A logo with text and a circle

Description automatically generated with medium confidenceFebruary 17, 2025

University of Sciences and Arts in Lebanon

Faculty of Sciences and Arts

**OWASP Mod-Security**

**Ethical Hacking Project Report**

Student Name – Anwar Ali Zeidani

Instructor: Dr. Ahmad Fadlallah.

# Abstract

In this report, we will talk about implementing a WAF mod-security that’s acts as a reverse proxy to protect a web server from OWASP top 10 web attack vulnerabilities. This project contains a docker-compose container that container 3 containers (PHP-Apache, database, and phpMyAdmin container) and an apache2 with mod-security CRS (Core Rule Set) acts as a WAF to protect the docker web server. We will aim on how we implement the docker-compose environment that contains these three containers connected on the same bridge network, and How to setup and configure the apache2 server with mod-security to acts as a WAF. This containers can communicate with layer 3 IP protocol that enabled by the docker0 interface, and this network can be reached from the host machine network. Therefore, the WAF server on the host machine can communicate with the containers network. Coraza WAF depends on mod-security rules and modules that developed to prevent the OWASP top 10 web vulnerabilities.

The abstract is a concise summary of the entire report. It provides a brief overview of the project, including its objectives, methodology, key findings, and conclusions. The abstract should give readers a clear understanding of the report's purpose and findings without having to read the entire document.

# Table of Contents

The table of contents is a list of the main sections and subsections of the report, along with their respective page numbers. It serves as a roadmap for readers, allowing them to quickly navigate through the document and locate specific information.

Sample Table of Contents:

[Abstract 2](#_Toc190654320)

[Table of Contents 3](#_Toc190654321)

[1 Introduction 4](#_Toc190654322)

[2 Product Presentation 5](#_Toc190654323)

[3 Attack Scenarios on Security Solutions 8](#_Toc190654324)

[4 Results of the Scenarios 14](#_Toc190654325)

[5 Conclusion 14](#_Toc190654326)

[6 References 15](#_Toc190654327)

[Appendix 16](#_Toc190654328)

# 1 Introduction

* This project aims to protect web application server by implementing a WAF server to capture, analyze, and take actions against http traffic.
* As we all know, security now is essential part in this huge digital word. Implementing a security solution that can defend against cyber attack is so important to protect a company business and network infrastructure. This measure could rescue the company from significant financial damage.
* In this report, we will talk about the implementation of docker-compose environment, implement an apache2 as a WAF server on the host, how these different components communicate with one another.
* Overview of the project and its objectives in the context of cybersecurity.
* Explanation of the importance of implementing defense security solutions.
* Report structure: A paragraph explaining what are the main sections of the report (one sentence for each section)

# 2 Product Presentation

* Web Application Firewall (WAF) is a server that responsible to protect the web application server by capture/filter the http traffic and take actions depends on pre-defined rules. Mod-Security is a powerful, open-source Web Application Firewall (WAF) engine. providing real-time monitoring and filtering of HTTP traffic.
* Mod-Security analyzes incoming and outgoing HTTP requests and responses, comparing them against a set of predefined rules. It can detect and block various web-based attacks, preventing them from reaching the web application. allowing administrators to customize rules and adapt to evolving threats. The Mod-Security rules are applied to prevent the OWASP top 10 web attacks. (SQLi, XSS, CSRF, broken-authentication, ..).
* Because the company deploy an E-commerce website and afraid of losing users data and a financial lose. The company decision to implement a WAF server that protect the web server from malicious attacks.

Figure1: As we see the web application firewall can prevent malicious traffic from reaching the destination web server.

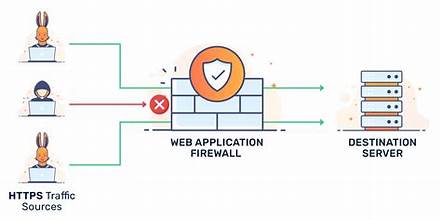


Figure2: This figure explains how the reverse proxy works with the servers and WAF.

