Kensei CyberSec / Vai Na Web 2025

Formação CyberSec

Relatório técnico de defesa e monitoramento

Web Application Firewall (**WAF**) e

Damn Vulnerable Web Application (**DVWA**)

Aluna: Dalila Salvatierra – **Data**: 19/09/2025

Versão: Projeto final Modulo 2 – Opção 1 (Hands On)

Sumário:

Este laboratório teve como propósito avaliar a segurança de uma aplicação web vulnerável (**DVWA**) utilizando o **WAF ModSecurity** com as regras do **OWASP CRS**. A intenção principal foi verificar como o sistema consegue identificar e bloquear ataques, além de monitorar em tempo real e aplicar respostas a incidentes seguindo boas práticas de proteção cibernética.

O ambiente foi montado com **containers Docker**, garantindo um espaço isolado e seguro para testes, além de facilitar a configuração. Os containers utilizados incluíram:

- **Kali Linux:** container dedicado a ataques controlados, executando testes de **SQL Injection (SQLi)** e **Cross-Site Scripting (XSS)**.
- **WAF ModSecurity:** firewall de aplicação web, configurado primeiro em **modo Detecção**, para registrar ataques sem bloqueio, e depois em **modo Blocking**, para impedir ações maliciosas.
- **DVWA:** aplicação vulnerável usada como alvo, ajustada para nível de segurança “Low” para permitir testes de exploração.
- **Dozzle:** ferramenta de monitoramento, permitiu a visualização de logs em tempo real, possibilitando acompanhar todas as detecções e bloqueios do WAF.

Durante a bateria de testes, foram realizados dois tipos de ataques principais:

1. **SQL Injection (SQLi):** tentativas de manipular dados do banco de forma não autorizada, enviando comandos maliciosos pelos parâmetros da aplicação.
2. **Cross-Site Scripting (XSS):** inserção de scripts maliciosos que poderiam ser executados no navegador, explorando vulnerabilidades da aplicação.

Resultados obtidos:

- No **modo Detecção**, os ataques foram registrados nos logs do WAF, mas a aplicação ainda respondeu normalmente (HTTP 302), permitindo estudar o comportamento dos ataques sem interrupções.

- No **modo Blocking**, os ataques foram efetivamente bloqueados, retornando HTTP 403, o que demonstra a capacidade do WAF de prevenir ações maliciosas antes que atinjam a aplicação.

- A observação em tempo real com o **Dozzle** permitiu identificar os **Rule IDs 942100 (SQLi) e 941100 (XSS)**, confirmando que as regras do OWASP CRS foram aplicadas corretamente.

- Logs detalhados e capturas de tela foram coletados como evidência do processo de detecção, bloqueio e resposta.

Conclusões:

O laboratório comprovou que um **WAF bem configurado** é eficaz na proteção contra ataques comuns em aplicações web. A experiência demonstrou a importância de manter regras atualizadas, monitorar logs constantemente e responder rapidamente a incidentes. Observou-se também que o **modo Detecção** é útil para análise e aprendizado, enquanto o **modo Blocking** se destacou como essencial em ambientes reais.

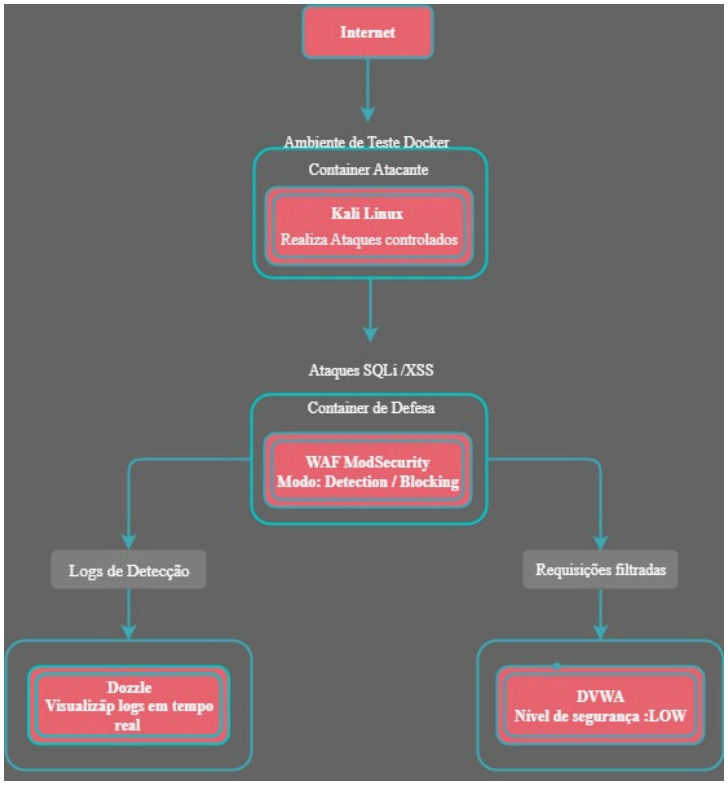
Objetivo e escopo:

Objetivo: Testar eficácia do WAF, monitoramento e resposta a incidentes.

Escopo:

- Protegido: DVWA (nível Low)
- Ataques: SQLi e XSS
- Limite: ambiente isolado em containers Docker

Arquitetura (Diagrama):



Metodologia:

A metodologia utilizada para este teste é baseada no tutorial "Lab de segurança WAF = DVWA" da Formação CyberSec, publicado no GitHub TUTORIAL-COMPLETO

1. Preparar ambiente Docker e containers.
2. Reconhecimento com **Nmap** para identificar serviços e portas.
3. Testar ataques:
   - Modo Detecção → registros em logs
   - Modo Blocking → bloqueio efetivo
4. Monitorar logs em Dozzle
5. Coletar evidências (logs, prints)
6. Aplicar resposta a incidentes NIST IR

Execução e Evidências:

- **SQLi e XSS:**
  - Modo Detecção → HTTP 302
  - Modo Blocking → HTTP 403
- **Logs:**
  - logs_waf_evidencias.txt
  - Rule IDs: 942100 (SQLi), 941100 (XSS)
- **Monitoramento:** prints do Dozzle mostrando detecção e bloqueio

Resposta a Incidente (NIST IR)

Fase NIST IR	Descrição/Ação	Exemplos de Logs/Evidência
Detecção	WAF identifica tráfego suspeito ou padrões de ataque, sem bloquear.	2025/09/18 16:28:05 [alert] [client 192.168.35.10] ModSecurity: Warning. Matched XSS **Attack via libinjection** in ARGS:name TX:ANOMALY_INBOUND_ANOMALY_SCORE: 3
Contenção	Ativação do modo **blocking** no WAF; ataques passam a ser bloqueados automaticamente.	2025/09/18 16:30:12 [notice] ModSecurity: SecRuleEngine set to "On" Indica que regras de bloqueio estão ativas
Erradicação	Requisições maliciosas são bloqueadas efetivamente pelo WAF.	2025/09/18 16:30:45 [error] [client 192.168.35.10] ModSecurity: **Access denied** with code 403 (phase 2). Pattern match "SQL Injection Attack" TX:ANOMALY_INBOUND_ANOMALY_SCORE: 8
Recuperação	Garantir que a aplicação continue disponível e monitorar logs para confirmar que ataques não retornaram.	Logs posteriores mostram requisições legítimas sem bloqueios indevidos 2025/09/18 16:35:10 [notice] [client 192.168.35.11] Request processed successfully

- **Detecção:** ataques registrados pelo WAF em modo DetectionOnly
- **Contenção:** ativação do WAF em modo Blocking
- **Erradicação:** ataques interrompidos e aplicação protegida
- **Recuperação:** ambiente mantido operacional
- **Lições Aprendidas:** importância de monitoramento contínuo, revisão de regras e automatização de alertas

Recomendações (80/20)

1. Manter WAF ativo em **modo Blocking** nos ambientes de produção.
2. Revisar regras **OWASP CRS** regularmente.
3. Monitorar logs continuamente.
4. Testar outros tipos de ataques em ambientes seguros.
5. Treinar equipe para resposta rápida.

