Informe Inyección SQL en Bee-Box.

Jose Rodriguez

Índice

Introducción

- This document demonstrates security testing performed in a controlled lab environment using intentionally vulnerable machines.
- All testing conducted on dedicated lab systems
- No real systems or data were compromised
- "Gaining machine control" refers to educational exercises
- Primary goal: Perform an SQL Injection on the target machine
- These exercises help understand attacker methodologies to build better defenses.

Informe de validación sobre una vulnerabilidad crítica de SQLi detectada en la aplicación Web DVWA. El análisis fue realizado durante una revisión programada, cumpliendo con el siguiente alcance:

Target:

http://10.0.2.6/dvwa/

El análisis se centró exclusivamente en la detección de fallos de inyección, priorizando la búsqueda de vectores de Inyección SQL (SQLi) en todos los campos de entrada de datos del usuario. Se hace constar que otras categorías de vulnerabilidades, tales como Cross-Site Scripting (XSS) o Cross-Site Request Forgery (CSRF), se definieron como fuera del alcance para los fines de esta evaluación específica.

La falla identificada representa un riesgo crítico de **exfiltración de datos**. Un agente de amenaza externo, sin necesidad de credenciales de acceso, podría comprometer la confidencialidad de la información almacenada en el servidor de base de datos, obteniendo registros sensibles de manera no autorizada.

Descripción del Incidente

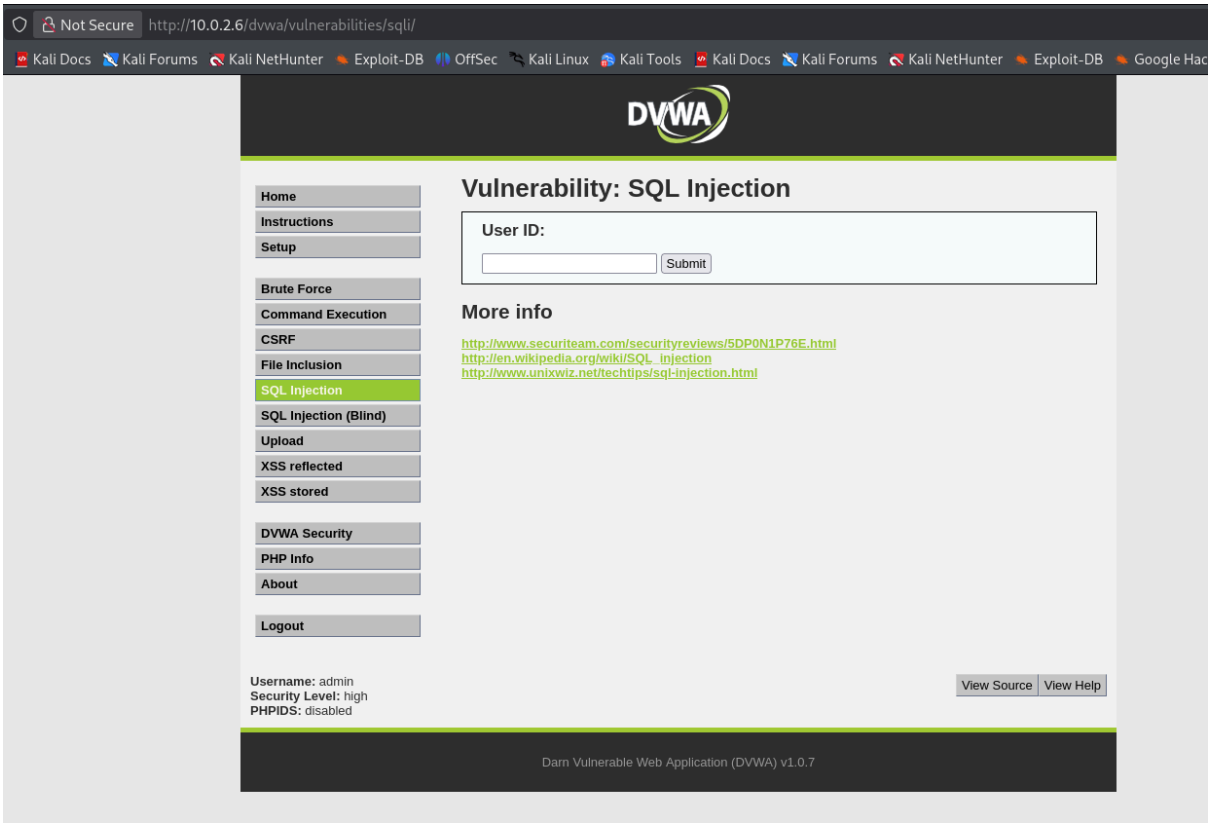
Se ha confirmado una vulnerabilidad de Inyección SQL Basada en Union (Union-Based) localizada en el módulo de consulta de usuarios. El punto de entrada específico es el parámetro asociado al campo 'User ID', el cual procesa entradas del lado del cliente de manera insegura.

