

## Annex 1: Instal·lacions i configuracions WAF's i eines

### Instal·lació Nginx proxy server i WAF ModSecurity amb OWASP Core Rule Set

Com a primera tasca, es realitza la desinstal·lació de la versió Nginx que proporciona la VM predefinida de Dojo.

```
apt purge nginx nginx-common
apt autoremove

apt remove --purge nginx*
apt autoremove
apt update
```

A continuació, s'instal·len les dependències necessàries per Nginx

```
apt install make gcc build-essential autoconf automake libtool libfuzzy-dev sdeep gettext pkg-config libcurl4-openssl-dev liblua5.3-dev libpcre3 libpcre3-dev libxml2 libxml2-dev libyajl-dev doxygen libcurl4 libgeoip-dev libssl-dev zlib1g-dev libxslt-dev liblmbd-dev libpcre++-dev libgd-dev
```

S'afegeix el repositori que conté el Nginx adequat, s'actualitza la llista de paquets i es realitza la instal·lació

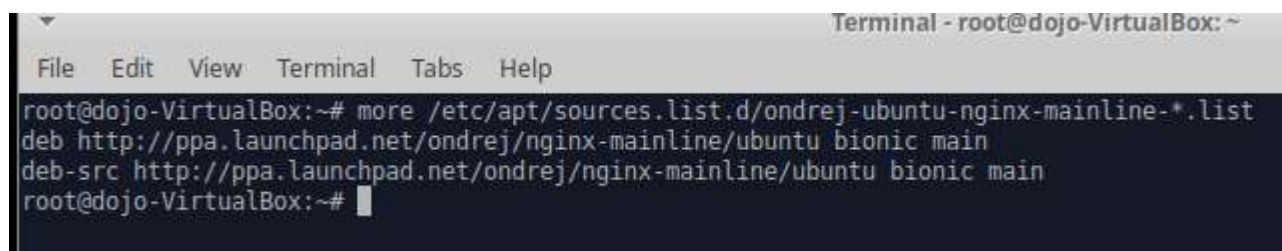
```
add-apt-repository ppa:ondrej/nginx-mainline -y

apt update
apt install nginx-core nginx-common nginx nginx-full
```

S'activa la possibilitat d'accedir al repositori de codi font del Nginx

```
vim /etc/apt/sources.list.d/ondrej-ubuntu-nginx-mainline-*.list
Locate and uncomment this line to enable the source code repository:

# deb-src http://ppa.launchpad.net/ondrej/nginx-mainline/ubuntu/ focal main
```



```
Terminal - root@dojo-VirtualBox: ~
File Edit View Terminal Tabs Help
root@dojo-VirtualBox:~# more /etc/apt/sources.list.d/ondrej-ubuntu-nginx-mainline-*.list
deb http://ppa.launchpad.net/ondrej/nginx-mainline/ubuntu bionic main
deb-src http://ppa.launchpad.net/ondrej/nginx-mainline/ubuntu bionic main
root@dojo-VirtualBox:~#
```

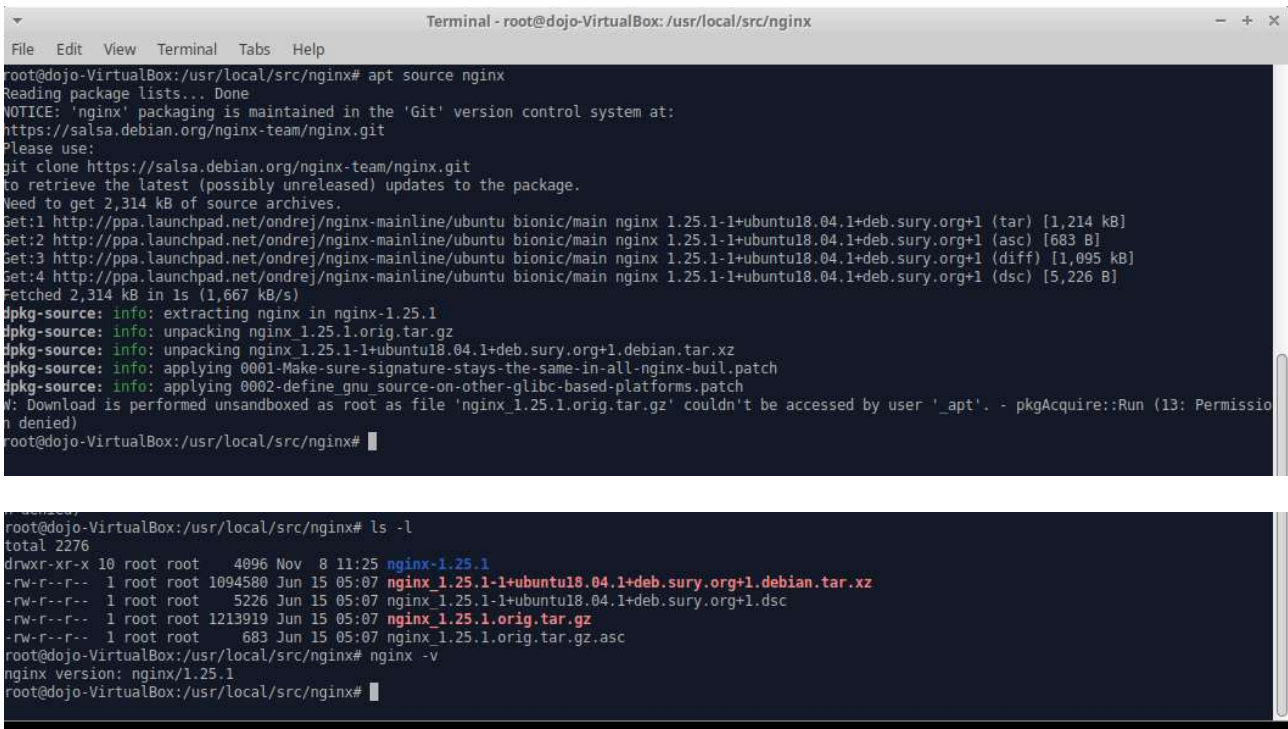
I s'actualitza el package index.

```
apt update
```

