Guide

Installing a Web Application Firewall on Ubuntu Server 22.04

- Nuno Romão

Objective:

In this lab you'll will be able to follow the step-by-step guide to install a Web Application Firewall to protect your web Apps, as well as set your Ubuntu Server as a reverse proxy in case your website is hosted in another server.

In this exercise we'll be using the following network design.

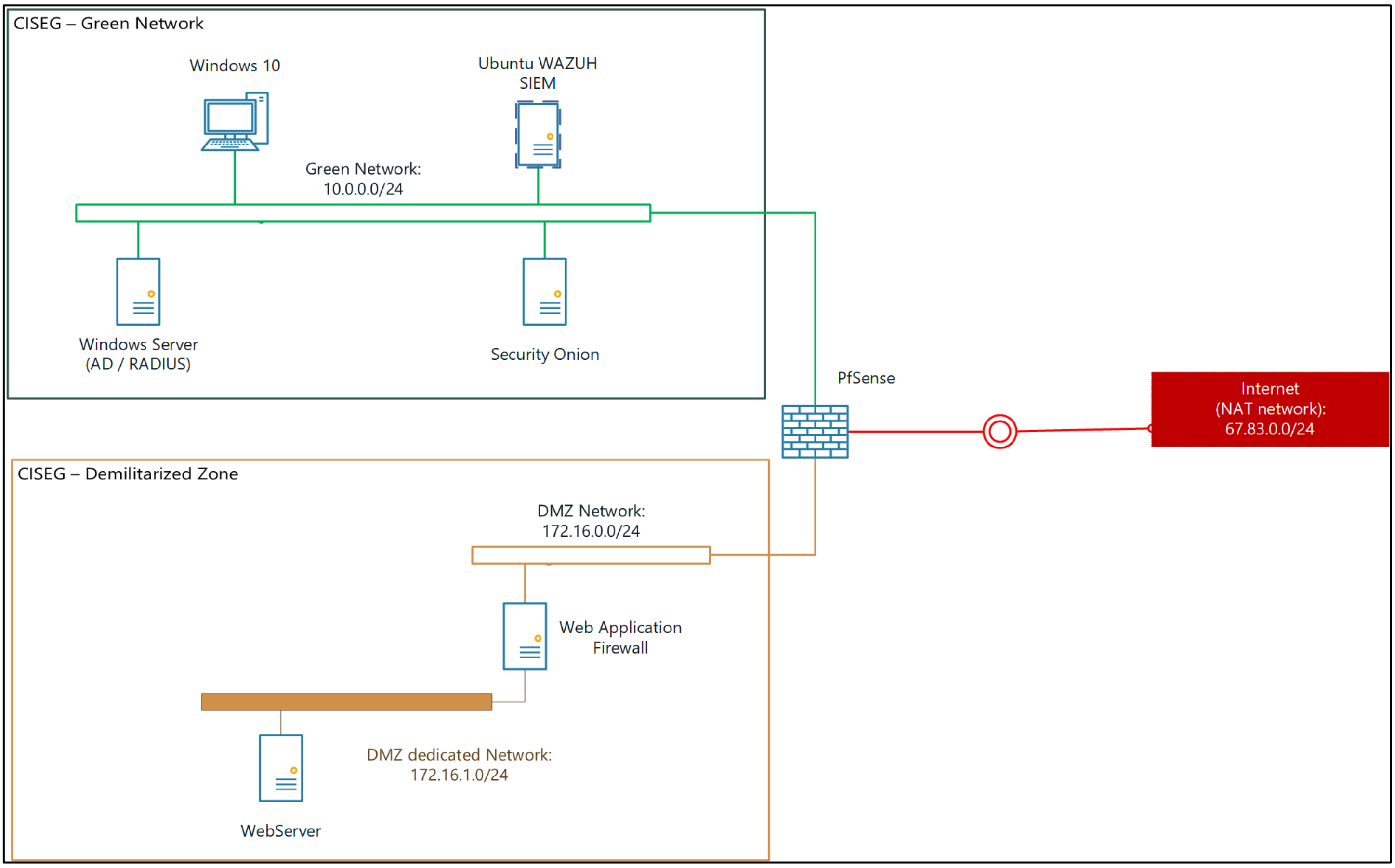

CISEG – Green Network
Windows 10
Ubuntu WAZUH SIEM
Green Network: 10.0.0.0/24
Windows Server (AD / RADIUS)
Security Onion
PfSense
Internet (NAT network): 67.83.0.0/24
CISEG – Demilitarized Zone
DMZ Network: 172.16.0.0/24
Web Application Firewall
DMZ dedicated Network: 172.16.1.0/24
WebServer

Figure 1 - Network typology

We'll be focusing mostly on the demilitarized zone.

Our Web Application Firewall is an Ubuntu Server 22.04 and our Web Server is a Windows Server 2022 with Internet Information Services installed.

The WebServer will be hosted behind the WAF server in it's own dedicated network, and our WAF will work in a reverse proxy manner.

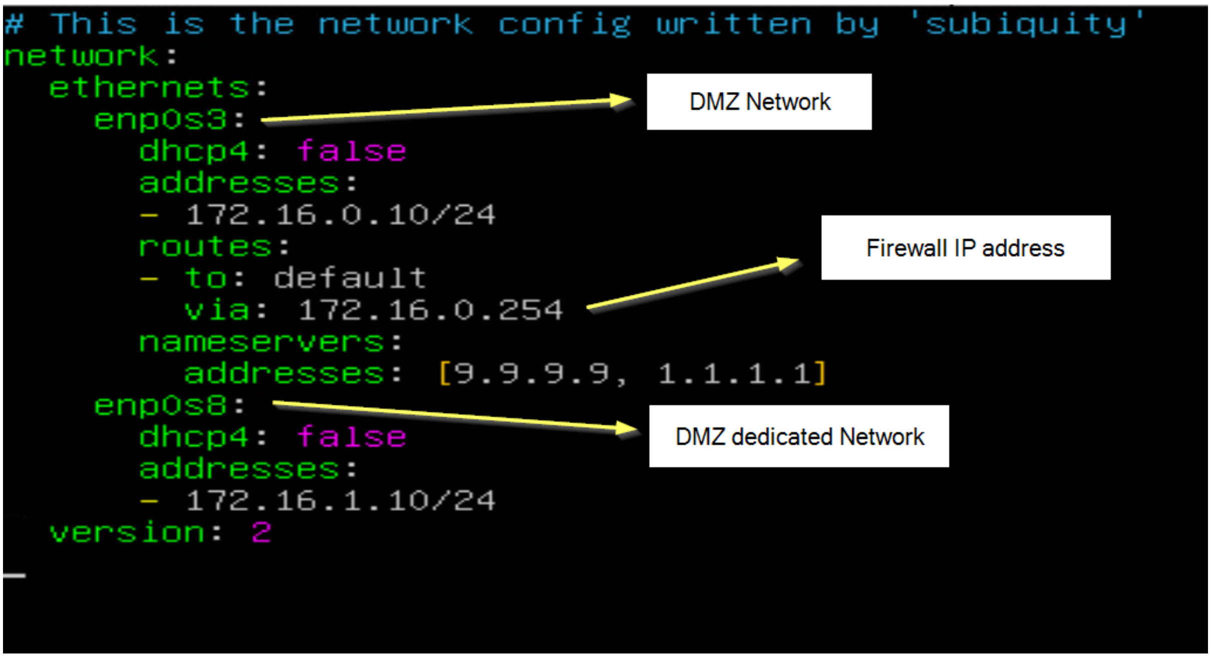
Here are the IP addresses used in the exercise:

Firewall (pfsense) – DMZ interface: 172.16.0.254

WAF DMZ network: 172.16.0.10

WAF DMZ dedicated network: 172.16.1.10

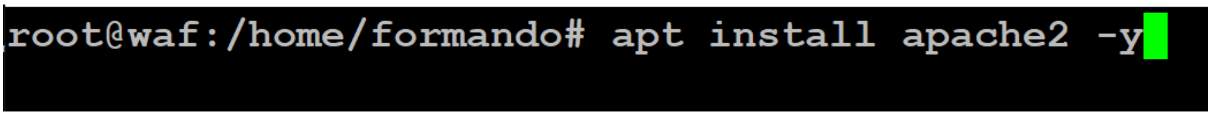
Web Server (Windows): 172.16.1.1



```
# This is the network config written by 'subiquity'
network:
  ethernets:
    enp0s3:
      dhcp4: false
      addresses:
      - 172.16.0.10/24
      routes:
      - to: default
        via: 172.16.0.254
      nameservers:
        addresses: [9.9.9.9, 1.1.1.1]
    enp0s8:
      dhcp4: false
      addresses:
      - 172.16.1.10/24
  version: 2
```

Figure 2 - WAF server network configuration

Temporarily we'll open up communications in our Firewall to install software and update our systems.

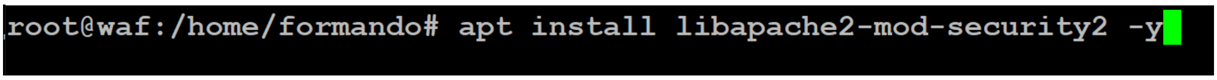


```
root@waf:/home/formando# apt install apache2 -y
```

Figure 3 - Apache2 installation

Install apache2 as the WAF we'll be using is ModSecurity and it works with Apache.

Then install ModSecurity module.



