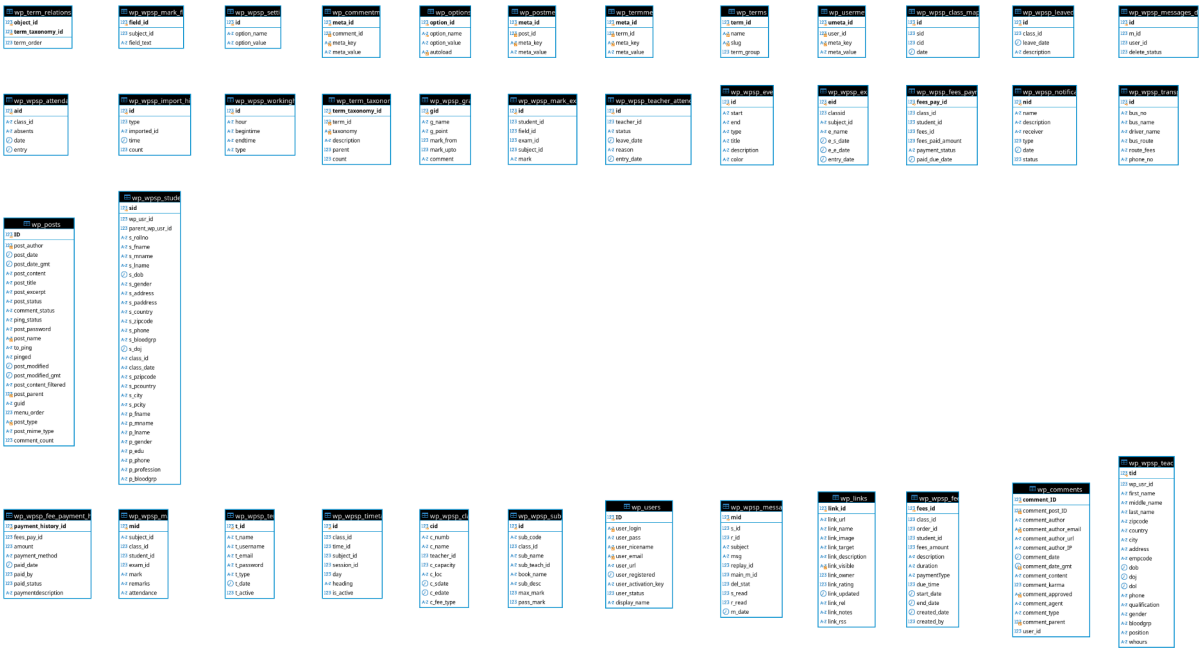
- Illustrate the flow of actions such as:
  - This sequence diagram shows the flow of actions in a school management system. The **Admin** manages **Students**, **Teachers**, **Classes**, and **Divisions**. The **Teacher** prepares exams and modifies **Subjects**. **Class** and **Subject** display relevant data. The **Exam** is written, and **Attendance** is marked. Each action is followed by a return indicating task

completion.



Database Diagram for Data Structure Description:

- Key tables to include:
  - **Users Table:** Contains user data (ID, username, role, password, etc.)
  - **Courses Table:** Information about courses (ID, name, teacher, credits)
  - **Grades Table:** Student grades for each course.
  - **Logs Table:** Security and system logs, especially during attack simulations.



System Performance Metrics

1. Performance

- **Response Time**: Time from request to response.
  Response Time = Time of Response - Time of Request
- **Throughput**: Number of requests handled per second.
  Throughput = Total Requests / Time Period

3. Resource Utilization

- **CPU Utilization**: CPU usage percentage.
  CPU Utilization = (CPU Time Used / Total CPU Time) * 100

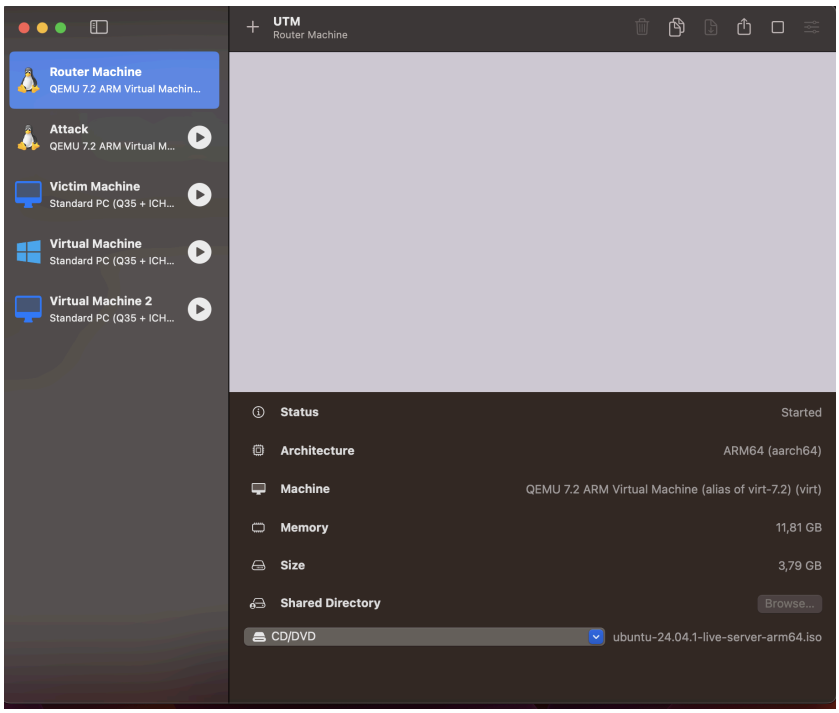
4. WordPress Specific

- **Page Load Time**: Time taken to load a page.
  Page Load Time = End Time - Start Time

Implementation.
I. Detail tools used for each components

1. Platform

1.Virtual Machine
set up



2. HTTP Server - Apache
Tools Used:

- **Apache HTTP Server**
  - **Apache** is a widely used web server, known for its modularity and extensibility, ideal for dynamic content served via PHP.

Configure Apache for WordPress

Create an Apache site for WordPress. Create /etc/apache2/sites-available/wordpress.conf with following lines:

```
<VirtualHost *:80>
    DocumentRoot /srv/www/wordpress
    <Directory /srv/www/wordpress>
        Options FollowSymLinks
        AllowOverride Limit Options FileInfo
```

```
        DirectoryIndex index.php
        Require all granted
    </Directory>
    <Directory /srv/www/wordpress/wp-content>
        Options FollowSymLinks
        Require all granted
    </Directory>
</VirtualHost>
```

Enable the site with:

sudo a2ensite wordpress
Enable URL rewriting with:

sudo a2enmod rewrite

Disable the default “It Works” site with:

sudo a2dissite 000-default

Or, instead of disabling the “it works” page, you may edit our configuration file to add a hostname that the WordPress installation will respond to requests for. This hostname must be mapped to your box somehow, e.g. via DNS, or edits to the client systems’ /etc/hosts file (on Windows the equivalent is C:\Windows\System32\drivers\etc\hosts). Add ServerName as below:

```
<VirtualHost *:80>
    ServerName hostname.example.com
    ... # the rest of the VHost configuration
</VirtualHost>
```