Per compilar el mòdul dinàmic de Modsecurity, cal descarregar el codi font de Nginx. Es crea un directori per allotjar aquest codi font i es descarrega en aquest directori mitjançant comanda apt i es verifica la versió instal·lada.

```
mkdir -p /usr/local/src/nginx
cd /usr/local/src/nginx

apt source nginx
```



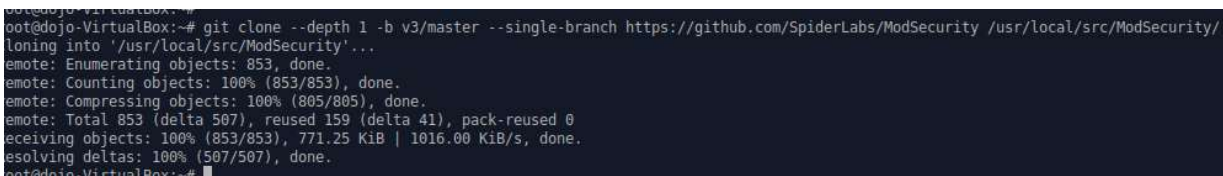
```
Terminal - root@dojo-VirtualBox: /usr/local/src/nginx
File Edit View Terminal Tabs Help

root@dojo-VirtualBox:/usr/local/src/nginx# apt source nginx
Reading package lists... Done
NOTICE: 'nginx' packaging is maintained in the 'Git' version control system at:
https://salsa.debian.org/nginx-team/nginx.git
Please use:
git clone https://salsa.debian.org/nginx-team/nginx.git
to retrieve the latest (possibly unreleased) updates to the package.
Need to get 2,314 kB of source archives.
Get:1 http://ppa.launchpad.net/ondrej/nginx-mainline/ubuntu bionic/main nginx 1.25.1-1+ubuntu18.04.1+deb.sury.org+1 (tar) [1,214 kB]
Get:2 http://ppa.launchpad.net/ondrej/nginx-mainline/ubuntu bionic/main nginx 1.25.1-1+ubuntu18.04.1+deb.sury.org+1 (asc) [683 B]
Get:3 http://ppa.launchpad.net/ondrej/nginx-mainline/ubuntu bionic/main nginx 1.25.1-1+ubuntu18.04.1+deb.sury.org+1 (diff) [1,095 kB]
Get:4 http://ppa.launchpad.net/ondrej/nginx-mainline/ubuntu bionic/main nginx 1.25.1-1+ubuntu18.04.1+deb.sury.org+1 (dsc) [5,226 B]
Fetched 2,314 kB in 1s (1,667 kB/s)
dpkg-source: info: extracting nginx in nginx-1.25.1
dpkg-source: info: unpacking nginx_1.25.1.orig.tar.gz
dpkg-source: info: unpacking nginx_1.25.1-1+ubuntu18.04.1+deb.sury.org+1.debian.tar.xz
dpkg-source: info: applying 0001-Make-sure-signature-stays-the-same-in-all-nginx-build.patch
dpkg-source: info: applying 0002-define-gnu-source-on-other-glibc-based-platforms.patch
W: Download is performed unsandboxed as root as file 'nginx_1.25.1.orig.tar.gz' couldn't be accessed by user '_apt'. - pkgAcquire::Run (13: Permission denied)
root@dojo-VirtualBox:/usr/local/src/nginx#

root@dojo-VirtualBox:/usr/local/src/nginx# ls -l
total 2276
drwxr-xr-x 10 root root 4096 Nov  8 11:25 nginx-1.25.1
-rw-r--r--  1 root root 1094580 Jun 15 05:07 nginx_1.25.1-1+ubuntu18.04.1+deb.sury.org+1.debian.tar.xz
-rw-r--r--  1 root root 5226 Jun 15 05:07 nginx_1.25.1-1+ubuntu18.04.1+deb.sury.org+1.dsc
-rw-r--r--  1 root root 1213919 Jun 15 05:07 nginx_1.25.1.orig.tar.gz
-rw-r--r--  1 root root 683 Jun 15 05:07 nginx_1.25.1.orig.tar.gz.asc
root@dojo-VirtualBox:/usr/local/src/nginx# nginx -v
nginx version: nginx/1.25.1
root@dojo-VirtualBox:/usr/local/src/nginx#
```

