Universidad Nacional
Autónoma de México
UNAM

DIRECCIÓN GENERAL DE CÓMPUTO Y DE
TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIÓN

03-marzo-2016

“PROYECTO” MODULO 2

Autor: Vega Tellez Jesus Antonio

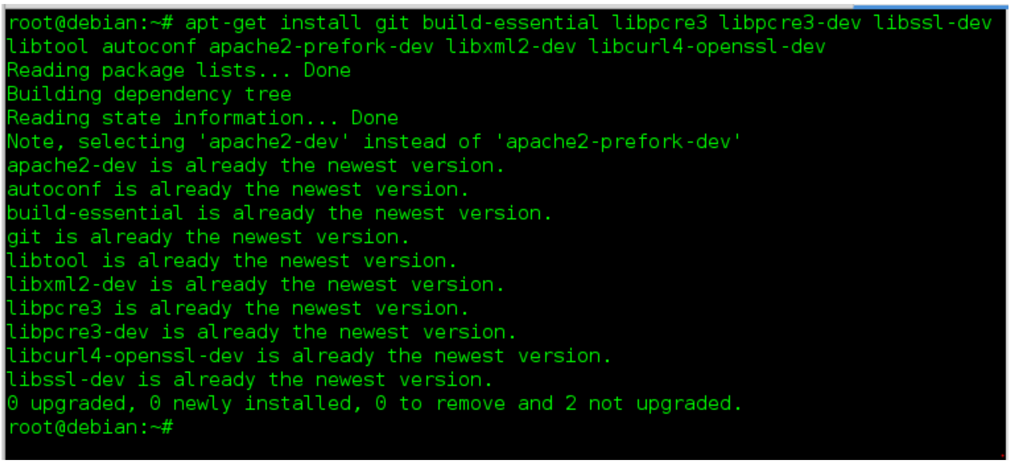
UNAM
CERT

Programa de Becas de Formación en Seguridad Informática 10ª Generación

1.- INSTALAR LAS DEPENDECIAS

Se instalaran todos los paquetes necesarios para compilar Nginx y ModSecurity.

apt-get install git build-essential libpcre3 libpcre3-dev libssl-dev libtool autoconf apache2-prefork-dev libxml2-dev libcurl4-openssl-dev

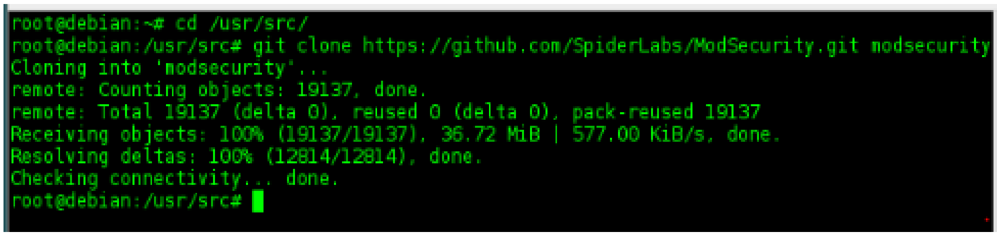


```
root@debian:~# apt-get install git build-essential libpcre3 libpcre3-dev libssl-dev
libtool autoconf apache2-prefork-dev libxml2-dev libcurl4-openssl-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'apache2-dev' instead of 'apache2-prefork-dev'
apache2-dev is already the newest version.
autoconf is already the newest version.
build-essential is already the newest version.
git is already the newest version.
libtool is already the newest version.
libxml2-dev is already the newest version.
libpcre3 is already the newest version.
libpcre3-dev is already the newest version.
libcurl4-openssl-dev is already the newest version.
libssl-dev is already the newest version.
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
root@debian:~#
```

2.- DESCARGAR MODSECURITY Y NGINX

Nos movemos al directorio **cd /usr/src** y allí clonaremos el siguiente repositorio.

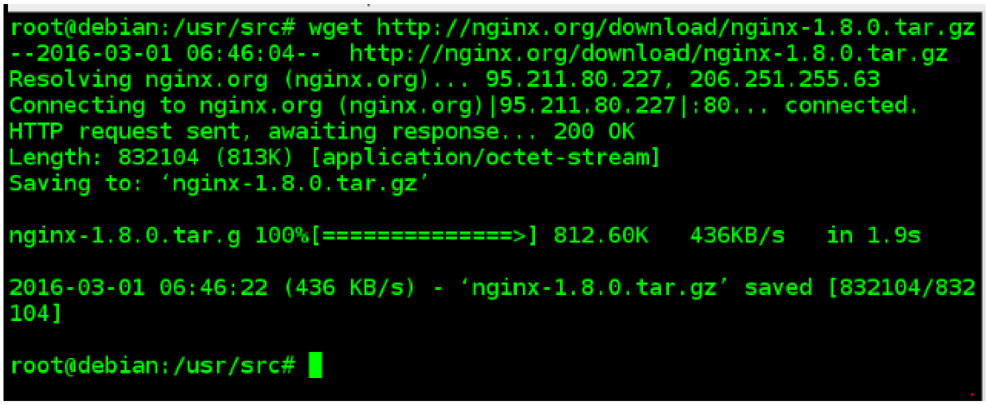
git clone https://github.com/SpiderLabs/ModSecurity.git modsecurity



```
root@debian:~# cd /usr/src/
root@debian:/usr/src# git clone https://github.com/SpiderLabs/ModSecurity.git modsecurity
Cloning into 'modsecurity'...
remote: Counting objects: 19137, done.
remote: Total 19137 (delta 0), reused 0 (delta 0), pack-reused 19137
Receiving objects: 100% (19137/19137), 36.72 MiB | 577.00 KiB/s, done.
Resolving deltas: 100% (12814/12814), done.
Checking connectivity... done.
root@debian:/usr/src# 
```

Una vez realizado esto, vamos a descargar Nginx con el comando wget usaremos la versión 1.8 con el comando:

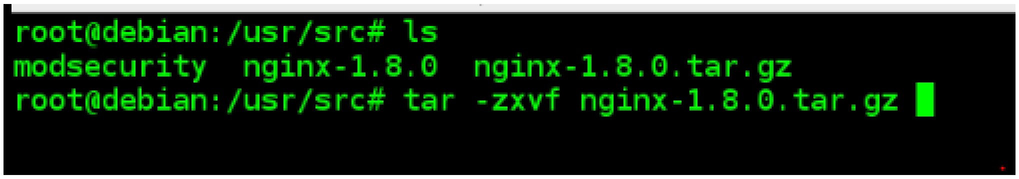
wget http://nginx.org/download/nginx-1.8.0.tar.gz



```
root@debian:/usr/src# wget http://nginx.org/download/nginx-1.8.0.tar.gz
--2016-03-01 06:46:04--  http://nginx.org/download/nginx-1.8.0.tar.gz
Resolving nginx.org (nginx.org)... 95.211.80.227, 206.251.255.63
Connecting to nginx.org (nginx.org)|95.211.80.227|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 832104 (813K) [application/octet-stream]
Saving to: ‘nginx-1.8.0.tar.gz’

nginx-1.8.0.tar.g 100%[===============>] 812.60K   436KB/s   in 1.9s

2016-03-01 06:46:22 (436 KB/s) - ‘nginx-1.8.0.tar.gz’ saved [832104/832
104]

root@debian:/usr/src# 
```