Finally, reload apache2 to apply all these changes:

sudo service apache2 reload

Tools Used:

MySQL is an open-source relational database management system (RDBMS) that is widely used for managing and storing data in a structured way. It is known for its fast performance, reliability, and ease of use. **MySQL** is often used in web applications, including **WordPress**, where it stores all content, user information, comments, settings, and more.
Use Case:

- **WordPress Database**: MySQL is used to store WordPress data

a. Configure database:

```
sudo mysql -u root
mysql> CREATE DATABASE wordpress;
mysql> CREATE USER wordpress@localhost IDENTIFIED BY 'hung';
mysql> GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,DROP,ALTER
    -> ON wordpress.*
    -> TO wordpress@localhost;
mysql> FLUSH PRIVILEGES;
```

b. **Configure WordPress to connect to the database**
Now, let's configure WordPress to use this database. First, copy the sample configuration file to wp-config.php:

```
sudo -u www-data cp /srv/www/wordpress/wp-config-sample.php
/srv/www/wordpress/wp-config.php
```

Next, set the database credentials in the configuration file (*do not* replace database_name_here or username_here in the commands below. *Do* replace <your-password> with your database password.):

```
sudo -u www-data sed -i 's/database_name_here/wordpress/' /srv/www/wordpress/wp-config.php
sudo -u www-data sed -i 's/username_here/wordpress/' /srv/www/wordpress/wp-config.php
sudo -u www-data sed -i 's/password_here/<your-password>/' /srv/www/wordpress/wp-config.php
```

Finally, in a terminal session open the configuration file in nano:

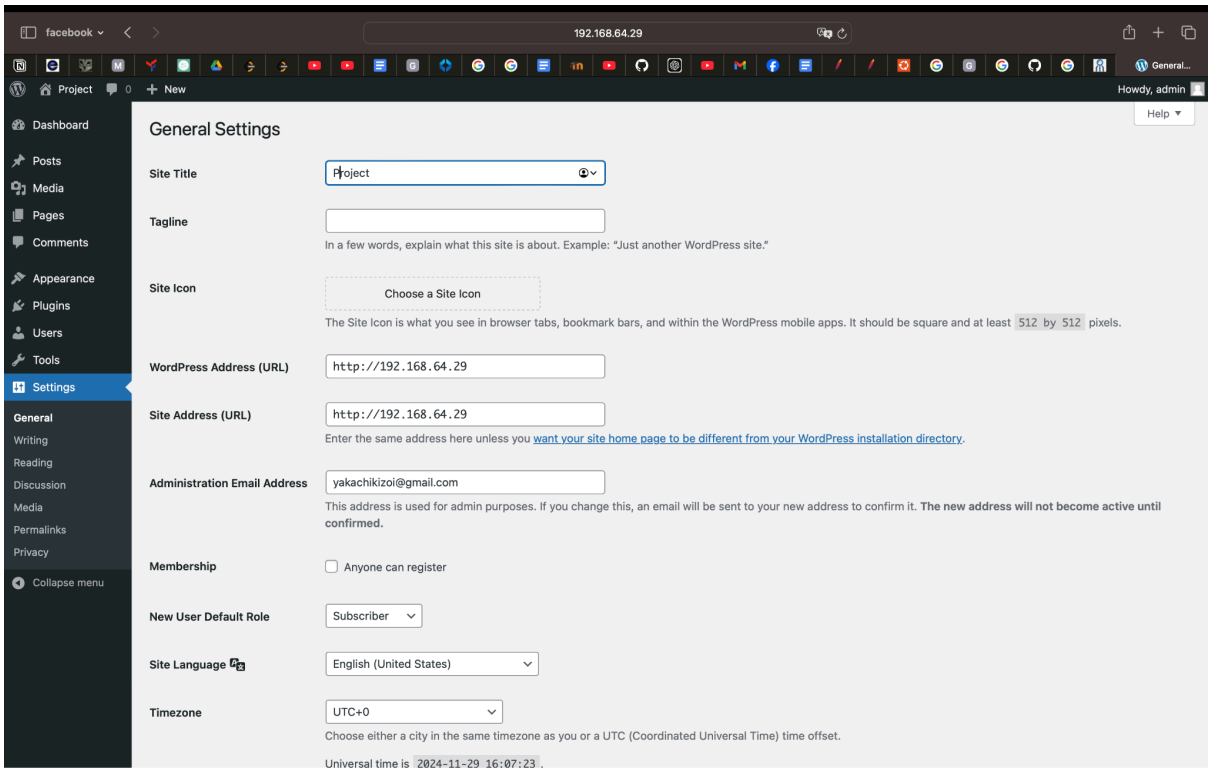
sudo -u www-data nano /srv/www/wordpress/wp-config.php

Find the following:

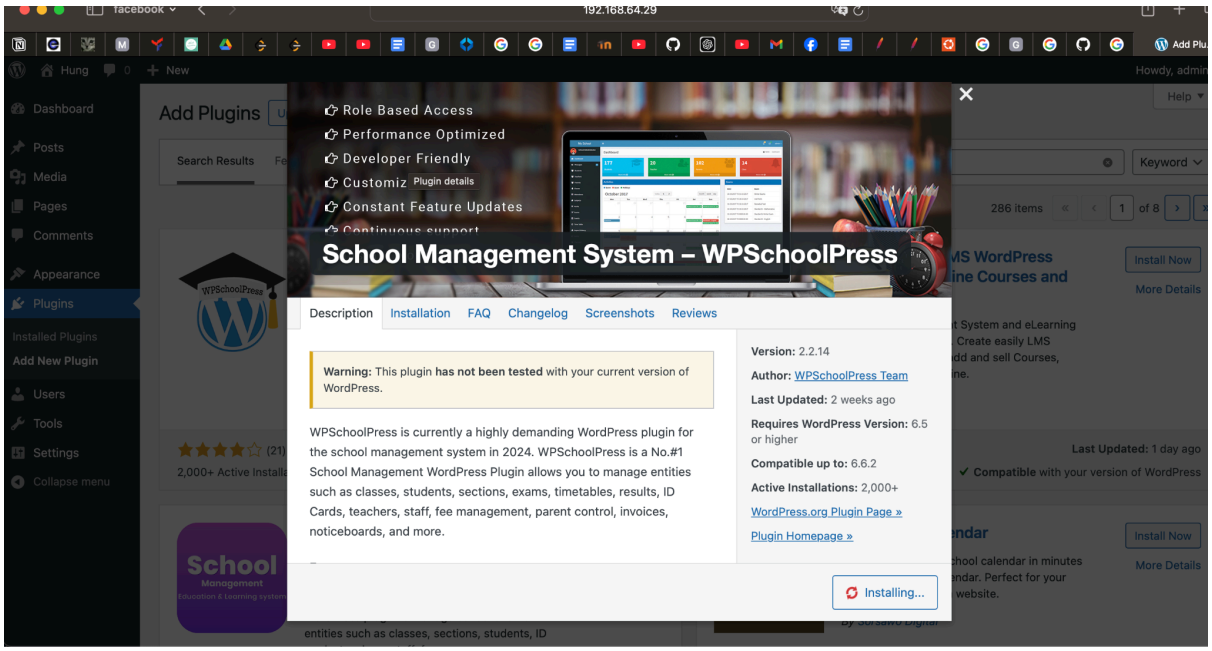
```
define( 'AUTH_KEY',         'put your unique phrase here' );
define( 'SECURE_AUTH_KEY',  'put your unique phrase here' );
define( 'LOGGED_IN_KEY',    'put your unique phrase here' );
define( 'NONCE_KEY',        'put your unique phrase here' );
define( 'AUTH_SALT',        'put your unique phrase here' );
define( 'SECURE_AUTH_SALT', 'put your unique phrase here' );
define( 'LOGGED_IN_SALT',   'put your unique phrase here' );
define( 'NONCE_SALT',       'put your unique phrase here' );
```

Then replace it with the content of WordPress.org Salt Generator.

