SECURING YOUR API GATEWAY USING MODSECURITY

Table of Content

This document is for customers hosting the API gateway on Nginx and concerned about security. The first thing you would like to implement is Web Application Firewall (WAF).

Mod Security is an Open Source WAF by Trustwave SpiderLabs and was made available for Nginx in 2012.

This guide, will explain how **download**, **install** and **configure** Mod Security with Nginx.

Download Nginx and ModSecurity

You can either download the nginx directly on your server or on your local PC then transfer it.

- Download the latest version from below link

http://nginx.org/en/download.html

- If you are directly downloading on server then can use wget as below

```
wget http://nginx.org/download/nginx-1.9.15.tar.gz
```

- Extract them by using gunzip command

```
gunzip -c nginx-1.9.15.tar.gz | tar xvf –
```

- You will see the new folder created

```
drwxr-xr-x 8 1001 1001    4096 Apr 19 12:02 nginx-1.9.15
```

- Download the latest version of Mod Security from below link

https://www.modsecurity.org/download.html

- You can use below commands from server directly

```
wget https://www.modsecurity.org/tarball/2.9.1/modsecurity-2.9.1.tar.gz
gunzip -c modsecurity-2.9.1.tar.gz | tar xvf –
```

Let’s get them installed

Install Nginx with Mod Security

It’s important to compile Nginx and mod security source code.

- Login into a server and ensure you have root permission.

Note: if you are doing on a brand new server then you may need to install following libraries.

```
yum install gcc make automake autoconf libtool pcre pcre-devel libxml2 libxml2-devel curl curl-devel httpd-devel
```

First, let's compile mod security. Go to **modsecurity-2.9.1** folder and use below commands.

```
./configure --enable-standalone-module

make
```

Next, install Nginx with mod security

```
./configure --add-module=../modsecurity-2.9.1/nginx/modsecurity

make

make install
```

This concludes Nginx is installed with Mod Security and it's time to configure it.

Configure Mod Security with Nginx

Copy **modsecurity.conf-recommended** & **unicode.mapping** file from extracted folder of above-downloaded ModSecurity source code to nginx conf folder. You may also use the find command.

```
find / -name modsecurity.conf-recommended

find / -name unicode.mapping

[root@GeekFlare-Lab conf]# cp /opt/nginx/binary/modsecurity-2.9.1/modsecurity.conf-recommended /usr/local/nginx/conf/

[root@GeekFlare-Lab conf]# cp /opt/nginx/binary/modsecurity-2.9.1/unicode.mapping /usr/local/nginx/conf/

[root@GeekFlare-Lab conf]#
```

Let's rename **modsecurity.conf-recommended** to **modsecurity.conf**

```
mv modsecurity.conf-recommended modsecurity.conf
```

- Take a backup of nginx.conf file
- Open nginx.conf file and add following under “location /” directive

```
ModSecurityEnabled on;
ModSecurityConfig modsecurity.conf;
```

So it should appear like this

```
location / {
ModSecurityEnabled on;
ModSecurityConfig modsecurity.conf;
}
```

Now, Mod Security is integrated with Nginx. Restart the Nginx to ensure it’s coming up without any error.

Let’s verify…

There are two possible methods to confirm Nginx is compiled with Mod Security.

First…

List the compiled module by using –V with nginx executable file.

```
[root@GeekFlare-Lab sbin]# ./nginx -V
nginx version: nginx/1.9.15
built by gcc 4.4.7 20120313 (Red Hat 4.4.7-16) (GCC)
configure arguments: --add-module=../modsecurity-2.9.1/nginx/modsecurity
[root@GeekFlare-Lab sbin]#
```

Second…

Go to logs folder and view the error file, you should see following

```
2016/05/21 21:54:51 [notice] 25352#0: ModSecurity for nginx (STABLE)/2.9.1 (http://www.modsecurity.org/) configured.

2016/05/21 21:54:51 [notice] 25352#0: ModSecurity: APR compiled version="1.3.9"; loaded version="1.3.9"

2016/05/21 21:54:51 [notice] 25352#0: ModSecurity: PCRE compiled version="7.8 "; loaded version="7.8 2008-09-05"

2016/05/21 21:54:51 [notice] 25352#0: ModSecurity: LIBXML compiled version="2.7.6"
```

This concludes you have successfully configured ModSecurity with Nginx.

By default configuration is in detect mode only that means it will not execute any action and protect your web applications.

Implement ModSecurity OWASP Core Rule-Set in Nginx

If you were securing Nginx with Mod Security then you would like to have OWASP core rule set (CRS) activated to protect from following threats.

- HTTP protocol violation protection
- Common web attacks
- Bots, crawlers, malicious activity protection
- Trojan protection
- Information leakage protection
- Cross Site Scripting attacks
- SQL injection attacks

ModSecurity is open source Web Application Firewall (WAF) and by default, it's configured to detect only. That means you need to enable the necessary configuration (**as following**) in order to start protecting your websites.