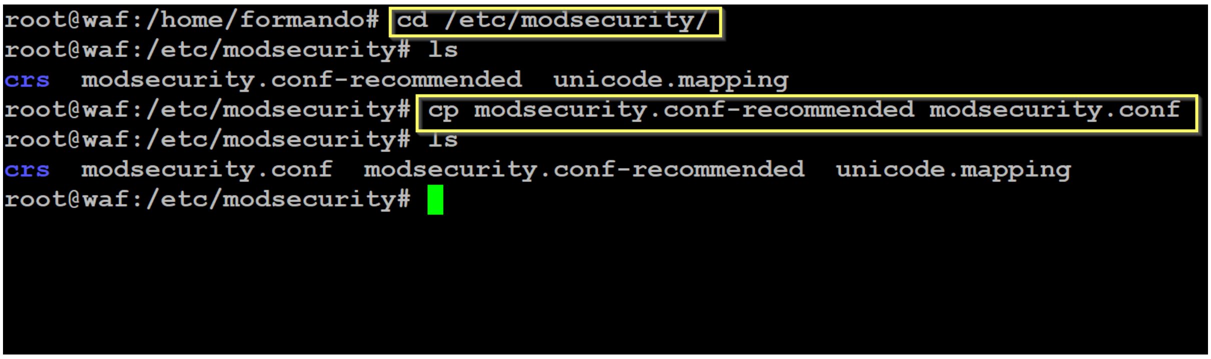
```
root@waf:/home/formando# apt install libapache2-mod-security2 -y
```

Figure 4 - ModSecurity2 installation command

After installing ModSecurity, we need to make sure “modsecurity.conf” file exists and it is recommended to change the way ModSecurity is working, as by default it will work in “DetectionOnly” and we need to change it to “On” in order to block malicious requests.

NOTE: After installation **modsecurity.conf** doesn't exists, the file has the name **modsecurity.conf-recommended** and it needs to be changed.

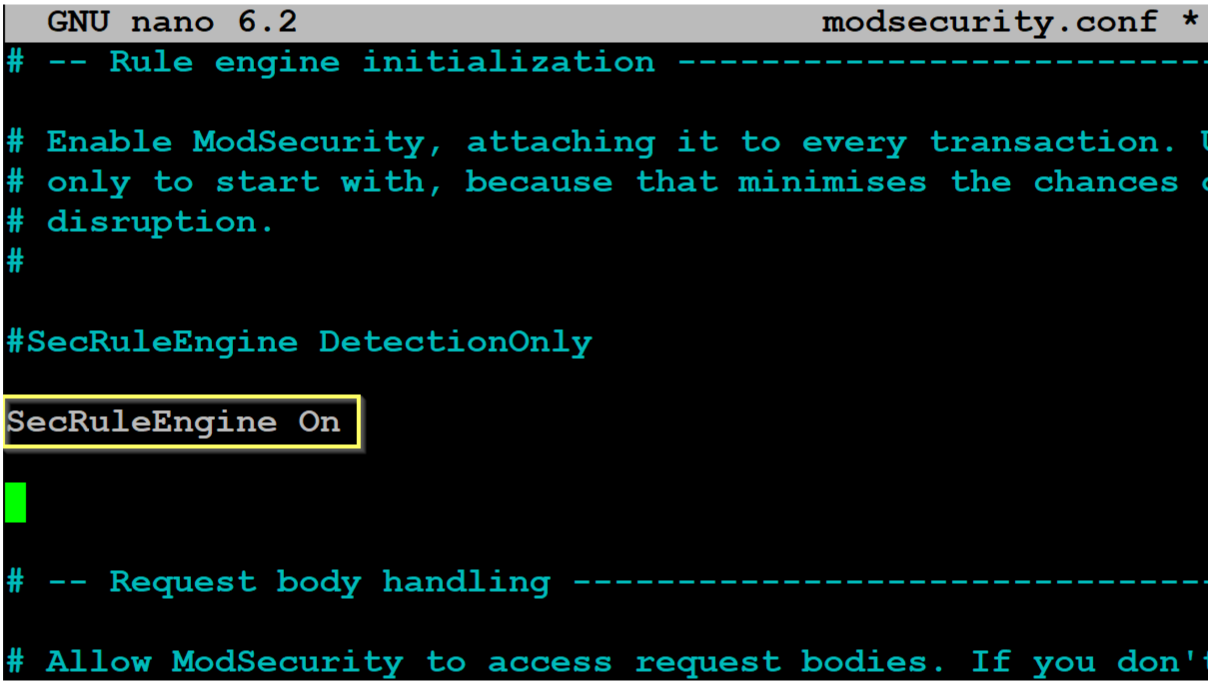
To perform this change go to /etc/modsecurity.



```
root@waf:/home/formando# cd /etc/modsecurity/
root@waf:/etc/modsecurity# ls
crs  modsecurity.conf-recommended  unicode.mapping
root@waf:/etc/modsecurity# cp modsecurity.conf-recommended modsecurity.conf
root@waf:/etc/modsecurity# ls
crs  modsecurity.conf  modsecurity.conf-recommended  unicode.mapping
root@waf:/etc/modsecurity# 
```

Figure 5 - modsecurity.conf creation

Edit **modsecurity.conf** and change the default value for SecRuleEngine.

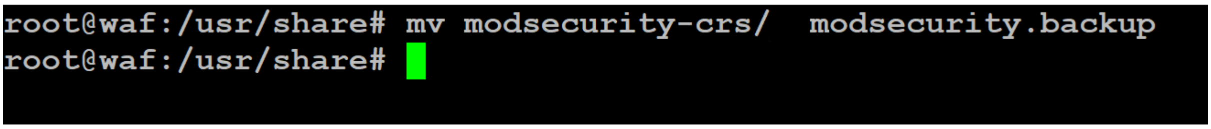


```
  GNU nano 6.2                          modsecurity.conf *
# -- Rule engine initialization ----------------------------

# Enable ModSecurity, attaching it to every transaction. U
# only to start with, because that minimises the chances o
# disruption.
#

#SecRuleEngine DetectionOnly

SecRuleEngine On



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

# Allow ModSecurity to access request bodies. If you don'
```

Figure 6 - Change SecRuleEngine to "On" to block any malicious requests

Once this is done, we can use the OWASP project Core Rule Set rules in our ModSecurity.

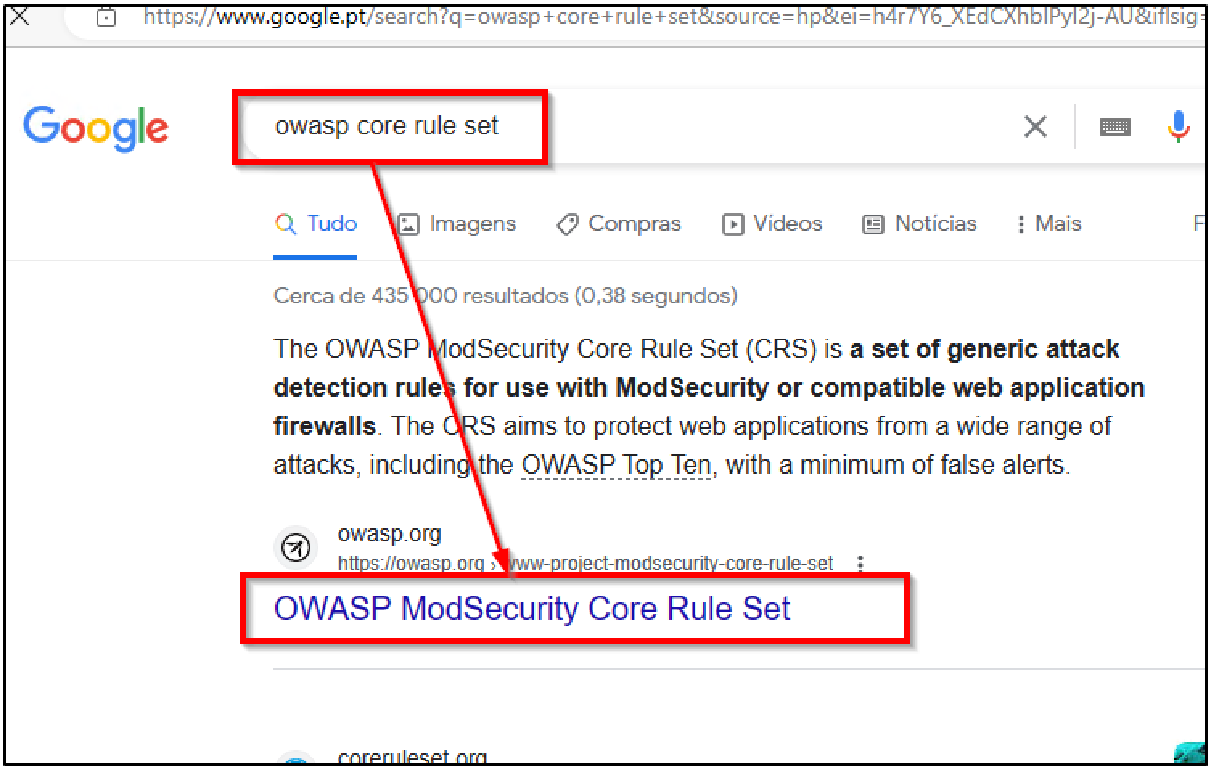
Before we begin, please create a backup of **modsecurity-crs** folder in **/usr/share**.



```
root@waf:/usr/share# mv modsecurity-crs/  modsecurity.backup
root@waf:/usr/share# 
```

Figure 7 - Create a backup of the original modsecurity core rule set folder

Go to OWASP core rule site GitHub.



https://www.google.pt/search?q=owasp+core+rule+set&source=hp&ei=h4r7Y6_XEdCXhbIPyI2j-AU&iflsig

owasp core rule set

Cerca de 435 000 resultados (0,38 segundos)

