Seguridad de la información – FCEN - DC.UBA.AR

Taller - Seguridad en Aplicaciones Web

Objetivos

El objetivo principal de este taller es aplicar de manera práctica los conceptos teóricos sobre seguridad en aplicaciones web. A través de una serie de ejercicios guiados, te enfrentarás a escenarios realistas donde deberás identificar, explotar y comprender diversas vulnerabilidades comunes.

Al finalizar esta guía, serás capaz de:

- Identificar fallos de seguridad en aplicaciones web.
- Utilizar herramientas profesionales para el análisis de seguridad.
- Comprender el impacto de vulnerabilidades como SQL Injection, fallos en el control de acceso, y manipulación de parámetros.
- Operar en un entorno controlado y seguro para la experimentación.

Entorno de Trabajo y Herramientas

Para realizar la práctica, utilizaremos un entorno virtualizado que ya contiene todas las herramientas necesarias. No necesitas instalar software adicional en tu máquina personal o del laboratorio.

Máquina Virtual (VM) del taller : `websecuritytdojo versión 3.4.1`.

Esta VM incluye un sistema operativo Linux, varias aplicaciones web vulnerables, y una colección de herramientas de seguridad. Se descarga de: https://sourceforge.net/projects/websecuritydojo/files/

Se recomienda tener a mano una copia del documento owasp top 10 2017:
https://wiki.owasp.org/images/5/5e/OWASP-Top-10-2017-es.pdf

Presentación

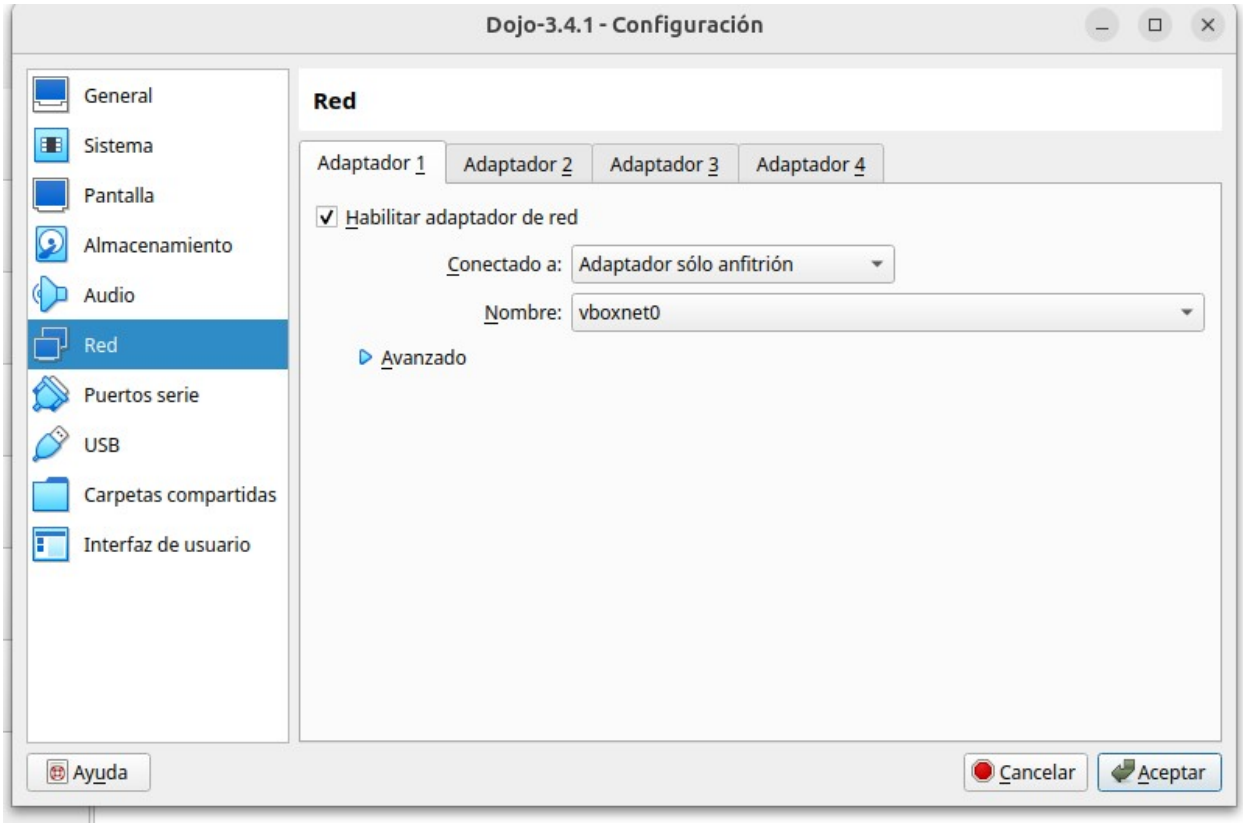
Webgoat será la principal herramienta a atacar en este taller. Es una aplicación deliberadamente insegura diseñada para enseñar sobre vulnerabilidades web en un ambiente controlado.
En cada ejercicio los usuarios deben demostrar la comprensión del problema planteado y explotar la vulnerabilidad propuesta. Por ejemplo, en una de las lecciones el usuario debe utilizar “SQL injection” para obtener números de tarjetas de crédito. La herramienta genera un ambiente de enseñanza realista y provee pistas a los usuarios.

