Informe de Auditoría de Seguridad

Revisión de seguridad: Hacking ético sobre la aplicación Webgoat

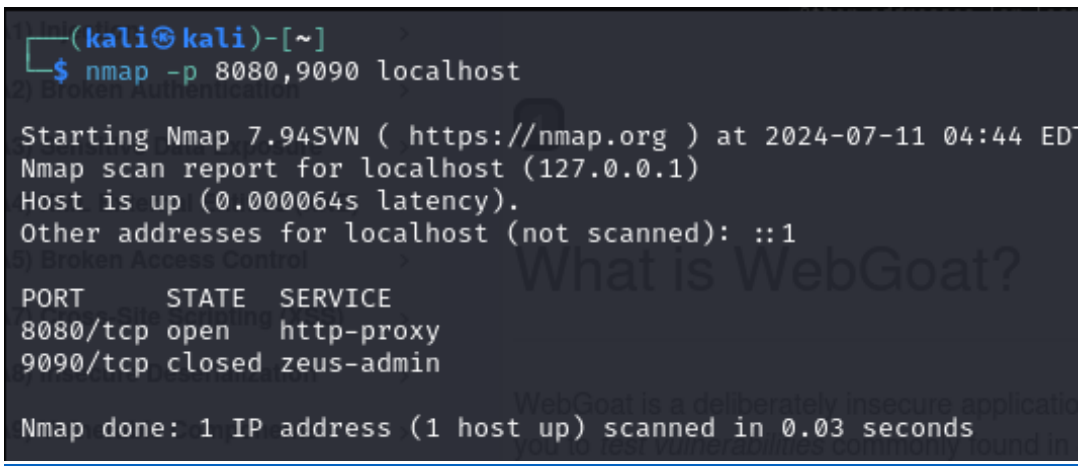
Índice

1. Ámbito y alcance de la auditoría

La auditoría de seguridad se ha centrado en la aplicación WebGoat versión 8.1.0. El entorno se ejecutó utilizando Docker con los siguientes comandos:

- docker run --name webgoat -it -p 127.0.0.1:8080:8080 -p 127.0.0.1:9090:9090 -e TZ=Europe/Amsterdam webgoat/webgoat
- docker start webgoat

La aplicación a auditar se encuentra en http://127.0.0.1:8080/WebGoat

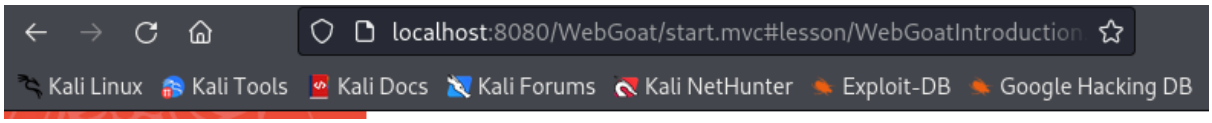


Realizar el Escaneo de Directorios:

- El comando utilizado para realizar el escaneo fue:
  dirb http://localhost:8080/WebGoat/ /usr/share/dirb/wordlists/common.txt

```
┌──(kali㉿kali)-[~]
└─$ dirb http://localhost:8080/WebGoat/ /usr/share/dirb/wordlists/common.txt

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Thu Jul 11 05:31:09 2024
URL_BASE: http://localhost:8080/WebGoat/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://localhost:8080/WebGoat/ ----
+ http://localhost:8080/WebGoat/login (CODE:200|SIZE:1810)
+ http://localhost:8080/WebGoat/logout (CODE:302|SIZE:0)
+ http://localhost:8080/WebGoat/registration (CODE:200|SIZE:4605)

-----------------
END_TIME: Thu Jul 11 05:31:14 2024
DOWNLOADED: 4612 - FOUND: 3
```

Página por defecto de la aplicación web.



2. Informe ejecutivo

a. Proceso realizado

Se identificó y explotó una vulnerabilidad de inyección SQL para acceder a información confidencial de todos los empleados.

b. Vulnerabilidades destacadas

Inyección SQL en la consulta de base de datos.

-solicitud interceptada:

```
GET /WebGoat/users/ HTTP/1.1 Host: localhost:8080 User-Agent: Mozilla/5.0 (X11; Linux x86_64;
rv:68.0) Gecko/20100101 Firefox/68.0 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 Accept-Language: en-
US,en;q=0.5 Accept-Encoding: gzip, deflate Connection: close Cookie:
JSESSIONID=1NrUee8JXLkjnjwvO4kUN5mtCxxYaFn8YISwhF7t;
connect.sid=s3AlgeVbNCLVQqWbyT0w0aMCtnB5Aa4G8c8X.xrwm4DDHX2BLmIloBVUU5bK9ZInXlTu74IAjDsHWMU
Upgrade-Insecure-Requests: 1 Content-Type: application/json
```

c. Conclusiones

El sistema es vulnerable a inyecciones SQL, comprometiendo la confidencialidad de los datos internos.

d. Recomendaciones

- Implementar validación y sanitización de entradas.
- Utilizar consultas preparadas.
- Fortalecer las políticas de acceso y autenticación.