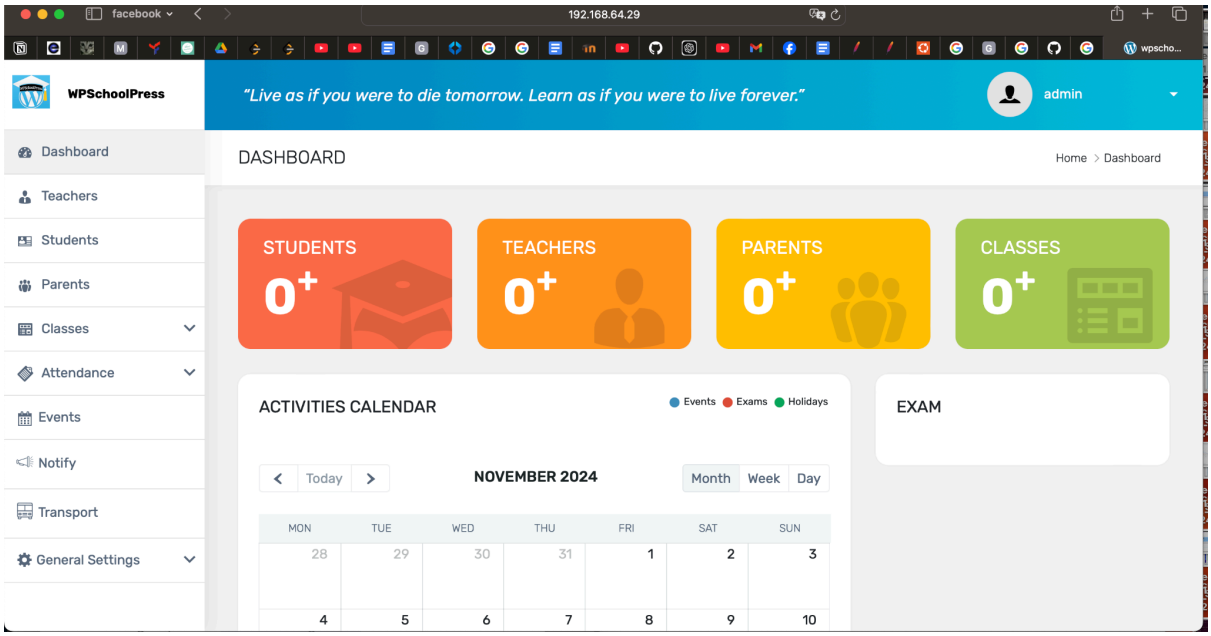
1. **Configure WordPress** **front-end**



install plugin WP school management

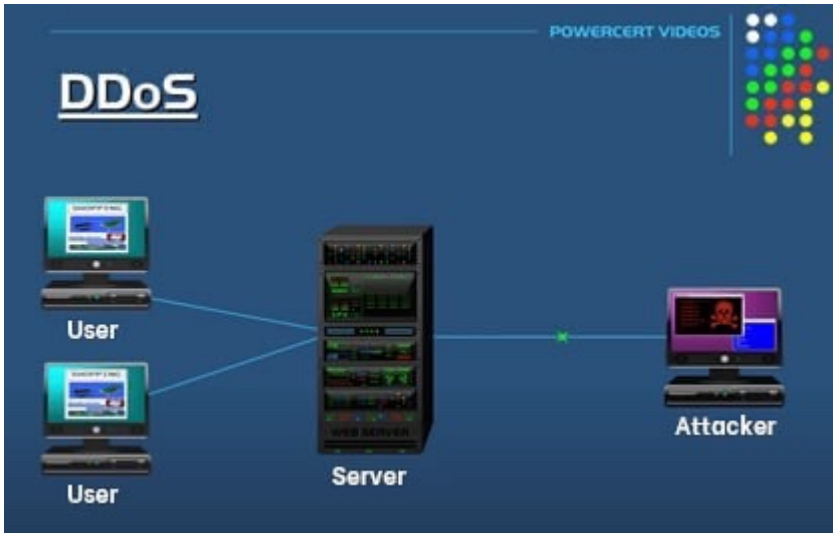


Open dashboard for admin



Attack and prevention

1. Network simulation



We are setting up 1 virtual model to be able to attack the ddos server and use a firewall to prevent this. Simulation has 2 users, 1 attacker and 1 server.

- First, we are setting up network router of user (ubuntu):

```
hung@hung:~$ sudo ip link set dev enp0s1 down
[sudo] password for hung:
hung@hung:~$ sudo ip addr add 10.10.1.1/24 dev enp0s1
hung@hung:~$ sudo ip link set dev enp0s1 up
hung@hung:~$ sudo ip link set dev enp0s2 down
hung@hung:~$ sudo ip addr add 172.16.1.1/24 dev enp0s2
hung@hung:~$ sudo ip link set dev enp0s2 up
```

- Next, we are setting up attack machine (kali linux):

