Babar tech association

Contents

Chapter:1
Introduction

Introduction
The net has become an vital element of contemporary lifestyles, serving as a critical device for appearing a mess of day by day responsibilities. Simultaneously the developing recognition of internet applications has furnished unparalleled get entry to to information and services. However with this increase in internet software usage comes a corresponding uptick in assaults targeting them. Malicious actors have emerge as increasingly more sophisticated, exploiting vulnerabilities in net applications to have interaction in sports such as records robbery or different nefarious actions. One powerful means of safeguarding net packages towards such assaults is through the implementation of a web utility Firewall (WAF). This file outlines the stairs for putting in an open-supply WAF on a Linux-based totally machine and demonstrates how it is able to be leveraged to thwart attacks on an internet software.

1.1. Overview

Web Application firewall

A WAF is a type of firewall that is designed to protect web applications from attacks. It is places between the web server and the internet to filter the incoming web traffic to the application no one attack is to be performed.

A WAF can be used to protect against a variety of web based attacks like cross site scripting and file inclusion.

Overview of WAF (Web Application firewall)

In this we perform hands on practical to see WAF is properly working. Know our first step is to install the WAF and turn on it know see it block the malicious traffic. The next step is turn off the WAF and see the traffic is moving or it cannot block the traffic.

ModSecurity

Modsecurity is open source web application that is used to protect our system from other side of attacks like web based attacks. Know that is used in every where and it also support like web server, Nginx and apache. we can install modsecurity in every operating system like window, linux and unix. It can see the receiving traffic and implement the rules that we already give him to work on the following rules.

Steps to install WAF(Web Application firewall) and implement

The process that is to follow

1. Install ModSecurity

2. Configuring ModSecurity
3. Setting Up the OWASP ModSecurity Core Rule Set
4. Enabling ModSecurity in Apache 2
5. Testing ModSecurity on DVWA.

Install modsecurity

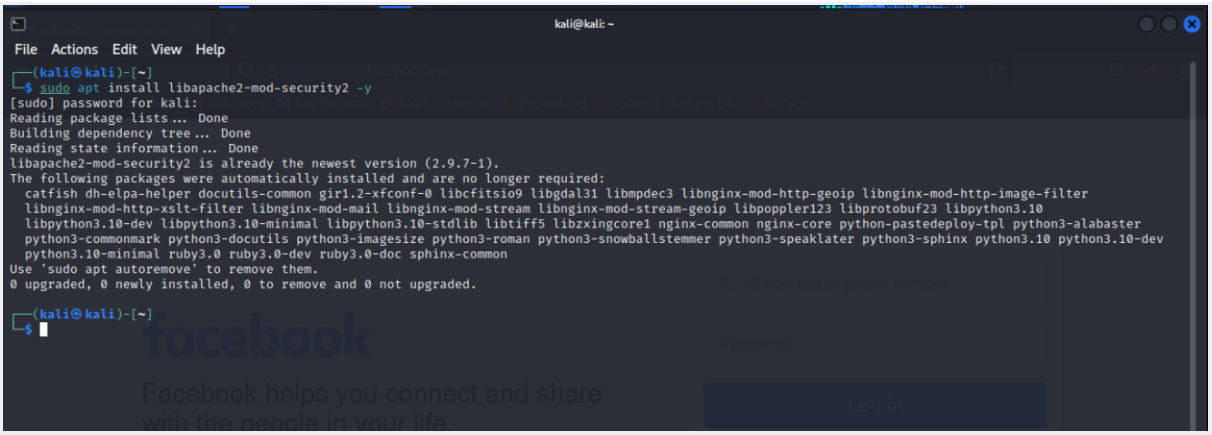
ModeSecurity is a open souce Web Application firewall that can be used to protect web application from attacks like file inclusion and cross site scripting. Modsecurity operates as an apache module and can be used with any web server that support apache interface.

This analyzing the receiving traffic and inspecting the HTTP requests and responses. It see the receiving traffic rules and compare the rules with that we implement in set of instructions. If the traffic is harm for our computer that we defined in rules it block the traffic.