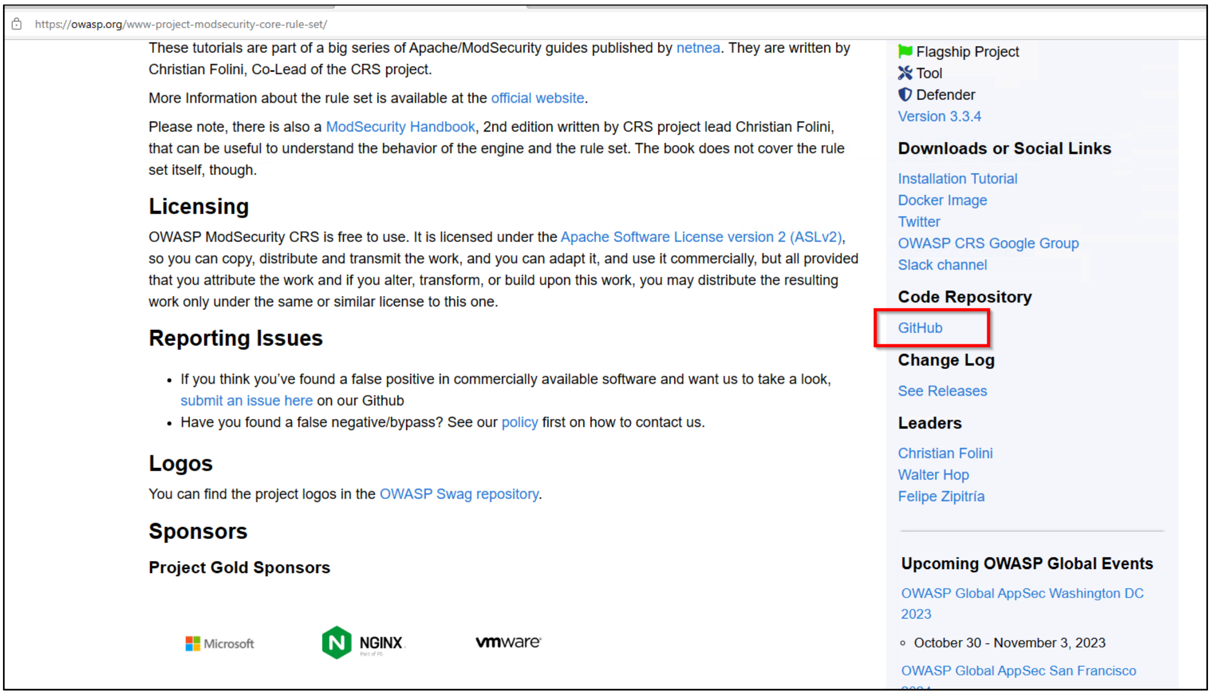
The OWASP ModSecurity Core Rule Set (CRS) is **a set of generic attack detection rules for use with ModSecurity or compatible web application firewalls**. The CRS aims to protect web applications from a wide range of attacks, including the OWASP Top Ten, with a minimum of false alerts.

owasp.org
https://owasp.org › www-project-modsecurity-core-rule-set

OWASP ModSecurity Core Rule Set

coreruleset.org

Figure 8 - Search for owasp core rule set



https://owasp.org/www-project-modsecurity-core-rule-set/

These tutorials are part of a big series of Apache/ModSecurity guides published by netnea. They are written by Christian Folini, Co-Lead of the CRS project.

More Information about the rule set is available at the official website.

Please note, there is also a ModSecurity Handbook, 2nd edition written by CRS project lead Christian Folini, that can be useful to understand the behavior of the engine and the rule set. The book does not cover the rule set itself, though.

Licensing

OWASP ModSecurity CRS is free to use. It is licensed under the Apache Software License version 2 (ASLv2), so you can copy, distribute and transmit the work, and you can adapt it, and use it commercially, but all provided that you attribute the work and if you alter, transform, or build upon this work, you may distribute the resulting work only under the same or similar license to this one.

Reporting Issues

- If you think you've found a false positive in commercially available software and want us to take a look, submit an issue here on our Github
- Have you found a false negative/bypass? See our policy first on how to contact us.

Logos

You can find the project logos in the OWASP Swag repository.

Sponsors

Project Gold Sponsors

Flagship Project
Tool
Defender
Version 3.3.4

Downloads or Social Links

Installation Tutorial
Docker Image
Twitter
OWASP CRS Google Group
Slack channel

Code Repository

GitHub

Change Log

See Releases

Leaders

Christian Folini
Walter Hop
Felipe Zipitría

Upcoming OWASP Global Events

OWASP Global AppSec Washington DC 2023

- October 30 - November 3, 2023

OWASP Global AppSec San Francisco

Figure 9 - Go to Github page for this project

Copy the git link.

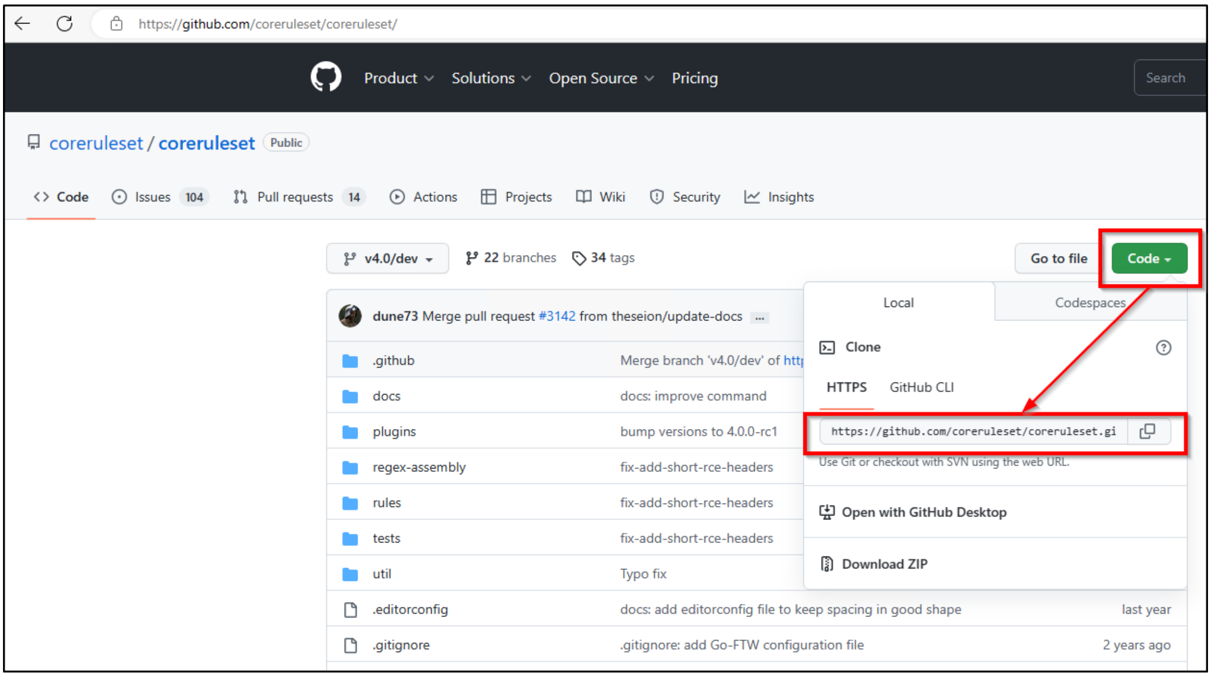
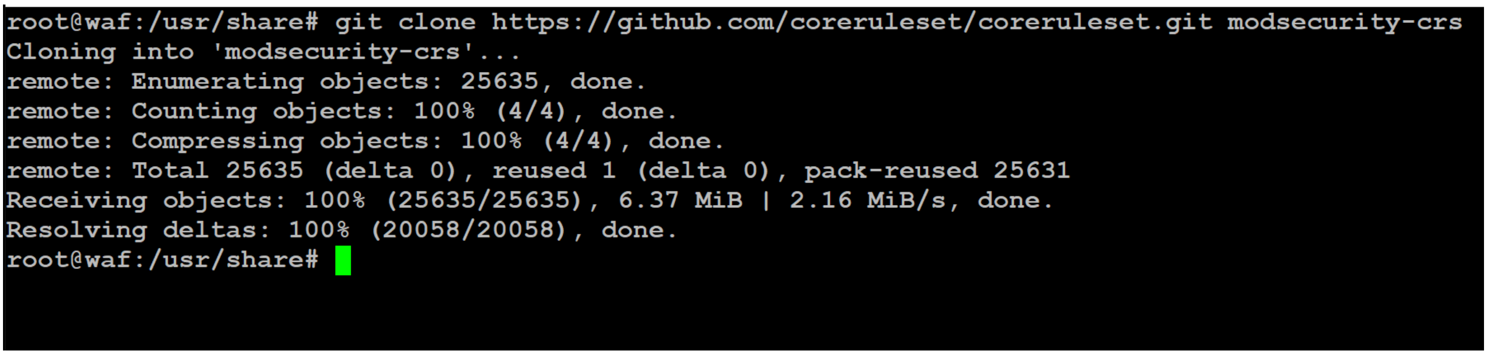

https://github.com/coreruleset/coreruleset/
Product
Solutions
Open Source
Pricing
Search
coreruleset / coreruleset
Public
Code
Issues 104
Pull requests 14
Actions
Projects
Wiki
Security
Insights
v4.0/dev
22 branches
34 tags
Go to file
Code
Local
Codespaces
dune73 Merge pull request #3142 from theseion/update-docs ...
Clone
HTTPS
GitHub CLI
https://github.com/coreruleset/coreruleset.gi
Use Git or checkout with SVN using the web URL.
Open with GitHub Desktop
Download ZIP
.github
Merge branch 'v4.0/dev' of http
docs
docs: improve command
plugins
bump versions to 4.0.0-rc1
regex-assembly
fix-add-short-rce-headers
rules
fix-add-short-rce-headers
tests
fix-add-short-rce-headers
util
Typo fix
.editorconfig
docs: add editorconfig file to keep spacing in good shape
last year
.gitignore
.gitignore: add Go-FTW configuration file
2 years ago

Figure 10 - Github project cloning

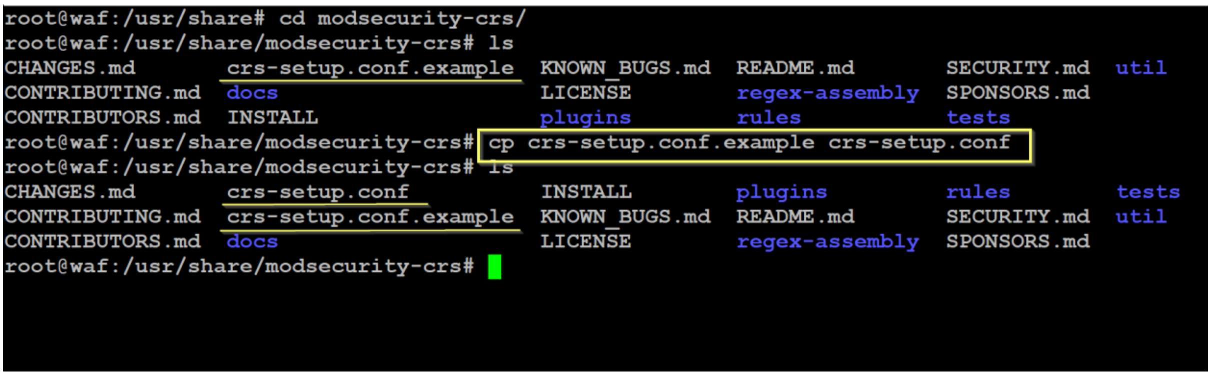
Paste the link and give **modsecurity-crs** as the name of the folder where all git content should be cloned to.



```
root@waf:/usr/share# git clone https://github.com/coreruleset/coreruleset.git modsecurity-crs
Cloning into 'modsecurity-crs'...
remote: Enumerating objects: 25635, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 25635 (delta 0), reused 1 (delta 0), pack-reused 25631
Receiving objects: 100% (25635/25635), 6.37 MiB | 2.16 MiB/s, done.
Resolving deltas: 100% (20058/20058), done.
root@waf:/usr/share# 
```

Figure 11 - Clone github repository to modsecurity-crs folder

Now create a **crs-setup.conf** file from the **crs-setup.conf.example** document.