Y se descomprime con:

tar –zvxf nginx-1.8.0.tar.gz

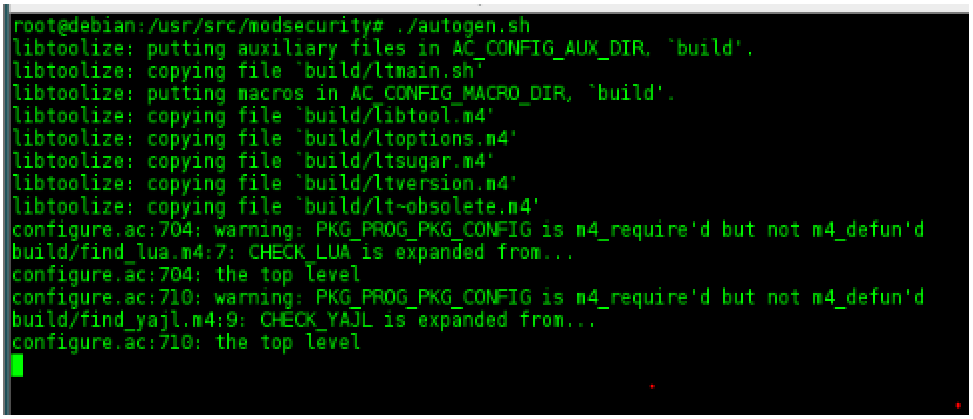


```
root@debian:/usr/src# ls
modsecurity  nginx-1.8.0  nginx-1.8.0.tar.gz
root@debian:/usr/src# tar -zxvf nginx-1.8.0.tar.gz
```

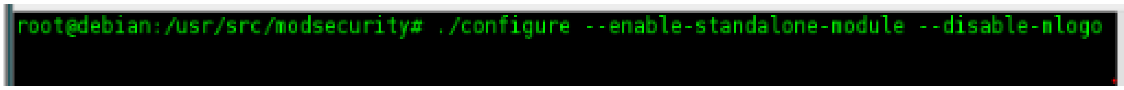
3.- INSTALAREMOS MODSECURITY Y NGINX

Vamos al directorio **cd /usr/src/modsecurity** dentro de allí vamos a compilar el modulo independiente en el servidor, por lo que podemos incluirlo a Nginx.

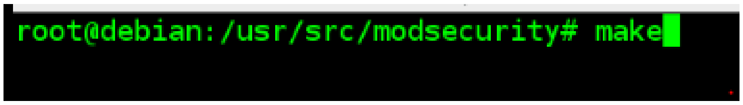
./autogen.sh
./configure --enable-standalone-module --disable-mlogc
make



```
root@debian:/usr/src/modsecurity# ./autogen.sh
libtoolize: putting auxiliary files in AC_CONFIG_AUX_DIR, `build'.
libtoolize: copying file `build/ltmain.sh'
libtoolize: putting macros in AC_CONFIG_MACRO_DIR, `build'.
libtoolize: copying file `build/libtool.m4'
libtoolize: copying file `build/ltoptions.m4'
libtoolize: copying file `build/ltsugar.m4'
libtoolize: copying file `build/ltversion.m4'
libtoolize: copying file `build/lt~obsolete.m4'
configure.ac:704: warning: PKG_PROG_PKG_CONFIG is m4_require'd but not m4_defun'd
build/find_lua.m4:7: CHECK_LUA is expanded from...
configure.ac:704: the top level
configure.ac:710: warning: PKG_PROG_PKG_CONFIG is m4_require'd but not m4_defun'd
build/find_yajl.m4:9: CHECK_YAJL is expanded from...
configure.ac:710: the top level
```

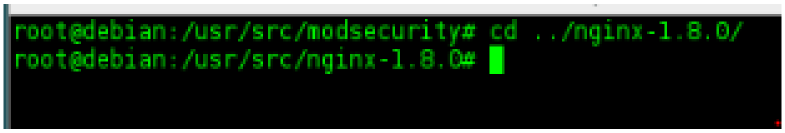


```
root@debian:/usr/src/modsecurity# ./configure --enable-standalone-module --disable-mlogo
```



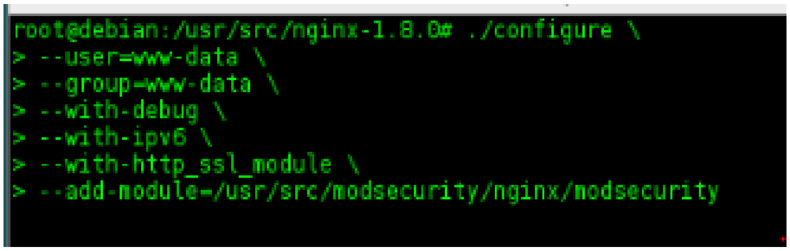
```
root@debian:/usr/src/modsecurity# make
```

Ahora nos situamos en directorio nginx **cd ../nginx-1.8.0** para compilar e incluir el módulo de ModSecurity.



```
root@debian:/usr/src/modsecurity# cd ../nginx-1.8.0/
root@debian:/usr/src/nginx-1.8.0#
```

```
./configure \
  --user=www-data \
  --group=www-data \
  --with-debug \
  --with-ipv6 \
  --with-http_ssl_module \
  --add-module=/usr/src/modsecurity/nginx/modsecurity
```

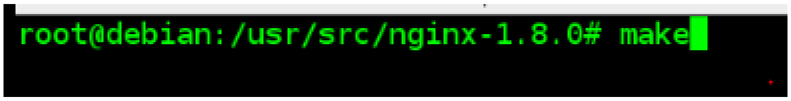


```
root@debian:/usr/src/nginx-1.8.0# ./configure \
> --user=www-data \
> --group=www-data \
> --with-debug \
> --with-ipv6 \
> --with-http_ssl_module \
> --add-module=/usr/src/modsecurity/nginx/modsecurity
```

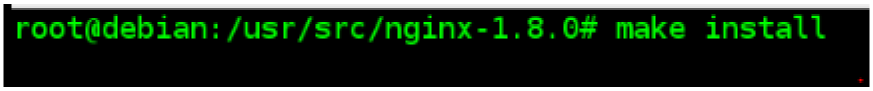
Nota: Nginx se ejecutará con el usuario y el grupo " www -data" , y activar los módulos de depuración , IPv6 y SSL . Y, finalmente, se incluye el módulo de ModSecurity en Nginx .