A continuació, s'instal·la la llibreria Libmodsecurity que gestiona el filtrat de les peticions HTTP. Es baixa la versió específica mitjançant comanda wget.

```
git clone --depth 1 -b v3/master --single-branch https://github.com/SpiderLabs/ModSecurity /usr/local/src/ModSecurity/
```



```
root@dojo-VirtualBox:~# git clone --depth 1 -b v3/master --single-branch https://github.com/SpiderLabs/ModSecurity /usr/local/src/ModSecurity/
Cloning into '/usr/local/src/ModSecurity'...
remote: Enumerating objects: 853, done.
remote: Counting objects: 100% (853/853), done.
remote: Compressing objects: 100% (805/805), done.
remote: Total 853 (delta 507), reused 159 (delta 41), pack-reused 0
Receiving objects: 100% (853/853), 771.25 KiB | 1016.00 KiB/s, done.
Resolving deltas: 100% (507/507), done.
root@dojo-VirtualBox:~#
```

Des de el directori clonat, s'instal·len els submòduls.

```
cd /usr/local/src/ModSecurity/

git submodule init
git submodule update
```

```

root@dojo-VirtualBox:~# cd /usr/local/src/ModSecurity/
root@dojo-VirtualBox:usr/local/src/ModSecurity# git submodule init
Submodule 'bindings/python' (https://github.com/SpiderLabs/ModSecurity-Python-bindings.git) registered for path 'bindings/python'
Submodule 'others/libinjection' (https://github.com/libinjection/libinjection.git) registered for path 'others/libinjection'
Submodule 'test/test-cases/secrules-language-tests' (https://github.com/SpiderLabs/secrules-language-tests) registered for path 'test/test-cases/secrules-language-tests'
root@dojo-VirtualBox:usr/local/src/ModSecurity# git submodule update
Cloning into '/usr/local/src/ModSecurity/bindings/python'...
Cloning into '/usr/local/src/ModSecurity/others/libinjection'...
Cloning into '/usr/local/src/ModSecurity/test/test-cases/secrules-language-tests'...
Submodule path 'bindings/python': checked out 'bc625d5bb0bac6a64bce8dc9902208612399348'
Submodule path 'others/libinjection': checked out 'bfba51f5af8f1f6cf5d6c4bf862f1e2474e018e3'
Submodule path 'test/test-cases/secrules-language-tests': checked out 'a3d4405e5a2c90488c387e589c5534974575e35b'
root@dojo-VirtualBox:usr/local/src/ModSecurity#
  
```

I es crea l'entorn a partir de les següents comandes.

```

./build.sh
./configure

Nota:
Ignore the error displayed below.
fatal: No names found, cannot describe anything.
  
```

Es compila el codi font i s'instal·len altres utilitats amb comanda make. Després s'instal·len les llibreries.

```

$ sudo make -j4
$ sudo make install
  
```

La següent tasca es baixar i compilar el conector de Modsecurity amb Nginx (ModSecurity v3 Nginx Connector). Aquest connector permet entrellçar la llibreria LibModSecurity amb el servidor web Nginx.

```

git clone --depth 1 https://github.com/SpiderLabs/ModSecurity-nginx.git /usr/local/src/ModSecurity-nginx/
  
```

```

root@dojo-VirtualBox:usr/local/src/ModSecurity# git clone --depth 1 https://github.com/SpiderLabs/ModSecurity-nginx.git /usr/local/src/ModSecurity-nginx/
Cloning into '/usr/local/src/ModSecurity-nginx'...
remote: Enumerating objects: 40, done.
remote: Counting objects: 100% (40/40), done.
remote: Compressing objects: 100% (38/38), done.
remote: Total 40 (delta 11), reused 11 (delta 0), pack-reused 0
Unpacking objects: 100% (40/40), done.
root@dojo-VirtualBox:usr/local/src/ModSecurity#
  
```

Des de el directori clonat, es construeixen les dependències.

```

cd /usr/local/src/nginx/nginx-1.21.3/

apt build-dep nginx
apt install uuid-dev
  
```

Es compila el modul connector amb l'opció --with compat per que aquest sigui compatible amb la llibreria Nginx descarregada. A continuació es construeix el modul amb comanda "make" i es copia l'objecte creat al directori dels moduls de Nginx.

```

./configure --with-compat --add-dynamic-module=/usr/local/src/ModSecurity-nginx
make modules
cp objs/ngx_http_modsecurity_module.so /usr/share/nginx/modules/
  
```

S'accedeix a la configuració de Nginx per carregar el modul connector acabat de crear. A més, s'incorpora en la section `http{...}` les línies que activen Modsecurity per tots els virtual hosts definits a Nginx.

