🛠️ Ejercicio Práctico

📌 **Título:** Instalación de un Entorno de Pruebas Vulnerable con DVWA y Docker

---

🎯 Objetivo del ejercicio:

Aprender a configurar un entorno controlado para pruebas de seguridad usando **Docker** y **DVWA (Damn Vulnerable Web Application)** desde Kali Linux u otra distribución compatible.

📝 Escenario:

Trabajas como estudiante de ciberseguridad y necesitas un entorno seguro donde practicar inyecciones SQL, escaneo de vulnerabilidades y explotación web. Para ello, usarás **Docker** para instalar una aplicación vulnerable que funcione en segundos.

🔍 Tu tarea:

✅ Paso 1 – Instalar Docker

1. Abre la terminal y ejecuta el siguiente comando para instalar Docker:

```
sudo apt update
sudo apt install docker.io -y
```

2. Verifica que Docker esté funcionando:

```
sudo systemctl start docker
sudo systemctl enable docker
```

3. Comprueba su estado:

sudo systemctl status docker

✅ Paso 2 – Ejecutar DVWA en Docker

1. Descarga y ejecuta DVWA con el siguiente comando:

sudo docker run --rm -it -d -p 80:80 vulnerables/web-dvwa

2. Abre tu navegador y accede a:

http://localhost

3. Inicia sesión en la interfaz DVWA con:

- Usuario: `admin`
- Contraseña: `password`

✅ Paso 3 – Explorar la interfaz de DVWA

1. Cambia el nivel de seguridad a “low” en la pestaña **DVWA Security**
2. Navega por las secciones disponibles (SQL Injection, File Upload, XSS, etc.)

✅ Paso 4 – Verificación de funcionamiento

1. Prueba insertar un valor como `' OR '1'='1` en el campo de “User ID” en la sección de SQL Injection.
2. Observa si retorna múltiples usuarios.
3. Si lo hace, el entorno ha sido correctamente configurado.

🧪 Resultado Esperado:

- Docker instalado y ejecutándose
- DVWA accesible vía navegador
- Prueba de SQL Injection realizada con éxito
- Seguridad configurada en nivel bajo para entrenamiento

🧠 Reflexión Final:

¿Por qué es importante tener un entorno aislado para practicar ataques?
¿Qué aprendiste sobre la instalación y uso de DVWA con Docker?

🛠️ Solución – Ejercicio Práctico

📌 Instalación de un Entorno de Pruebas Vulnerable con DVWA y Docker

---

🎯 Objetivo cumplido:

Se logró instalar y ejecutar **Docker** en Kali Linux, se desplegó la aplicación vulnerable **DVWA**, se accedió correctamente vía navegador, y se validó una vulnerabilidad de inyección SQL simple como prueba funcional del entorno.

✅ Paso 1 – Instalación de Docker

Comandos ejecutados:

sudo apt update

sudo apt install docker.io -y

sudo systemctl start docker

sudo systemctl enable docker

sudo systemctl status docker

Resultado:
Docker instalado correctamente y en ejecución. El servicio se inicia automáticamente con el sistema operativo.

✅ Paso 2 – Ejecución de DVWA en Docker

Comando utilizado:

sudo docker run --rm -it -d -p 80:80 vulnerables/web-dvwa

Verificación:
Se accede a DVWA desde el navegador en `http://localhost`.

Credenciales utilizadas:

- Usuario: `admin`
- Contraseña: `password`

Resultado:
Acceso exitoso al panel de DVWA.

✅ Paso 3 – Exploración de la Interfaz

- Se cambió el nivel de seguridad a **Low** desde el panel de configuración
- Se revisaron los módulos: *SQL Injection*, *Command Injection*, *XSS*, *File Upload*

Observación:
La plataforma respondió correctamente al cambio de configuración y navegación por módulos.

✅ Paso 4 – Prueba de Vulnerabilidad

Prueba ejecutada en módulo SQL Injection:
Se ingresó el siguiente payload en el campo de “User ID”:

' OR '1'='1

Resultado obtenido:
El sistema devolvió múltiples registros, confirmando la vulnerabilidad.

🧪 Resultado esperado alcanzado:

✔ Docker en ejecución
✔ DVWA desplegado correctamente
✔ Acceso web funcional
✔ Explotación exitosa de vulnerabilidad básica (SQLi)

🧠 Reflexión Final

Este ejercicio permitió familiarizarme con la configuración de un entorno vulnerable seguro usando Docker. Aprendí la importancia de contar con plataformas prácticas para entrenar habilidades ofensivas sin afectar sistemas reales. Además, comprobé de forma controlada cómo funcionan las inyecciones SQL más básicas, lo que me motivó a seguir explorando técnicas más avanzadas.
