

Guide Installation ModSecurity

Les sites web de Periscope sont développés via le CMS de Wordpress et tournent donc sous serveur Apache2. Afin de renforcer la sécurité de ces sites, le module ModSecurity de Apache permet de configurer un WAF (Web Application Firewall) qui filtre le trafic du site.

ModSecurity fonctionne avec un Connector qui diffère en fonction du serveur web utilisé (dans notre cas Apache), il permet l'utilisation des librairies ModSecurity avec les règles de détection et blocage de ModSecurity. Les règles de détection et blocage sont définies dans le Core Rule Set de l'OWASP qui permet d'éliminer la très grosse majorité des vulnérabilités que l'on peut avoir sur un site web.

Pour relancer Apache → **httpd -k restart**

Pour relancer le php → **php-fpm**

Instructions pour activer le module :

Installation des dépendances pour ModSecurity :

apk update

apk add --no-cache --virtual .build-modsec automake autoconf build-base git libxml2-dev libtool nano linux-headers pcre-dev apache2-dev libmaxminddb-dev

Récupération des librairies de ModSecurity depuis le GitHub officiel :

cd /opt

git clone --depth 1 https://github.com/SpiderLabs/ModSecurity

cd ModSecurity

Initialisation du module :

git submodule init

git submodule update

Configuration de ModSecurity :

mv /opt/ModSecurity/modsecurity.conf-recommended /opt/ModSecurity/modsecurity.conf

Modifier le fichier /opt/ModSecurity/modsecurity.conf :

SecRuleEngine DetectionOnly → **SecRuleEngine On** pour la détection en temps réel

Ajouter juste en dessous du **SecRuleEngine On** :

Include "/etc/apache2/modsecurity.d/crs-setup.conf"

Include "/etc/apache2/modsecurity.d/rules/*.conf"

Installation des librairies de ModSecurity :

```
./build.sh
```

```
./configure
```

```
make
```

```
make install
```

Récupération et Installation du Connector Apache :

```
cd ..
```

```
git clone --depth 1 https://github.com/SpiderLabs/ModSecurity-apache.git
```

```
cd ModSecurity-apache
```

```
./autogen.sh
```

```
./configure --with-libmodsecurity=/usr/local/modsecurity/
```

```
make
```

```
make install
```

Récupération du Core Rule Set OWASP :

```
git clone https://github.com/coreruleset/coreruleset.git /etc/apache2/modsecurity.d
```

Configuration du fichier d'utilisation des règles :

```
mv /etc/apache2/modsecurity.d/crs-setup.conf.example  
/etc/apache2/modsecurity.d/crs-setup.conf
```

Configuration des fichier de règles permanentes :

```
mv  
/etc/apache2/modsecurity.d/rules/REQUEST-900-EXCLUSION-RULES-BEFORE-CRS.conf.  
example  
/etc/apache2/modsecurity.d/rules/REQUEST-900-EXCLUSION-RULES-BEFORE-CRS.conf
```

```
mv  
/etc/apache2/modsecurity.d/rules/RESPONSE-999-EXCLUSION-RULES-AFTER-CRS.conf.  
example  
/etc/apache2/modsecurity.d/rules/RESPONSE-999-EXCLUSION-RULES-AFTER-CRS.conf
```

Activation des logs Apache (pas mis dans le Dockerfile) :

```
touch /var/log/apache2/error.log
```

```
touch /var/log/apache2/access.log
```

```
chown www-data:www-data /var/log/apache2/error.log
```

```
chown www-data:www-data /var/log/apache2/access.log
```

Implémentation de ModSecurity dans la configuration Apache :

Ajouter à la fin du fichier `/etc/apache2/httpd.conf` :

```
ErrorLog /var/log/apache2/error.log
CustomLog /var/log/apache2/access.log combined
LoadModule security3_module modules/mod_security3.so
modsecurity_rules_file /opt/ModSecurity/modsecurity.conf
```


Commande de test de ModSecurity :

```
curl -X GET "http://www.mileade-rh.test/" -d "parametre=<script>alert('XSS')</script>"
```

Les logs d'alertes sont dans `/var/log/modsec_audit.log`

Exemple d'alerte :

rh-develop-wordpress-1




 [rh-develop-wordpress](#)

82289b4045db

[80:80](#) [9003:9003](#)

STATUS

Running (2 minutes ago)

Logs

Inspect

Terminal

Files

Stats

[Open in external terminal](#)

```
/var/www/html # cat /var/log/modsec_audit.log
---wkj7WwTS---A--
[10/Aug/2023:15:24:40 +0200] 169167388045.969041 172.23.0.3 34590 localhost 80
---wkj7WwTS---B--
GET // HTTP/1.1
Host: www.mileade-rh.test
User-Agent: curl/7.83.1
Accept: */*
Content-Length: 39
Content-Type: application/x-www-form-urlencoded

---wkj7WwTS---D--

---wkj7WwTS---F--
HTTP/1.1 403

---wkj7WwTS---H--
ModSecurity: Warning. Matched "Operator 'Rx' with parameter '^0?$' against variable 'REQUEST_HEADERS:Content-Length' (Value: '39' ) [file "/etc/apache2/modsecurity.d/rules/REQUEST-920-PROTOCOL-ENFORCEMENT.conf"] [line "171"] [id "920170"] [rev "" ] [msg "GET or HEAD Request with Body Content"] [data "39"] [severity "2"] [ver "OWASP_CRS/4.0.0-rc1"] [maturity "0"] [accuracy "0"] [tag "application-multi"] [tag "language-multi"] [tag "platform-multi"] [tag "attack-protocol"] [tag "paranoia-level/1"] [tag "OWASP_CRS"] [tag "cape c/1000/210/272"] [hostname "localhost"] [uri "/" ] [unique_id "169167388045.969041"] [ref "o0,3v0,3v94,2"]
ModSecurity: Access denied with code 403 (phase 2). Matched "Operator 'Ge' with parameter '5' against variable 'TX:BLOCKING_INBOUND_ANOMALY_SCORE' (Value: '5' ) [file "/etc/apache2/modsecurity.d/rules/REQUEST-949-BLOCKING-EVALUATION.conf"] [line "176"] [id "949110"] [rev "" ] [msg "Inbound Anomaly Score Exceeded (Total Score: 5)"] [data "" ] [severity "0"] [ver "OWASP_CRS/4.0.0-rc1"] [maturity "0"] [accuracy "0"] [tag "anomaly-evaluation"] [hostname "localhost"] [uri "/" ] [unique_id "169167388045.969041"] [ref "" ]

---wkj7WwTS---I--

---wkj7WwTS---J--
```