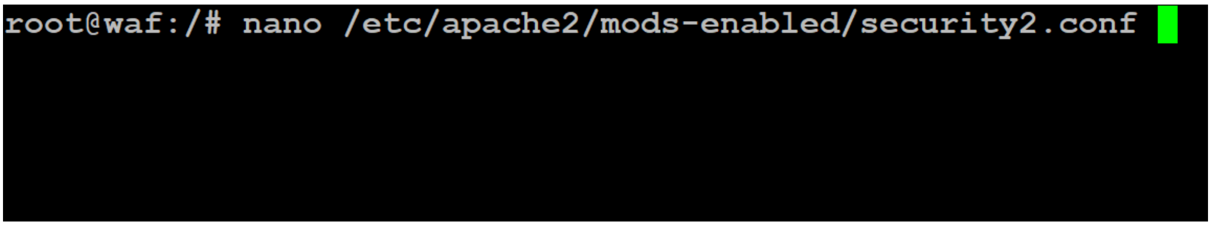


```
root@waf:/usr/share# cd modsecurity-crs/
root@waf:/usr/share/modsecurity-crs# ls
CHANGES.md        crs-setup.conf.example  KNOWN_BUGS.md  README.md       SECURITY.md  util
CONTRIBUTING.md   docs                    LICENSE        regex-assembly  SPONSORS.md
CONTRIBUTORS.md   INSTALL                 plugins        rules           tests
root@waf:/usr/share/modsecurity-crs# cp crs-setup.conf.example crs-setup.conf
root@waf:/usr/share/modsecurity-crs# ls
CHANGES.md        crs-setup.conf          INSTALL        plugins         rules        tests
CONTRIBUTING.md   crs-setup.conf.example  KNOWN_BUGS.md  README.md       SECURITY.md  util
CONTRIBUTORS.md   docs                    LICENSE        regex-assembly  SPONSORS.md
root@waf:/usr/share/modsecurity-crs# 
```

Figure 12 - Creation of crs-setup.conf

Once that is done, it is time to include all the rules present in /usr/share/modsecurity-crs/rules on our apache module.

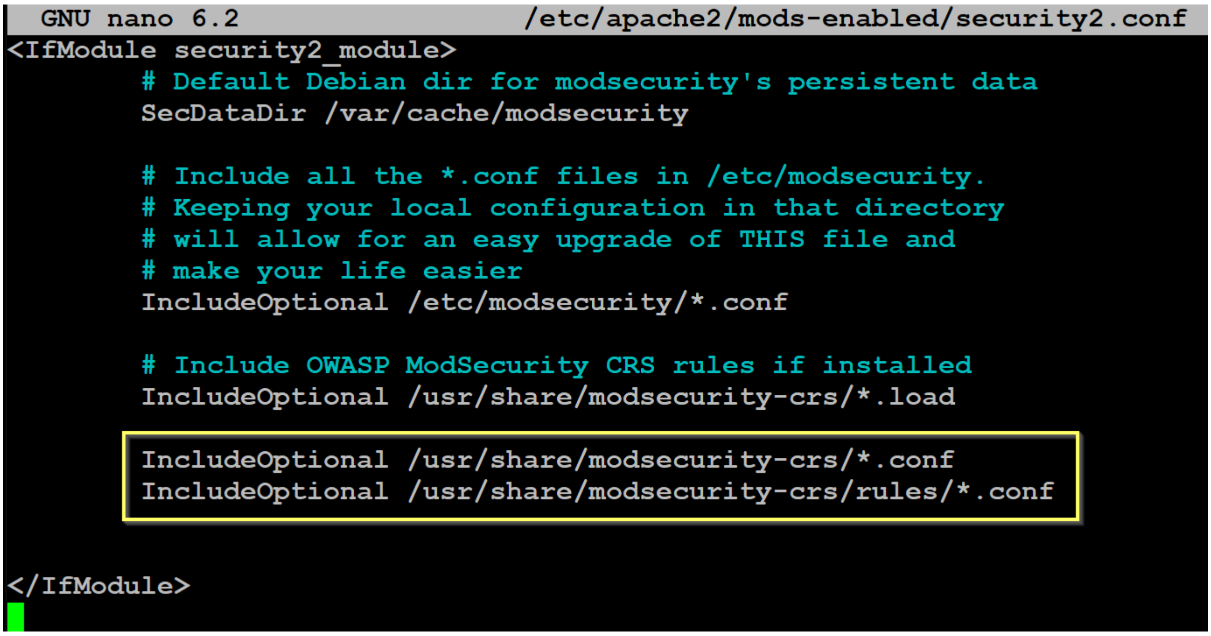
Edit the security2.conf file at **/etc/apache2/mods-enabled/security2.conf**.



```
root@waf:/# nano /etc/apache2/mods-enabled/security2.conf
```

Figure 13 - ModSecurity2 mod configuration file

Add the following lines.

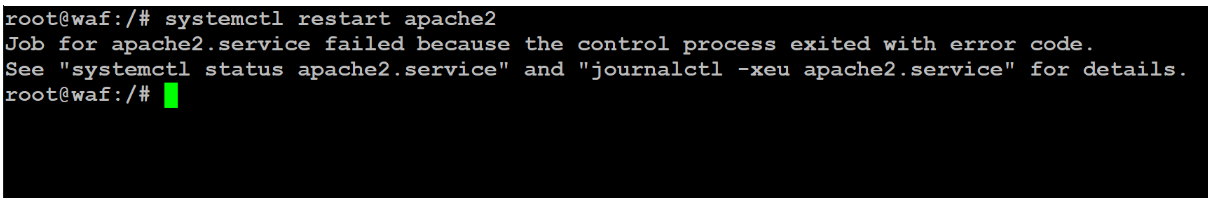


```
  GNU nano 6.2                    /etc/apache2/mods-enabled/security2.conf
<IfModule security2_module>
        # Default Debian dir for modsecurity's persistent data
        SecDataDir /var/cache/modsecurity

        # Include all the *.conf files in /etc/modsecurity.
        # Keeping your local configuration in that directory
        # will allow for an easy upgrade of THIS file and
        # make your life easier
        IncludeOptional /etc/modsecurity/*.conf

        # Include OWASP ModSecurity CRS rules if installed
        IncludeOptional /usr/share/modsecurity-crs/*.load

        IncludeOptional /usr/share/modsecurity-crs/*.conf
        IncludeOptional /usr/share/modsecurity-crs/rules/*.conf


</IfModule>
```

Figure 14 - Identify files to be loaded with ModSecurity2

You may restart your Apache2, and be aware it may throw an error.

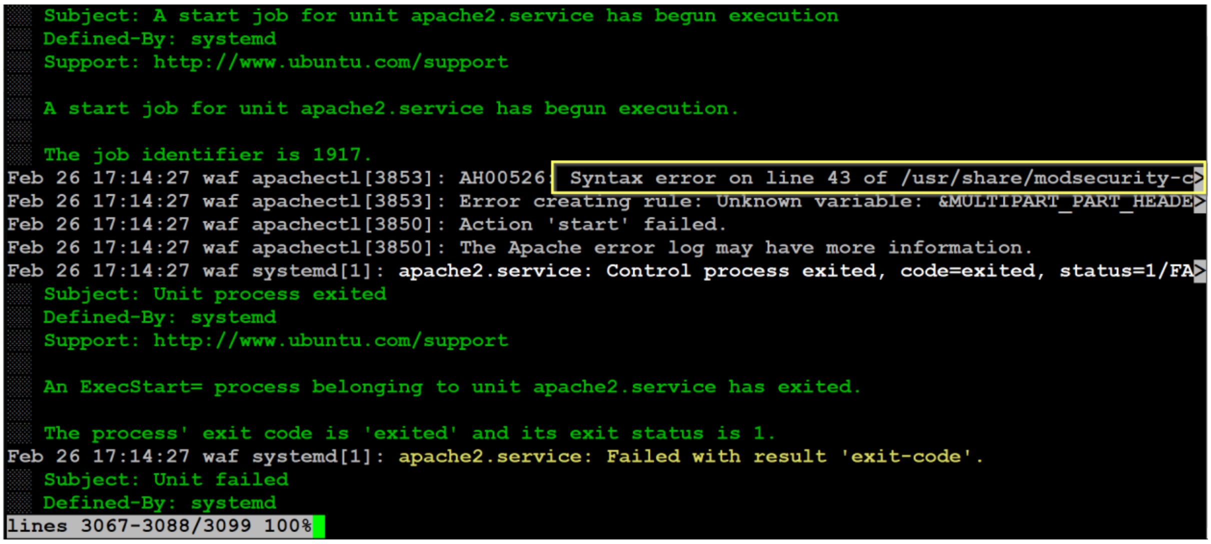
Don’t worry, we’ll see what the problem is with the rules and since it is a rule that is compatible with modsecurity version 3, we’ll need to remove this file for the time being as at the time of writing, modsecurity3 is still not available from the repositories and it would be necessary to compile it from source. We’ll not be focusing on that process in this guide, so we are just going to move our rule to another folder.



```
root@waf:/# systemctl restart apache2
Job for apache2.service failed because the control process exited with error code.
See "systemctl status apache2.service" and "journalctl -xeu apache2.service" for details.
root@waf:/#
```