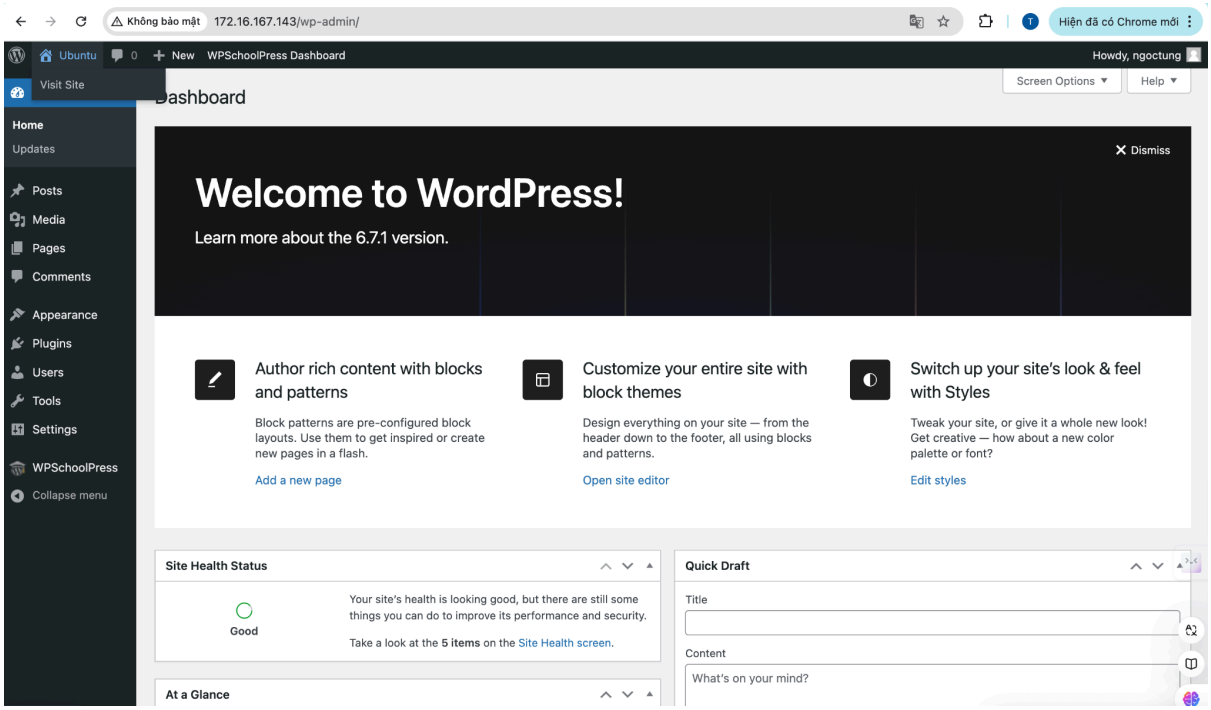
```
┌──(ngoctung㉿ngoctung)-[~]
└─$ sudo ip link set dev eth0 down

┌──(ngoctung㉿ngoctung)-[~]
└─$ sudo ip addr add 10.10.1.2/24 dev eth0

┌──(ngoctung㉿ngoctung)-[~]
└─$ sudo ip link set dev eth0 up

┌──(ngoctung㉿ngoctung)-[~]
└─$ sudo ip route add default via 10.10.1.1

┌──(ngoctung㉿ngoctung)-[~]
└─$ sudo ip r
default via 10.10.1.1 dev eth0
default via 192.168.64.1 dev eth2 proto dhcp src 192.168.64.15 metric 100
10.10.1.0/24 dev eth0 proto kernel scope link src 10.10.1.2
192.168.64.0/24 dev eth2 proto kernel scope link src 192.168.64.15 metric 100
```

- finally, we are setting up network server (ubuntu server):

```
ngoctung@ngoctung:~$ sudo ip link set dev enp0s1 down
[sudo] password for ngoctung:
ngoctung@ngoctung:~$ sudo ip addr add 172.16.1.2/24 dev enp0s1
ngoctung@ngoctung:~$ sudo ip link set dev enp0s1 up
ngoctung@ngoctung:~$ sudo ip route add default via 172.16.1.1
```

2. Attack

- Before the attack and not the firewall on wordpress.



We execute the command above on the Ubuntu server to monitor the number of packets captured, those received by the filter, and those dropped by the kernel.

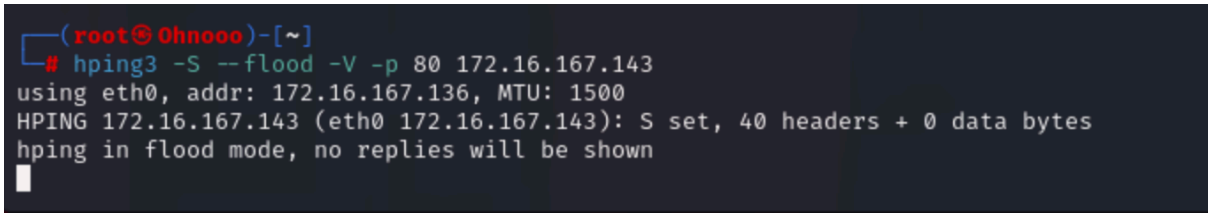
```
ngoctung@ngoctung:~$ sudo tcpdump -n "tcp[tcpflags] & (tcp-syn) != 0 and tcp[tcpflags] & (tcp-ack) == 0"
[sudo] password for ngoctung:
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on ens160, link-type EN10MB (Ethernet), snapshot length 262144 bytes
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel
```

- We are using hping3 for an attack server.
  sudo apt install hping3

```
┌──(root㉿Ohnooo)-[~]
└─# apt install hping3
hping3 is already the newest version (3.a2.ds2-11~kali1).
Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 2123
```

- hping3 is an open-source command-line tool used primarily to send custom TCP/IP packets for network testing, security testing, or analysis.It can be used to send packets to servers or network devices with different parameters, such as IP addresses, ports, and packet types.
- We are using a machine kali attack on a server. ip server 172.16.167.143

sudo hping3 -S –flood -V -p 80 172.16.167.143



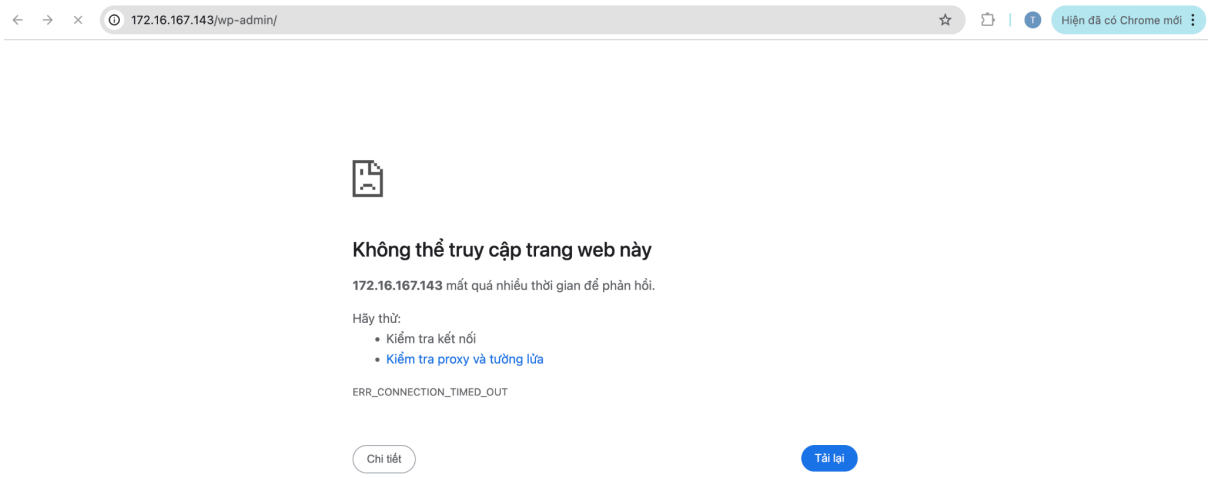
-S: This option tells hping3 to send TCP packets with the SYN flag set. The SYN flag is used in the TCP 3-way handshake to initiate a connection.

--flood: This option makes hping3 send packets as fast as possible, simulating a flood attack. This is typically used in Denial of Service (DoS) attacks, especially a SYN flood, where the target server is overwhelmed with an excessive amount of SYN requests.

-V: This enables verbose mode, which provides detailed information about the packets being sent. It helps the user to see the packets' progress and responses from the target machine.