Conclusão:

A partir dos testes de laboratório realizados foi constatado que o WAF é eficaz contra ataques comuns (SQLi e XSS), que o monitoramento em tempo real é essencial e que a metodologia NIST IR ajuda a organizar a resposta a incidentes.

Anexos:

Configurações Docker/WAF:

Figura 1 – Iniciando containers, verificação de status de containers e teste de conectividade

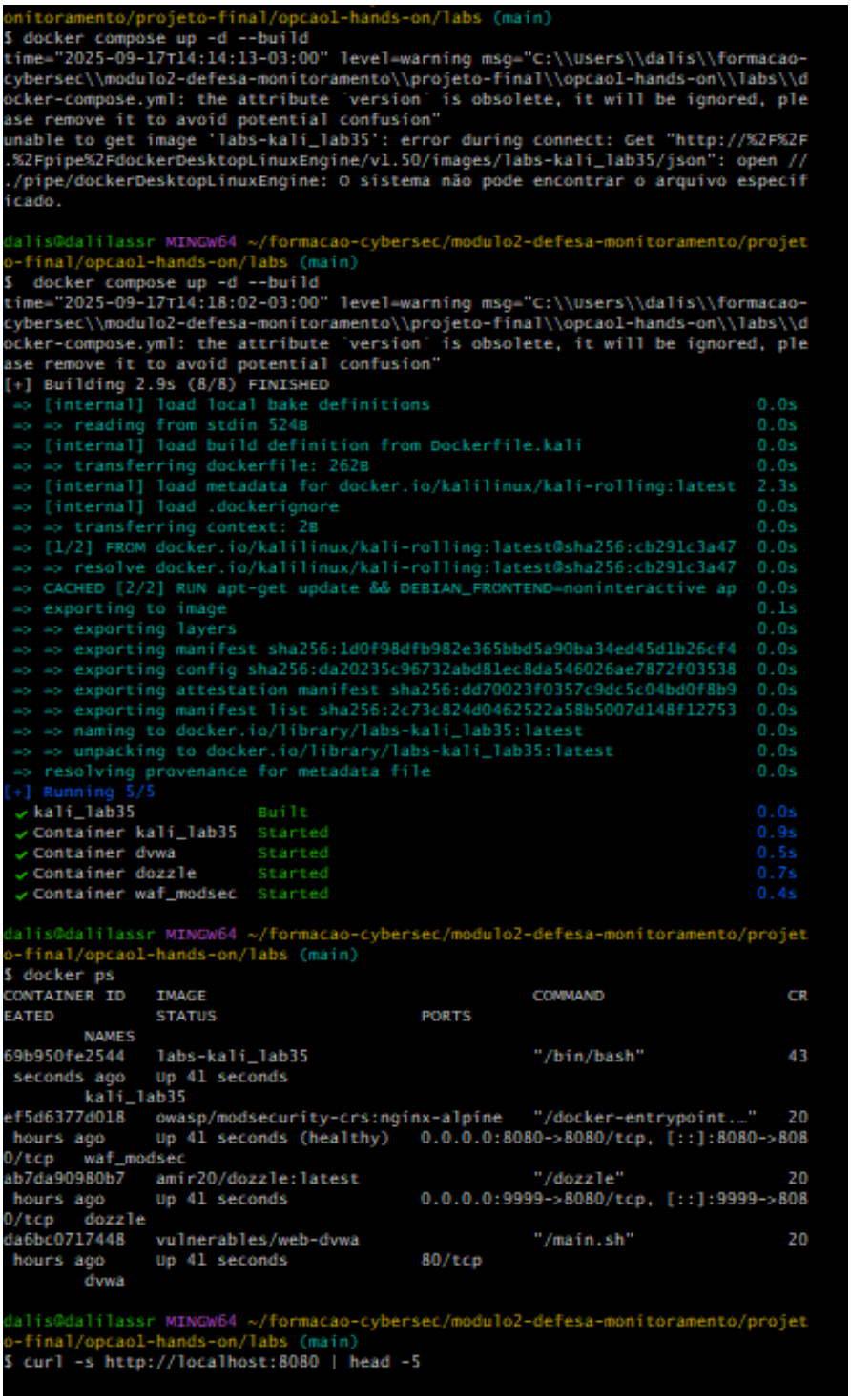
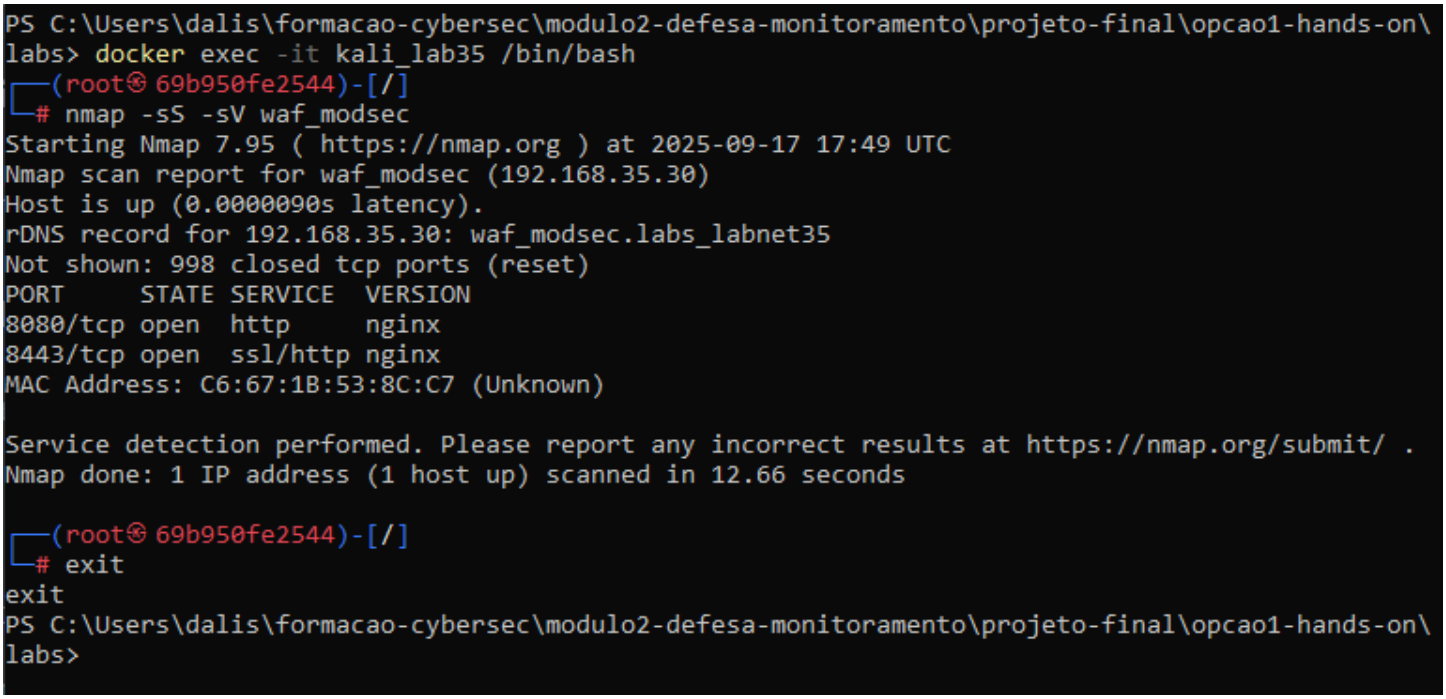