Process

In this know we see how to install first we open our terminal in kali linux and put the following command in terminal:

```
sudo apt install libapache2-mod-security2
-y
```



After installing ModSecurity, enable the Apache 2 by running the following command:

```
sudo a2enmod headers
```

```
┌──(kali㉿kali)-[~]
└─$ sudo a2enmod headers
Module headers already enabled
```

After installing ModSecurity and enabling the header module you need to restart the apache2 service this can be done be this command:

```
sudo systemctl restart apache2
```

```
┌──(kali㉿kali)-[~]
└─$  sudo systemctl restart apache2
```

Configuring ModSecurity

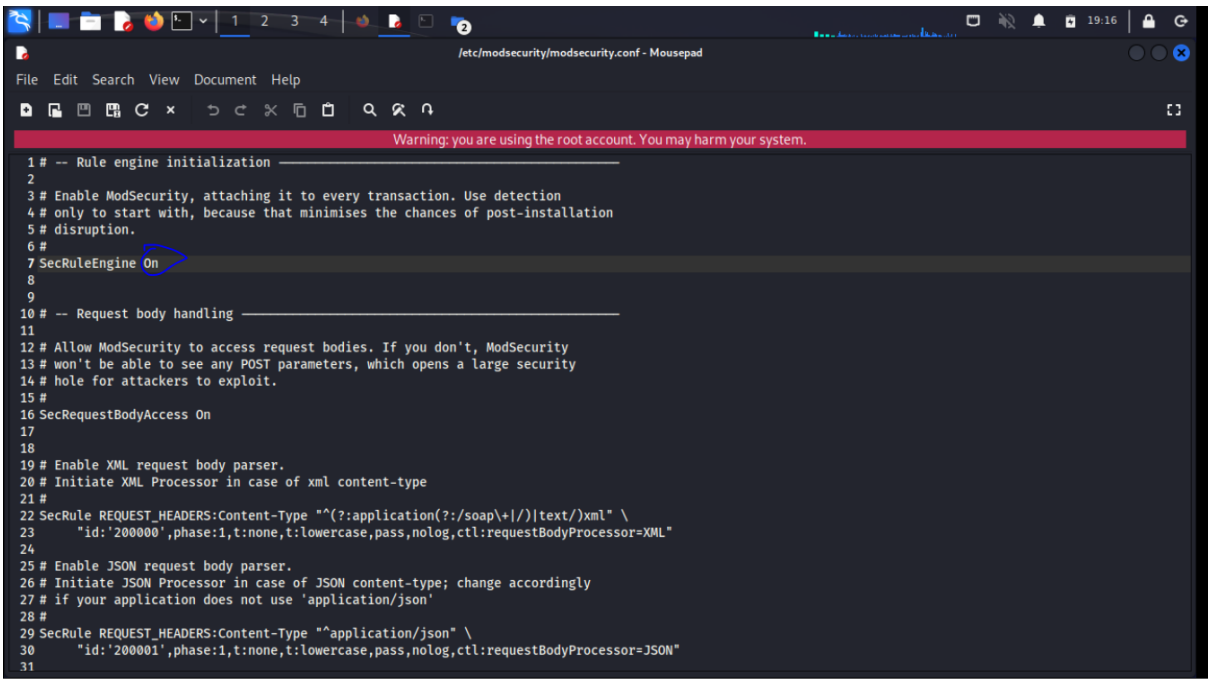
ModSecurity is a firewall and that requires rules to function. Here we see how to implement OWASP Core Rule Set. First we have need to prepare the ModSecurity configure file.

As per instruction we need no remove the .recommended extension from the modsecurity configuration file with the following command:

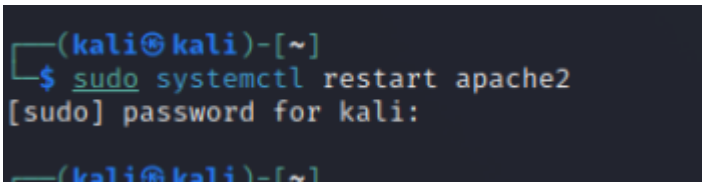
```
sudo cp
/etc/modsecurity/modsecurity.conf-recommended
/etc/modsecurity/modsecurity.conf
```

```
┌──(kali㉿kali)-[~]
└─$ sudo cp /etc/modsecurity/modsecurity.conf-recommended /etc/modsecurity/modsecurity.conf
[sudo] password for kali:
```

From know we gives the rules by OWASP Core Rule Set on firewall we should modify the file which has this way:



After that we restart the apache to apply changes by following command:



Setting Up the OWASP ModSecurity Core Rule Set

The OWASP ModSecurity core Rule Set (CRS) is a hard and fast of accepted assault detection regulations for use with ModSecurity or compatible internet software firewalls. The CRS objectives to protect internet packages from a huge variety of assaults, such as the OWASP top Ten, with at the least fake alerts. The CRS gives protection in opposition to many commonplace assault categories, along with square Injection, pass site Scripting, and local document Inclusion.