Ahora instalaremos Nginx

make
make install

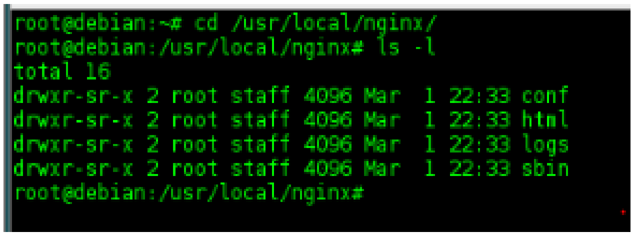


```
root@debian:/usr/src/nginx-1.8.0# make
```



```
root@debian:/usr/src/nginx-1.8.0# make install
```

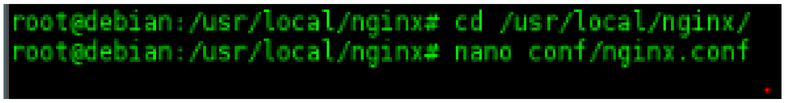
Cuando el comando make install está terminado, se puede ver que Nginx se instala en el directorio " **/ usr / local / nginx** "



```
root@debian:~# cd /usr/local/nginx/
root@debian:/usr/local/nginx# ls -l
total 16
drwxr-sr-x 2 root staff 4096 Mar  1 22:33 conf
drwxr-sr-x 2 root staff 4096 Mar  1 22:33 html
drwxr-sr-x 2 root staff 4096 Mar  1 22:33 logs
drwxr-sr-x 2 root staff 4096 Mar  1 22:33 sbin
root@debian:/usr/local/nginx#
```

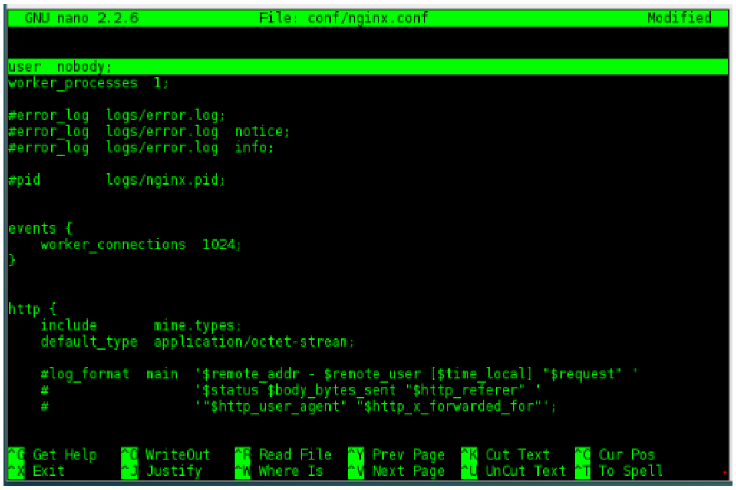
4.-CONFIGURACION NGINX

Ahora vamos al directorio **cd /usr/local/nginx/conf** y vamos a editar el archivo *nginx.conf.*

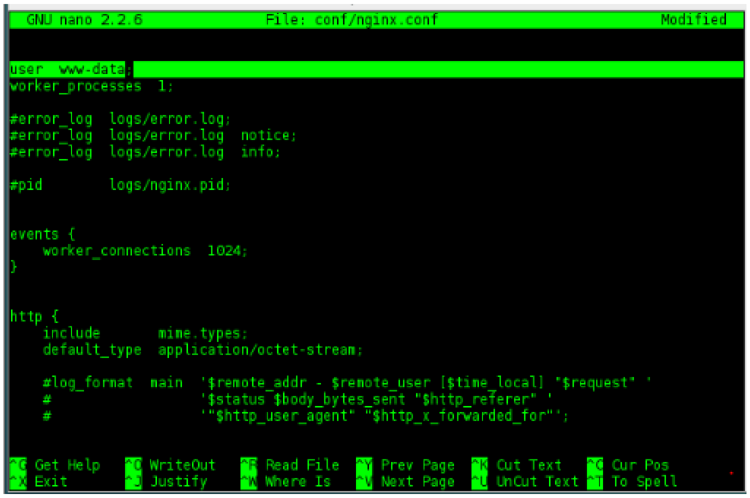


```
root@debian:/usr/local/nginx# cd /usr/local/nginx/
root@debian:/usr/local/nginx# nano conf/nginx.conf
```

Cambiaremos la primera línea de *user nobody* → *user www-data guardamos y salimos*

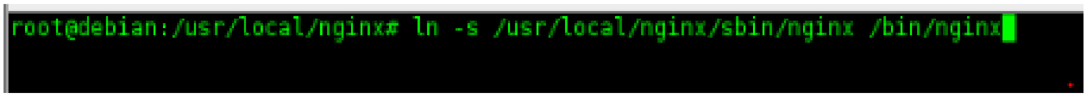


```
  GNU nano 2.2.6              File: conf/nginx.conf                    Modified


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;

    #log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
    #                  '$status $body_bytes_sent "$http_referer" '
    #                  '"$http_user_agent" "$http_x_forwarded_for"';

^G Get Help   ^O WriteOut   ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit       ^J Justify    ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```



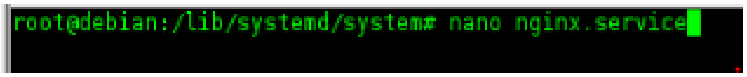
```
  GNU nano 2.2.6              File: conf/nginx.conf                    Modified


user  www-data;
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;

    #log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
    #                  '$status $body_bytes_sent "$http_referer" '
    #                  '"$http_user_agent" "$http_x_forwarded_for"';

^G Get Help   ^O WriteOut   ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit       ^J Justify    ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```

Crearemos un enlace simbólico para el binario nginx para que podamos sacar el comando "nginx" directamente **ln -s /usr/local/nginx/sbin/nginx /bin/nginx**