Guía de la práctica

Webgoat
Para realizar los ejercicios propuestos, se debe preparar el equipamiento siguiendo los siguientes pasos.

Importar en virtualbox la máquina virtual (Menu Archivo → Importar servicio virtualizado)

ANTES DE INICIAR LA VM POR PRIMERA VEZ, ir a configuración → Red , y en el Adaptador 1 cambiar “Conectado a:” para que use “Adaptador sólo anfitrión” para evitar problemas con las actualizaciones automáticas:



También se recomienda subir la memoria a 4GB, para mejorar la performance.

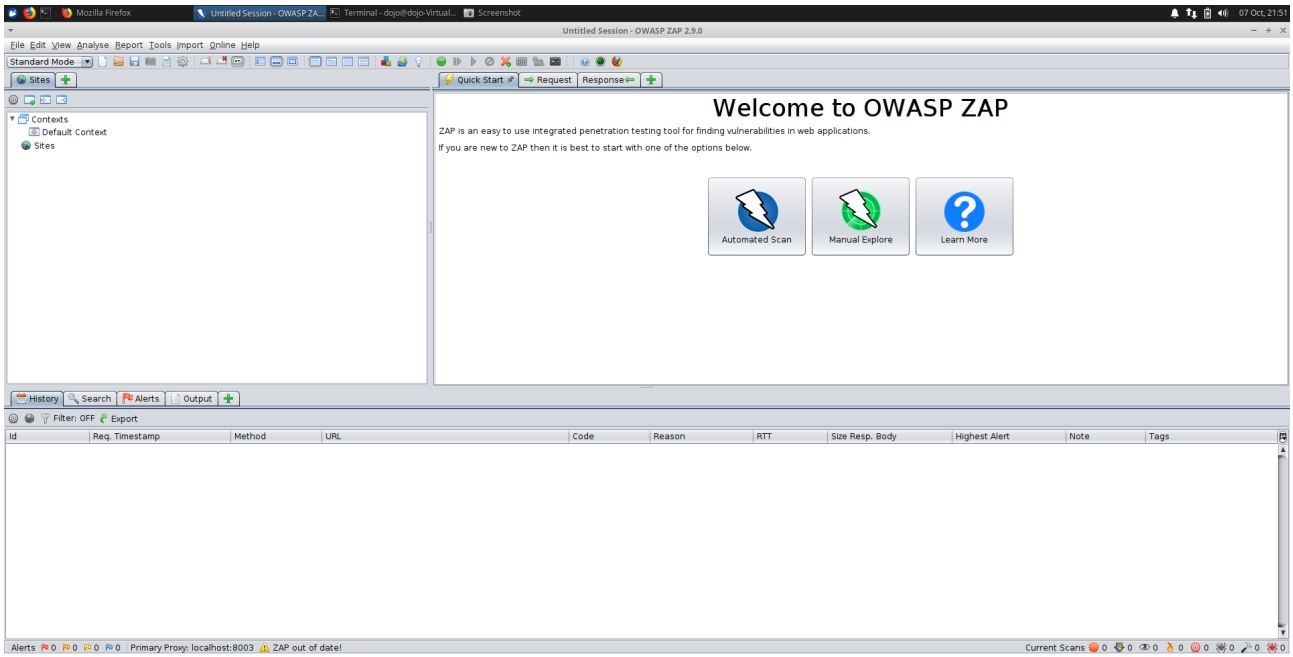
Iniciar la Máquina Virtual.