Know we first delete the current rule set that comes prepackaged with ModSecurity by running the following command:

```
sudo rm -rf /usr/share/modsecurity-crs
```

```
┌──(kali㉿kali)-[~]
└─$ sudo rm -rf /usr/share/modsecurity-crs
[sudo] password for kali:
```

Know we install git:

```
sudo apt install git
```

```
┌──(kali㉿kali)-[~]
└─$ sudo apt install git
[sudo] password for kali:
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
git is already the newest version (1:2.39.2-1.1).
git set to manually installed.
The following packages were automatically installed and are no longer required:
  catfish dh-elpa-helper docutils-common gir1.2-xfconf-0 libcfitsio9 libgdal31 libmpdec3 libnginx-mod-http-geoip libnginx-mod-http-image-filter
  libnginx-mod-http-xslt-filter libnginx-mod-mail libnginx-mod-stream libnginx-mod-stream-geoip libpoppler123 libprotobuf23 libpython3.10
  libpython3.10-dev libpython3.10-minimal libpython3.10-stdlib libtiff5 libzxingcore1 nginx-common nginx-core python-pastedeploy-tpl python3-alabaster
  python3-commonmark python3-docutils python3-imagesize python3-roman python3-snowballstemmer python3-speaklater python3-sphinx python3.10 python3.10-dev
  python3.10-minimal ruby3.0 ruby3.0-dev ruby3.0-doc sphinx-common
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

Know we clone the OWASP CRS Github repository by this command:

```
sudo git clone
https://github.com/coreruleset/coreruleset
/usr/share/modsecurity-crs
```

```
┌──(kali㉿kali)-[~]
└─$ sudo git clone https://github.com/coreruleset/coreruleset /usr/share/modsecurity-crs
Cloning into '/usr/share/modsecurity-crs' ...
remote: Enumerating objects: 25911, done.
remote: Counting objects: 100% (102/102), done.
remote: Compressing objects: 100% (59/59), done.
remote: Total 25911 (delta 50), reused 81 (delta 43), pack-reused 25809
Receiving objects: 100% (25911/25911), 6.52 MiB | 295.00 KiB/s, done.
Resolving deltas: 100% (20241/20241), done.
```

know we rename the “crs-setup.conf.example” to “crs-setup.conf”:

```
sudo mv
/usr/share/modsecurity-crs/crs-setup.conf.exa
mple
/usr/share/modsecurity-crs/crs-setup.conf
```

```
┌──(kali㉿kali)-[~]
└─$ sudo mv /usr/share/modsecurity-crs/crs-setup.conf.example /usr/share/modsecurity-crs/crs-setup.conf
```

know we rename the default request exclusion rule file by this command:

```
sudo mv
/usr/share/modsecurity-crs/rules/REQUEST-900-EXCLU
SION-RULES-BEFORE-CRS.conf.example
/usr/share/modsecurity-crs/rules/REQUEST-900-EXCLU
SION-RULES-BEFORE-CRS.conf
```

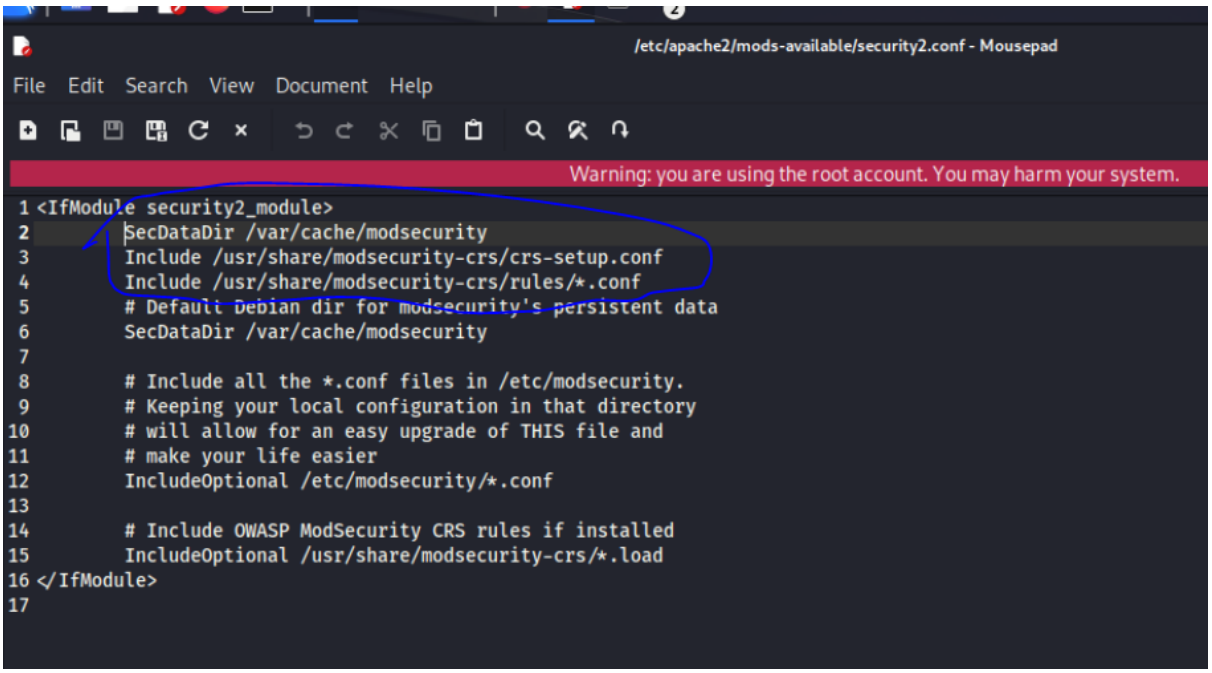
Know enabling the ModSecurity in Apache2:

To use ModSecurity we enable the apache configuration file by following steps:

```
/etc/apache2/mods-available/security2.conf
```

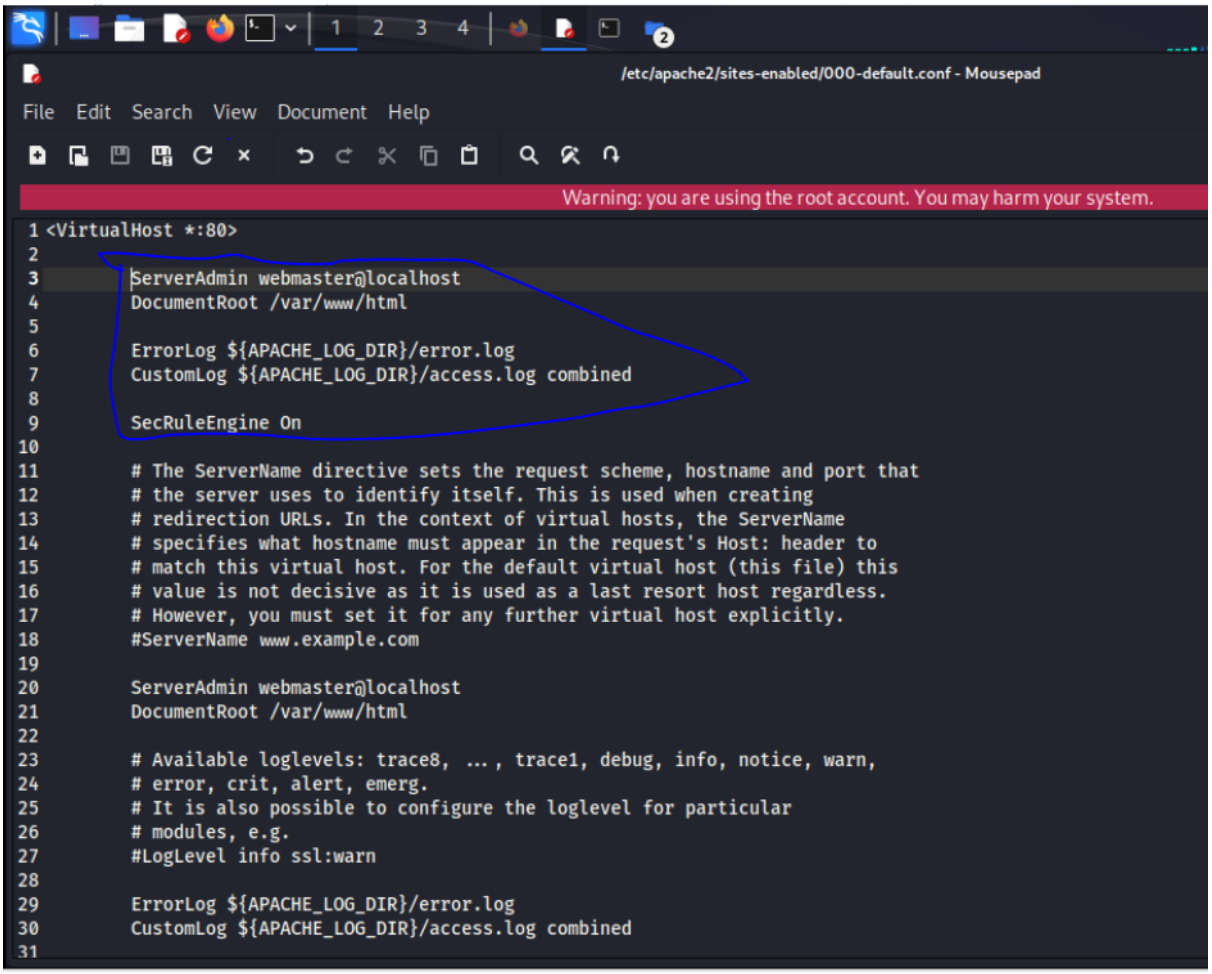
Know the following code is past in file:

```
 SecDataDir /var/cache/modsecurity

        Include
/usr/share/modsecurity-crs/crs-setup.conf

        Include
/usr/share/modsecurity-crs/rules/*.conf
```



Know we go into this file `/etc/apache2/sites-enabled/000-default.conf` and past the following file virtualhost block and also include the SecRuleEngine detective to On.

```
ServerAdmin webmaster@localhost

        DocumentRoot /var/www/html



        ErrorLog ${APACHE_LOG_DIR}/error.log

        CustomLog ${APACHE_LOG_DIR}/access.log combined



        SecRuleEngine On
```



After all that know we must restart the apache2 service to configure by the following command:

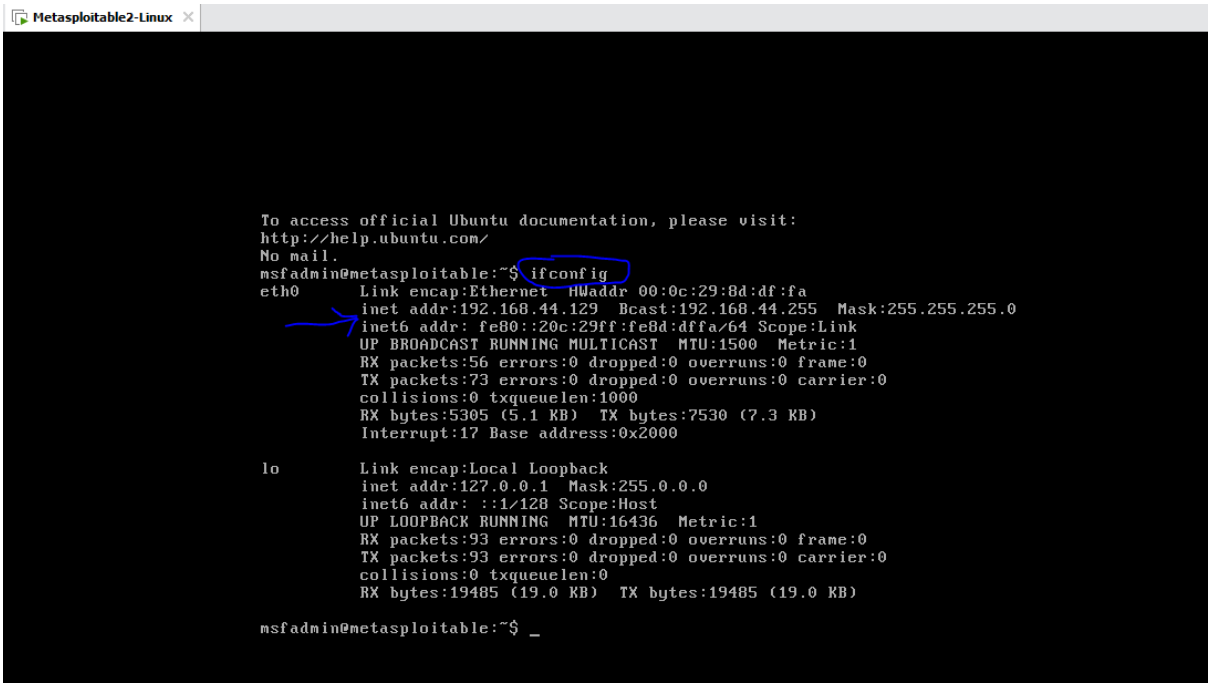
```
sudo systemctl restart apache2
```

Download DVWA in localhost:

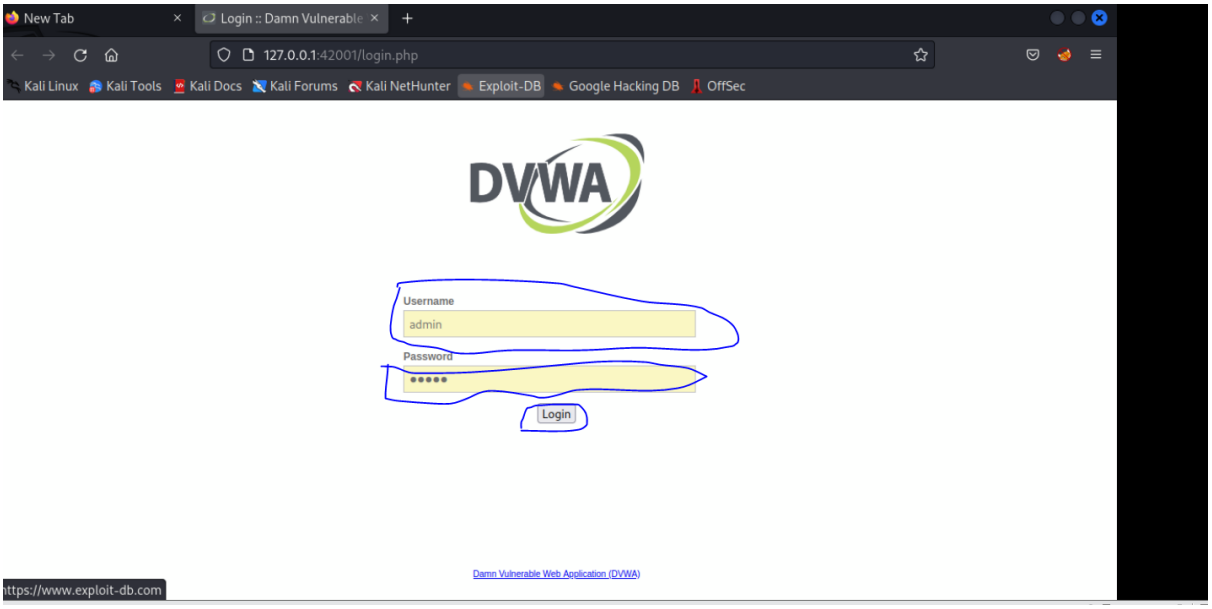
Here we see the some steps to install the DVWA in our localhost.

First we download metasploitable2 with this link.