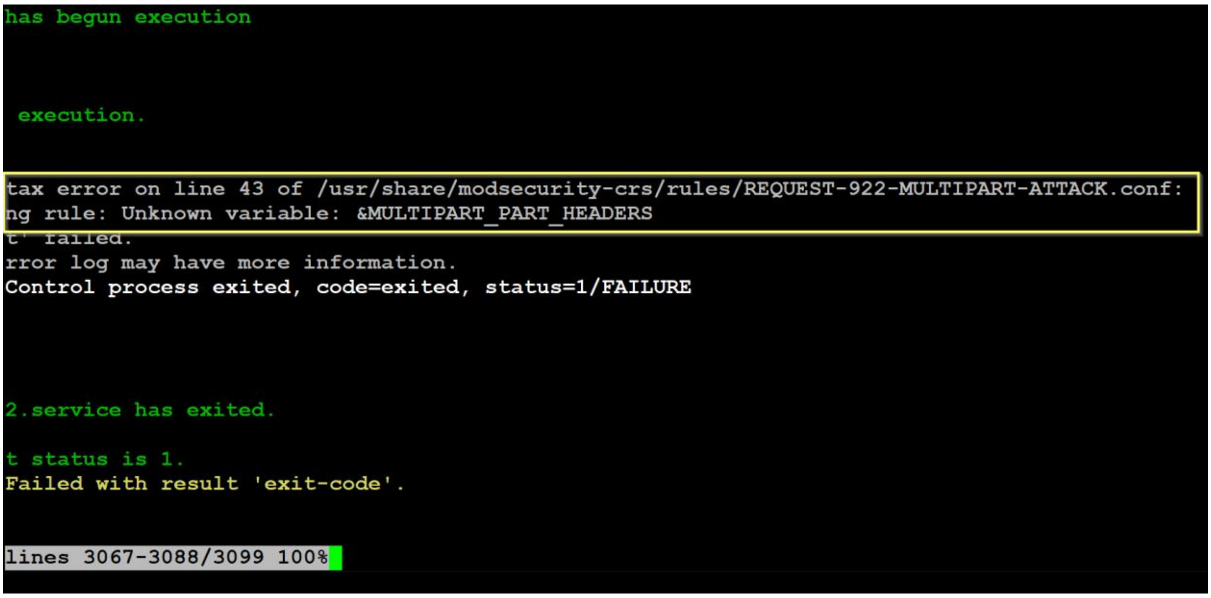
Figure 15 - Apache2 error when restarting the service

Use **journalctl -xe** to view which rule is giving us problems.



```
░ Subject: A start job for unit apache2.service has begun execution
░ Defined-By: systemd
░ Support: http://www.ubuntu.com/support
░
░ A start job for unit apache2.service has begun execution.
░
░ The job identifier is 1917.
Feb 26 17:14:27 waf apachectl[3853]: AH00526: Syntax error on line 43 of /usr/share/modsecurity-c>
Feb 26 17:14:27 waf apachectl[3853]: Error creating rule: Unknown variable: &MULTIPART_PART_HEADE>
Feb 26 17:14:27 waf apachectl[3850]: Action 'start' failed.
Feb 26 17:14:27 waf apachectl[3850]: The Apache error log may have more information.
Feb 26 17:14:27 waf systemd[1]: apache2.service: Control process exited, code=exited, status=1/FA>
░ Subject: Unit process exited
░ Defined-By: systemd
░ Support: http://www.ubuntu.com/support
░
░ An ExecStart= process belonging to unit apache2.service has exited.
░
░ The process' exit code is 'exited' and its exit status is 1.
Feb 26 17:14:27 waf systemd[1]: apache2.service: Failed with result 'exit-code'.
░ Subject: Unit failed
░ Defined-By: systemd
lines 3067-3088/3099 100%
```

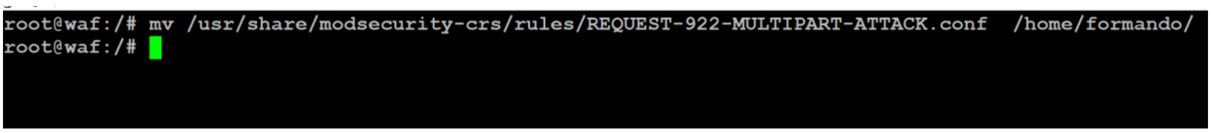
Figure 16 - journalctl -xe output



```
has begun execution



 execution.


tax error on line 43 of /usr/share/modsecurity-crs/rules/REQUEST-922-MULTIPART-ATTACK.conf:
ng rule: Unknown variable: &MULTIPART_PART_HEADERS
t' failed.
rror log may have more information.
Control process exited, code=exited, status=1/FAILURE




2.service has exited.

t status is 1.
Failed with result 'exit-code'.


lines 3067-3088/3099 100%
```

Figure 17 - journalctl -xe output

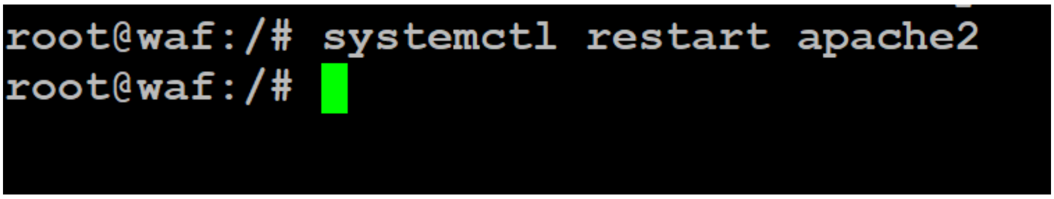
Let us move the rule to the home folder of our user.



```
root@waf:/# mv /usr/share/modsecurity-crs/rules/REQUEST-922-MULTIPART-ATTACK.conf  /home/formando/
root@waf:/# 
```

Figure 18 – Moving the rule that is giving an error to another folder

Let us restart apache again.

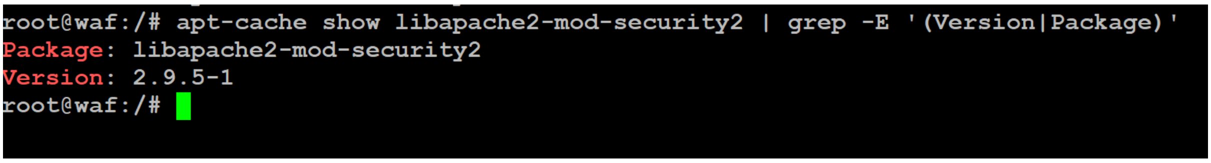


```
root@waf:/# systemctl restart apache2
root@waf:/# 
```

Figure 19 - Apache2 restarted successfully

If you want to know which version of modsecurity you are using, you can use the following command:

apt-cache show libapache2-mod-security2 | grep -E '(Version|Package)'



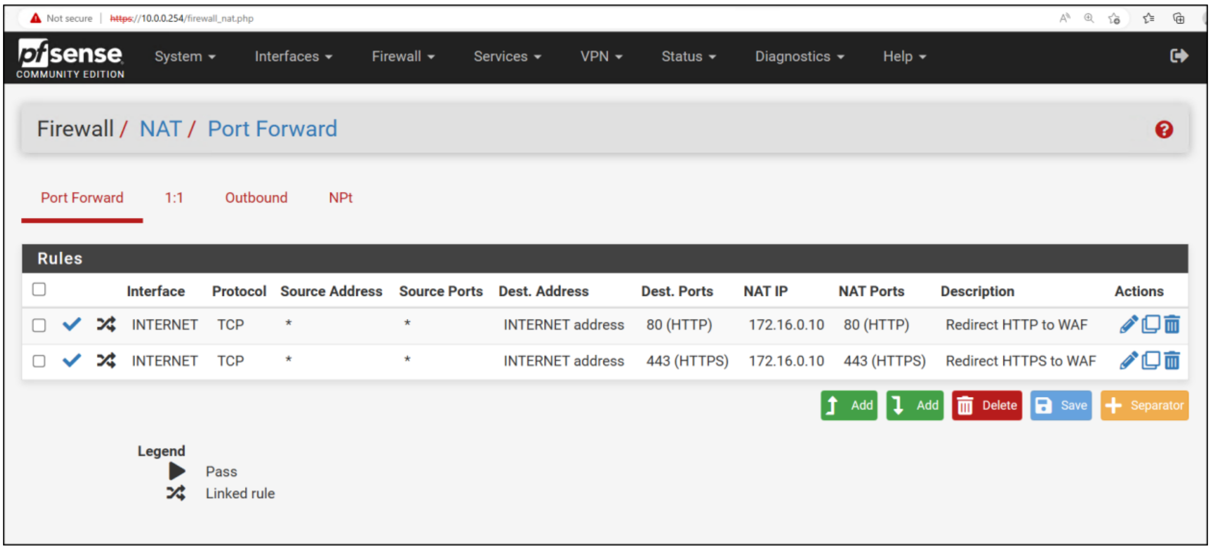
```
root@waf:/# apt-cache show libapache2-mod-security2 | grep -E '(Version|Package)'
Package: libapache2-mod-security2
Version: 2.9.5-1
root@waf:/# 
```

Figure 20 - View which version of apache2 you are currently using

Now we need to make sure that ModSecurity is working.