Figura 2 – Dentro do Container Kali, executar Scan Nmap (identificação do WAF rodando nginx nas portas 8080(HTTP) e 8443(HTTPS).



Configuração WAF para Modo Detecção:

Figura 3 – No arquivo docker-compose.yml, Modo Detecção está assim: **-MODSEC_RULE_ENGINE=DetectionOnly**

#modo detecção apenas

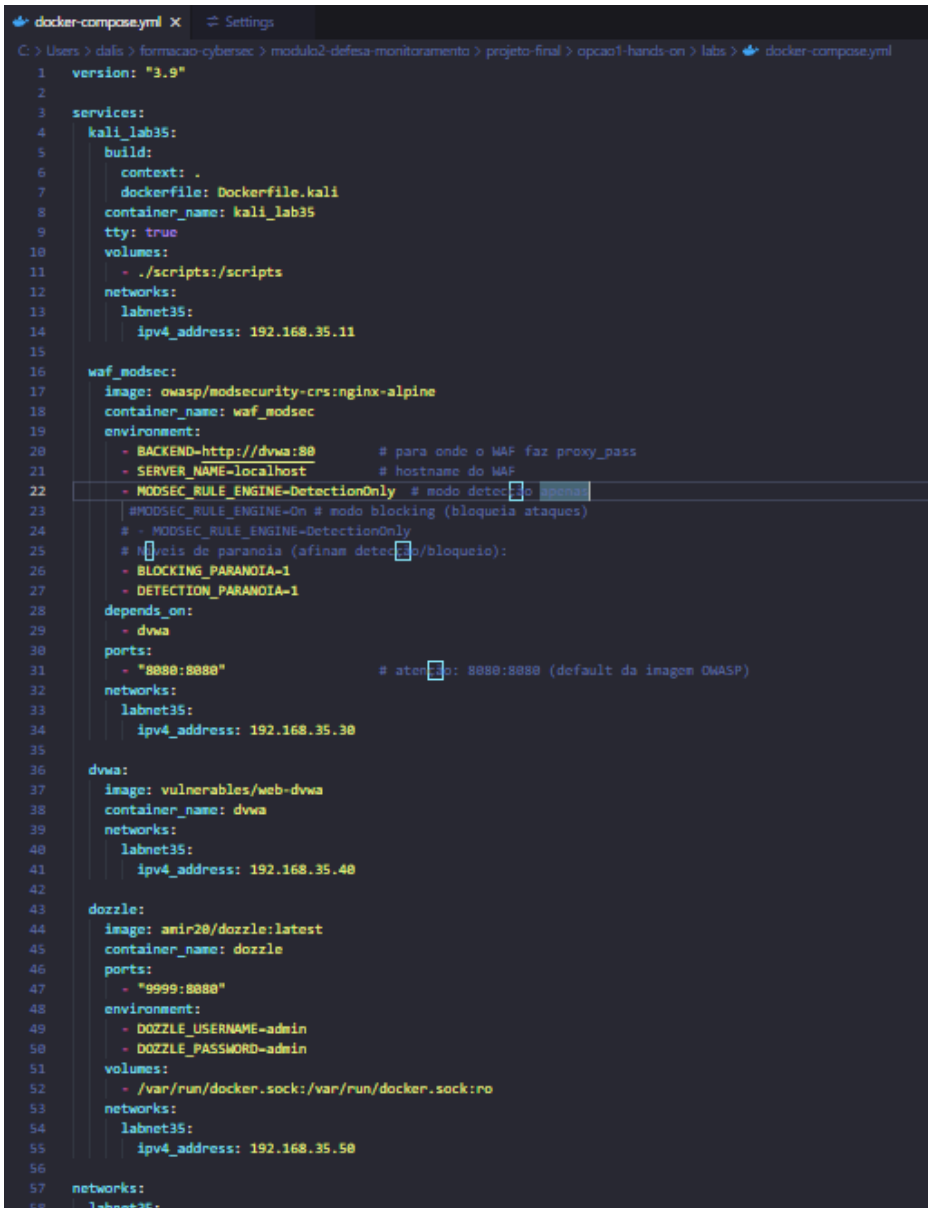


Figura 4 – Container WAF

```
dalis@dalilassr MINGW64 ~/formacao-cybersec/modulo2-defesa-monitoramento/projeto-final/opcao1-hands-on/labs (main)
$ docker compose up -d --force-recreate waf_modsec
time="2025-09-17T15:23:51-03:00" level=warning msg="C:\\Users\\dalis\\formacao-cybersec\\modulo2-defesa-monitoramento\\proj
eto-final\\opcao1-hands-on\\labs\\docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remov
e it to avoid potential confusion"
[+] Running 2/2
 ✓Container dvwa        Running    0.0s
1'+OR+'1'='1'--+-&Submit=Submit" \
  -H "Host: dvwa" \
  -H "Cookie: PHPSESSID=test; security=low" \
  -w "Status: %{http_code}\n"
Status: 302
```