Iniciar el zap Proxy: Start (el start es el boton de arriba a la izquierda) → tools → Zed Attack Proxy

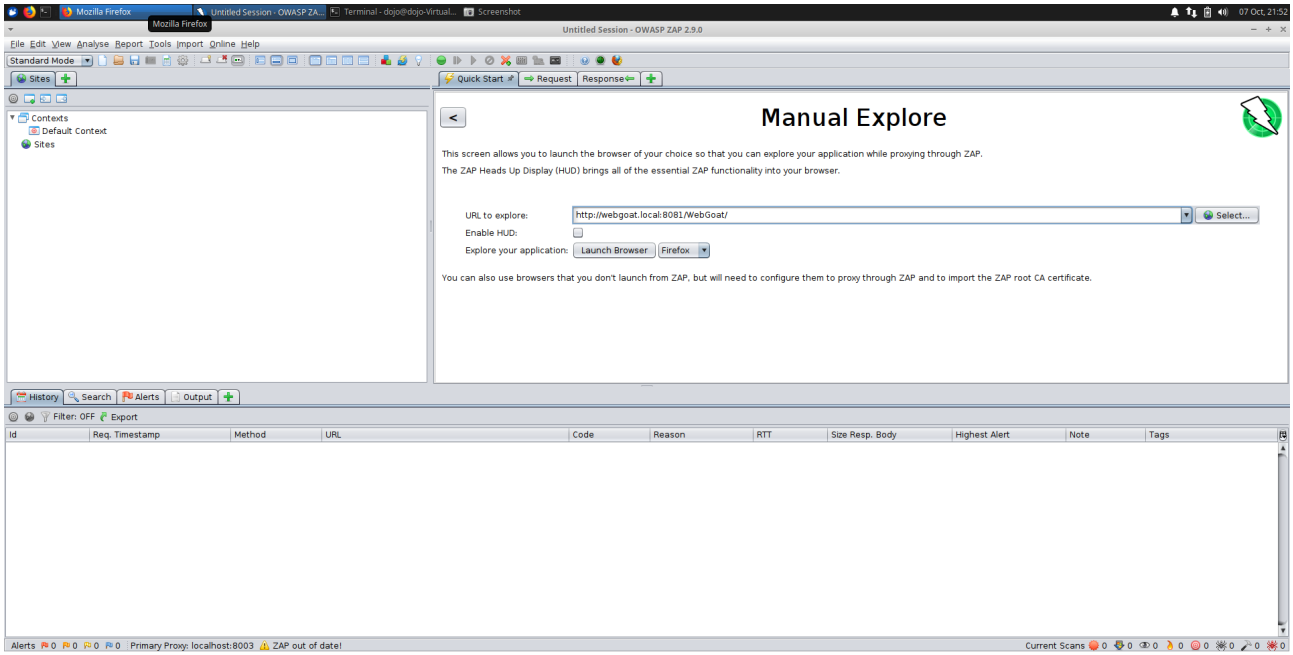
Iniciar webgoat: Start → Targets → Webgoat NG start

Esto último abre el firefox en la pantalla de login de webgoat. Pueden ingresar utilizado usuario: guest y clave: guest.
Sin embargo, esa versión de firefox al rato deshabilita las extensiones para configurar el proxy de manera sencilla, por lo cual no la usaremos.