-p 80: This specifies the destination port to which the packets will be sent. In this case, port 80 (HTTP) is used. The target web server will receive the SYN packets on this port.

- After an attack on the server. The server just can’t be accessed anymore.



- The number of packets has increased a lot, causing the system to be overloaded, leading to the inability to load.

```
14:44:01.879084 IP 172.16.167.136.50770 > 172.16.167.143.80: Flags [S], seq 1822648421, win 512, length 0
14:44:01.879084 IP 172.16.167.136.50771 > 172.16.167.143.80: Flags [S], seq 72825575, win 512, length 0
14:44:01.879084 IP 172.16.167.136.50772 > 172.16.167.143.80: Flags [S], seq 861432063, win 512, length 0
14:44:01.879084 IP 172.16.167.136.50773 > 172.16.167.143.80: Flags [S], seq 1499862957, win 512, length 0
14:44:01.879084 IP 172.16.167.136.50774 > 172.16.167.143.80: Flags [S], seq 1692814560, win 512, length 0
14:44:01.879125 IP 172.16.167.136.50775 > 172.16.167.143.80: Flags [S], seq 416335838, win 512, length 0
14:44:01.879125 IP 172.16.167.136.50776 > 172.16.167.143.80: Flags [S], seq 681968156, win 512, length 0
14:44:01.879125 IP 172.16.167.136.50777 > 172.16.167.143.80: Flags [S], seq 1131334669, win 512, length 0
14:44:01.879125 IP 172.16.167.136.50778 > 172.16.167.143.80: Flags [S], seq 1732842020, win 512, length 0
14:44:01.879125 IP 172.16.167.136.50779 > 172.16.167.143.80: Flags [S], seq 1158479864, win 512, length 0
14:44:01.879125 IP 172.16.167.136.50780 > 172.16.167.143.80: Flags [S], seq 1377624414, win 512, length 0
14:44:01.879125 IP 172.16.167.136.50781 > 172.16.167.143.80: Flags [S], seq 183129883, win 512, length 0
14:44:01.879125 IP 172.16.167.136.50782 > 172.16.167.143.80: Flags [S], seq 907357214, win 512, length 0
14:44:01.923476 IP 172.16.167.136.50783 > 172.16.167.143.80: Flags [S], seq 1372196383, win 512, length 0
14:44:01.923476 IP 172.16.167.136.50784 > 172.16.167.143.80: Flags [S], seq 1183734734, win 512, length 0
14:44:01.923476 IP 172.16.167.136.50785 > 172.16.167.143.80: Flags [S], seq 2036052516, win 512, length 0
14:44:01.923476 IP 172.16.167.136.50786 > 172.16.167.143.80: Flags [S], seq 1327099726, win 512, length 0
14:44:01.923476 IP 172.16.167.136.50787 > 172.16.167.143.80: Flags [S], seq 643245725, win 512, length 0
14:44:01.923476 IP 172.16.167.136.50788 > 172.16.167.143.80: Flags [S], seq 2057047937, win 512, length 0
14:44:01.923476 IP 172.16.167.136.50789 > 172.16.167.143.80: Flags [S], seq 1805166000, win 512, length 0
14:44:01.923476 IP 172.16.167.136.50790 > 172.16.167.143.80: Flags [S], seq 583948300, win 512, length 0
14:44:01.923518 IP 172.16.167.136.50791 > 172.16.167.143.80: Flags [S], seq 1493728312, win 512, length 0
14:44:01.923518 IP 172.16.167.136.50792 > 172.16.167.143.80: Flags [S], seq 339270583, win 512, length 0
14:44:01.923518 IP 172.16.167.136.50793 > 172.16.167.143.80: Flags [S], seq 1787253174, win 512, length 0
14:44:01.923518 IP 172.16.167.136.50794 > 172.16.167.143.80: Flags [S], seq 669927874, win 512, length 0
14:44:01.923518 IP 172.16.167.136.50795 > 172.16.167.143.80: Flags [S], seq 317621306, win 512, length 0
14:44:01.923518 IP 172.16.167.136.50796 > 172.16.167.143.80: Flags [S], seq 376373198, win 512, length 0
14:44:01.923518 IP 172.16.167.136.50797 > 172.16.167.143.80: Flags [S], seq 553919670, win 512, length 0
14:44:01.923518 IP 172.16.167.136.50798 > 172.16.167.143.80: Flags [S], seq 1013282598, win 512, length 0
14:44:01.952574 IP 172.16.167.136.50799 > 172.16.167.143.80: Flags [S], seq 40714300, win 512, length 0
14:44:01.952574 IP 172.16.167.136.50800 > 172.16.167.143.80: Flags [S], seq 637518254, win 512, length 0
14:44:01.952574 IP 172.16.167.136.50801 > 172.16.167.143.80: Flags [S], seq 331375586, win 512, length 0
14:44:01.952574 IP 172.16.167.136.50802 > 172.16.167.143.80: Flags [S], seq 1342595518, win 512, length 0
14:44:01.952574 IP 172.16.167.136.50803 > 172.16.167.143.80: Flags [S], seq 237642817, win 512, length 0
14:44:01.952574 IP 172.16.167.136.50804 > 172.16.167.143.80: Flags [S], seq 663864089, win 512, length 0
14:44:01.952574 IP 172.16.167.136.50805 > 172.16.167.143.80: Flags [S], seq 19106598, win 512, length 0
14:44:01.952574 IP 172.16.167.136.50806 > 172.16.167.143.80: Flags [S], seq 867189829, win 512, length 0
14:44:01.952613 IP 172.16.167.136.50807 > 172.16.167.143.80: Flags [S], seq 1742598291, win 512, length 0
14:44:01.952613 IP 172.16.167.136.50808 > 172.16.167.143.80: Flags [S], seq 1629183005, win 512, length 0
14:44:01.952613 IP 172.16.167.136.50809 > 172.16.167.143.80: Flags [S], seq 112302343, win 512, length 0
14:44:01.952613 IP 172.16.167.136.50810 > 172.16.167.143.80: Flags [S], seq 497919817, win 512, length 0
14:44:01.952613 IP 172.16.167.136.50811 > 172.16.167.143.80: Flags [S], seq 2016176195, win 512, length 0
14:44:01.952613 IP 172.16.167.136.50812 > 172.16.167.143.80: Flags [S], seq 555234063, win 512, length 0
14:44:01.952613 IP 172.16.167.136.50813 > 172.16.167.143.80: Flags [S], seq 1317417932, win 512, length 0
14:44:01.952613 IP 172.16.167.136.50814 > 172.16.167.143.80: Flags [S], seq 1379956889, win 512, length 0
^C
2320 packets captured
2416 packets received by filter
0 packets dropped by kernel
```

3. Configuration firewall

1. Installing and Configuring UFW (Uncomplicated Firewall)

Install UFW on the WordPress server:

