**RELATÓRIO – Opção 1 (Hands-on)**

**Nome:** Yúri Cordeiro  
**Data:** 16/09/2025

**1. Sumário Executivo**

Este exercício simulou ataques de aplicação web contra o ambiente **DVWA** protegido por um **WAF (ModSecurity + OWASP CRS)** em contêiner Docker.  
Foram realizados testes em **modo DetectionOnly** (alerta sem bloqueio) e em **modo Blocking** (negação ativa).  
Ataques simulados: **SQL Injection (SQLi)** e **Cross-Site Scripting (XSS)**. O ambiente detectou corretamente no modo de detecção e bloqueou no modo ativo, comprovando a eficácia do WAF.

**2. Objetivo e Escopo**

* **Objetivo:** Avaliar a capacidade do WAF em detectar e bloquear ataques de injeção em aplicações web.
* **Escopo:**
  + Aplicação vulnerável **DVWA** exposta via proxy WAF.
  + Ataques simulados: SQLi e XSS (futuros: Command Injection, File Inclusion).
  + Não incluído: análise de evasão avançada, fuzzing, escalonamento de privilégios.

**3. Arquitetura (Diagrama)**

Fluxo simplificado:

Attacker (Kali) → WAF (ModSecurity + CRS) → DVWA (App Vulnerável)

* **Camada de detecção:** ModSecurity com regras CRS.
* **Camada de contenção:** futura integração com iptables.
* **Camada de resposta:** tuning + automação de bloqueio.

**4. Metodologia**

1. Configuração inicial do WAF em **DetectionOnly**.
2. Testes de ataque com curl a partir do contêiner Kali.
3. Coleta de evidências em logs (docker logs e Dozzle).
4. Reconfiguração para **Blocking**.
5. Reexecução dos ataques para validação do bloqueio.
6. Organização das evidências em diretório ./evidencias/.

Critério de sucesso:

* **DetectionOnly** → resposta 302 + log do ataque.
* **Blocking** → resposta 403 Forbidden + log detalhado CRS.

**5. Execução e Evidências**

**5.1 SQLi**

* Payload:
* id=1'+OR+'1'='1'--+-
* Resultado:
  + DetectionOnly → Status 302 Found
  + Blocking → Status 403 Forbidden
* Log ModSecurity:
  + 949110 SQL Injection Attack Detected

**5.2 XSS**

* Payload:
* <script>alert("XSS")</script>
* Resultado:
  + DetectionOnly → Status 302 Found
  + Blocking → Status 403 Forbidden
* Log ModSecurity:
  + 941100 XSS Attack Detected

**5.3 Evidências coletadas**

* logs\_waf\_evidencias.txt
* Screenshots Dozzle mostrando entradas CRS.
* nmap confirmando portas 8080 (HTTP) e 8443 (HTTPS).

**6. Resposta a Incidente (NIST IR)**

* **Detecção:** ataques SQLi e XSS identificados em logs CRS.
* **Análise:** verificação dos payloads e resposta da aplicação.
* **Contenção:** troca de modo para **Blocking** e bloqueio ativo.
* **Erradicação:** ataques não mais executados com sucesso.
* **Recuperação:** serviço DVWA permanece operacional, apenas ataques foram bloqueados.
* **Lições Aprendidas:** ModSecurity é efetivo, mas tuning é essencial para reduzir falsos positivos.

**7. Recomendações (80/20)**

| **Ação** | **Esforço** | **Impacto** | **Relação impacto/esforço** | **Prioridade** |
| --- | --- | --- | --- | --- |
| Expandir testes: Command Injection e File Inclusion | Médio | Médio | Médio | 4 |
| Configurar iptables no host (contenção) | Baixo | Alto | Alto | 2 |
| Automatizar resposta (scripts: tcpdump, alertas + bloqueio) | Médio | Alto | Médio | 3 |
| **Tuning das regras CRS / ModSecurity** | Baixo | **Alto** | **Muito Alto** | **1** |
| **Integrar logs com SIEM (ELK/Wazuh)** | Médio | **Alto** | **Alto** | **1** |

**8. Conclusão**

O exercício demonstrou que o **WAF em modo DetectionOnly** gera alertas confiáveis e que o **modo Blocking** efetivamente mitiga ataques simples de injeção.  
A maturidade demonstrada é nível inicial, mas já há evidência clara de defesa em camadas.  
Próximos passos incluem: expandir ataques simulados, adicionar contenção via firewall, e automatizar resposta para aumentar a eficácia operacional.

**Anexos**

* logs\_waf\_evidencias.txt
* Capturas de tela (Dozzle e terminal)