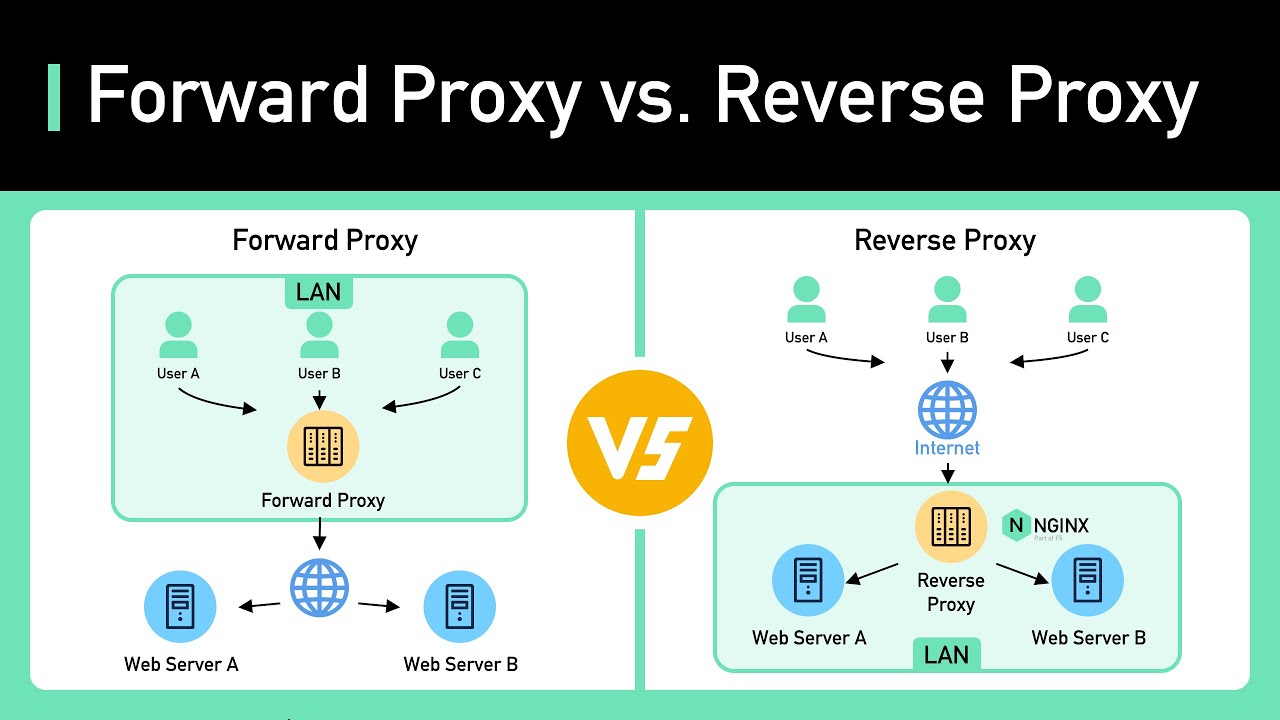
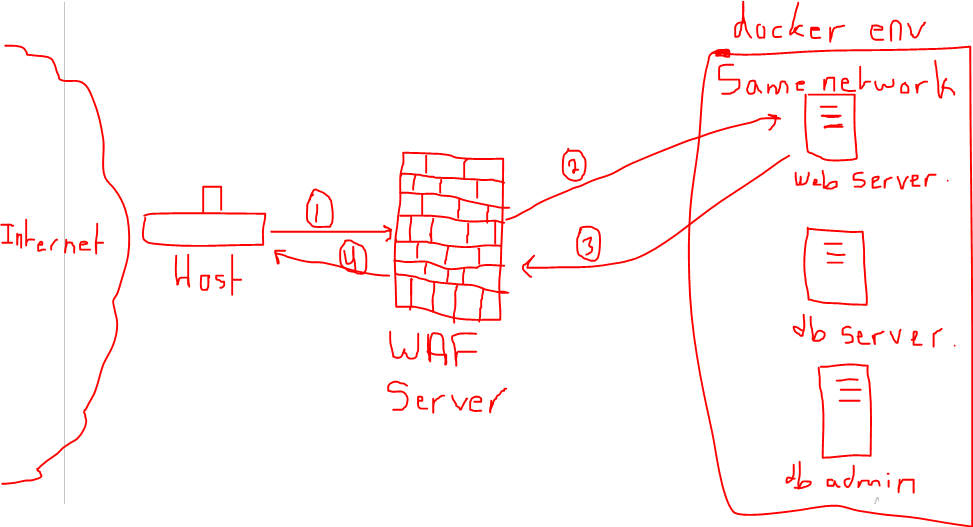


Figure3: This figure shows the solution and deployment structure of the project.



# Attack Scenarios on Security Solutions

* We will begin with the most popular web attack: SQL Injection is the art of inject quires to the database server that if it execute it can steal all db information and cause a huge impact. By running SQLmap for automated the attack by launching a huge number of injected quires.

Figure4: This figure shows the sqlmap running command.



Figure5 : This figure represents a WARNING that traffic is being Forbidden.