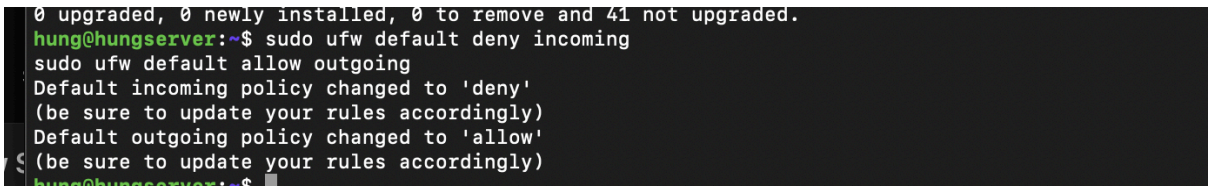
sudo apt update

sudo apt install ufw

Set default UFW policies:

sudo ufw default deny incoming

sudo ufw default allow outgoing



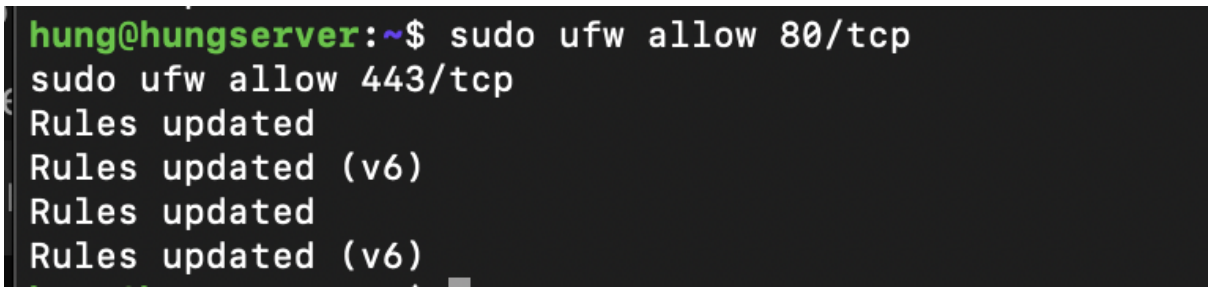
2. Firewall Rules Based on User Access Control

Allow HTTP and HTTPS for All Users (Normal Page Access)

This rule allows access to the WordPress public pages:

sudo ufw allow 80/tcp # Allow HTTP

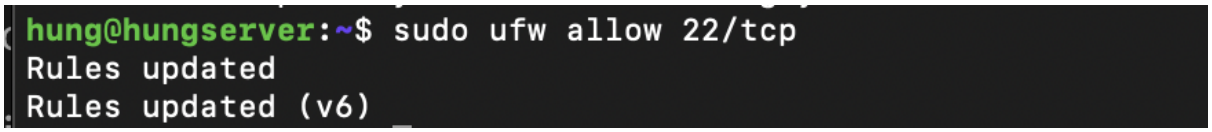
sudo ufw allow 443/tcp # Allow HTTPS



Allow SSH Access Only for Sysadmin (hung)

Restrict SSH access to the Sysadmin (hung) user with rate limiting to prevent DoS attacks:

sudo ufw limit 22/tcp



- Alternatively, configure SSH to only allow the hung and admin users:

Open SSH configuration:

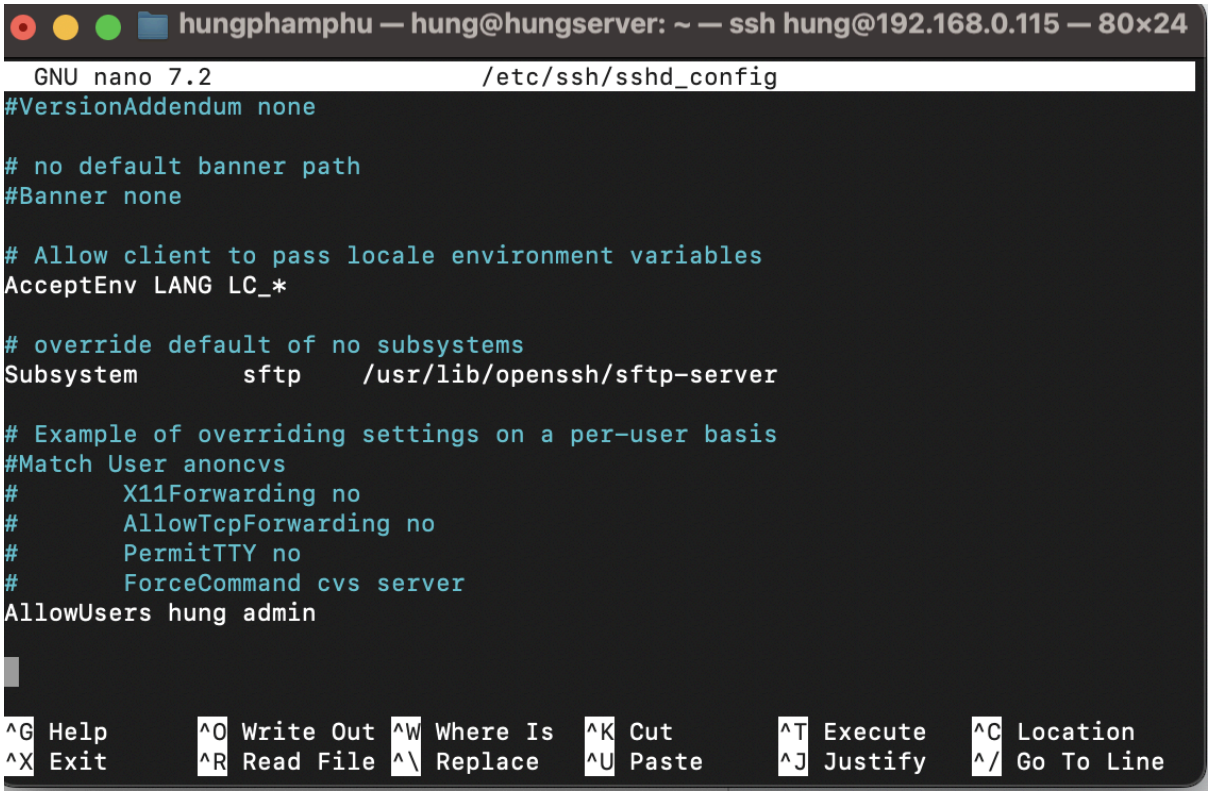
sudo nano /etc/ssh/sshd_config

1.

Add the following line to limit SSH access:

plaintext

AllowUsers hung admin



2.

Restart SSH to apply changes:
sudo systemctl restart ssh

3.