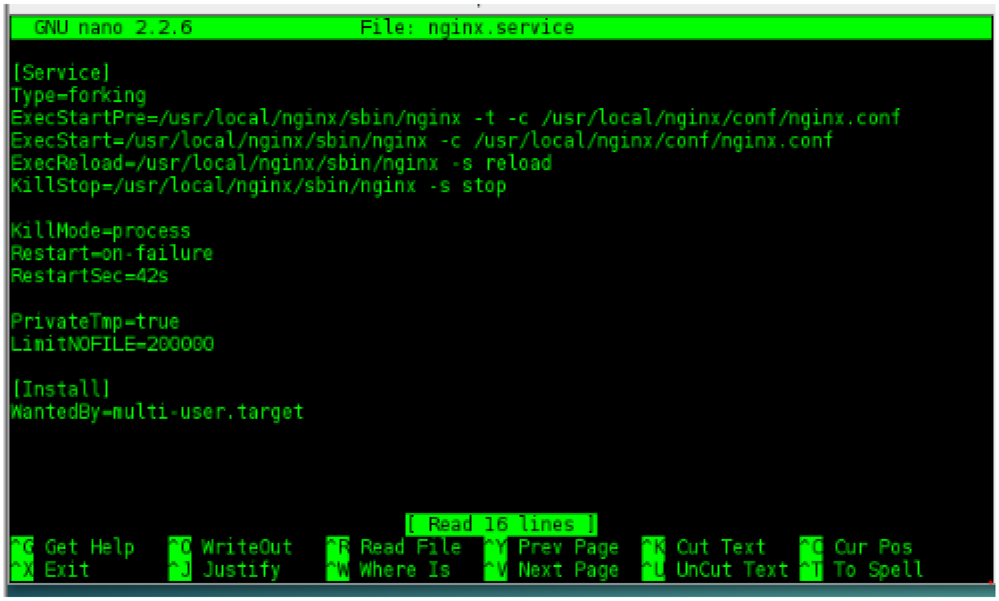
```
root@debian:/usr/local/nginx# ln -s /usr/local/nginx/sbin/nginx /bin/nginx
```

El siguiente paso es cambiarnos de directorio a **cd /lib/systemd/system/** y dentro de allí editar el archivo nginx.service (agregar lo siguiente al código) cuando se haya hecho guardar y salir.



```
root@debian:/lib/systemd/system# nano nginx.service
```

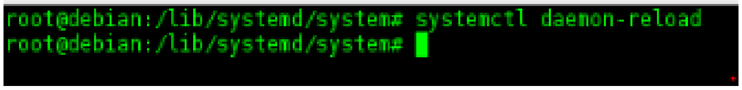
```
[Service]
Type=forking
ExecStartPre=/usr/local/nginx/sbin/nginx -t -c /usr/local/nginx/conf/nginx.conf
ExecStart=/usr/local/nginx/sbin/nginx -c /usr/local/nginx/conf/nginx.conf
ExecReload=/usr/local/nginx/sbin/nginx -s reload
KillStop=/usr/local/nginx/sbin/nginx -s stop

KillMode=process
Restart=on-failure
RestartSec=42s

PrivateTmp=true
LimitNOFILE=200000

[Install]
WantedBy=multi-user.target
```



```
  GNU nano 2.2.6              File: nginx.service

[Service]
Type=forking
ExecStartPre=/usr/local/nginx/sbin/nginx -t -c /usr/local/nginx/conf/nginx.conf
ExecStart=/usr/local/nginx/sbin/nginx -c /usr/local/nginx/conf/nginx.conf
ExecReload=/usr/local/nginx/sbin/nginx -s reload
KillStop=/usr/local/nginx/sbin/nginx -s stop

KillMode=process
Restart=on-failure
RestartSec=42s

PrivateTmp=true
LimitNOFILE=200000

[Install]
WantedBy=multi-user.target



                                [ Read 16 lines ]
^G Get Help   ^O WriteOut   ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit       ^J Justify    ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```

Ahora recargaremos systemd-daemon para que el systemd cargue nuestro archivo de servicio NGINX.

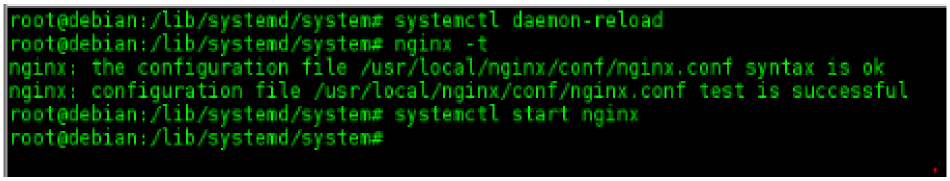
systemctl daemon-reload



```
root@debian:/lib/systemd/system# systemctl daemon-reload
root@debian:/lib/systemd/system# █
```

Se verificara la configuración de Nginx y se reiniciar el servicio

nginx -t
systemctl start nginx

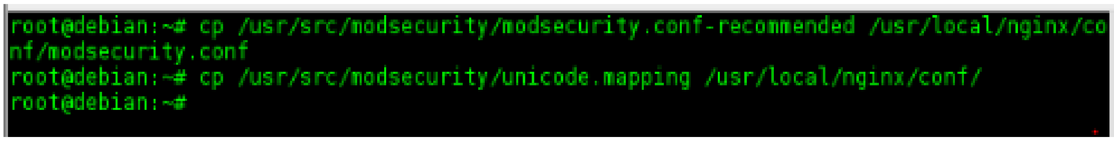


```
root@debian:/lib/systemd/system# systemctl daemon-reload
root@debian:/lib/systemd/system# nginx -t
nginx: the configuration file /usr/local/nginx/conf/nginx.conf syntax is ok
nginx: configuration file /usr/local/nginx/conf/nginx.conf test is successful
root@debian:/lib/systemd/system# systemctl start nginx
root@debian:/lib/systemd/system#
```

5.-CONFIGURANDO MODSECURITY

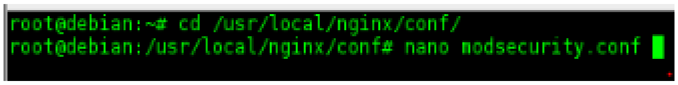
Copiaremos el archivo de configuración de ModSecurity al directorio Nginx con el nombre de “modsecurity.conf*”*

cp /usr/src/modsecurity/modsecurity.conf-recommended /usr/local/nginx/conf/modsecurity.conf
cp /usr/src/modsecurity/unicode.mapping /usr/local/nginx/conf/



```
root@debian:~# cp /usr/src/modsecurity/modsecurity.conf-recommended /usr/local/nginx/co
nf/modsecurity.conf
root@debian:~# cp /usr/src/modsecurity/unicode.mapping /usr/local/nginx/conf/
root@debian:~#
```

Cambiamos al directorio **cd /usr/local/nginx/conf** y editamos el archivo modsecurity.conf en las siguientes líneas:



```
root@debian:~# cd /usr/local/nginx/conf/
root@debian:/usr/local/nginx/conf# nano modsecurity.conf
```

Línea 7 cambiamos “Detection Only” → “Detection On”

Línea 38 aumentamos el valor a: SecRequestBodyLimit 13107200 → SecRequestBodyLimit 100000000

Línea 192 cambiamos el valor de: SecAuditLogType Serial → SecAuditLogTypeSerial Concurret

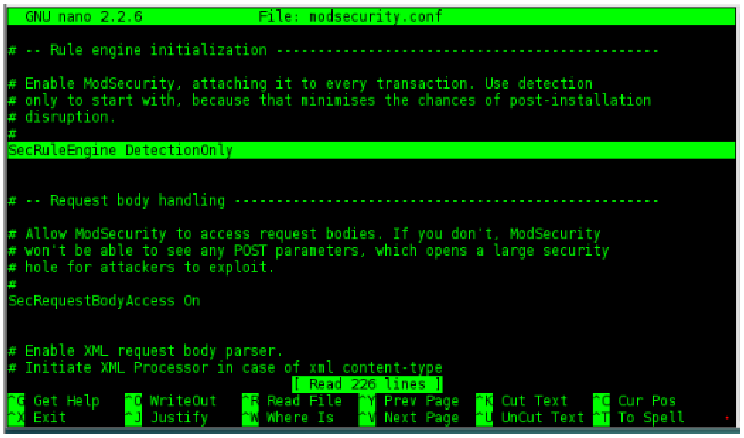
Línea 193 la comentamos

SecAuditLog /var/log/modsec_audit.log → # SecAuditLog /var/log/modsec_audit.log

Línea 196 se descomenta la línea

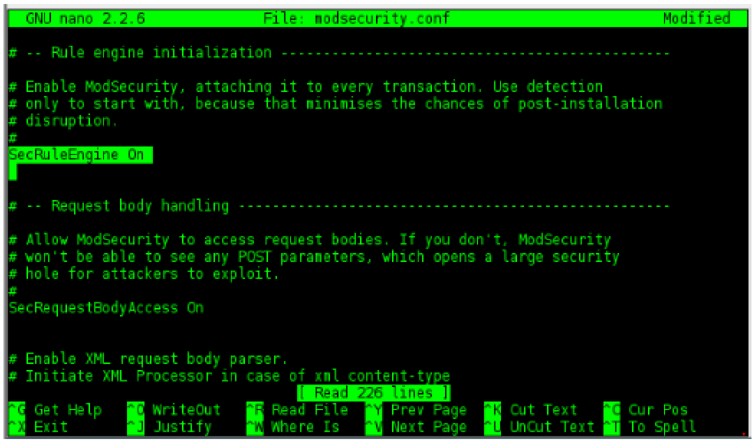
#SecAuditLogStorageDir /opt/modsecurity/var/audit/ → SecAuditLogStorageDir /opt/modsecurity/var/audit/

Guardamos y salimos



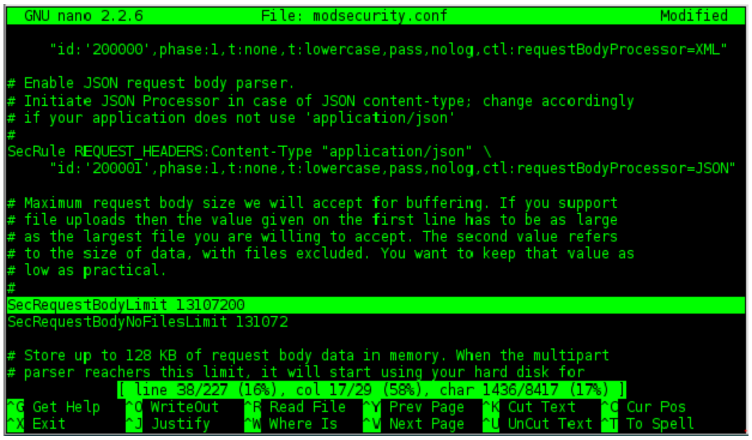
```
  GNU nano 2.2.6              File: modsecurity.conf

# -- 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 DetectionOnly


# -- 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


# Enable XML request body parser.
# Initiate XML Processor in case of xml content-type
                                  [ Read 226 lines ]
```

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell



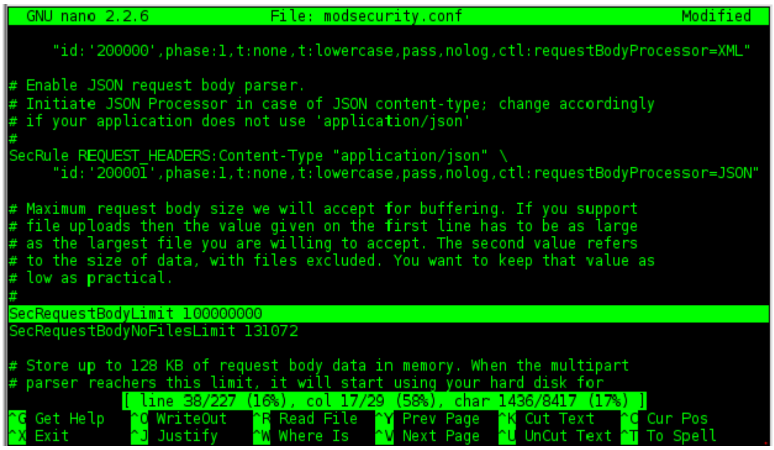
```
  GNU nano 2.2.6              File: modsecurity.conf                       Modified

# -- 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


# Enable XML request body parser.
# Initiate XML Processor in case of xml content-type
                                  [ Read 226 lines ]
```

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell



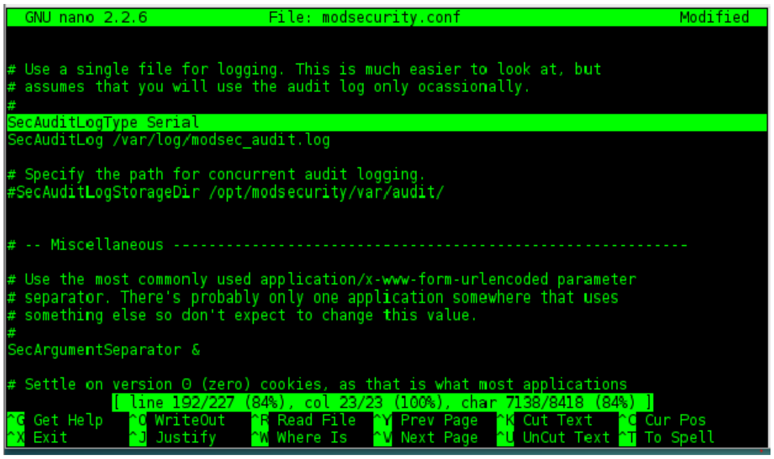
```
  GNU nano 2.2.6              File: modsecurity.conf                       Modified

     "id:'200000',phase:1,t:none,t:lowercase,pass,nolog,ctl:requestBodyProcessor=XML"

# Enable JSON request body parser.
# Initiate JSON Processor in case of JSON content-type; change accordingly
# if your application does not use 'application/json'
#
SecRule REQUEST_HEADERS:Content-Type "application/json" \
     "id:'200001',phase:1,t:none,t:lowercase,pass,nolog,ctl:requestBodyProcessor=JSON"