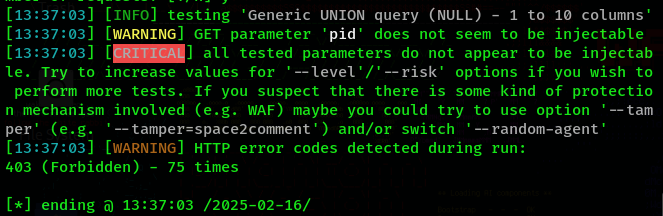


Figure6: As this figure shown, the WAF server detect the malicious activities and it in the /var/log/apache2/modsec\_audit.log.

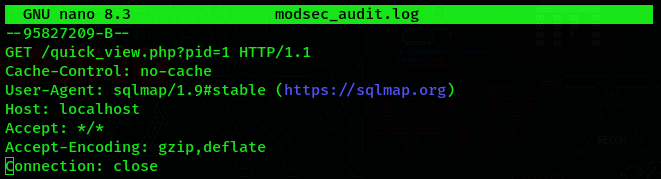
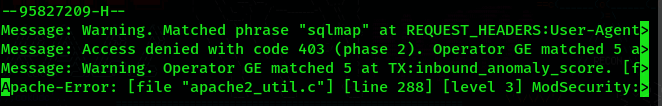


Figure7: WAF detect this attack because it matches one or more of the CRS and custom rules.



SQL Injection targeted the code that communicate with the database like (Forms (sign-in/login), get productid ,.. ).

* Now let’s try to perform a Cross-site Scripting (XSS attack). This attack aims to inject a JavaScript code into the web application.

Figure8: This image shows a way to perform the XSS attack by inject the code in the contact Us message .

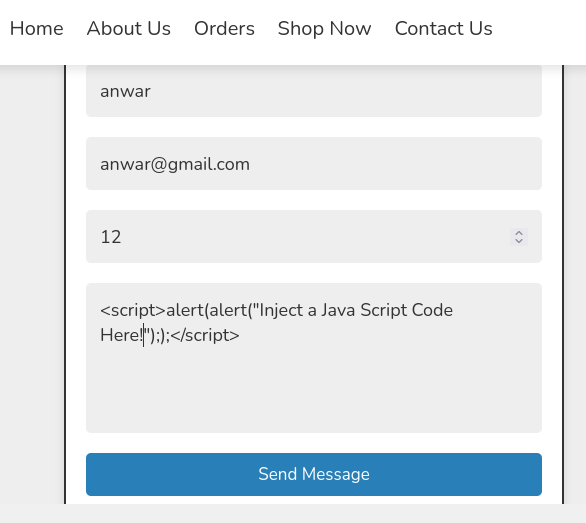


Figure9: This figure shows that the WAF server send a 403 Forbidden to this request.



Figure10: This figure shows that the WAF server detect and prevent this XSS attack.

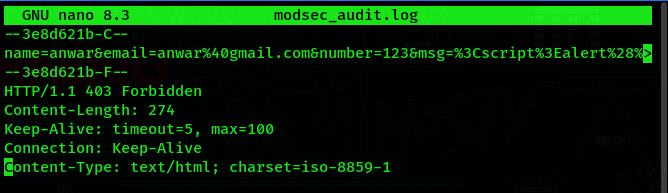
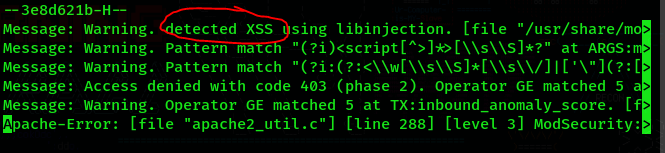


Figure11: This figure shows that the WAF message that tills the administrator that there is XSS attack are performed.



Cross-site scripting attacks targets the functionality of user can comment or send a message, If the XSS attack success then every browser opens the webpage the browser executes the injected JavaScript code.

* Nmap vulnerability scanning: Nmap is a powerful network discovery tool that can scan the entire network and detect weaknesses. Let’s try to perform Nmap vuln scanning against our website.

Figure12: In this picture we see that Nmap cannot detect any type of vulnerabilities on port 80.

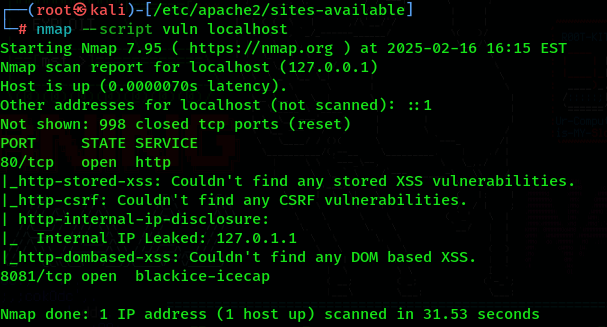


Figure13: This picture shows that the request send from curl is blocked by the WAF server.