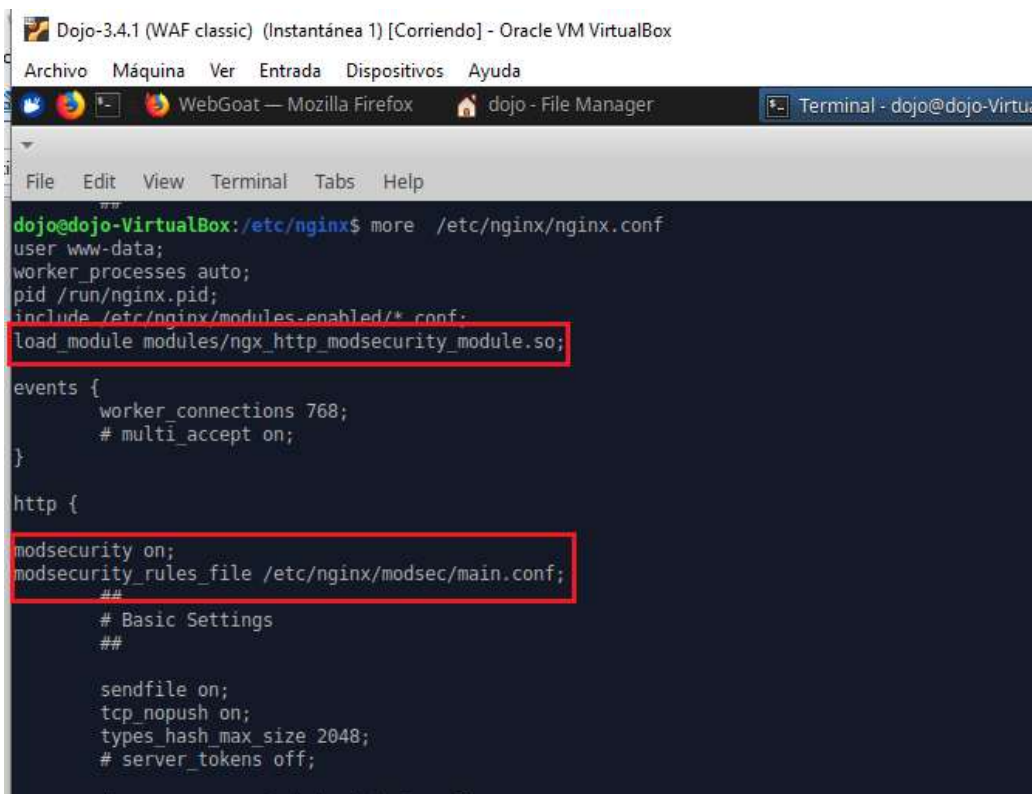
```
vim /etc/nginx/nginx.conf

#Append the following line just below the first few lines

load_module modules/ngx_http_modsecurity_module.so;

#In addition, append the following lines in the http {...} section. This enables ModSecurity for
#all Nginx virtual hosts.

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



```
Dojo-3.4.1 (WAF classic) (Instantánea 1) [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
WebGoat — Mozilla Firefox  dojo - File Manager  Terminal - dojo@dojo-Virtua
File  Edit  View  Terminal  Tabs  Help
dojo@dojo-VirtualBox:/etc/nginx$ more /etc/nginx/nginx.conf
user www-data;
worker_processes auto;
pid /run/nginx.pid;
include /etc/nginx/modules-enabled/*.conf;
load_module modules/ngx_http_modsecurity_module.so;

events {
    worker_connections 768;
    # multi_accept on;
}

http {

modsecurity on;
modsecurity_rules_file /etc/nginx/modsec/main.conf;
##
# Basic Settings
##

    sendfile on;
    tcp_nopush on;
    types_hash_max_size 2048;
    # server_tokens off;
```

A continuació es crea el directori que contindrà la configuració per ModSecurity i es copia la configuració recomanada en aquest directori. Es varia la directiva "SecRuleEngine" per que bloquegi els atacs web.

```
# Next up, create the /etc/nginx/modsec/ directory which will store ModSecurity configuration.
mkdir /etc/nginx/modsec/

#Next, copy the ModSecurity configuration file as follows.
cp /usr/local/src/ModSecurity/modsecurity.conf-recommended /etc/nginx/modsec/modsecurity.conf

#Then open the configuration file.
```

```
vim /etc/nginx/modsec/modsecurity.conf
#Locate the line beginning with the SecRuleEngine directive.

SecRuleEngine DetectionOnly

#Change the line to the line below

SecRuleEngine On
```

Es crea el fitxer main.conf i s'afegeix la referència del fitxer modsecurity.conf

```
vim /etc/nginx/modsec/main.conf

#Append this line to reference the /etc/nginx/modsec/modsecurity.conf configuration file.

Include /etc/nginx/modsec/modsecurity.conf
```

Es copia el fitxer de mapeig d'Unicode i es testeja la que la configuració de Nginx sigui correcta.

```
cp /usr/local/src/ModSecurity/unicode.mapping /etc/nginx/modsec/

#Then test Nginx configuration.

nginx -t
```

```
root@dojo-VirtualBox:/usr/local/src/nginx/nginx-1.25.1# nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
root@dojo-VirtualBox:/usr/local/src/nginx/nginx-1.25.1#
```

Es configura el servidor web Nginx perquè escolti pel port 8888 i derivi les peticions a les aplicacions que calgui en cada cas.

Al fitxer `/etc/nginx/sites-enabled/default` es canvia

```
server {
    listen      80;
    server_name localhost;

    #access_log /var/log/nginx/host.access.log main;

    location / {
        root    /usr/share/nginx/html;
        index  index.html index.htm;
    }
}
```

per

```
server {
    listen      8888;
    server_name localhost;

    #access_log /var/log/nginx/host.access.log main;
```

```

location / {
    root    /usr/share/nginx/html;
    index  index.html index.htm;
}
location /WebGoat { proxy_pass http://webgoat.local:8081/WebGoat; }

```

Es re-arrenca el Nginx per aplicar tots els canvis efectuats.