En su lugar, copiaremos la url de acceso, cerraremos el firefox, y accederemos vía la opción “Manual Explore” de Zap:



En la siguiente pantalla, se pega la url de webgoat copiada antes (http://webgoat.local:8001/WebGoat/), se deshabilita HUD si está habilitado, y se da en launch browser:



Eso abre un nuevo firefox configurado para navegar a través de zap. Recordar ingresar a WebGoat con usuario guest, clave guest y recordar usar el botón verde en ZAP para pausar el reenvio de pedidos entre el browser y el servidor.

Alternativamente, se puede usar la url http://targets.local para acceder al listado de otras herramientas vulnerables (que no están levantadas por defecto)

Sugerencia: Para todos los problemas se recomienda comenzar leyendo el “Lesson Plans” y, de ser necesario, hacer uso de las pistas (Hints). Si no pueden resolver alguno en particular, lean la solución y sigan los pasos descritos.

Selección de ejercicios a resolver en WebGoat

Módulo	Ejercicio a resolver
Code Quality	Discover Clues in the HTML
Parameter Tampering	Exploit hidden fields
Parameter Tampering	Exploit unchecked email
Parameter Tampering	Bypass Client Side JavaScript Validation
Access Control Flaws	Using an Access Control Matrix
Access Control Flaws	Bypass a Path Based Access Control Scheme
Authentication Flaws	Forgot Password
Session Management Flaws	Spoof and Authentication Cookie
Injection Flaws	Command Injection
Injection Flaws	Numeric SQL Injection
Injection Flaws	Log Spoofing
Injection Flaws	String Sql Injection
Injection Flaws	Database Backdoors
Improper Error Handling	Fail Open Authentication Scheme

Luego, con la herramienta sqlmap, hacer el ejercicio de Blind numeric sql injection y listar las bases de datos disponibles.

Opcionalmente, se puede resolver con la herramienta SoapUI (descargarla de https://www.soapui.org/) los ejercicios “Create a SOAP Request” y “Web Service SQL Injection” del módulo de web services.

Hacme casino (opcional)

Esta es otra aplicación web con vulnerabilidades, que simula un casino online.

Se recomienda cerrar el webgoat antes de empezar.
Luego hay que levantar la aplicación : Start->Targets→Hacme Casino Start.
En Start→Documentation hay una guía “Hacme Casino Users Guide” que describe los desafíos y también las soluciones.

Algunas cosas que se pueden hacer:
1- Ingresar al casino a través del navegador (se abre solo el firefox al levantar la app, pero usen el que se ejecuta desde zap)
2- Hacer un sql injection para saltear formulario de login (ojo que las queries pueden tener paréntesis).
3- El usuario al que accedemos (Andy Aces) no tiene plata. Loguearse con username:
bobby_blackjack , clave twenty_one y transferir plata.

4- Volver como usuario Andy Aces, y entrar al blackjack (o jugar con Bobby_blackjack si no pudieron hacer el sql injection). (reglas del juego: el usuario tiene que juntar cartas que sumen un valor cercano o igual a 21, sin pasarse. El as puede valer 1 u 11, las figuras valen 10. Cuando uno elige “Hit”, pide otra carta).
5- Ver como evitar perder con cartas malas, o de cómo cobrar más de una vez una jugada ganada.
6- Encontrar otros problemas en esta aplicación.

Juicy Shop (opcional)

Esta es otra aplicación con vulnerabilidades, usando tecnologías como Node.js y Angular.
La aplicación simula una tienda online de venta de jugos de frutas y productos relacionados.
Sirve como ejemplo de una aplicación con mucho uso de javascript para el frontend y APIs REST.
Se puede mirar la guía incluida en la VM (Start → Documentation → OWASP Juicy Shop Documentation) para ver cuales son los desafíos.