# Maximum request body size we will accept for buffering. If you support
# file uploads then the value given on the first line has to be as large
# as the largest file you are willing to accept. The second value refers
# to the size of data, with files excluded. You want to keep that value as
# low as practical.
#
SecRequestBodyLimit 13107200
SecRequestBodyNoFilesLimit 131072

# Store up to 128 KB of request body data in memory. When the multipart
# parser reachers this limit, it will start using your hard disk for
              [ line 38/227 (16%), col 17/29 (58%), char 1436/8417 (17%) ]
```

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell

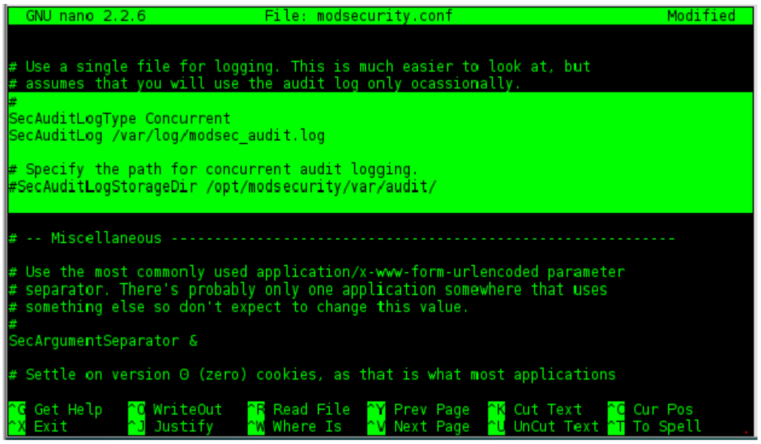


```
  GNU nano 2.2.6              File: modsecurity.conf                       Modified

     "id:'200000',phase:1,t:none,t:lowercase,pass,nolog,ctl:requestBodyProcessor=XML"

# Enable JSON request body parser.
# Initiate JSON Processor in case of JSON content-type; change accordingly
# if your application does not use 'application/json'
#
SecRule REQUEST_HEADERS:Content-Type "application/json" \
     "id:'200001',phase:1,t:none,t:lowercase,pass,nolog,ctl:requestBodyProcessor=JSON"

# Maximum request body size we will accept for buffering. If you support
# file uploads then the value given on the first line has to be as large
# as the largest file you are willing to accept. The second value refers
# to the size of data, with files excluded. You want to keep that value as
# low as practical.
#
SecRequestBodyLimit 100000000
SecRequestBodyNoFilesLimit 131072

# Store up to 128 KB of request body data in memory. When the multipart
# parser reachers this limit, it will start using your hard disk for
              [ line 38/227 (16%), col 17/29 (58%), char 1436/8417 (17%) ]
```

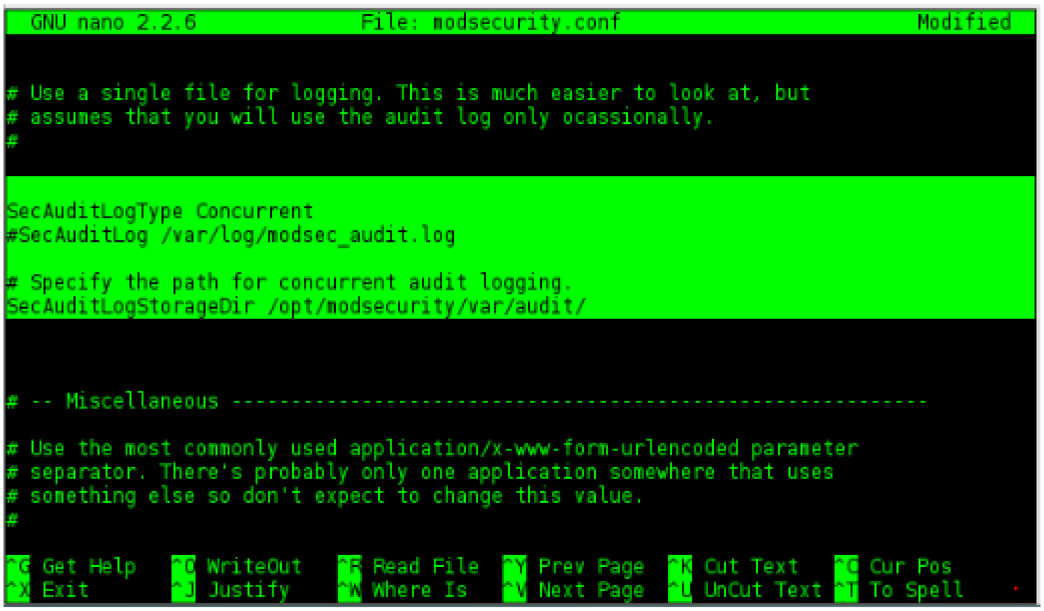
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell



```
# Use a single file for logging. This is much easier to look at, but
# assumes that you will use the audit log only ocassionally.
#
SecAuditLogType Serial
SecAuditLog /var/log/modsec_audit.log

# Specify the path for concurrent audit logging.
#SecAuditLogStorageDir /opt/modsecurity/var/audit/


# -- Miscellaneous -----------------------------------------------------------

# Use the most commonly used application/x-www-form-urlencoded parameter
# separator. There's probably only one application somewhere that uses
# something else so don't expect to change this value.
#
SecArgumentSeparator &

# Settle on version 0 (zero) cookies, as that is what most applications
            [ line 192/227 (84%), col 23/23 (100%), char 7138/8418 (84%) ]
```



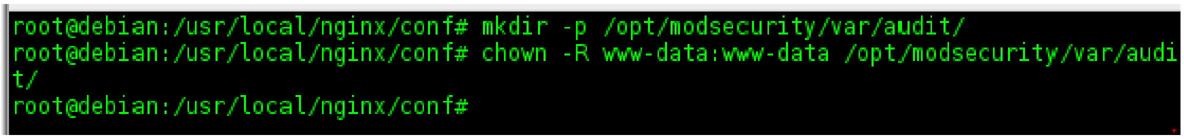
```
# Use a single file for logging. This is much easier to look at, but
# assumes that you will use the audit log only ocassionally.
#
SecAuditLogType Concurrent
SecAuditLog /var/log/modsec_audit.log

# Specify the path for concurrent audit logging.
#SecAuditLogStorageDir /opt/modsecurity/var/audit/

# -- Miscellaneous -----------------------------------------------------------

# Use the most commonly used application/x-www-form-urlencoded parameter
# separator. There's probably only one application somewhere that uses
# something else so don't expect to change this value.
#
SecArgumentSeparator &

# Settle on version 0 (zero) cookies, as that is what most applications
```



```
# Use a single file for logging. This is much easier to look at, but
# assumes that you will use the audit log only ocassionally.
#