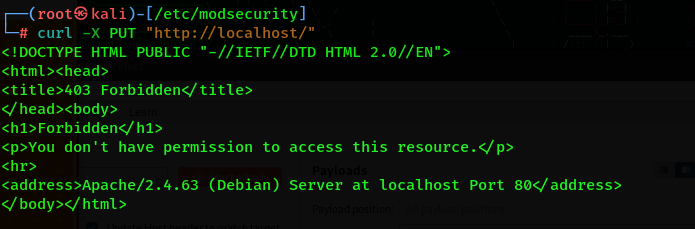


Figure14: Another curl blocked request.

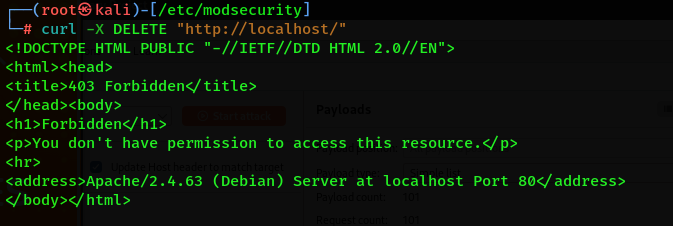


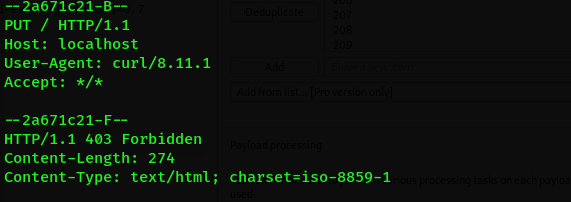
Figure15: This image shows the log of the WAF server when detect the curl request.

Figure16: This message sends by the matched rules.

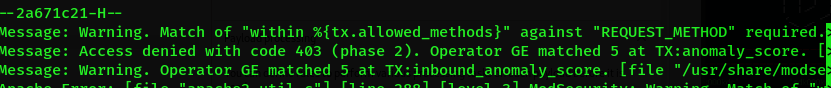


Figure17: This curl request focusses on sending a sensitive data to the web server.

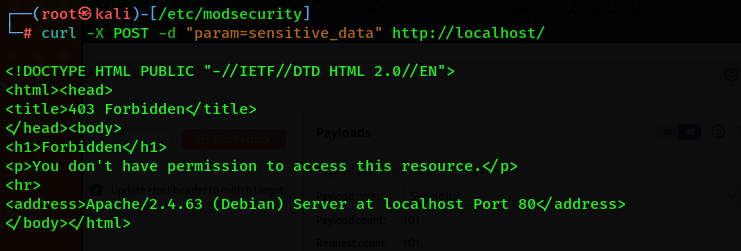
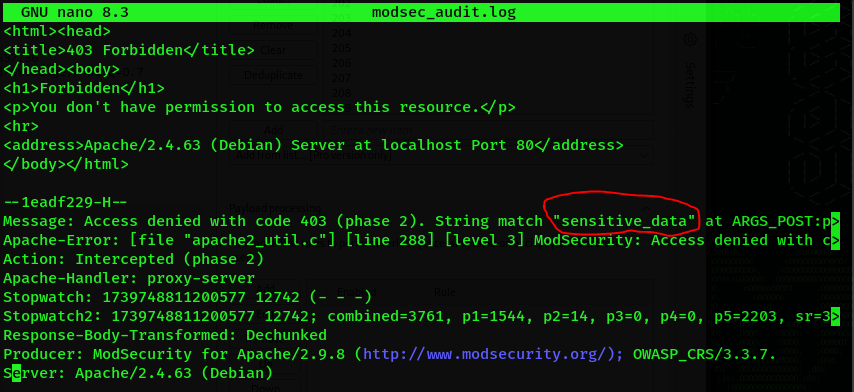


Figure18: As we see, the rule match the sensitive data content and deny the request by sending a forbidden response.



# Results of the Scenarios

* This WAF is very powerful against the OWASP top 10 web attacks because the OWASP developers developed the core rule set CRS to match the known web application attacks.
* This WAF is build on pre-defined rules that can only prevent the listed and known attack signatures. Therefore, this WAF is not able to detect advanced attacks that developed specially for the target server.
* It provides a minimum security needed by the web server but it is effective when the attack has known signature or if the administrator is professional, he can generate many rules to specific the allowed and not allowed actions.
* We can add a **black-list IP** and domain names by adding a black-list.txt file and generate a rule that compare the sender with this black-list.txt content.
* It is effective for a small website that doesn’t need for a paid WAF engine.