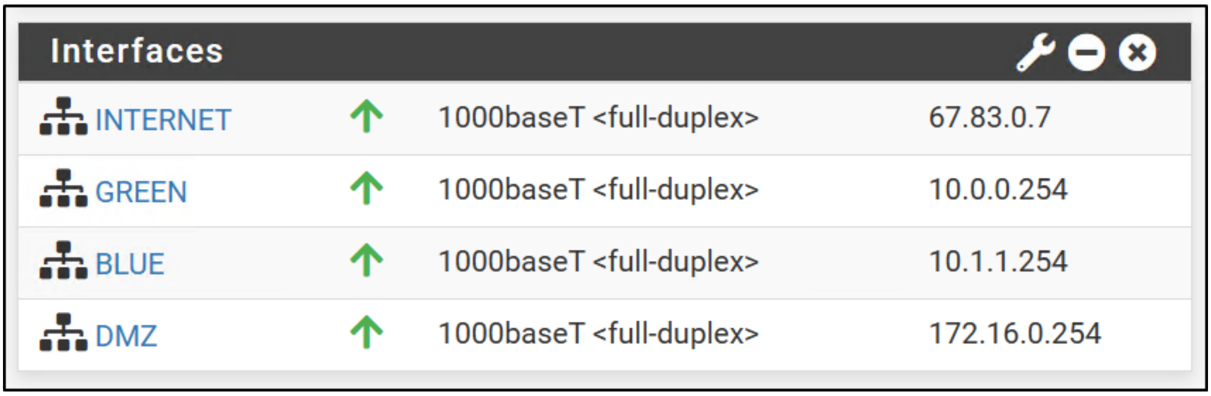
To test if the system is working, we'll be using a Kali Linux machine on the "INTERNET netwok (NAT Network)" showed previously in our network typology.

Keep in mind that our Firewall already has packet forwarding setup on it's NAT configuration.



□			Interface	Protocol	Source Address	Source Ports	Dest. Address	Dest. Ports	NAT IP	NAT Ports	Description	Actions
□	✓	⤨	INTERNET	TCP	*	*	INTERNET address	80 (HTTP)	172.16.0.10	80 (HTTP)	Redirect HTTP to WAF	
□	✓	⤨	INTERNET	TCP	*	*	INTERNET address	443 (HTTPS)	172.16.0.10	443 (HTTPS)	Redirect HTTPS to WAF	

Figure 21 - Port forwarding configuration in our PfSense



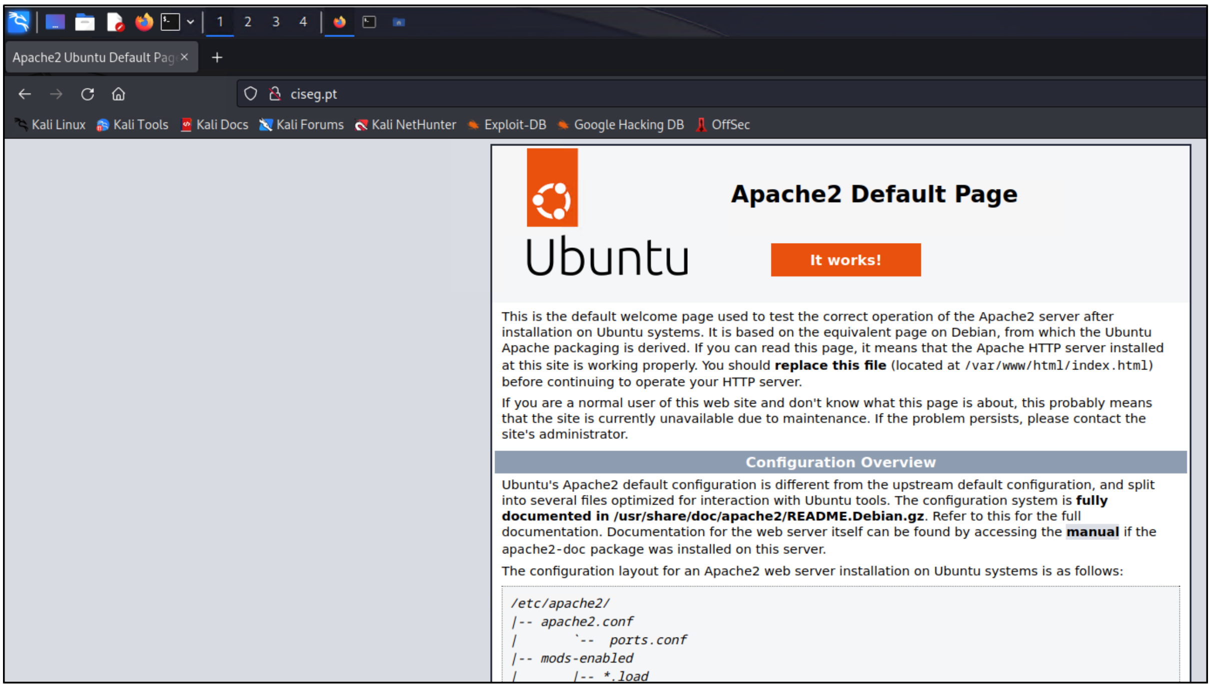
Interfaces			
INTERNET	↑	1000baseT <full-duplex>	67.83.0.7
GREEN	↑	1000baseT <full-duplex>	10.0.0.254
BLUE	↑	1000baseT <full-duplex>	10.1.1.254
DMZ	↑	1000baseT <full-duplex>	172.16.0.254

Figure 22 - Firewall networks

From what we can see in our firewall, the 67.83.0.7 is our public interface, and Kali will try to connect to our WAF through that IP address.

NOTE: This "public" IP address is set in a controlled environment and is not in any way connected or related to the real 67.83.0.7 IP address in the real world.

In Kali we can confirm that we can connect ("ciseg.pt" is defined in our hosts for this lab)



Apache2 Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. 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.

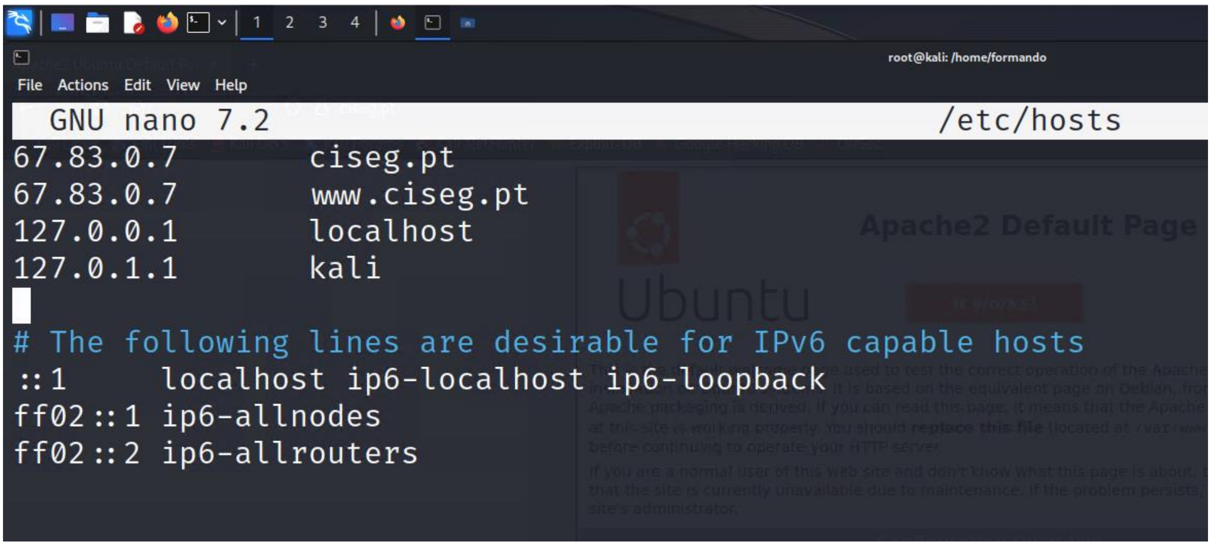
Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu 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 Ubuntu systems is as follows:

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

Figure 23 - Web page access from Kali Linux

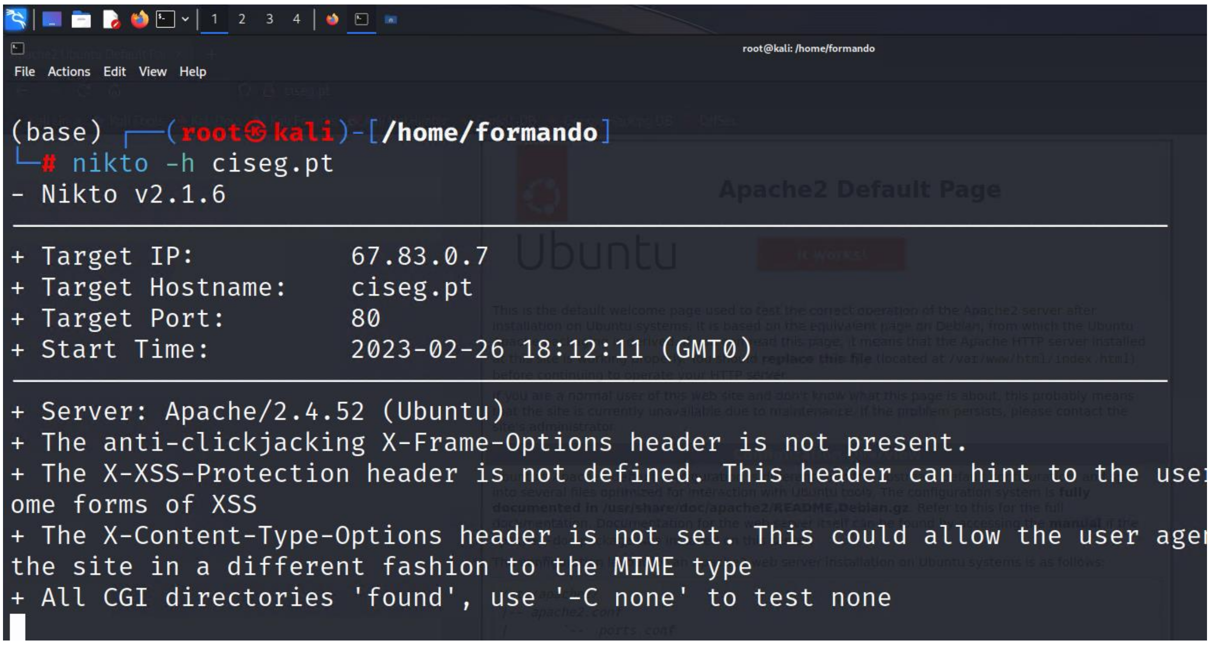


```
  GNU nano 7.2                                      /etc/hosts
67.83.0.7       ciseg.pt
67.83.0.7       www.ciseg.pt
127.0.0.1       localhost
127.0.1.1       kali

# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Figure 24 - hosts configuration on Kali Linux

Let us do a scan with Nikto and see if our ModSecurity is stopping malicious requests.

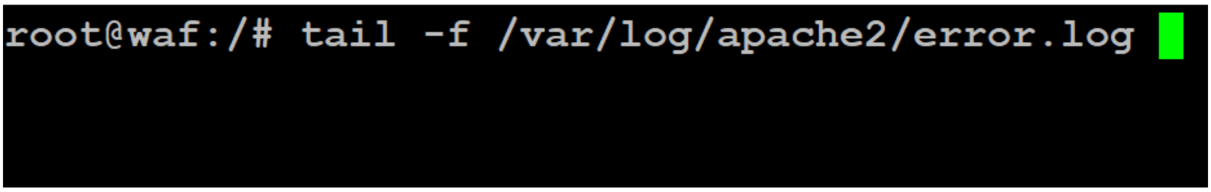
On kali run the scan.