Download ModSecurity CRS

- Download latest CRS zip file from the following link and transferred to the server

https://github.com/SpiderLabs/owasp-modsecurity-crs/zipball/master

- unzip the file

```
unzip SpiderLabs-owasp-modsecurity-crs-2.2.9-26-gf16e0b1.zip
```

- Copy following to nginx conf folder

```
modsecurity_crs_10_setup.conf.example

base_rules
```

Configure Nginx to Integrate OWASP ModSecurity CRS

Since you have decided to use OWASP CRS, you need to merge the conf file included in SpiderLabs OWASP CRS, which you just copied (modsecurity_crs_10_setup.conf.example) under nginx folder.

Nginx doesn't support multiple ModSecurityConfig directives like Apache so you need to put all rules conf together in a single file.

Let's do it…

- Add base_rules & modsecurity_crs_10_setup.conf.example to modsecurity.conf file

```
cat modsecurity_crs_10_setup.conf.example base_rules/*.conf >>/usr/local/nginx/conf/modsecurity.conf
```

You also need to copy all ***.data** file to nginx conf folder

```
cp base_rules/*.data /usr/local/nginx/conf/
```

Quick verification:

Ensure you have added ModSecurityEnabled and ModSecurityConfig directive in nginx.conf file under location. If not, add them like below.

```
location / {
ModSecurityEnabled on;
ModSecurityConfig modsecurity.conf;
}
```

- Restart Nginx

By doing above all means you have successfully integrated OWASP CRS in Mod Security on Nginx. It’s time to do the little essential tweaking.

Configuring OWASP Core Rule Set to Start Protecting

In this section, all modifications will be in **modsecurity.conf** file so remembers to take a backup.

First thing first

Enable Audit Logging

It’s essential to generate logs so you know what’s being blocked. Add SecAuditLog directive if doesn’t exist.

```
SecAuditLog /usr/local/nginx/logs/modsec_audit.log
```

Restart Nginx and you will see the log file generated

```
-rw-r----- 1 root root 0 May 22 07:54 /usr/local/nginx/logs/modsec_audit.log
```

Enable Security Rule Engine

Begin Mod Security protection by enabling rule engine as below

```
SecRuleEngine On
```

Enable Default Action as Deny

Configure default action as “block” for any request matching with the rules.

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

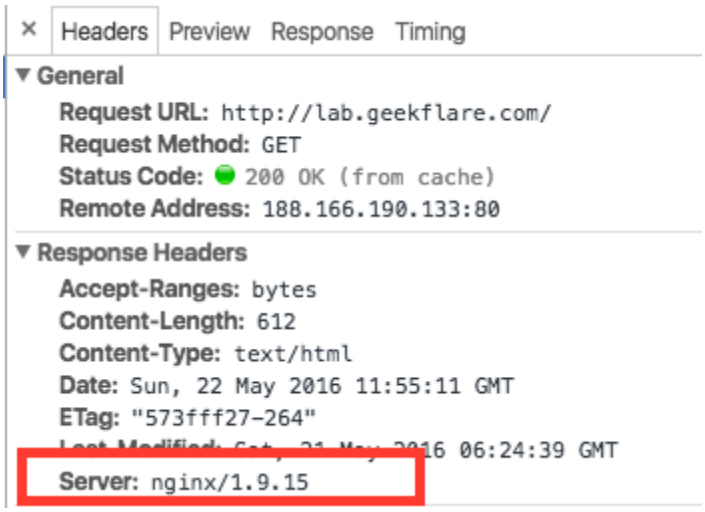
Above three configurations is **essential** and now ModSecurity is ready to execute the action and protect.

Here is one more configuration you may like.

Change Server Header Banner

Default Nginx configuration will expose server information with its version, which is highly recommended to mask it if you are working in PCI-DSS environment.

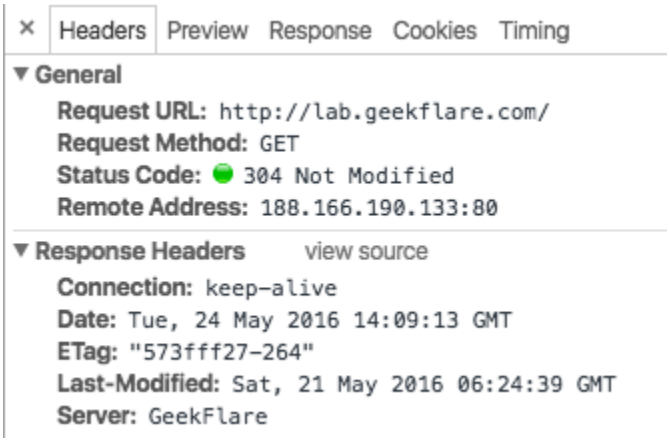
Default header:



You can do this easily by adding below a line.

```
SecServerSignature GeekFlare
```

And now it looks like:



I hope above instruction helps you in integrating OWASP Core Rule Set with Nginx web server for better protection.