“Penetration Testing & Web Application Firewall Evaluation using DVWA and ModSecurity”

Sri Lakshmi

Intern @Elevate Labs

◆ Introduction

This project focuses on testing web application security using a lab environment. I set up **Damn Vulnerable Web Application (DVWA)** on Ubuntu as the target system and configured **Mod Security WAF** for defence. From a **Kali Linux attacker machine**, I launched different attacks such as SQL Injection, XSS, and Command Injection. The goal was to compare results with and without the WAF, showing how attacks succeed on DVWA but are blocked when ModSecurity is enabled. This project demonstrates both offensive and defensive security skills in a controlled setup.

◆ Abstract

This project demonstrates a practical approach to web application security testing. A vulnerable web application, **DVWA**, was deployed on Ubuntu and protected with **ModSecurity WAF**. Using **Kali Linux**, various attacks such as SQL Injection, XSS, and Command Injection were executed. The results highlight how vulnerabilities can be exploited when no defenses are in place and how ModSecurity effectively blocks such attacks. This work showcases both offensive testing and defensive mitigation in a controlled lab setup.

◆ Tools Used

Tool	Purpose in Project
DVWA (Damn Vulnerable Web Application)	Target vulnerable web application for testing common attacks.
Ubuntu Server	Hosted DVWA with Apache, MariaDB, and ModSecurity WAF.

Tool	Purpose in Project
Kali Linux	Attacker machine used to perform penetration testing and launch exploits.
ModSecurity (with OWASP CRS)	Web Application Firewall used to detect and block malicious traffic.
Sqlmap	Automated SQL Injection tool to exploit and test DVWA database vulnerabilities.
Burp Suite	Intercepted and modified HTTP requests/responses, analyzed cookies, and validated WAF.
Web Browser (Firefox/Chrome)	Used to access DVWA, perform manual attacks, and observe application behavior.

◆ Steps Involved in Building the Project

1. **Environment Setup** – Installed Ubuntu server with DVWA, Apache, MariaDB, PHP, and ModSecurity; configured Kali Linux as attacker.
2. **Deploy DVWA** – Set up DVWA, configured database, and verified login.
3. **Attack Phase (Without WAF)** – Launched SQLi, XSS, Command Injection, and File Upload attacks from Kali Linux; attacks succeeded.
4. **Enable ModSecurity** – Activated ModSecurity with OWASP CRS and checked logs.
5. **Attack Phase (With WAF)** – Repeated same attacks; most requests blocked by ModSecurity (403 Forbidden).
6. **Analysis** – Compared results with and without WAF; validated logs using Burp Suite and documented findings.
7. **Documentation** – Compiled screenshots, results, and recommendations into a professional report.

◆ Conclusion

This project proved how critical it is to secure web applications against common vulnerabilities. By setting up DVWA as a target and launching real-world attacks from Kali Linux, it was demonstrated that applications without protection are highly exposed to SQL Injection, XSS, Command Injection, and File Upload exploits. After enabling **Mod Security with OWASP CRS**, most of these attacks were successfully blocked, showing the effectiveness of a Web Application Firewall as a defensive layer.