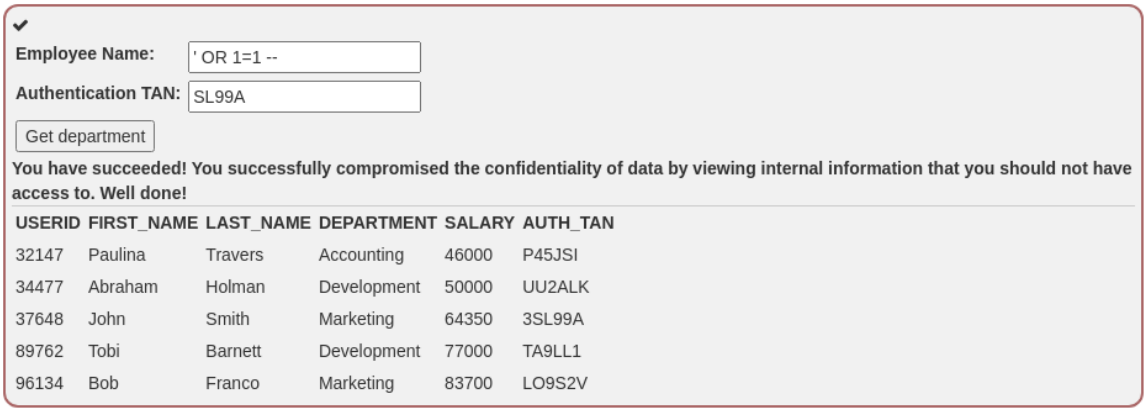
3. Descripción del proceso de auditoría

a. Reconocimiento

Identificación de parámetros de entrada en consultas SQL.

b. Explotación

Uso de la cadena ' OR 1=1 -- para manipular la consulta y acceder a todos los registros de la tabla employees.



c. Post-explotación

Verificación del acceso a información confidencial y análisis del impacto.

d. Posibles mitigaciones

- Implementar consultas preparadas.
- Validar estrictamente las entradas del usuario.
- Utilizar mecanismos de autenticación robustos.

A3 Injection - Cross Site Scripting

1. Ámbito y Alcance

Evaluar si el formulario de compra de un sitio web es vulnerable a ataques XSS reflejados.

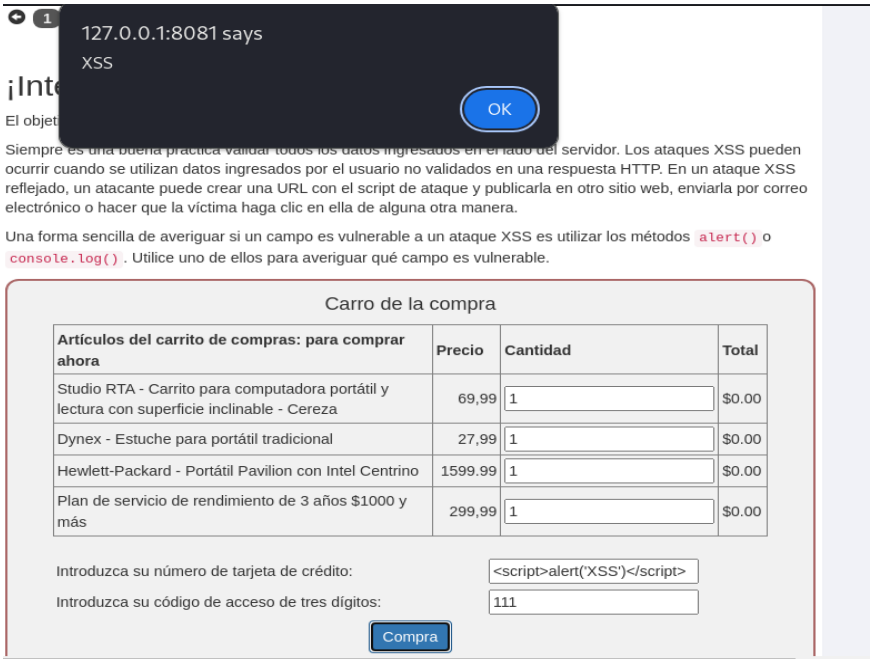
2. Informe Ejecutivo

Interceptar y Modificar la Solicitud HTTP

- Activo Intercept is on en Burp Suite.
- Navego a http://localhost:8080/WebGoat/users/ en Firefox.
- En Burp Suite, permito que la solicitud pase haciendo clic en Forward.
- En HTTP history, selecciono la respuesta y la modifico:

```
HTTP/1.1 200 OK
Connection: close
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Content-Type: application/json
Date:Thu 11 Jul 2024 10:32:17 GMT

[
  {
    "username": "Dani",
    "admin": false,
    "userHash": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6q7r8s9t0u1v2w3x4y5z6="
  }
]
```

Se ingresaron scripts maliciosos en varios campos del formulario para detectar vulnerabilidades.