# Conclusion

* We discuss about the WAF and what WAF responsible for, How the WAF work and we ensure that the WAF detect the attacking techniques on an efficient way.
* The PHP web application is a weak website as a standalone because of many poor security in coding, but with WAF the attack vector is reduced to the half because it can prevent the malicious behaviors before it reach the web server and access to the code.
* Implementing the Docker environment proved to be quite difficult, as I was unfamiliar with it. Similarly, setting up the web application firewall and figuring out the required configurations and integration was a major challenge. I faced over 100 errors, but I gained a lot of valuable experience.
* For recommendations for further enhancements this project structure need to add a docker containers security and fix bugs on PHP code to work smoothly with the web server (apache2, nginx, or caddy ,..). And monitor the network behaviors to apply a set of new rules to reduce the attack vector.

# 6 References

References List:

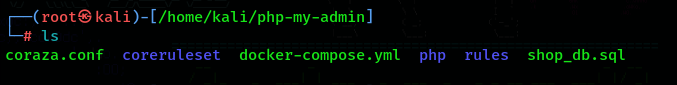
|  |  |
| --- | --- |
| [1] | https://docs.docker.com/compose/ |
| [2] | https://hub.docker.com/ |
| [3] | https://modsecurity.org/ |
| [4] | https://gemini.google.com/ |
| [5] | https://en.wikipedia.org/wiki/ModSecurity |

# Appendix

* Install and configure the docker containers environment:



First, we install docker and docker-compose to our machine.



Second, we create a directory for the whole project for example called php-my-admin.

Before using the docker-compose there are two steps that must be done:

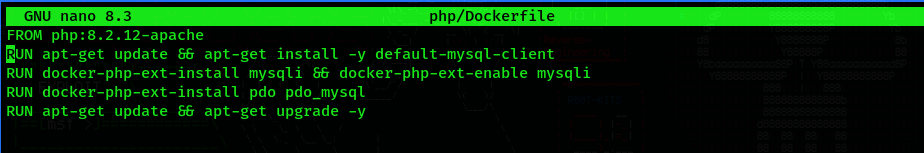
Create and configure the docker-compose.yml and Dockerfile.

Ther docker-compose.yml is run by default when the docker-compose up command are executed to run the project. And the DockerFile is responsible for installing the dependencies and correct versions by running a commands while the docker-compose build process is executes.

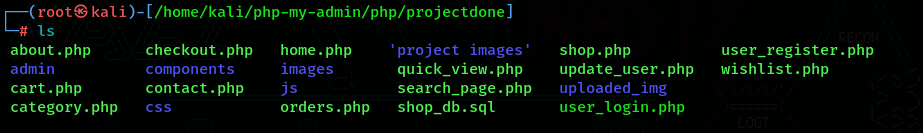




This is the docker-compose.yml content that define each container with name, volumes that send from the host files to the container files, docker-image, ports that map from the host machine to the container machine , …



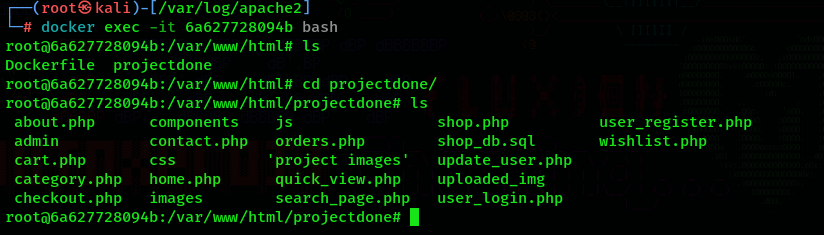
This is the dockerfile that has the instructions that must executes while the building process executes.



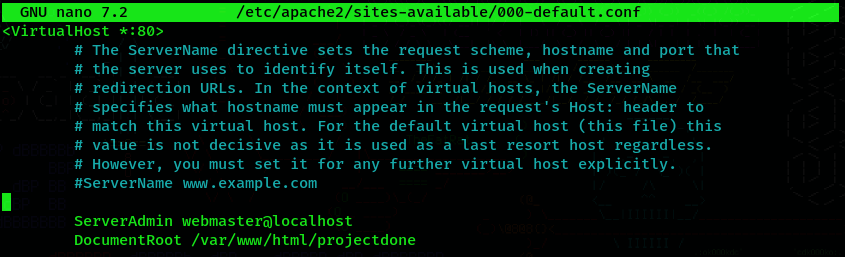
This is my php project that I deployed on an apache web server inside docker container.