```
(base) ┌──(root㉿kali)-[/home/formando]
└─# nikto -h ciseg.pt
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          67.83.0.7
+ Target Hostname:    ciseg.pt
+ Target Port:        80
+ Start Time:         2023-02-26 18:12:11 (GMT0)
---------------------------------------------------------------------------
+ Server: Apache/2.4.52 (Ubuntu)
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the use
ome forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user ager
the site in a different fashion to the MIME type
+ All CGI directories 'found', use '-C none' to test none
```

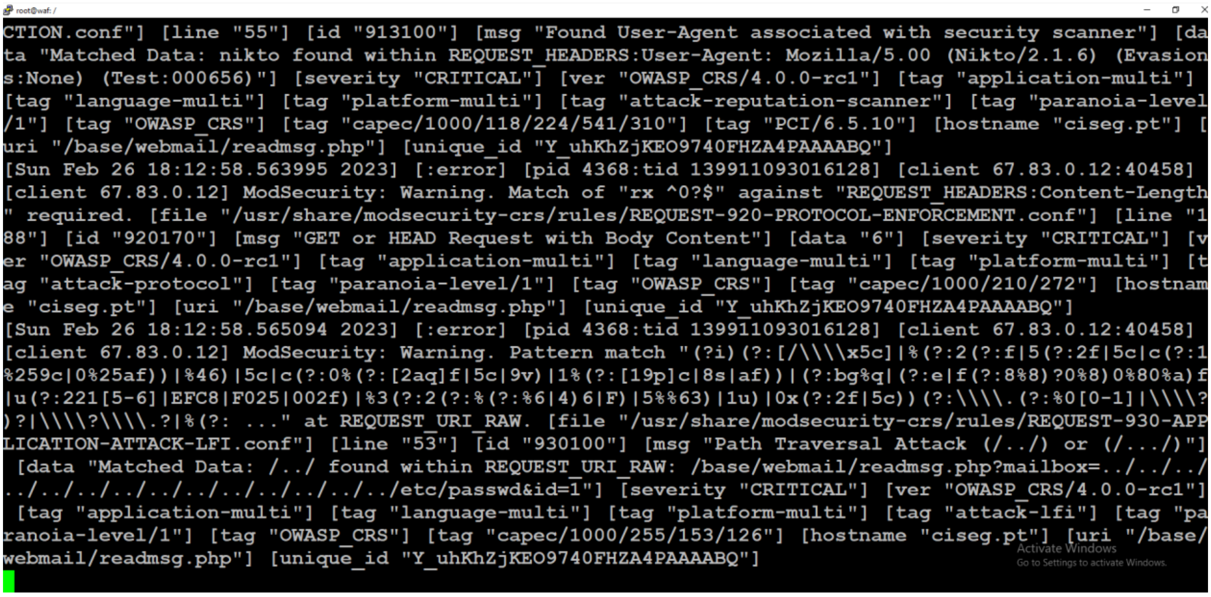
Figure 25 - Nikto attempting to scan the ciseg.pt website

We can see the logs generated by modsecurity in the following **error.log** file.



```
root@waf:/# tail -f /var/log/apache2/error.log
```

Figure 26 - See apache2 error logs

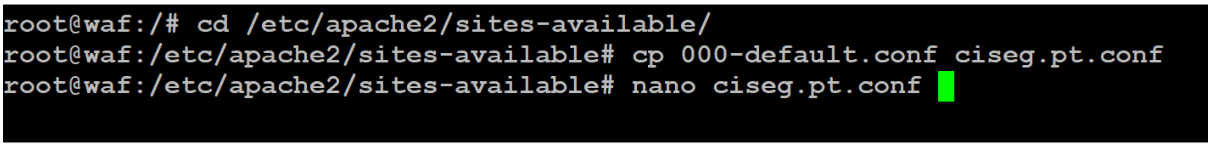


```
CTION.conf"] [line "55"] [id "913100"] [msg "Found User-Agent associated with security scanner"] [da
ta "Matched Data: nikto found within REQUEST_HEADERS:User-Agent: Mozilla/5.00 (Nikto/2.1.6) (Evasion
s:None) (Test:000656)"] [severity "CRITICAL"] [ver "OWASP_CRS/4.0.0-rc1"] [tag "application-multi"]
[tag "language-multi"] [tag "platform-multi"] [tag "attack-reputation-scanner"] [tag "paranoia-level
/1"] [tag "OWASP_CRS"] [tag "capec/1000/118/224/541/310"] [tag "PCI/6.5.10"] [hostname "ciseg.pt"] [
uri "/base/webmail/readmsg.php"] [unique_id "Y_uhKhZjKEO9740FHZA4PAAAABQ"]
[Sun Feb 26 18:12:58.563995 2023] [:error] [pid 4368:tid 139911093016128] [client 67.83.0.12:40458]
[client 67.83.0.12] ModSecurity: Warning. Match of "rx ^0?$" against "REQUEST_HEADERS:Content-Length
" required. [file "/usr/share/modsecurity-crs/rules/REQUEST-920-PROTOCOL-ENFORCEMENT.conf"] [line "1
88"] [id "920170"] [msg "GET or HEAD Request with Body Content"] [data "6"] [severity "CRITICAL"] [v
er "OWASP_CRS/4.0.0-rc1"] [tag "application-multi"] [tag "language-multi"] [tag "platform-multi"] [t
ag "attack-protocol"] [tag "paranoia-level/1"] [tag "OWASP_CRS"] [tag "capec/1000/210/272"] [hostnam
e "ciseg.pt"] [uri "/base/webmail/readmsg.php"] [unique_id "Y_uhKhZjKEO9740FHZA4PAAAABQ"]
[Sun Feb 26 18:12:58.565094 2023] [:error] [pid 4368:tid 139911093016128] [client 67.83.0.12:40458]
[client 67.83.0.12] ModSecurity: Warning. Pattern match "(?i)(?:[/\\\\x5c]|%(?:2(?:f|5(?:2f|5c|c(?:1
%259c|0%25af))|%46)|5c|c(?:0%(?:[2aq]f|5c|9v)|1%(?:[19p]c|8s|af))|(?:bg%q|(?:e|f(?:8%8)?0%8)0%80%a)f
|u(?:221[5-6]|EFC8|F025|002f)|%3(?:2(?:%(?:%6|4)6|F)|5%%63)|1u)|0x(?:2f|5c))(?:\\\\.(?:%0[0-1]|\\\\?
)?|\\\\?\\\\.?|%(?: ..." at REQUEST_URI_RAW. [file "/usr/share/modsecurity-crs/rules/REQUEST-930-APP
LICATION-ATTACK-LFI.conf"] [line "53"] [id "930100"] [msg "Path Traversal Attack (/../) or (/.../)"]
 [data "Matched Data: /../ found within REQUEST_URI_RAW: /base/webmail/readmsg.php?mailbox=../../../
../../../../../../../../../../etc/passwd&id=1"] [severity "CRITICAL"] [ver "OWASP_CRS/4.0.0-rc1"]
 [tag "application-multi"] [tag "language-multi"] [tag "platform-multi"] [tag "attack-lfi"] [tag "pa
ranoia-level/1"] [tag "OWASP_CRS"] [tag "capec/1000/255/153/126"] [hostname "ciseg.pt"] [uri "/base/
webmail/readmsg.php"] [unique_id "Y_uhKhZjKEO9740FHZA4PAAAABQ"]
```

Figure 27 - Sample of messages being generated in error.log

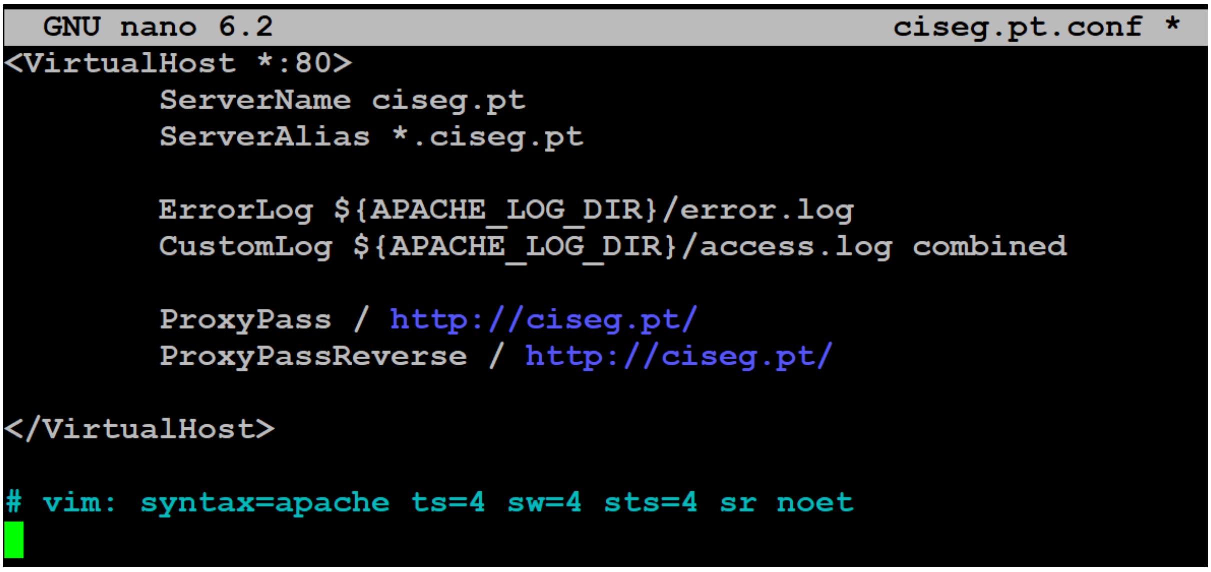
When we run the tail command the output will run very fast in your screen, while nikto is still running.

Since it is working, we should configure our reverse proxy and send allowed communications to the IIS server.