https://www.vulnhub.com/entry/metasploitable-2,29/ . Open this in you vm wear or in virtual box and here default username and password is “msfadmin”.



Know we goto in linux terminal and type the following command "dvwa-start" Default username is "admin" and password is "password".

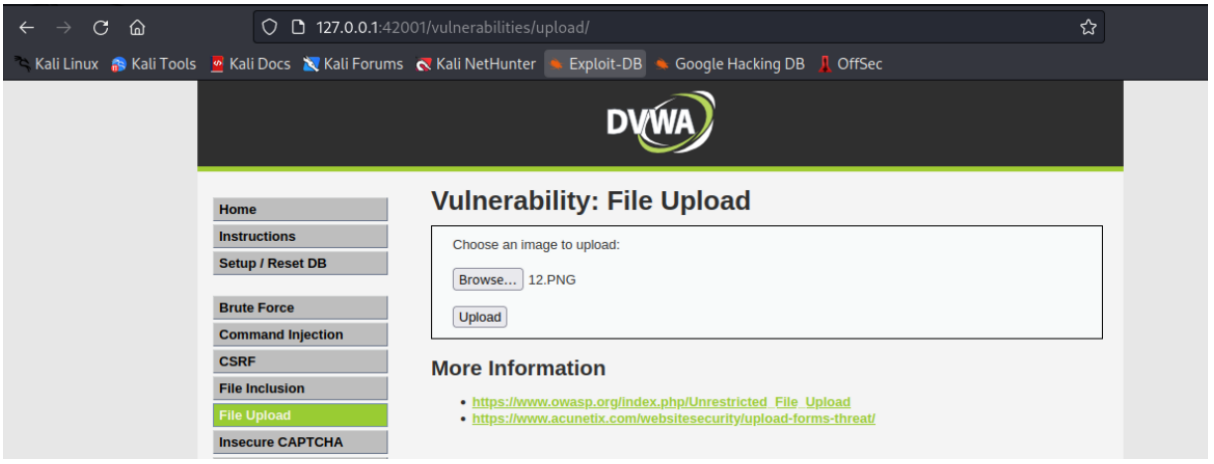


Testing ModSecurity

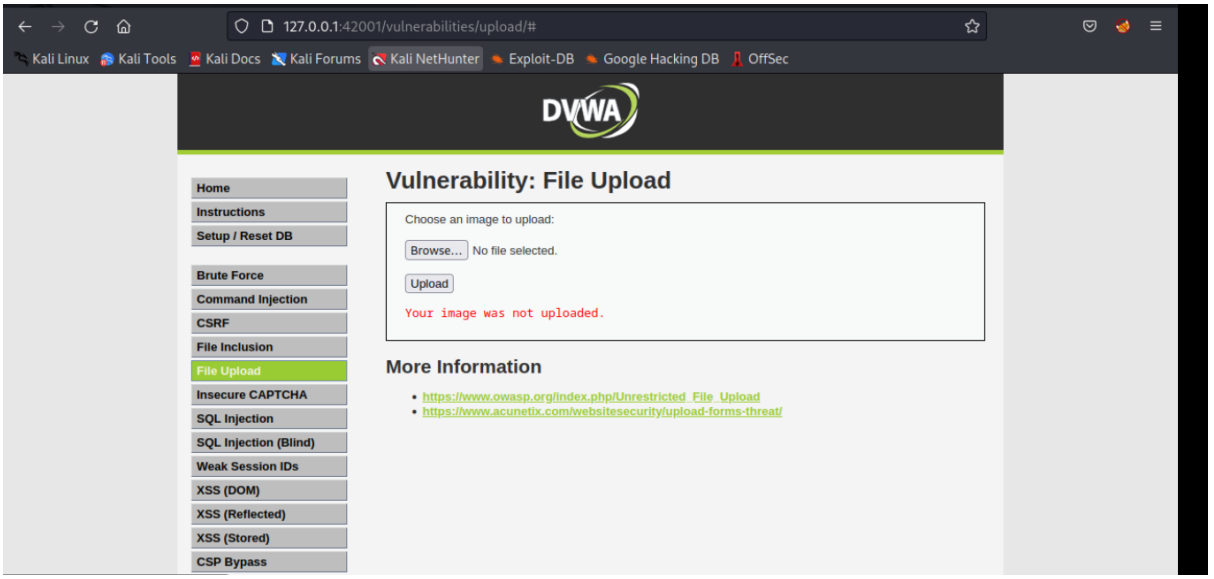
After all the installation know we are ready to perform and attack to see what things are working. To perform this operation we use DVWA.

Malicious file upload:

During the on firewall we can upload the image.PNG this restrict or not allow the image to upload. We can also upload the php, python, JPG and .PNG in this but know the condition this allow to upload jpg and png.

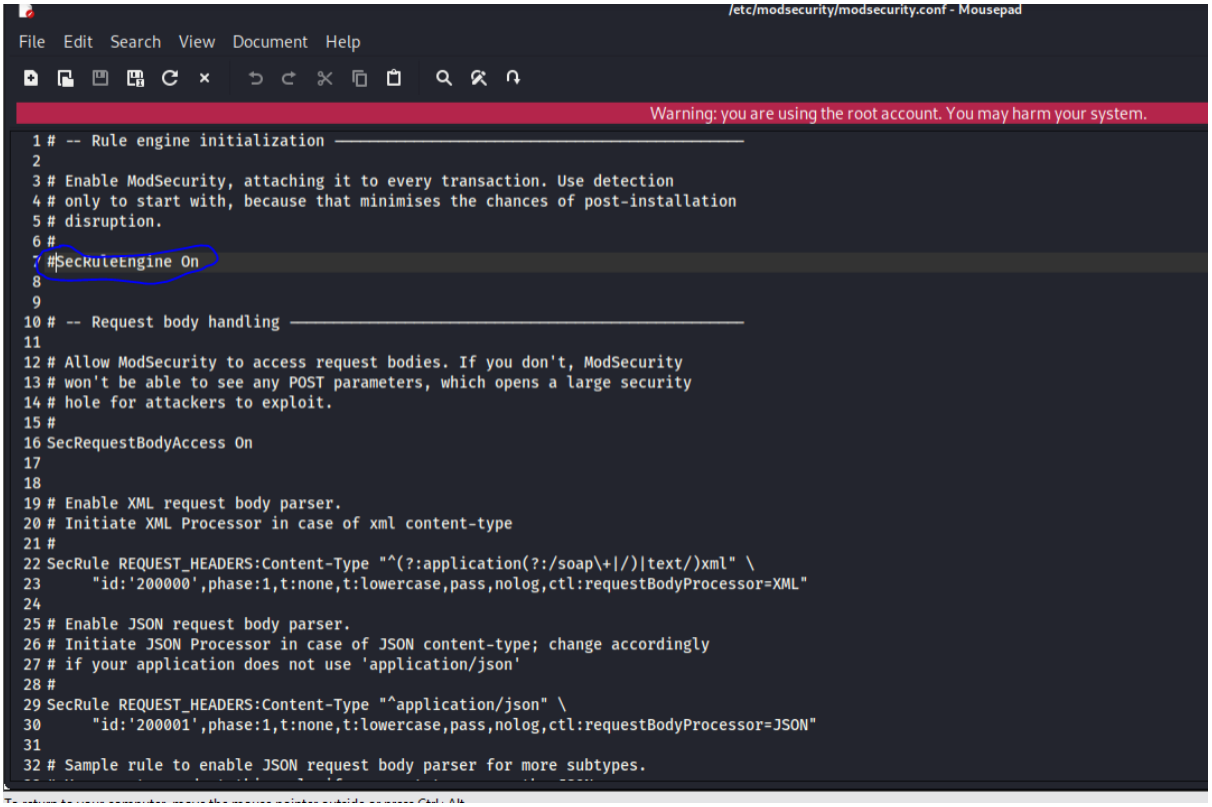


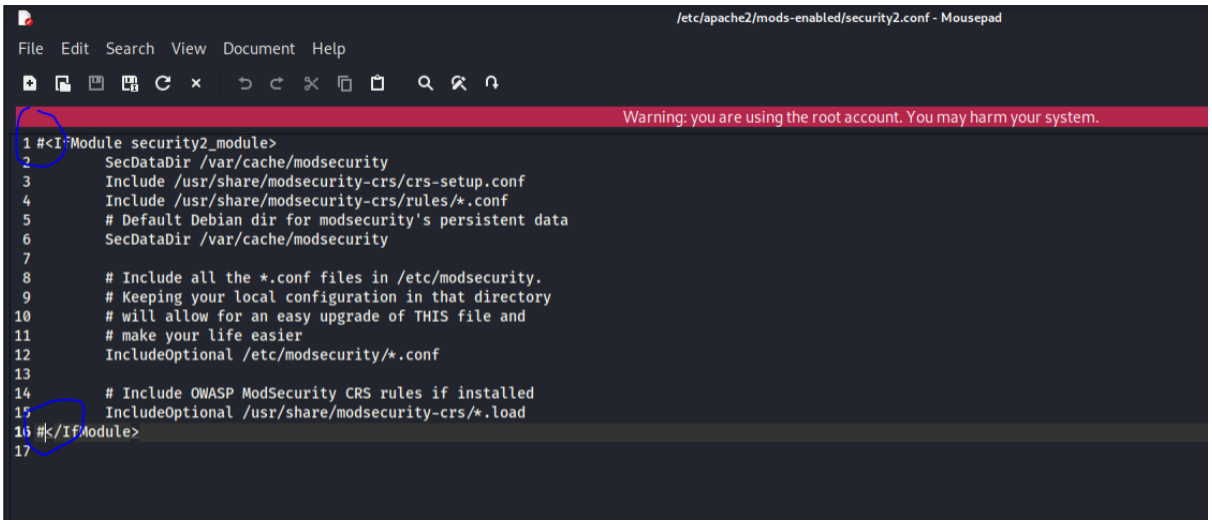
After uploading .jpg



Testing ModSecurity

At the last we off the modsecurity and upload the file and we see the file is successfully uploaded.