```
systemctl restart nginx
```

Com a tasca final, previa al testeig final, s'incorporen les regles del OWASP Corerule Set. Es baixen de Github a partir de comanda "wget".

```
wget https://github.com/coreruleset/coreruleset/archive/v3.3.5.tar.gz
```

```

root@dojo-VirtualBox:~# wget https://github.com/coreruleset/coreruleset/archive/v3.3.5.tar.gz
--2023-11-09 03:29:56-- https://github.com/coreruleset/coreruleset/archive/v3.3.5.tar.gz
Resolving github.com (github.com)... 140.82.121.4
Connecting to github.com (github.com)|140.82.121.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://codeload.github.com/coreruleset/coreruleset/tar.gz/refs/tags/v3.3.5 [following]
--2023-11-09 03:29:56-- https://codeload.github.com/coreruleset/coreruleset/tar.gz/refs/tags/v3.3.5
Resolving codeload.github.com (codeload.github.com)... 140.82.121.10
Connecting to codeload.github.com (codeload.github.com)|140.82.121.10|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/x-gzip]
Saving to: 'v3.3.5.tar.gz'

v3.3.5.tar.gz          [ <=> ] 292.44K  1.57MB/s  in 0.2s

2023-11-09 03:29:57 (1.57 MB/s) - 'v3.3.5.tar.gz' saved [299458]

root@dojo-VirtualBox:~#

```

Es descomprimeix el gzip i es mou el directori al path adient. S'usa el exemple del crs-setup com a fitxer inicial de configuració de regles.

```

tar xvf v3.3.5.tar.gz

#Ensure to move the uncompressed directory to the /etc/nginx/modsec/ path.

mv coreruleset-3.3.5/ /etc/nginx/modsec/

#Then rename the crs-setup.conf.example file to crs-setup.conf.

mv /etc/nginx/modsec/coreruleset-3.3.5/crs-setup.conf.example
/etc/nginx/modsec/coreruleset-3.3.5/crs-setup.conf

```

Es configura el main.conf del servidor Nginx incorporant els directoris/fitxers de regles.

```

vim /etc/nginx/modsec/main.conf
# And append the following lines.

Include /etc/nginx/modsec/coreruleset-3.3.5/crs-setup.conf
Include /etc/nginx/modsec/coreruleset-3.3.5/rules/*.conf

```

Es re-arrenca el Nginx per aplicar tots els canvis efectuats.



```
systemctl restart nginx
```

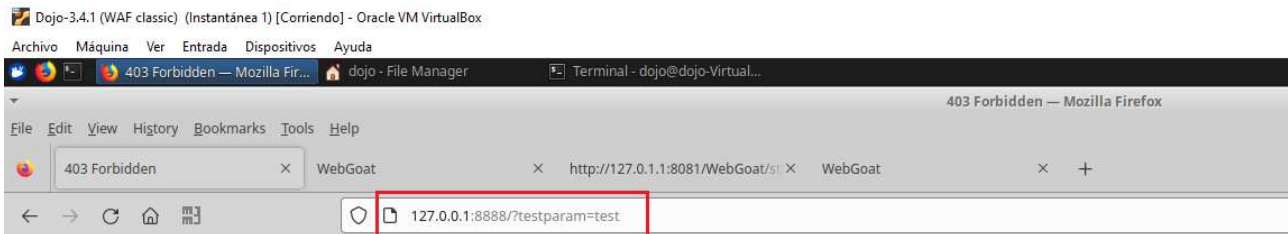
Per comprovar el correcte funcionament de Modsecurity, confirmant que detecta i bloqueja el transit sospitós, es crea una regla que bloqueja una URL concreta quan s'accedeix a través del navegador.

```
#vim /etc/nginx/modsec/modsecurity.conf
Add this line just below the SecRuleEngine On directive

SecRule ARGS:testparam "@contains test" "id:254,deny,status:403,msg:'Test Successful'"
```

S'accedeix a la URL

```
http://127.0.0.1:8888/?testparam=test
```



**403 Forbidden**

nginx/1.25.1

I es comprova també en el log del Nginx



Referència: <https://www.tecmint.com/install-modsecurity-nginx-debian-ubuntu/>

## Instal·lació Nginx proxy server i WAF open-appsec

El WAF open-appsec requereix de una versió de Nginx 1.22 o superior.

Per això, es baixa la versió específica mitjançant comanda wget:

```
wget https://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.22.0-1~bionic_amd64.deb
```

A continuació, s'instal·la la versió baixada:

```
sudo dpkg -i nginx_1.22.0-1~bionic_amd64.deb
```

Es configura el servidor web Nginx perquè escolti pel port 8888 i derivi les peticions a les aplicacions que calgui en cada cas.

Al fitxer `/etc/nginx/conf.d/default.conf` es canvia

```
server {
    listen      80;
    server_name localhost;

    #access_log /var/log/nginx/host.access.log main;

    location / {
        root    /usr/share/nginx/html;
        index  index.html index.htm;
    }
}
```

per

```
server {
    listen      8888;
    server_name localhost;

    #access_log /var/log/nginx/host.access.log main;

    location / {
        root    /usr/share/nginx/html;
        index  index.html index.htm;
    }
    location /WebGoat { proxy_pass http://webgoat.local:8081/WebGoat; }
```

Posteriorment, es realitza la instal·lació del WAF open-appsec. Es baixa mitjançant comanda wget.