```
root@waf:/# cd /etc/apache2/sites-available/
root@waf:/etc/apache2/sites-available# cp 000-default.conf ciseg.pt.conf
root@waf:/etc/apache2/sites-available# nano ciseg.pt.conf
```

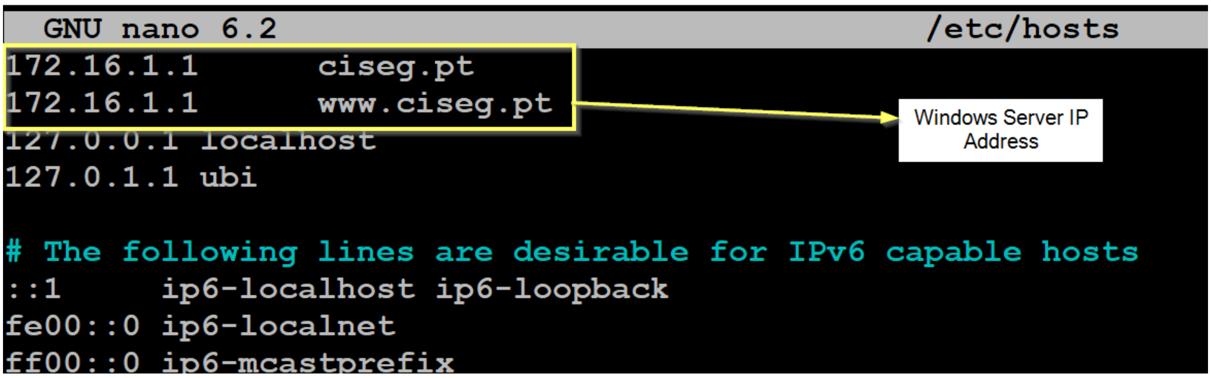
Figure 28 - Creation of ciseg.pt.conf configuration file



```
<VirtualHost *:80>
        ServerName ciseg.pt
        ServerAlias *.ciseg.pt

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

        ProxyPass / http://ciseg.pt/
        ProxyPassReverse / http://ciseg.pt/

</VirtualHost>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

Figure 29 - Reverse proxy configuration for HTTP connections

Make sure your WAF has hosts file configured.



```
  GNU nano 6.2                                          /etc/hosts
172.16.1.1      ciseg.pt
172.16.1.1      www.ciseg.pt
127.0.0.1 localhost
127.0.1.1 ubi

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
```

Windows Server IP Address

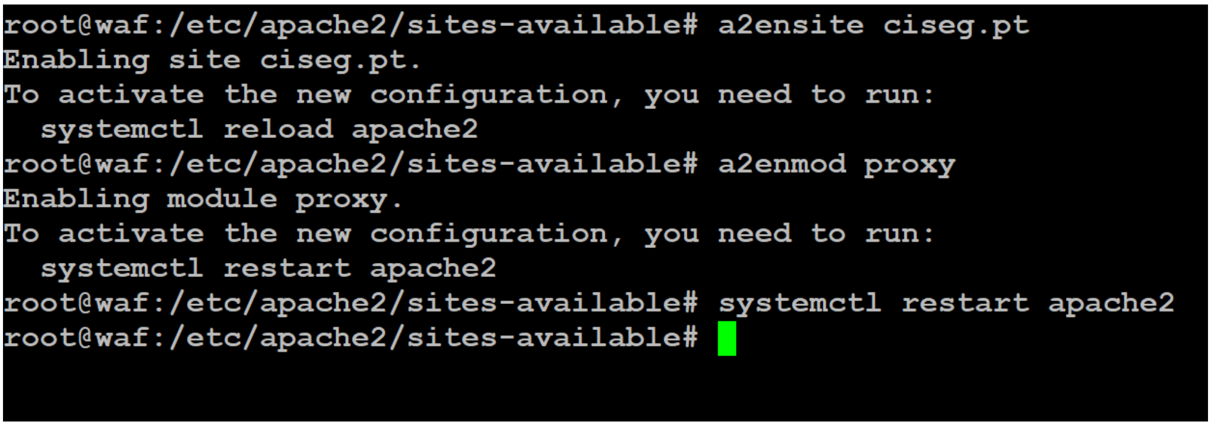
Figure 30 - WAF server hosts file configuration

Let us enable proxy module and our website, with

a2ensite ciseg.pt

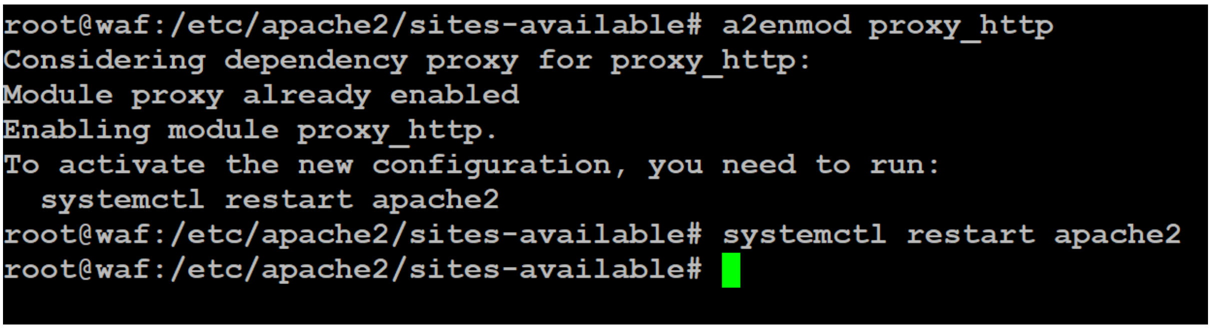
a2enmod proxy

a2enmod proxy_http



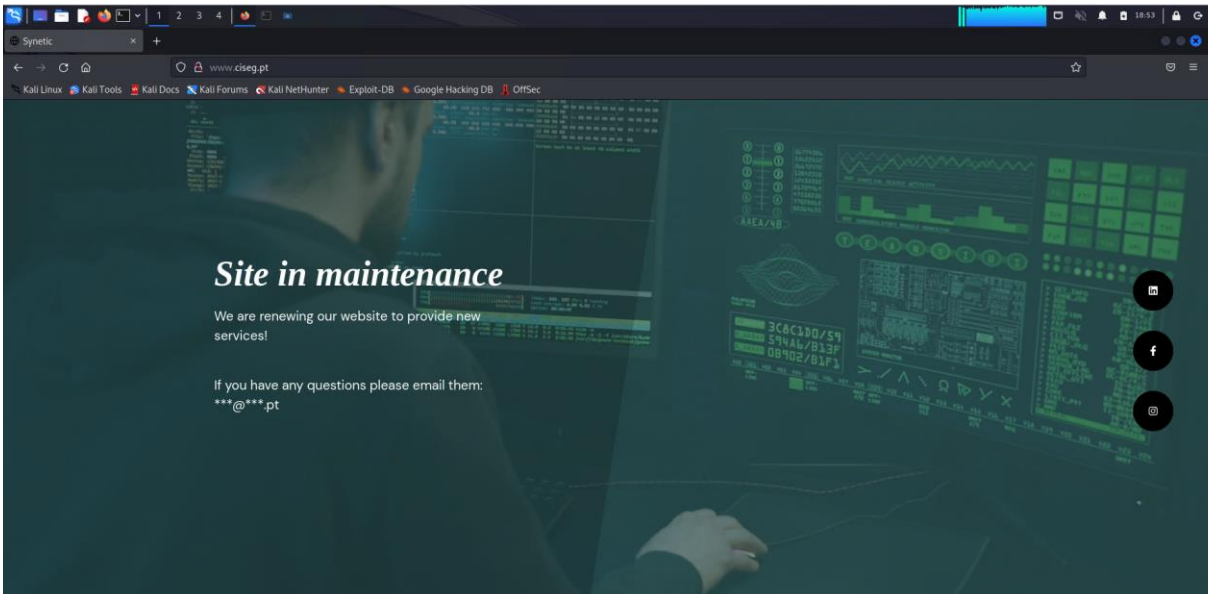
```
root@waf:/etc/apache2/sites-available# a2ensite ciseg.pt
Enabling site ciseg.pt.
To activate the new configuration, you need to run:
  systemctl reload apache2
root@waf:/etc/apache2/sites-available# a2enmod proxy
Enabling module proxy.
To activate the new configuration, you need to run:
  systemctl restart apache2
root@waf:/etc/apache2/sites-available# systemctl restart apache2
root@waf:/etc/apache2/sites-available# 
```

Figure 31 - Enabling website and proxy module



```
root@waf:/etc/apache2/sites-available# a2enmod proxy_http
Considering dependency proxy for proxy_http:
Module proxy already enabled
Enabling module proxy_http.
To activate the new configuration, you need to run:
  systemctl restart apache2
root@waf:/etc/apache2/sites-available# systemctl restart apache2
root@waf:/etc/apache2/sites-available# 
```

Figure 32 - enabling proxy_http module



Site in maintenance

We are renewing our website to provide new services!

If you have any questions please email them: ***@***.pt

Figure 33 - Accessing website from Kali Linux machine

Even with a proxy enabled you'll be able to connect to your website and ModSecurity will still be able to block malicious connections to our website.

This is just a small feature to improve your security posture, but take into consideration that we should activate Transport Layer Security.