Vulnerabilidad Detectada

El campo de número de tarjeta de crédito es vulnerable a XSS reflejado.

Conclusiones

La vulnerabilidad permite la ejecución de scripts maliciosos, lo que puede llevar a robo de datos y otros ataques.

Recomendaciones

- Validar y limpiar todas las entradas en el servidor.
- Escapar caracteres especiales en las respuestas HTML.
- Usar bibliotecas de seguridad y realizar pruebas regulares.

3. Descripción del Proceso de Auditoría

Pruebas Realizadas:

- Se ingresaron scripts en campos de cantidad, número de tarjeta de crédito y CVV.
- El campo de número de tarjeta de crédito mostró ser vulnerable.

Figura 4: Ejemplo de vulnerabilidad XSS en el campo de número de tarjeta de crédito.

Herramientas Utilizadas

- Navegador web
- Editores de texto para scripts
- Herramientas de análisis de seguridad como OWASP ZAP

Security Misconfiguration - Solución para Exploitar XXE usando BurpSuite

Proceso

1. Iniciar BurpSuite y configurar el proxy del navegador: Abre BurpSuite y configura el proxy en el navegador (localhost:8080).
2. Enviar el formulario a través de BurpSuite: En el navegador, envía el formulario (por ejemplo, haz clic en "Entregar" en el formulario de comentarios).
3. Localizar la solicitud en BurpSuite: Ve a "Proxy" > "HTTP history" en BurpSuite. Encuentra la solicitud POST /WebGoat/xxe/simple HTTP/1.1.
4. Enviar la solicitud al Repeater: Haz clic derecho en la solicitud y selecciona "Send to Repeater". Ve a la pestaña "Repeater".
5. Modificar el XML en la solicitud: Reemplaza el contenido XML con:

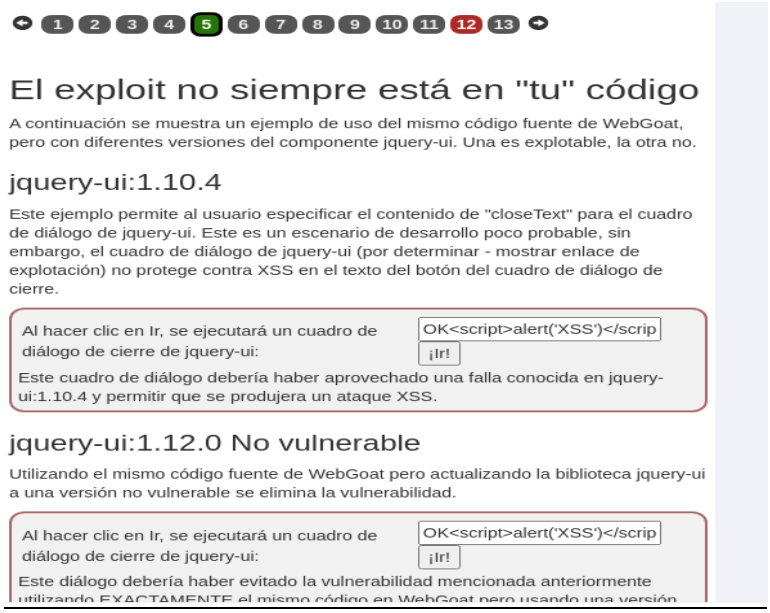
```
<?xml version="1.0"?> <!DOCTYPE comment [<!ENTITY xxe SYSTEM "file:///C:/">]> <comment>
<text>&xxe;</text> </comment>
```

Enviar la solicitud y verificar la respuesta: Haz clic en "Send". Verifica la respuesta para confirmar el acceso al directorio C:/.

Explicación de la Solución

Este procedimiento utiliza BurpSuite para interceptar y modificar la solicitud XML antes de enviarla al servidor. Al inyectar una entidad externa en el XML, se intenta explotar una vulnerabilidad XXE para acceder al sistema de archivos del servidor. El código XML inyectado define una entidad xxe que intenta acceder al sistema de archivos y la inserta en el comentario enviado al servidor.