```
wget https://downloads.openappsec.io/open-appsec-install && chmod +x open-appsec-install
```



```

Terminal - root@dojo-VirtualBox: -
File Edit View Terminal Tabs Help
root@dojo-VirtualBox:~# wget https://downloads.openappsec.io/open-appsec-install && chmod +x open-appsec-install
-2023-10-24 11:31:08-- https://downloads.openappsec.io/open-appsec-install
resolving downloads.openappsec.io (downloads.openappsec.io)... 108.157.109.57, 108.157.109.73, 108.157.109.18, ...
connecting to downloads.openappsec.io (downloads.openappsec.io)|108.157.109.57|443... connected.
HTTP request sent, awaiting response... 200 OK
length: 17712 (17K) [binary/octet-stream]
saving to: 'open-appsec-install'

open-appsec-install 100%[=====] 17.30K --KB/s in 0s
-2023-10-24 11:31:09 (191 MB/s) - 'open-appsec-install' saved [17712/17712]
root@dojo-VirtualBox:~#

```

I, s'instal·la el WAF integrant-se automaticament amb el servidor Nginx.

```

./open-appsec-install --auto --prevent

```

```

root@dojo-VirtualBox:~# ./open-appsec-install --auto --prevent
open-appsec for NGINX and Kong Installer v1.2245.1
For release notes and known limitations check:
https://docs.openappsec.io/release-notes
Searching local NGINX...
NGINX version found: 1.22.0-1-bionic
Downloading open-appsec NGINX attachment... stored in '/tmp/open-appsec'
Downloading open-appsec agent... stored in '/tmp/open-appsec'
Add your email to receive important security updates and so you can approach us with technical questions (enter IGNORE to ignore):
IGNORE
Installing open-appsec for NGINX...
Updating NGINX server configuration...
Starting open-appsec installation...
Setting mode to prevent-learn...
Successfully installed open-appsec for NGINX and Kong...
For release notes and known limitations check: https://docs.openappsec.io/release-notes
For troubleshooting and support: https://openappsec.io/support
root@dojo-VirtualBox:~#

```

Referència: <https://medium.com/@bhojport/installing-a-specific-version-of-nginx-on-ubuntu-or-any-other-platform-4fed8e859534>

## Instal·lació eina waf comparison project

En primera instancia, es baixa el projecte desde Github mitjançant comanda "git clone":

```
Terminal - root@dojo-VirtualBox: ~
File Edit View Terminal Tabs Help

root@dojo-VirtualBox:~# git clone https://github.com/openappsec/waf-comparison-project.git
Cloning into 'waf-comparison-project'...
remote: Enumerating objects: 54, done.
remote: Counting objects: 100% (54/54), done.
remote: Compressing objects: 100% (40/40), done.
remote: Total 54 (delta 23), reused 32 (delta 10), pack-reused 0
Unpacking objects: 100% (54/54), done.
root@dojo-VirtualBox:~#
```

Posteriorment, es realitza la instal·lació dels requeriments de Python necessaris per l'eina.

```
Terminal - root@dojo-VirtualBox: ~/waf-comparison-project
File Edit View Terminal Tabs Help

