Docs Home > Guides > Security, Upgrades & Backups > Security Basics

Securing Apache 2 With ModSecurity

Updated Thursday, March 9, 2023, by HackerSploit

This guide was written for Apache 2. Other web servers are available: Apache 2

What is ModSecurity?

ModSecurity is a free and open source web application that started out as an Apache module and grew to a fully-fledged web application firewall. It works by inspecting requests sent to the web server in real time against a predefined rule set, preventing typical web application attacks like XSS and SQL Injection.

Prerequisites & Requirements

In order to install and configure ModSecurity, you need to have a Linux server with the following services running:

- Apache 2

For instructions, see our guide on How to Install Apache Web Server on Ubuntu 18.04 LTS. Installation instructions for several other Linux distributions are also accessible from this guide.

> **Note**
>
> This demonstration has been performed on Ubuntu 18.04. However, all techniques demonstrated are distribution agnostic with the exception of package names and package managers.

Installing ModSecurity

1. ModSecurity can be installed by running the following command in your terminal:

```
sudo apt install libapache2-mod-security2 -y
```

2. Alternatively, you can also build ModSecurity manually by cloning the official ModSecurity Github repository.
3. After installing ModSecurity, enable the Apache 2 `headers` module by running the following command:

```
sudo a2enmod headers
```

After installing ModSecurity and enabling the header module, you need to restart the apache2 service, this can be done by running the following command:

```
sudo systemctl restart apache2
```

You should now have ModSecurity installed. The next steps involves enabling and configuring ModSecurity and the OWASP-CRS.

Configuring ModSecurity

ModSecurity is a firewall and therefore requires rules to function. This section shows you how to implement the OWASP Core Rule Set. First, you must prepare the ModSecurity configuration file.

1. Remove the `.recommended` extension from the ModSecurity configuration file name with the following command:

```
sudo cp /etc/modsecurity/modsecurity.conf-recommended /etc/modsecurity/modsecurity.conf
```

2. With a text editor such as vim, open `/etc/modsecurity/modsecurity.conf` and change the value for `SecRuleEngine` to `On`:

File: /etc/modsecurity/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 On
...

```

3. Restart Apache to apply the changes:

```
sudo systemctl restart apache2
```

ModSecurity should now be configured to run. The next step in the process is to set up a rule set to actively prevent your web server from attacks.

Setting Up the OWASP ModSecurity Core Rule Set

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. The CRS provides protection against many common attack categories, including SQL Injection, Cross Site Scripting, and Local File Inclusion.

To set up the OWASP-CRS, follow the procedures outlined below.

1. First, delete the current rule set that comes prepackaged with ModSecurity by running the following command:

```
sudo rm -rf /usr/share/modsecurity-crs
```

2. Ensure that git is installed:

```
sudo apt install git
```

3. Clone the OWASP-CRS GitHub repository into the `/usr/share/modsecurity-crs` directory:

```
sudo git clone https://github.com/coreruleset/coreruleset /usr/share/modsecurity-crs
```

4. Rename the `crs-setup.conf.example` to `crs-setup.conf`:

```
sudo mv /usr/share/modsecurity-crs/crs-setup.conf.example /usr/share/modsecurity-crs/crs-setup.conf
```

5. Rename the default request exclusion rule file:

```
sudo mv /usr/share/modsecurity-crs/rules/REQUEST-900-EXCLUSION-RULES-BEFORE-CRS.conf.example /usr/share/modsecurity-crs/rules/
```

You should now have the OWASP-CRS setup and ready to be used in your Apache configuration.

Enabling ModSecurity in Apache 2

To begin using ModSecurity, enable it in the Apache configuration file by following the steps outlined below:

1. Using a text editor such as vim, edit the `/etc/apache2/mods-available/security2.conf` file to include the OWASP-CRS files you have downloaded:

File: /etc/apache2/mods-available/security2.conf

```
<IfModule security2_module>
        SecDataDir /var/cache/modsecurity
        Include /usr/share/modsecurity-crs/crs-setup.conf
        Include /usr/share/modsecurity-crs/rules/*.conf
</IfModule>

```

2. In `/etc/apache2/sites-enabled/000-default.conf` file `VirtualHost` block, include the `SecRuleEngine` directive set to `On`.

File: /etc/apache2/sites-enabled/000-default.conf

```
<VirtualHost *:80>
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html

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

        SecRuleEngine On
</VirtualHost>

```

If you are running a website that uses SSL, add `SecRuleEngine` directive to that website's configuration file as well. See our guide on SSL Certificates with Apache on Debian & Ubuntu for more information.

3. Restart the apache2 service to apply the configuration:

```
sudo systemctl restart apache2
```

ModSecurity should now be configured and running to protect your web server from attacks. You can now perform a quick test to verify that ModSecurity is running.

Testing ModSecurity

Test ModSecurity by performing a simple local file inclusion attack by running the following command:

```
curl http://<SERVER-IP/DOMAIN>/index.php?exec=/bin/bash
```

If ModSecurity has been configured correctly and is actively blocking attacks, the following error is returned:

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>403 Forbidden</title>
</head><body>
<h1>Forbidden</h1>
<p>You don't have permission to access this resource.</p>
<hr>
<address>Apache/2.4.25 (Debian) Server at 96.126.105.75 Port 80</address>
</body></html>
```

This page was originally published on Friday, March 26, 2021.

SECURITY WEB SERVER APACHE

Why Choose Us

- Why Choose Us
- Global Infrastructure
- Cloud Simplified
- Predictable Pricing
- Support Experience
- Free Bundled Services
- Customer Stories
- Cloud for Business
- Our Approach
- What is Cloud Computing?

Company

- About
- Partners
- Press Center
- Careers
- Legal

Products

- Products Overview
- Dedicated CPU
- Shared CPU
- High Memory
- GPU
- Kubernetes
- Block Storage
- Object Storage
- Backups
- Managed Databases
- Cloud Firewall
- DDoS Protection
- DNS Manager
- NodeBalancers
- VLAN
- Managed
- Professional Services
- Cloud Manager
- API
- CLI
- Terraform Provider
- Ansible Collection

Industries

- Digital Agencies
- Ecommerce
- Education
- Gaming
- Managed Hosting
- Managed Service Providers
- Media
- SaaS

Marketplace

- Browse Marketplace
- Submit Marketplace App

Pricing

- Pricing List
- Cloud Estimator
- Cloud Pricing Calculator

Community

- Community Overview
- Q&A
- Developer Portal
- Affiliate Program
- Beta Program
- Customer Referral Program
- Partner Program
- Startup Program
- Blog
- Content Resources
- Events
- Promotional Offers
- Distributions
- Kernels

Contact

- Support
- System Status
- Log in