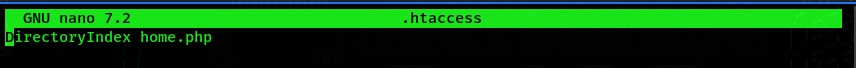
This is the command that will run all the environment by run the docker-compose.yml and dockerfile content and configurations.

* Now I need to deploy the web application on the container

Now we are inside the apache web server container. As we see the project is exists on the /var/www/html/ directory.

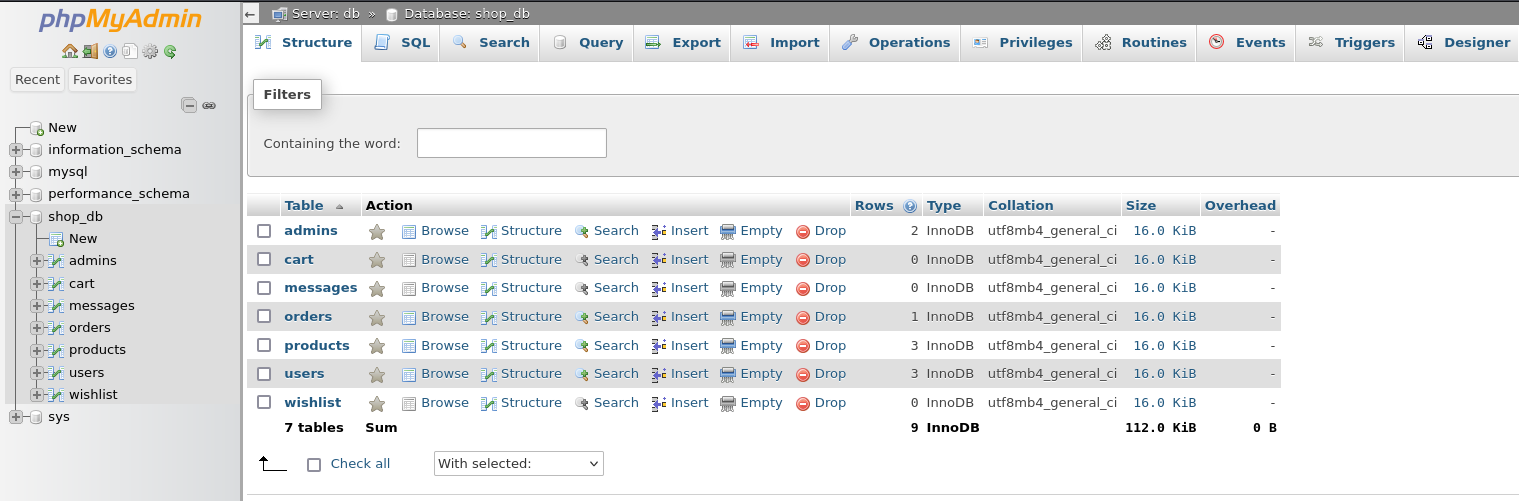


In this step we change the DocumentRoot to match with our project directory root (/etc/pache2/sites-available/000-default.conf)