SecAuditLogType Concurrent
#SecAuditLog /var/log/modsec_audit.log

# Specify the path for concurrent audit logging.
SecAuditLogStorageDir /opt/modsecurity/var/audit/



# -- Miscellaneous -----------------------------------------------------------

# Use the most commonly used application/x-www-form-urlencoded parameter
# separator. There's probably only one application somewhere that uses
# something else so don't expect to change this value.
#
```

Ahora crearemos un nuevo directorio para el registro de Modsecurity y cambiar el propietario a www-data

mkdir -p /opt/modsecurity/var/audit/
chown -R www-data:www-data /opt/modsecurity/var/audit/

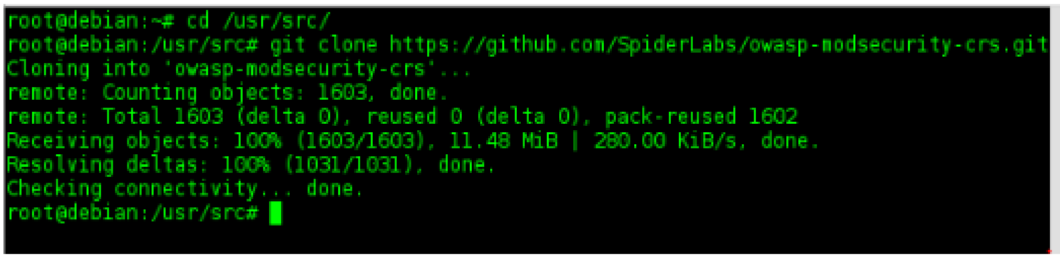


```
root@debian:/usr/local/nginx/conf# mkdir -p /opt/modsecurity/var/audit/
root@debian:/usr/local/nginx/conf# chown -R www-data:www-data /opt/modsecurity/var/audi
t/
root@debian:/usr/local/nginx/conf#
```

6.- CONFIGURANDO OWASP Core Rule Set (CRS)

Nos cambiamos de directorio a **cd /usr/src** y clonamos el siguiente repositorio:

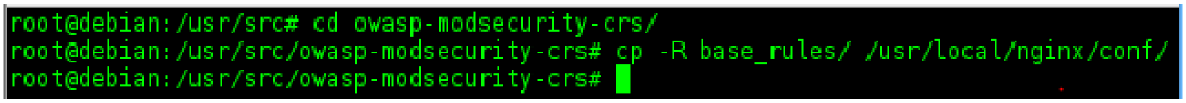
git clone https://github.com/SpiderLabs/owasp-modsecurity-crs.git



```
root@debian:~# cd /usr/src/
root@debian:/usr/src# git clone https://github.com/SpiderLabs/owasp-modsecurity-crs.git
Cloning into 'owasp-modsecurity-crs'...
remote: Counting objects: 1603, done.
remote: Total 1603 (delta 0), reused 0 (delta 0), pack-reused 1602
Receiving objects: 100% (1603/1603), 11.48 MiB | 280.00 KiB/s, done.
Resolving deltas: 100% (1031/1031), done.
Checking connectivity... done.
root@debian:/usr/src#
```

Luego vamos al directorio **cd owasp-modsecurity-crs** y copiamos el directorio “base_rules” al directio nginx.

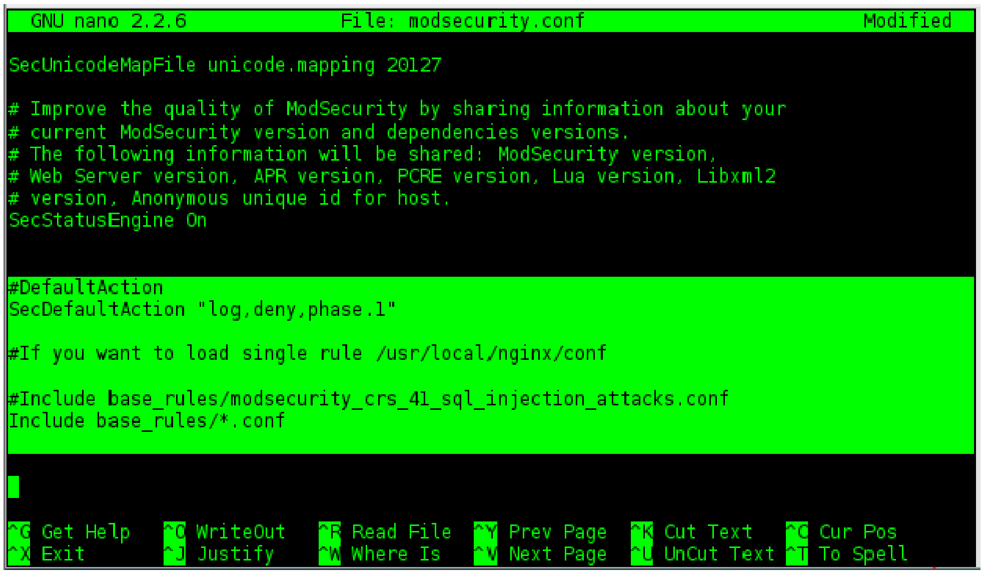
cp -R base_rules/ /usr/local/nginx/conf/



```
root@debian:/usr/src# cd owasp-modsecurity-crs/
root@debian:/usr/src/owasp-modsecurity-crs# cp -R base_rules/ /usr/local/nginx/conf/
root@debian:/usr/src/owasp-modsecurity-crs#
```

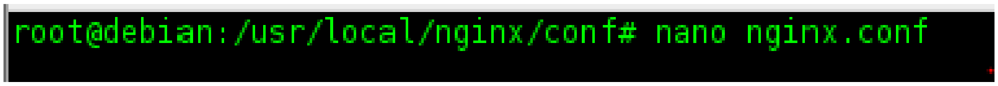
Editamos modsecurity.conf que está dentro del directorio **cd /usr/local/nginx/conf**/ y agregamos OWASP CRS al final del archivo

```
#DefaultAction
SecDefaultAction "log,deny,phase:1"

#If you want to load single rule /usr/loca/nginx/conf
#Include base_rules/modsecurity_crs_41_sql_injection_attacks.conf