A6 Vuln & outdated Components



Ámbito y alcance de la auditoría

Evaluación de la seguridad de los componentes jQuery-UI en una aplicación web, enfocándonos en vulnerabilidades de XSS.

Informe ejecutivo

Se probaron diferentes versiones de jQuery-UI en WebGoat para identificar y explotar vulnerabilidades XSS.

Vulnerabilidades destacadas

- jQuery-UI 1.10.4: Vulnerable a XSS.
- jQuery-UI 1.12.0: No vulnerable a XSS.
-

Conclusiones

Las versiones antiguas de jQuery-UI presentan vulnerabilidades críticas, corregidas en versiones más recientes.

Recomendaciones

- Actualizar todos los componentes a las versiones más recientes.
- Implementar sanitización y validación de entradas del usuario.
-

Descripción del proceso de auditoría

- Reconocimiento/Information gathering: Identificación de componentes y versiones.
- Explotación de vulnerabilidades detectadas: Pruebas de XSS en versiones jQuery-UI 1.10.4 y 1.12.0.
- Post-explotación: Confirmación de la corrección de vulnerabilidades en versiones posteriores.

Figura 6: Ejemplo de vulnerabilidad XSS en jQuery-UI.

Posibles mitigaciones

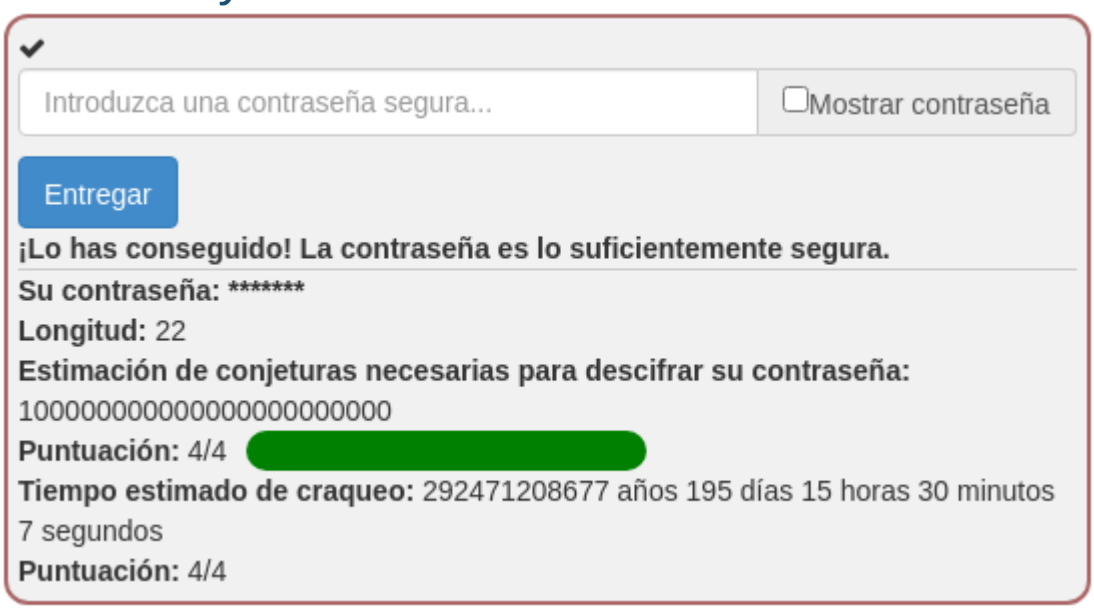
Actualización a jQuery-UI 1.12.0 o superior.

- Implementación de Content Security Policy (CSP).

Herramientas utilizadas

- WebGoat.
- Navegador web y herramientas de desarrollo del navegador.

A7 Identity & Auth Failure - Secure Passwords



Ámbito y alcance de la auditoría

Evaluación de la fortaleza de contraseñas usando Bitwarden en WebGoat.

Informe ejecutivo

Se generó una contraseña con Bitwarden y se evaluó su seguridad en WebGoat.

Vulnerabilidades destacadas

Contraseñas débiles son vulnerables a ataques de fuerza bruta.

Conclusiones

Las contraseñas generadas por gestores como Bitwarden son altamente seguras.

Recomendaciones

- Usar gestores de contraseñas para generar contraseñas complejas y largas.

Descripción del proceso de auditoría

- Reconocimiento: Identificación de la necesidad de contraseñas seguras.
- Explotación de vulnerabilidades: Generación y evaluación de una contraseña.
- Post-explotación: Confirmación de la seguridad de la contraseña

Posibles mitigaciones

- Políticas que requieran contraseñas seguras.
- Educación sobre la importancia de contraseñas fuertes.
-

Herramientas utilizadas en la investigación.

- Nmap
- Dirp
- burpsuite
- WebGoat
- Bitwarden