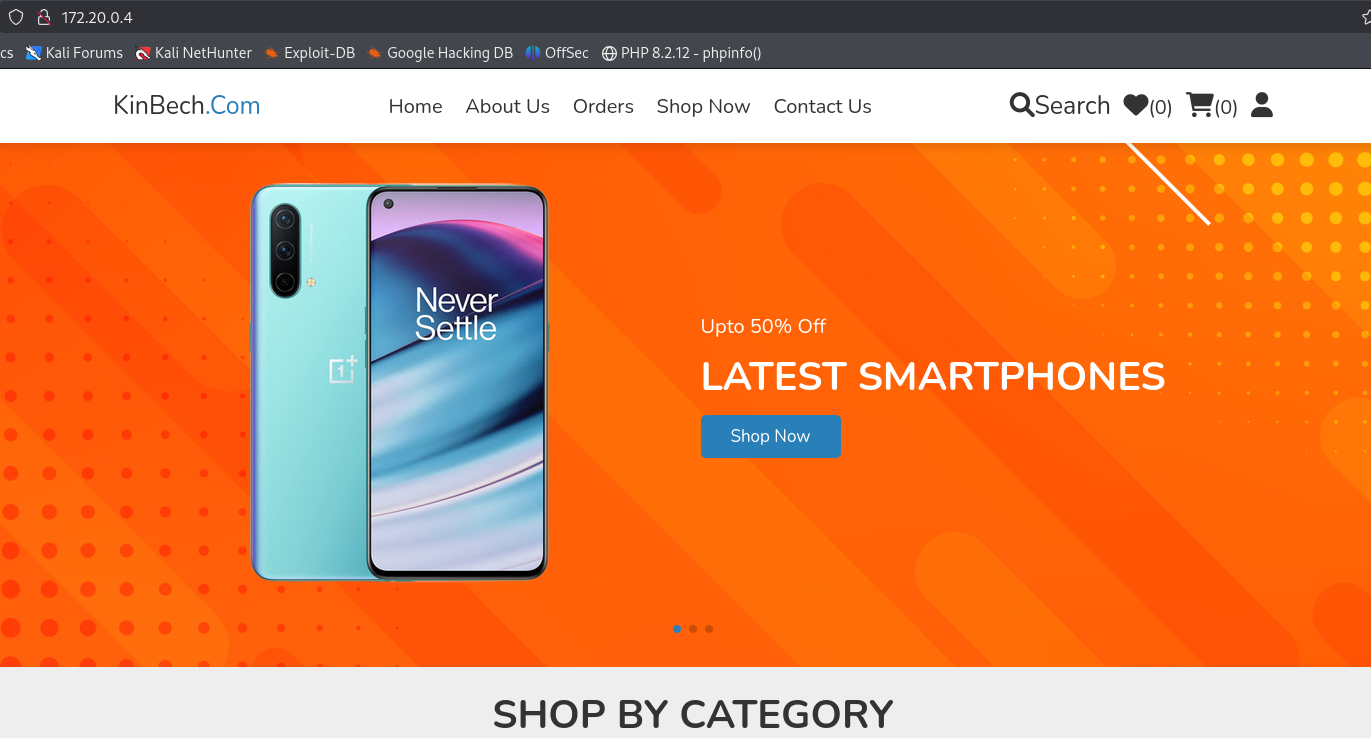


Now we create the .htaccess file inside our documentroot directory. This file allow the apache web server to render the home.php page as a default.

* Now I implement the docker phpMyAdmin server to manage the database of the php web application.



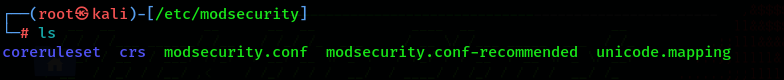
Now our website render smoothly.



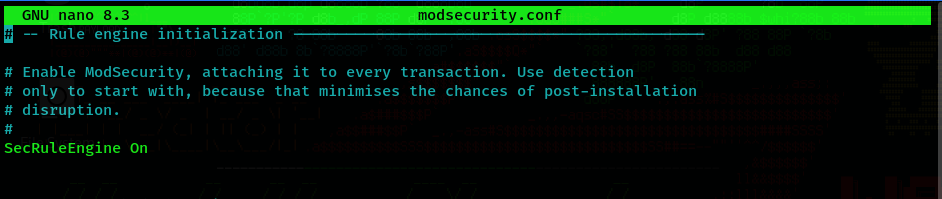
* Now we need to configure the Apache2 server running the mod-security rules to acts as a reverse proxy WAF. By install the apache2 service and the mod-security modules.



This is apache2 service.



This is the mod security folder.



We enable the SecRuleEngine on the modsecurity.conf file.

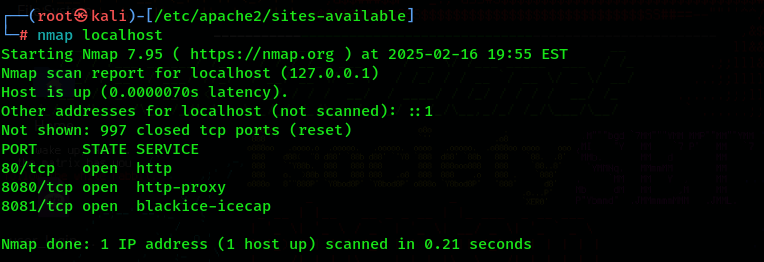


We enable the reverse proxy on the apache server at(/etc/apache2/site-available/000-default.conf) by sets the IP:Port of the web server docker container.

* After this configuration the apache2 service need to restart.