root@dojo-VirtualBox:~# cd waf-comparison-project/
root@dojo-VirtualBox:~/waf-comparison-project# pip install -r requirements.txt
Collecting SQLAlchemy (from -r requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/4a/e8/79a08e64a6440c632f3d657fe76a2eb8cf29571cfa3318c44aa669193682/SQLAlchemy-1.4.49-cp27-cp27mu-manylinux_2_5_x86_64.manylinux1_x86_64.whl (1.6MB)
  100% |#####| 1.6MB 232kB/s
Requirement already satisfied: requests in /usr/local/lib/python2.7/dist-packages (from -r requirements.txt (line 2))
Collecting colorlog (from -r requirements.txt (line 3))
  Downloading https://files.pythonhosted.org/packages/32/e6/e9ddc6f1104fda718338b341e4b3dc1cd8039ab29e52fc73b508515361/colorlog-5.0.1-py2.py3-none-any.whl
  Downloading https://files.pythonhosted.org/packages/db/83/7d4808ffcc2988066ff37f6a0bb6d7b68822367dcb36ba5e39aa7801fda54/pandas-0.24.2-cp27-cp27mu-manylinux1_x86_64.whl (10.1MB)
  100% |#####| 10.1MB 80kB/s
Collecting tqdm (from -r requirements.txt (line 5))
  Downloading https://files.pythonhosted.org/packages/47/bb/849011636c4da2e44f1253cd927cfb20ada4374d8b3a4e425416e84900cc/tqdm-4.64.1-py2.py3-none-any.whl (78kB)
  100% |#####| 81kB 1.2MB/s
Collecting plotly (from -r requirements.txt (line 6))
  Downloading https://files.pythonhosted.org/packages/1f/f6/bd3c17c8003b6641df1228e80e1acac97ed8402635e46c2571f8e1ef63af/plotly-4.14.3-py2.py3-none-any.whl (13.2MB)
  100% |#####| 13.2MB 62kB/s
Requirement already satisfied: importlib-metadata; python version < "3.8" in /usr/local/lib/python2.7/dist-packages (from SQLAlchemy->-r requirements.txt (line 1))
Requirement already satisfied: urllib3!=1.25.0,!1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python2.7/dist-packages (from requests->-r requirements.txt (line 2))
Requirement already satisfied: certifi=2017.4.17 in /usr/local/lib/python2.7/dist-packages (from requests->-r requirements.txt (line 2))
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in /usr/local/lib/python2.7/dist-packages (from requests->-r requirements.txt (line 2))
Requirement already satisfied: idna<2.9,>=2.5 in /usr/lib/python2.7/dist-packages (from requests->-r requirements.txt (line 2))
Collecting numpy=1.12.0 (from pandas->-r requirements.txt (line 4))
  Downloading https://files.pythonhosted.org/packages/3a/5f/47e578b3ae79e2624e205445ab77a1848acdaa2929a00eeef6b16eaaeb20/numpy-1.16.6-cp27-cp27mu-manylinux1_x86_64.whl (17.0MB)
  100% |#####| 17.0MB 48kB/s
Requirement already satisfied: pytz=2011k in /usr/local/lib/python2.7/dist-packages (from pandas->-r requirements.txt (line 4))
Requirement already satisfied: python-dateutil=2.5.0 in /usr/local/lib/python2.7/dist-packages (from pandas->-r requirements.txt (line 4))
Collecting importlib-resources; python version < "3.7" (from tqdm->-r requirements.txt (line 5))
  Downloading https://files.pythonhosted.org/packages/6d/94/2f6ee0e4e630ff0177c87e68d21e937a19fbc77c4739755b49f5adb04/importlib-resources-3.3.1-py2.py3-none-any.whl
Requirement already satisfied: six in /usr/lib/python2.7/dist-packages (from plotly->-r requirements.txt (line 6))
Collecting retrying=1.3.3 (from plotly->-r requirements.txt (line 6))
  Downloading https://files.pythonhosted.org/packages/ce/70/15ce8551d65b324e18c5aa6ef699880f21ead51ebe5ed743c0950d7d9dd/retrying-1.3.4.tar.gz
Requirement already satisfied: zipp=0.5 in /usr/local/lib/python2.7/dist-packages (from importlib-metadata; python version < "3.8">-SQLAlchemy->-r requirements.txt (line 1))
Requirement already satisfied: configparser=3.5; python version < "3" in /usr/local/lib/python2.7/dist-packages (from importlib-metadata; python version < "3.8">-SQLAlchemy->-r requirements.txt (line 1))
Requirement already satisfied: contextlib2; python version < "3" in /usr/local/lib/python2.7/dist-packages (from importlib-metadata; python version < "3.8">-SQLAlchemy->-r requirements.txt (line 1))
Requirement already satisfied: pathlib2; python version < "3" in /usr/local/lib/python2.7/dist-packages (from importlib-metadata; python version < "3.8">-SQLAlchemy->-r requirements.txt (line 1))
Collecting typing; python version < "3.5" (from importlib-resources; python version < "3.7">-tqdm->-r requirements.txt (line 5))
  Downloading https://files.pythonhosted.org/packages/0b/cb/da856e81731833b94da70a08712f658416266a5fb2a9d9e426c8061becf/typing-3.10.0-py2-none-any.whl
Requirement already satisfied: singledispatch; python version < "3.4" in /usr/local/lib/python2.7/dist-packages (from importlib-resources; python version < "3.7">-tqdm->-r requirements.txt (line 5))
Requirement already satisfied: scandir; python version < "3.5" in /usr/local/lib/python2.7/dist-packages (from pathlib2; python version < "3">-importlib-metadata; python version < "3.8">-SQLAlchemy->-r requirements.txt (line 1))
Building wheels for collected packages: retrying
  Running setup.py bdist_wheel for retrying ... done
  Stored in directory: /root/.cache/pip/wheels/22/f5/12/b3b4d1c867e4dd0870b548a43335d1c6d32da50c9b181d5ef
Successfully built retrying
Installing collected packages: SQLAlchemy, colorlog, numpy, pandas, typing, importlib-resources, tqdm, retrying, plotly
Successfully installed SQLAlchemy-1.4.49 colorlog-5.0.1 importlib-resources-3.3.1 numpy-1.16.6 pandas-0.24.2 plotly-4.14.3 retrying-1.3.4 tqdm-4.64.1 typing-3.10.0
root@dojo-VirtualBox:~/waf-comparison-project#
```

La configuració es realitza en el fitxer config.py. S'especifica el engine de base de dades, les ubicacions dels payloads i les URL's dels WAF's en estudi.