Figura 5 – Ataque SQLi detectado, mas não bloqueado. Status 302

```
dalis@dalilassr MINGW64 ~/formacao-cybersec/modulo2-defesa-monitoramento/projeto-final/opcao1-hands-on/labs (main)
$ docker exec kali_lab35 bash -lc 'while true; do curl -s "http://waf_modsec:8080/vulnerabilities/sqli/?id=1'\''+OR+'\''1
'\''='\''1'\''--+-&Submit=Submit" -H "Host: dvwa" -H "Cookie: PHPSESSID=test; security=low" -w "\nStatus: %{http_code}\n";
sleep 2; done'

Status: 302
```

Figura 6 – Ataque XSS detectado, mas não bloqueado. Status 302

```
dalis@dalilassr MINGW64 ~/formacao-cybersec/modulo2-defesa-monitoramento/projeto-final/opcao1-hands-on/labs (main)
$ docker exec kali_lab35 curl -s "http://waf_modsec:8080/vulnerabilities/xss_r/?name=%3Cscript%3Ealert%28%22XSS%22%29%3C/sc
ript%3E" \
  -H "Host: dvwa" \
  -H "Cookie: security=low" \
  -w "Status: %{http_code}\n"
Status: 302

dalis@dalilassr MINGW64 ~/formacao-cybersec/modulo2-defesa-monitoramento/projeto-final/opcao1-hands-on/labs (main)
$
```

Configuração WAF para Modo Blocking:

Figura 7 – No arquivo docker-compose.yml, alteração para Modo Blocking:– **MODSEC_RULE_ENGINE =On**

#modo blocking (bloqueia ataques)

```yaml
services:
  kali_lab35:
    build:
      context: .
      dockerfile: Dockerfile.kali
    container_name: kali_lab35
    tty: true
    volumes:
      - ./scripts:/scripts
    networks:
      labnet35:
        ipv4_address: 192.168.35.11

  waf_modsec:
    image: owasp/modsecurity-crs:nginx-alpine
    container_name: waf_modsec
    environment:
      - BACKEND=http://dvwa:80        # para onde o WAF faz proxy_pass
      - SERVER_NAME=localhost         # hostname do WAF
      - MODSEC_RULE_ENGINE=On # modo blocking (bloqueia ataques)
      # - MODSEC_RULE_ENGINE=DetectionOnly
      # Níveis de paranoia (afinam detecção/bloqueio):
      - BLOCKING_PARANOIA=1
      - DETECTION_PARANOIA=1
    depends_on:
      - dvwa
    ports:
      - "8080:8080"                   # atenção: 8080:8080 (default da imagem OWASP)
    networks:
      labnet35:
        ipv4_address: 192.168.35.30

  dvwa:
    image: vulnerables/web-dvwa
    container_name: dvwa
    networks:
      labnet35:
        ipv4_address: 192.168.35.40

  dozzle:
    image: amir20/dozzle:latest
    container_name: dozzle
    ports:
      - "9999:8080"
    environment:
      - DOZZLE_USERNAME=admin
      - DOZZLE_PASSWORD=admin
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock:ro
```

Figura 8 – Recriando container WAF

```
PS C:\Users\dalis\formacao-cybersec\modulo2-defesa-monitoramento\projeto-fi
nal\> docker compose up -d --force-recreate waf_modsec
time="2025-09-17T15:06:00-03:00" level=warning msg="C:\\Users\\dalis\\forma
cao-cybersec\\modulo2-defesa-monitoramento\\projeto-final\\opcao1-hands-on\
\labs\\docker-compose.yml: the attribute `version` is obsolete, it will be
ignored, please remove it to avoid potential confusion"
[+] Running 2/2
 ✔Container dvwa        Running                                    0.0s
 ✔Container waf_modsec  Started                                    1.5s
PS C:\Users\dalis\formacao-cybersec\modulo2-defesa-monitoramento\projeto-fi
nal\opcao1-hands-on\labs>
```