#Load all Rule
Include base_rules/*.conf
```



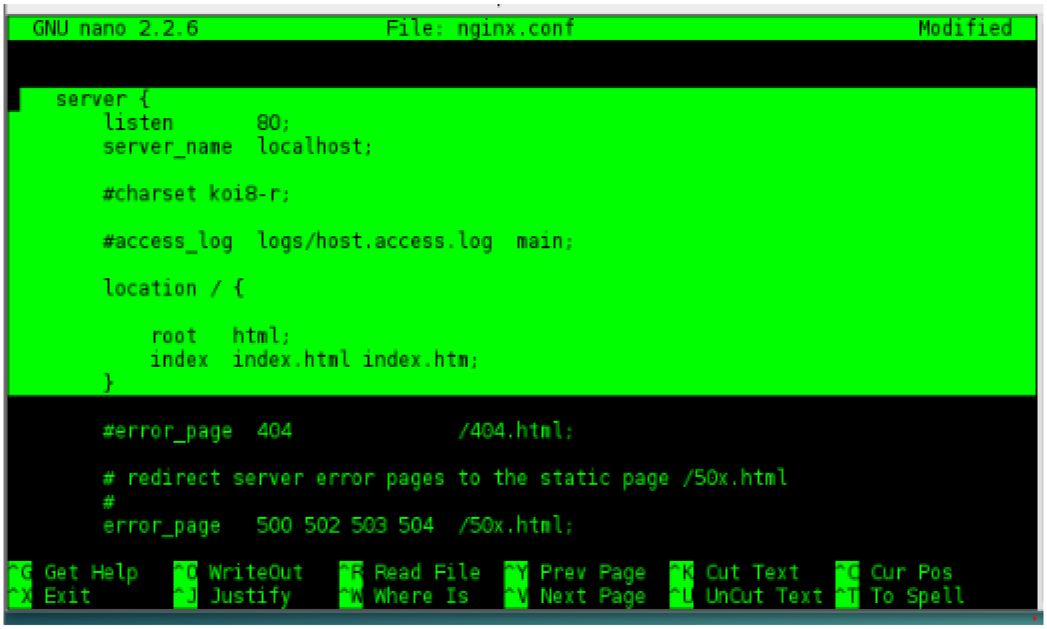
```
  GNU nano 2.2.6              File: modsecurity.conf                      Modified

SecUnicodeMapFile unicode.mapping 20127

# Improve the quality of ModSecurity by sharing information about your
# current ModSecurity version and dependencies versions.
# The following information will be shared: ModSecurity version,
# Web Server version, APR version, PCRE version, Lua version, Libxml2
# version, Anonymous unique id for host.
SecStatusEngine On


#DefaultAction
SecDefaultAction "log,deny,phase.1"

#If you want to load single rule /usr/local/nginx/conf

#Include base_rules/modsecurity_crs_41_sql_injection_attacks.conf
Include base_rules/*.conf



^G Get Help    ^O WriteOut    ^R Read File   ^Y Prev Page   ^K Cut Text    ^C Cur Pos
^X Exit        ^J Justify     ^W Where Is    ^V Next Page   ^U UnCut Text  ^T To Spell
```

Ingresar a la ruta **cd /usr/local/nginx/conf** y agregar las siguientes líneas en el archivo nginx.conf.



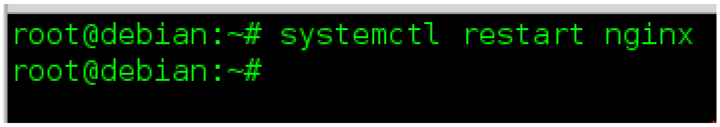
```
root@debian:/usr/local/nginx/conf# nano nginx.conf
```

```
[.....]

#Enable ModSecurity
ModSecurityEnabled on;
ModSecurityConfig modsecurity.conf;

root html;
index index.php index.html index.htm;

[.....]
```



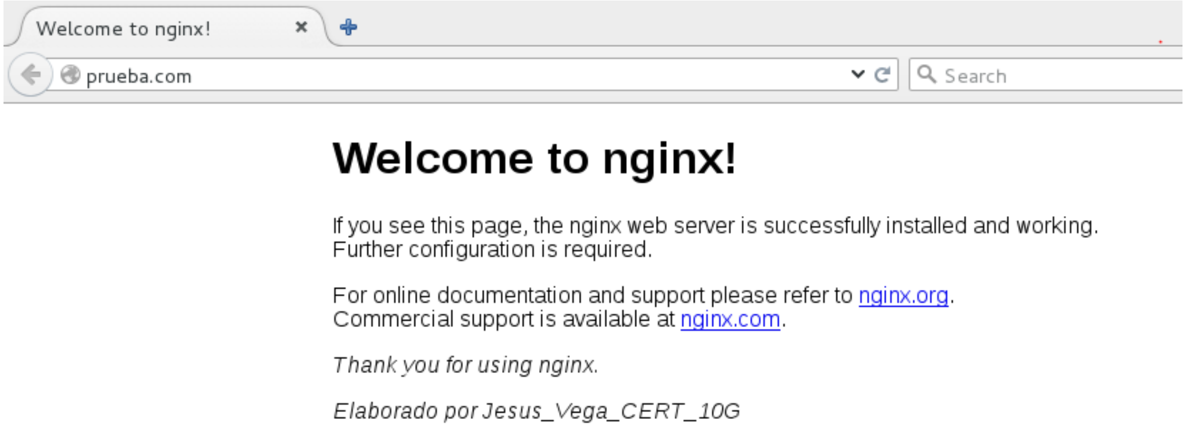
```
  GNU nano 2.2.6                File: nginx.conf                          Modified


    server {
        listen       80;
        server_name  localhost;

        #charset koi8-r;

        #access_log  logs/host.access.log  main;

        location / {

            root   html;
            index  index.html index.htm;
        }

        #error_page  404              /404.html;

        # redirect server error pages to the static page /50x.html
        #
        error_page   500 502 503 504  /50x.html;
```

Ultimo paso (reiniciar nginx para aplicar los cambias): **Systemctl** restart nginx



```
root@debian:~# systemctl restart nginx
root@debian:~#
```

6.-Probamos que nuestro nginx esté funcionando, ingresando al nombre del server que se le puso en este caso es “prueba.com” y nos debe mandar una pantalla así .



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Elaborado por Jesus_Vega_CERT_10G

→Para quitar la versión de nginx al producir un error se modificará el siguiente archivo

→

“REFERENCIAS”

Risager, T. (2015). *Compiling Nginx with ModSecurity on Ubuntu 14.04 LTS*. *Stickleback*. Retrieved 29 February 2016, from https://blog.stickleback.dk/nginx-modsec-on-ubuntu-14-04-lts/

Howtoforge.com,. (2016). *Installing Nginx With PHP5 (And PHP-FPM) And MySQL Support (LEMP) On Ubuntu 14.04 LTS*. Retrieved 2 March 2016, from https://www.howtoforge.com/installing-nginx-with-php5-fpm-and-mysql-on-ubuntu-14.04-lts-lemp

Ueland, C. (2013). *How to install Mod_Security on Nginx | Nginx Tips*. *ScaleScale.com*. Retrieved 2 March 2016, from https://www.scalescale.com/tips/nginx/how-to-install-mod_security-on-nginx/

Arul, M. (2015). *How to Install Nginx with ModSecurity on Ubuntu 15.04*. *Howtoforge.com*. Retrieved 2 March 2016, from https://www.howtoforge.com/tutorial/install-nginx-with-mod_security-on-ubuntu-15-04/