File Edit Search View Document Help

Warning: you are using the root account. You may harm your system.

```
<VirtualHost *:80>

        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

        #SecRuleEngine On

        # The ServerName directive sets the request scheme, hostname and port that
        # the server uses to identify itself. This is used when creating
        # redirection URLs. In the context of virtual hosts, the ServerName
        # specifies what hostname must appear in the request's Host: header to
        # match this virtual host. For the default virtual host (this file) this
        # value is not decisive as it is used as a last resort host regardless.
        # However, you must set it for any further virtual host explicitly.
        #ServerName www.example.com

        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html

        # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
        # error, crit, alert, emerg.
        # It is also possible to configure the loglevel for particular
        # modules, e.g.
        #LogLevel info ssl:warn

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

        # For most configuration files from conf-available/, which are
```

comment it

return to your computer, move the mouse pointer outside or press Ctrl+Alt.

127.0.0.1:42001/vulnerabilities/upload/#

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

DVWA

Home
Instructions
Setup / Reset DB

Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection

Vulnerability: File Upload

Choose an image to upload:

Browse... No file selected.

Upload

../../hackable/uploads/ccsrf.html succesfully uploaded!

More Information

- https://www.owasp.org/index.php/Unrestricted_File_Upload
- https://www.acunetix.com/websitesecurity/upload-forms-threat/

Babar tech association

Thanks