Figura 9 – Ataque SQLi Bloqueado. Status 403 + página "403 Forbidden"

```
dalis@dalilassr MINGW64 ~/formacao-cybersec/modulo2-defesa-monitoramento/projeto-final/opcao1-hands-on/labs (main)
$ docker exec kali_lab35 curl -s "http://waf_modsec:8080/vulnerabilities/sqli/?id=1'+OR+'1'='1'--+-&Submit=Submit" \
  -H "Host: dvwa" \
  -H "Cookie: PHPSESSID=test; security=low" \
  -w "Status: %{http_code}\n"
<html>
<head><title>403 Forbidden</title></head>
<body>
<center><h1>403 Forbidden</h1></center>
<hr><center>nginx</center>
</body>
</html>
Status: 403
```

Figura 10 – Ataque XSS Bloqueado. Status 403 + página "403 Forbidden"

```
$ docker exec kali_lab35 curl -s "http://waf_modsec:8080/vulnerabilities/xss_r/?name=%3Cscript%3Ealert%28%22XSS%22%29%3C/sc
ript%3E" \
  -H "Host: dvwa" \
  -H "Cookie: security=low" \
  -w "Status: %{http_code}\n"
<html>
<head><title>403 Forbidden</title></head>
<body>
<center><h1>403 Forbidden</h1></center>
<hr><center>nginx</center>
</body>
</html>
Status: 403
```

Interface Dozzle

Logs do WAF, container “waf-modsec”:

Figura 11 - Nginx e ModSecurity foram iniciados corretamente.

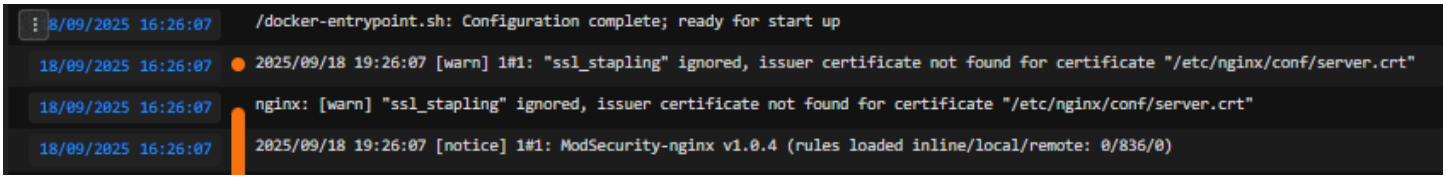


Figura 12 - Acesso normal - Detecção sem bloqueio

Figura 13 - SQL Injection (SQLi) - Bloqueado

Figura 14 - XSS (Cross-Site Scripting)- Bloqueado

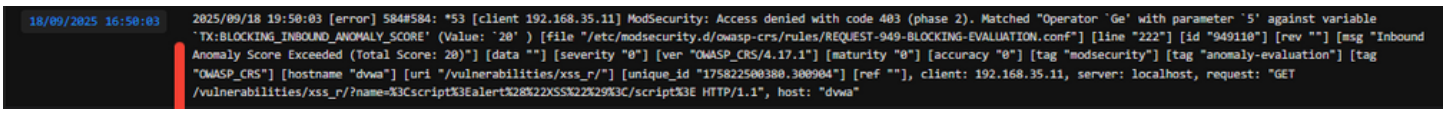


Figura 15 – Nmap: nmap_waf_evidencias.txt

```
┌──(root㉿69b950fe2544)-[/]
└─# cat nmap_waf_evidencias.txt
Starting Nmap 7.95 ( https://nmap.org ) at 2025-09-17 19:55 UTC
Nmap scan report for waf_modsec (192.168.35.30)
Host is up (0.000017s latency).
rDNS record for 192.168.35.30: waf_modsec.labs_labnet35
Not shown: 998 closed tcp ports (reset)
PORT     STATE SERVICE  VERSION
8080/tcp open  http     nginx
8443/tcp open  ssl/http nginx
MAC Address: EE:C5:B6:6A:94:1D (Unknown)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
```