WAF Setup

WE Innovate X Zero$ploit

Prepared by : Omar Hassan

Supervised by : Eng.Hazem Abdelrahman

Task:

- Set up a **Web Application Firewall (WAF)** on Ubuntu using **Nginx** & **ModSecurity** with **OWASP Core Rule Set (CRS)** so it can detect common web attacks.

***Setup** :

- **Setting Up Ubuntu machine**

Setting	Recommended
RAM	2-4 GB
Disk	10-20 GB
CPU	1-2 Cores
Network	NAT/Bridged

***Installing Prerequisites** :

Installing required packages & dependencies

```
sudo apt update
sudo apt install -y git gcc g++ make automake autoconf libtool libpcre3 libpcre3-dev \
libxml2 libxml2-dev libyajl-dev pkgconf zlib1g zlib1g-dev libcurl4-openssl-dev \
libgeoip-dev liblmdb-dev libfuzzy-dev liblua5.3-dev libpcre2-dev
```

Installing Modsecurity

```
git clone --depth 1 -b v3/master https://github.com/SpiderLabs/ModSecurity
cd ModSecurity
```

Build & install

```
git submodule init
git submodule update
./build.sh
./configure
make -j$(nproc)
sudo make install
cd ..
```

Download Nginx and ModSecurity connector

wget http://nginx.org/download/nginx-1.24.0.tar.gz
tar -xvzf nginx-1.24.0.tar.gz
git clone --depth 1 https://github.com/SpiderLabs/ModSecurity-nginx.git
sudo apt install nginx -y

Build Nginx with ModSecurity module

cd nginx-1.24.0
./configure --with-compat --add-dynamic-module=../ModSecurity-nginx
make modules
sudo mkdir -p /etc/nginx/modules
sudo cp objs/ngx_http_modsecurity_module.so /etc/nginx/modules

---

***Configuration** :

Enabling Modsecurity

sudo nano ~/nginx-1.24.0/conf/nginx.conf

Add this inside the file :

First Line in the file

load_module /home/omar/nginx-1.24.0/objs/ngx_http_modsecurity_module.so;

Inside Http{}

modsecurity on;
modsecurity_rules_file /etc/nginx/modsec/main.conf;

```
load_module /home/omar/nginx-1.24.0/objs/ngx_http_modsecurity_module.so;
#user  nobody;
worker_processes  1;

#error_log  logs/error.log;
#error_log  logs/error.log  notice;
#error_log  logs/error.log  info;

#pid        logs/nginx.pid;


events {
    worker_connections  1024;
}


http {
    include       mime.types;
    default_type  application/octet-stream;


    modsecurity on;
    modsecurity_rules_file /etc/nginx/modsec/main.conf;
```

Configuring Modsecurity

cd ~/ModSecurity
sudo cp modsecurity.conf-recommended /etc/nginx/modsec/modsecurity.conf
sudo nano /etc/nginx/modsec/modsecurity.conf

and edit SecRuleEngine to On instead of DetectionOnly

```
# -- Rule engine initialization ----------------------------------------------

# Enable ModSecurity, attaching it to every transaction. Use detection
# only to start with, because that minimises the chances of post-installation
# disruption.
#
SecRuleEngine On


# -- Request body handling ---------------------------------------------------

# Allow ModSecurity to access request bodies. If you don't, ModSecurity
# won't be able to see any POST parameters, which opens a large security
# hole for attackers to exploit.
#
SecRequestBodyAccess On
```

Installing OWASP Core Rule Set

cd /etc/nginx/modsec

sudo git clone https://github.com/coreruleset/coreruleset.git

sudo cp coreruleset/crs-setup.conf.example coreruleset/crs-setup.conf

Creating main.conf to load modsecurity + CRS

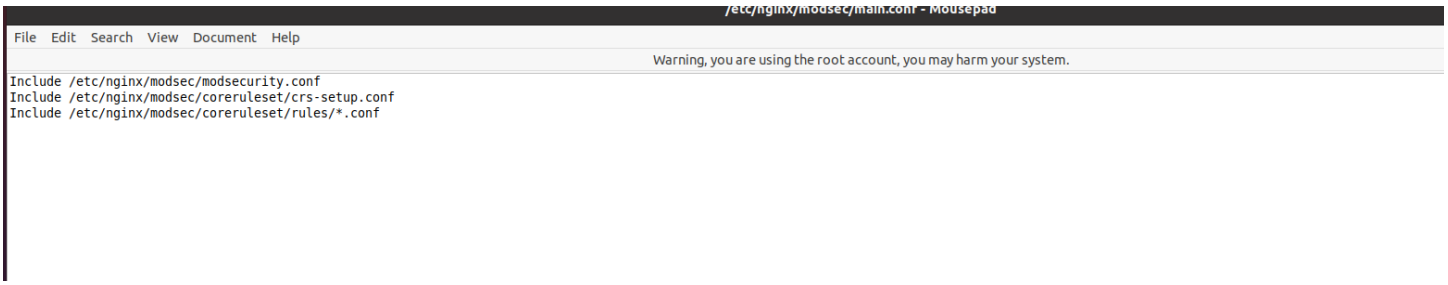
sudo nano /etc/nginx/modsec/main.conf

Add this inside

Include /etc/nginx/modsec/modsecurity.conf

Include /etc/nginx/modsec/coreruleset/crs-setup.conf

Include /etc/nginx/modsec/coreruleset/rules/.conf*



*****Testing** :***

sudo nginx -t

sudo systemctl restart nginx

Testing the rules using XSS

curl "http://localhost/?search=<script>alert(1)</script>"

```
omar@omar-ubunto:/etc/nginx/modsec$ curl "http://localhost/?search=<script>alert(1)</script>"
<html>
<head><title>403 Forbidden</title></head>
<body>
<center><h1>403 Forbidden</h1></center>
<hr><center>nginx/1.24.0</center>
</body>
</html>
omar@omar-ubunto:/etc/nginx/modsec$
```

A message of 403 forbidden means everything is working