```

Terminal - root@dojo-VirtualBox: ~/waf-comparison-project
File Edit View Terminal Tabs Help
root@dojo-VirtualBox:~/waf-comparison-project# more config.py
from sqlalchemy import create_engine
from pathlib import Path

# Database configuration
#engine = create_engine(rf"sqlite:///waf_comparison.db")
engine = create_engine("sqlite:///root/waf-comparison-project/waf_comparison.db")

# Data set paths
#LEGITIMATE_URL_PATH = "https://downloads.openappsec.io/waf-comparison-project/legitimate.zip"
#MALICIOUS_URL_PATH = "https://downloads.openappsec.io/waf-comparison-project/malicious.zip"
LEGITIMATE_URL_PATH = "http://localhost:8888/LegitimateMostres.zip"
MALICIOUS_URL_PATH = "http://localhost:8888/sqli.zip"

# Data set Path
DATA_PATH = Path('Data')
LEGITIMATE_PATH = DATA_PATH / 'Legitimate'
MALICIOUS_PATH = DATA_PATH / 'Malicious'

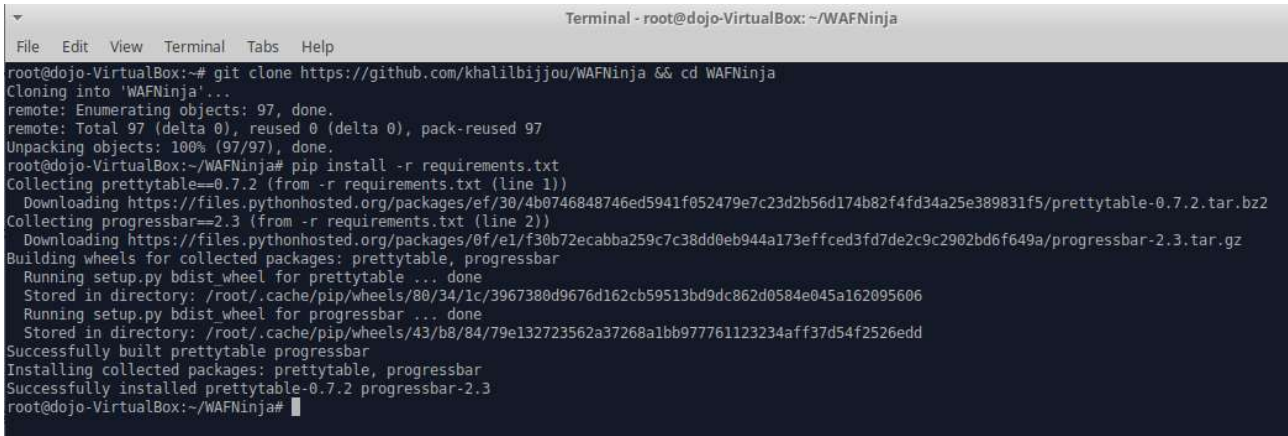
# WAF configuration
WAFS_DICT = {
    "open-appsec": 'http://127.0.0.1:8888/WebGoat',
    "nginx-Modsecurity": 'http://192.168.1.65:8888/WebGoat'
}
root@dojo-VirtualBox:~/waf-comparison-project#

```

## Instal·lació eina WafNinja

Per la baixada i instal·lació de l'eina, es clona mitjançant comanda git, i s'executen les dependències de Python.

```
git clone https://github.com/khalilbijjou/WAFNinja && cd WAFNinja
pip install -r requirements.txt
```



```
Terminal - root@dojo-VirtualBox: ~/WAFNinja
File Edit View Terminal Tabs Help
root@dojo-VirtualBox:~# git clone https://github.com/khalilbijjou/WAFNinja && cd WAFNinja
Cloning into 'WAFNinja'...
remote: Enumerating objects: 97, done.
remote: Total 97 (delta 0), reused 0 (delta 0), pack-reused 97
Unpacking objects: 100% (97/97), done.
root@dojo-VirtualBox:~/WAFNinja# pip install -r requirements.txt
Collecting prettytable==0.7.2 (from -r requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/ef/30/4b0746848746ed5941f052479e7c23d2b56d174b82f4fd34a25e389831f5/prettytable-0.7.2.tar.bz2
Collecting progressbar==2.3 (from -r requirements.txt (line 2))
  Downloading https://files.pythonhosted.org/packages/0f/e1/f30b72ecabba259c7c38dd0eb944a173effced3fd7de2c9c2902bd6f649a/progressbar-2.3.tar.gz
Building wheels for collected packages: prettytable, progressbar
  Running setup.py bdist_wheel for prettytable ... done
  Stored in directory: /root/.cache/pip/wheels/80/34/1c/3967380d9676d162cb59513bd9dc862d0584e045a162095606
  Running setup.py bdist_wheel for progressbar ... done
  Stored in directory: /root/.cache/pip/wheels/43/b8/84/79e132723562a37268a1bb977761123234aff37d54f2526edd
Successfully built prettytable progressbar
Installing collected packages: prettytable, progressbar
Successfully installed prettytable-0.7.2 progressbar-2.3
root@dojo-VirtualBox:~/WAFNinja#
```