3. Blocking Unauthorized Access and Invalid Packets

Drop Invalid Packets to prevent unauthorized access attempts:
sudo iptables -A INPUT -m conntrack --ctstate INVALID -j DROP

Prevent DoS Attacks by rate limiting HTTP and SSH connections:
sudo ufw limit 80/tcp
sudo ufw limit 22/tcp

```
hung@hungserver:~$ sudo ufw limit 80/tcp
sudo ufw limit 22/tcp
Rule updated
Rule updated (v6)
Skipping adding existing rule
Skipping adding existing rule (v6)
```

4. Saving and Enabling UFW

Enable UFW to apply these rules:
sudo ufw enable

```
hung@hungserver:~$ sudo ufw enable
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
hung@hungserver:~$
```

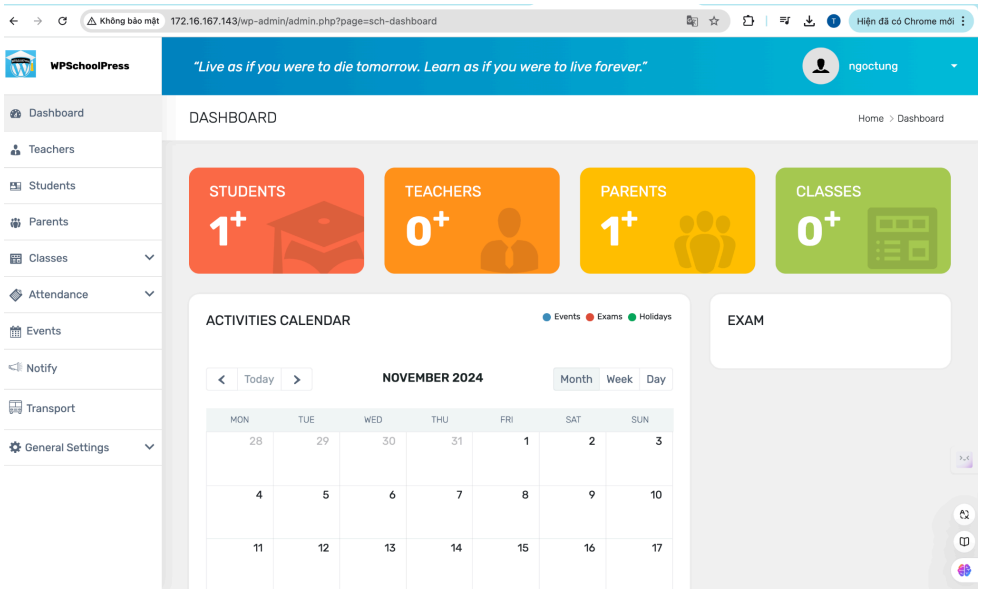
Check the status to verify the configuration:
sudo ufw status verbose

```
hung@hungserver:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
22/tcp                     LIMIT IN    Anywhere
80/tcp                     LIMIT IN    Anywhere
443/tcp                    ALLOW IN    Anywhere
22/tcp (v6)                LIMIT IN    Anywhere (v6)
80/tcp (v6)                LIMIT IN    Anywhere (v6)
443/tcp (v6)               ALLOW IN    Anywhere (v6)
```

- test after setup configuration firewall

```
ngoctung@ngoctung:~$ cd /var/log/
ngoctung@ngoctung:/var/log$ ls
README                  faillog                 vmware-network.2.log
alternatives.log        installer               vmware-network.3.log
apache2                 journal                 vmware-network.4.log
apport.log              kern.log                vmware-network.5.log
apt                     landscape               vmware-network.6.log
auth.log                lastlog                 vmware-network.7.log
bootstrap.log           mysql                   vmware-network.8.log
btmp                    private                 vmware-network.9.log
cloud-init-output.log   samba                   vmware-network.log
cloud-init.log          syslog                  vmware-vmsvc-root.log
dist-upgrade            sysstat                 vmware-vmtoolsd-root.log
dmesg                   ufw.log                 wtmp
dpkg.log                unattended-upgrades
fail2ban.log            vmware-network.1.log
ngoctung@ngoctung:/var/log$ tail -100f ufw.log
2024-11-27T15:09:07.788245+00:00 ngoctung kernel: [UFW LIMIT BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=4
7128 PROTO=TCP SPT=2926 DPT=80 WINDOW=512 RES=0x00 SYN URGP=0
2024-11-27T15:09:07.788735+00:00 ngoctung kernel: [UFW LIMIT BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=3
4974 PROTO=TCP SPT=2927 DPT=80 WINDOW=512 RES=0x00 SYN URGP=0
2024-11-27T15:09:14.679240+00:00 ngoctung kernel: [UFW LIMIT BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=6
3175 PROTO=TCP SPT=18324 DPT=80 WINDOW=512 RES=0x00 SYN URGP=0
2024-11-27T15:09:47.948535+00:00 ngoctung kernel: [UFW LIMIT BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=5
9795 PROTO=TCP SPT=1951 DPT=443 WINDOW=512 RES=0x00 SYN URGP=0
2024-11-27T15:09:54.788598+00:00 ngoctung kernel: [UFW LIMIT BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=6
0671 PROTO=TCP SPT=9695 DPT=443 WINDOW=512 RES=0x00 SYN URGP=0
2024-11-27T15:10:14.669118+00:00 ngoctung kernel: [UFW LIMIT BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=3
5142 PROTO=TCP SPT=33055 DPT=443 WINDOW=512 RES=0x00 SYN URGP=0
2024-11-27T15:17:01.921345+00:00 ngoctung kernel: [UFW BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=27379 P
ROTO=TCP SPT=3004 DPT=143 WINDOW=512 RES=0x00 SYN URGP=0
2024-11-27T15:17:01.922302+00:00 ngoctung kernel: [UFW BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=45862 P
ROTO=TCP SPT=3005 DPT=143 WINDOW=512 RES=0x00 SYN URGP=0
2024-11-27T15:17:01.922309+00:00 ngoctung kernel: [UFW BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=14191 P
ROTO=TCP SPT=3006 DPT=143 WINDOW=512 RES=0x00 SYN URGP=0
2024-11-27T15:17:01.922312+00:00 ngoctung kernel: [UFW BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=36241 P
ROTO=TCP SPT=3007 DPT=143 WINDOW=512 RES=0x00 SYN URGP=0
2024-11-27T15:17:01.922313+00:00 ngoctung kernel: [UFW BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=48009 P
ROTO=TCP SPT=3008 DPT=143 WINDOW=512 RES=0x00 SYN URGP=0
2024-11-27T15:17:01.922328+00:00 ngoctung kernel: [UFW BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=50964 P
ROTO=TCP SPT=3009 DPT=143 WINDOW=512 RES=0x00 SYN URGP=0
2024-11-27T15:17:01.922331+00:00 ngoctung kernel: [UFW BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=60762 P
ROTO=TCP SPT=3010 DPT=143 WINDOW=512 RES=0x00 SYN URGP=0
2024-11-27T15:17:01.922332+00:00 ngoctung kernel: [UFW BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=13889 P
ROTO=TCP SPT=3011 DPT=143 WINDOW=512 RES=0x00 SYN URGP=0
2024-11-27T15:17:01.922333+00:00 ngoctung kernel: [UFW BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=8519 PR
OTO=TCP SPT=3012 DPT=143 WINDOW=512 RES=0x00 SYN URGP=0
2024-11-27T15:17:01.922334+00:00 ngoctung kernel: [UFW BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=55568 P
ROTO=TCP SPT=3013 DPT=143 WINDOW=512 RES=0x00 SYN URGP=0
2024-11-27T15:17:21.939082+00:00 ngoctung kernel: [UFW BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=30045 P
ROTO=TCP SPT=12587 DPT=50 WINDOW=512 RES=0x00 SYN URGP=0
2024-11-27T15:20:43.499894+00:00 ngoctung kernel: [UFW BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=65073 P
ROTO=TCP SPT=2286 DPT=0 WINDOW=512 RES=0x00 URGP=0
2024-11-27T15:20:43.501415+00:00 ngoctung kernel: [UFW BLOCK] IN=ens160 OUT= MAC=00:0c:29:90:02:8e:00:0c:29:2e:ee:fc:08:00 SRC=172.16.167.136 DST=172.16.167.143 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=17386 P
ROTO=TCP SPT=2287 DPT=0 WINDOW=512 RES=0x00 URGP=0
```

We can see that the firewall has successfully blocked the attacker's attempt and users can access normally.




```
ngoctung@ngoctung:~$ sudo tcpdump -n "tcp[tcpflags] & (tcp-syn) != 0 and tcp[tcpflags] & (tcp-ack) == 0"
[sudo] password for ngoctung:
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on ens160, link-type EN10MB (Ethernet), snapshot length 262144 bytes
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel
```

We execute the command above on the Ubuntu server to monitor the number of packets captured, those received by the filter, and those dropped by the kernel.