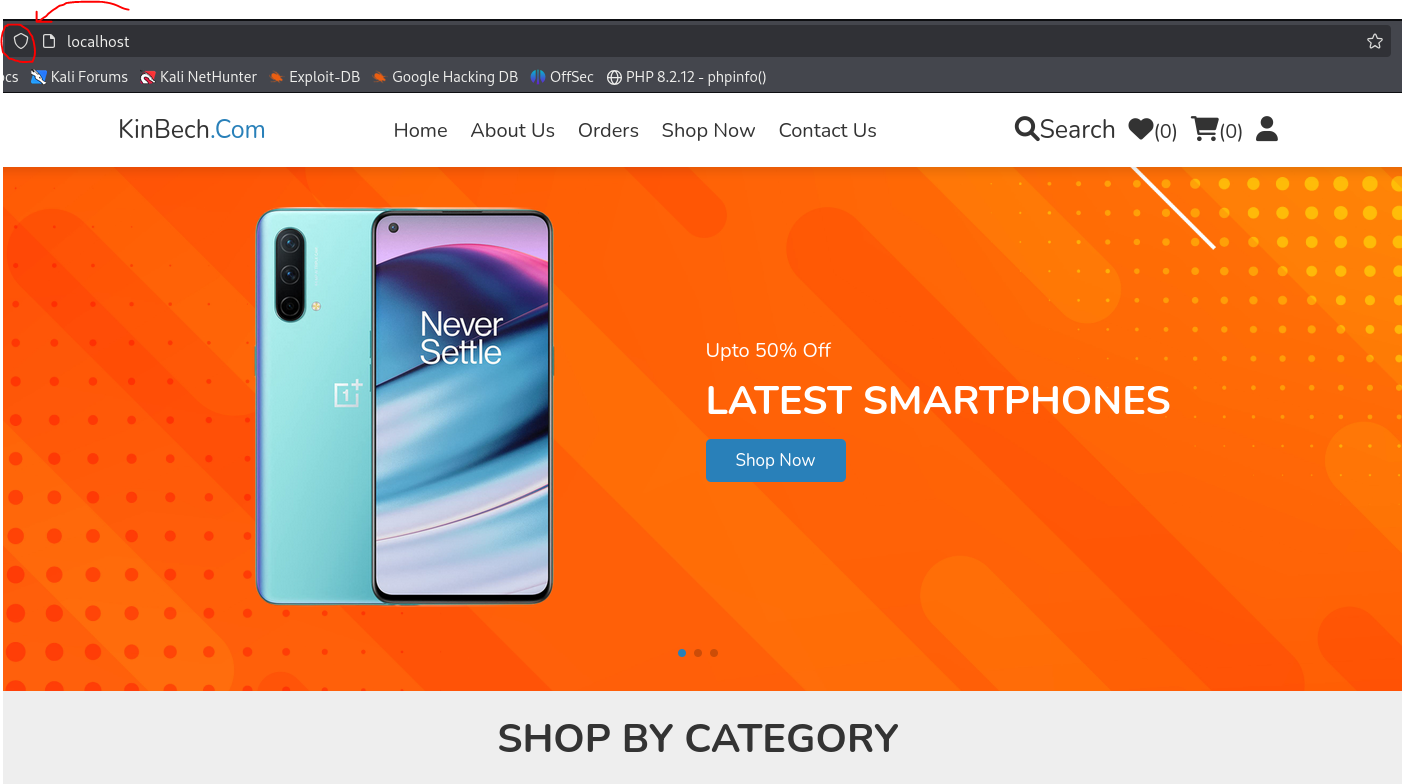


This command restarts the apache2 service.

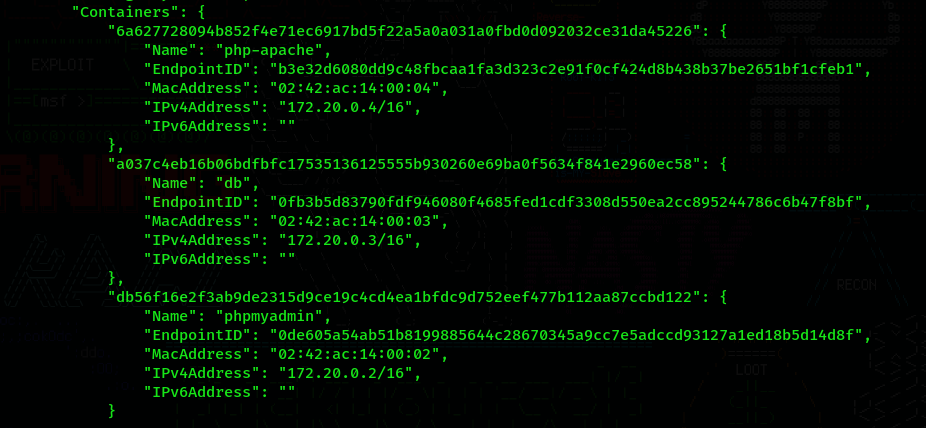


This Nmap scanning perform that port 80 are enabled on our localhost, port 8080 are responsible for http-proxy with the container web server.

* Now the website is redirected to our localhost:80. Lets try:



* At the end I should provide this picture:



As we notice the 3 containers are in the same network 172.20.0.0/16. This process is done by default when we use docker-compose to build a multi-container project.