1. Observation Techniques Using Datadog

- **Performance Metrics**:

```
ngoctung@ngoctung:~$ curl -w "Time of Request: %{time_starttransfer}\nTime of Response: %{time_total}\n" -o /dev/null -s 172.16.167.143
Time of Request: 0.109886
Time of Response: 0.109940
```

- **Resource Utilization**:
  - **Infrastructure Monitoring**: Monitor **CPU** in real-time using Datadog's infrastructure monitoring.
  -

- **WordPress Performance**:
  - **RUM**: Measure **page load time** and **database query performance** through Datadog Real User Monitoring and APM.

2. Evaluation Strategies

- **Experiment Time**:
  - **Baseline**: Test under normal conditions (low traffic) to gather initial metrics.
  - **Peak Load**: Perform load testing during high traffic or stress conditions to test scalability.
- **Experiment Duration**:
  - **Short-Term**: 1-2 hours for quick performance checks.
  - **Long-Term**: 24-48 hours for stability and resource monitoring.
- **Observation Frequency**:
  - **Continuous Monitoring**: Real-time monitoring of system health.
  - **Snapshot Testing**: Periodic checks (e.g., every 30 minutes) to identify anomalies.

- **Metrics Evaluation**:
  - Compare response times, throughput, and resource utilization against baseline performance.
  - Track security events and evaluate attack detection and prevention effectiveness.
  - Monitor CPU/memory to ensure system stability during peak loads.

Results.

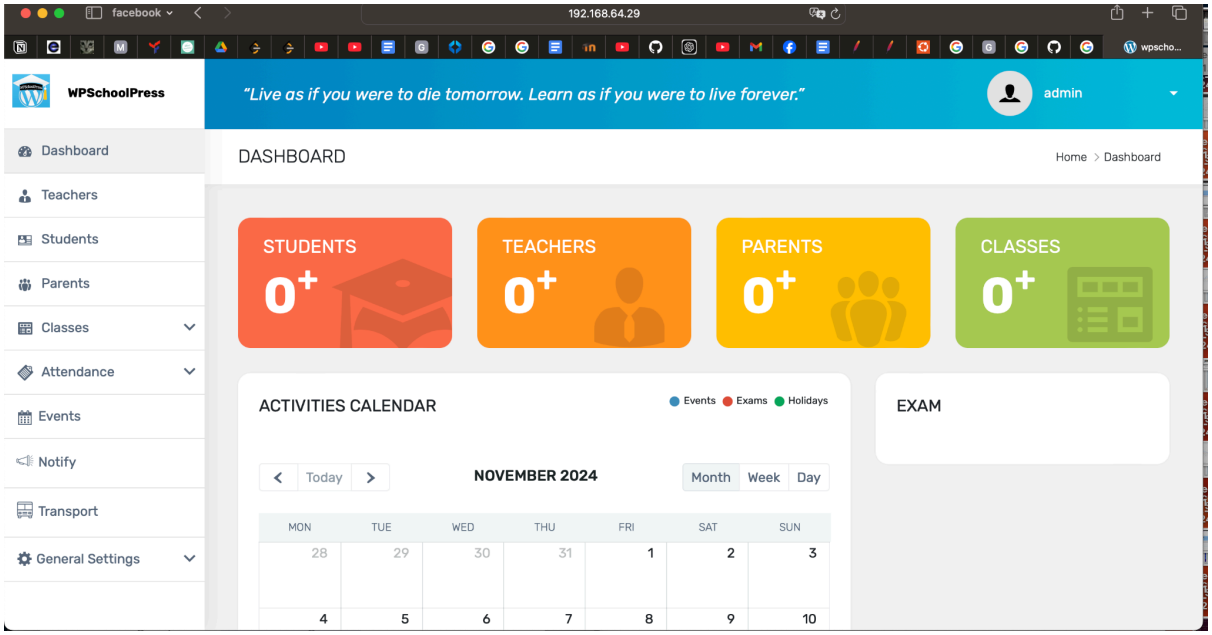
1. Proof of Work

This section should demonstrate that the WordPress system is functional, and the **DoS attack simulations** and **defense measures** are properly implemented and tested.

- **WordPress Installation and Configuration**: Show that WordPress is running on the server with proper configuration for database, PHP, and web server (Apache or Nginx).
- **DoS Attack Simulation**: Demonstrate the **attack** using tools like **hping3** on your **Kali Linux** machine, targeting the **WordPress server**, and show the result (server becomes unreachable).
- **Firewall Configuration**: After applying **UFW firewall rules**, show that the attack is blocked and normal user access is restored.

2. Image Capture for Front-End Configuration

- **WordPress admin dashboard** showing how it's set up, including the **plugin** used for school management.



3. Table for Quantitative Evaluation or API Implementation Results

Provide **quantitative metrics** comparing the system's performance **before** and **after** the attack and defense configuration:

Metric	Before Attack	During Attack	After Attack (with Firewall)
Response Time	0.01s	10s	0.02s

Page Load Time	0.65s	8.0s	1.5
CPU Utilization	14%	100%	75%

- **Before Attack**: Show the system's baseline metrics (normal load).
- **During Attack**: Show how the system is overwhelmed during the **DoS attack** (e.g., high CPU usage, increased page load time).
- **After Attack (with Firewall)**: Show the performance **after applying the firewall** and demonstrating how the system recovers.

4. Discuss the Results

- **Performance Impact**: Discuss the **response times** and **page load times** before and after the DoS attack. Highlight how the firewall helped in recovering from the attack and reducing load times.
- **Firewall Effectiveness**: Explain how the firewall's **rate-limiting** and **packet filtering** helped in blocking the attack and restoring system accessibility. Discuss whether the firewall successfully prevented the **DoS attack**.
- **CPU Utilization**: Show the CPU usage spikes during the attack and how the system stabilized after firewall rules were applied. This demonstrates the importance of resource management in high-load scenarios.
- **Throughput**: Discuss how the **requests per second (throughput)** were affected by the attack and how the system managed under normal conditions after the attack.
- **Security Measures**: Reflect on the **security measures** you implemented (firewall, SSH access control, etc.) and how they successfully prevented **unauthorized access** or minimized damage from the attack.

https://ubuntu.com/tutorials/install-and-configure-wordpress#1-overview
https://www.optimizesmart.com/wordpress-ninja-15-minutes/