Proceso de Reproducción

Un atacante puede explotar esta vulnerabilidad sin necesidad de herramientas especializadas, utilizando únicamente un navegador web.

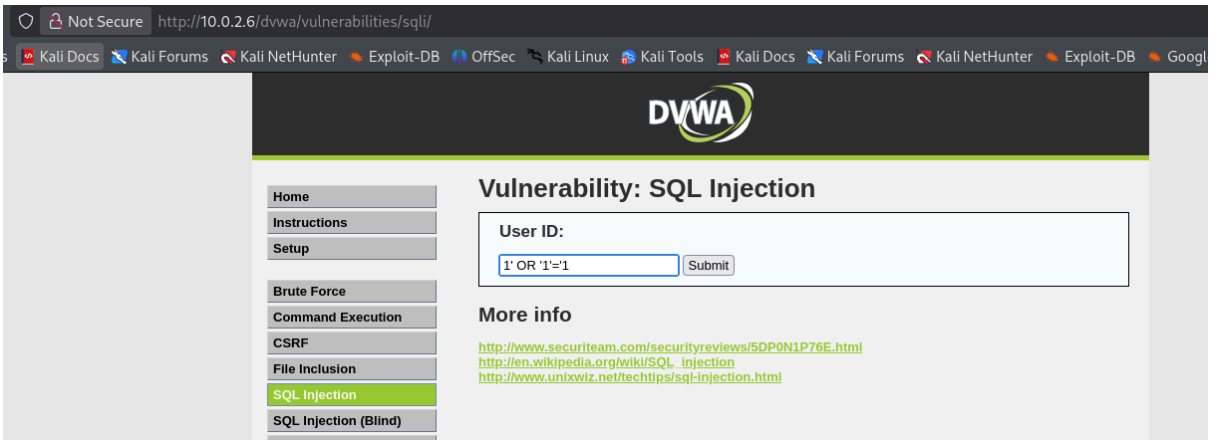
Navegue a la página web:

http://10.0.2.6/dvwa/vulnerabilities/sqli/

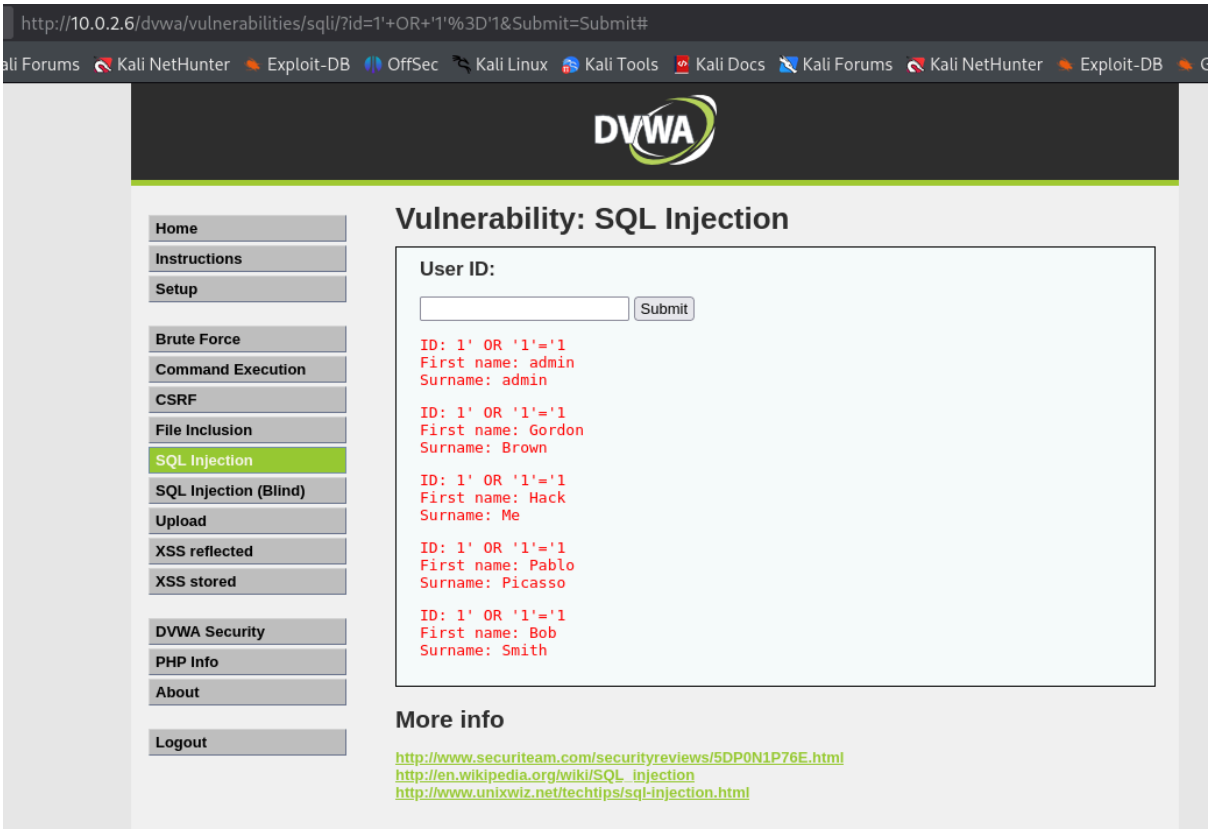


En el cuadro de búsqueda, introduzca el siguiente payload:

1' OR '1'='1



Clic en Submit. En lugar de los resultados de productos, la página de respuesta de la aplicación mostrará una lista de nombres de usuario del sistema y hashes de contraseñas de la tabla users.


http://10.0.2.6/dvwa/vulnerabilities/sqli/?id=1'+OR+'1'%3D'1&Submit=Submit#
Kali Forums
Kali NetHunter
Exploit-DB
OffSec
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
DVWA
Home
Instructions
Setup
Brute Force
Command Execution
CSRF
File Inclusion
SQL Injection
SQL Injection (Blind)
Upload
XSS reflected
XSS stored
DVWA Security
PHP Info
About
Logout
Vulnerability: SQL Injection
User ID:
Submit
ID: 1' OR '1'='1
First name: admin
Surname: admin
ID: 1' OR '1'='1
First name: Gordon
Surname: Brown
ID: 1' OR '1'='1
First name: Hack
Surname: Me
ID: 1' OR '1'='1
First name: Pablo
Surname: Picasso
ID: 1' OR '1'='1
First name: Bob
Surname: Smith
More info
http://www.securiteam.com/securityreviews/5DP0N1P76E.html
http://en.wikipedia.org/wiki/SQL_injection
http://www.unixwiz.net/techtips/sql-injection.html

Impacto del Incidente

Confidencialidad: Impacto Crítico. Los atacantes pueden exfiltrar todos los datos almacenados en la base de datos conectada, incluyendo credenciales de usuario (contraseñas), información de identificación personal (PII), registros financieros y datos comerciales patentados.

Integridad: Impacto Alto. Aunque se demostró como un vector de robo de datos, la misma falla podría permitir potencialmente la modificación o eliminación de registros de la base de datos (INSERT, UPDATE, DELETE), lo que llevaría a la corrupción o pérdida de datos.

Disponibilidad: Impacto Medio. Los payloads de SQL malformados pueden causar errores en la base de datos y denegación de servicio, haciendo que la función de búsqueda de productos no esté disponible.

Riesgo de Negocio: Este fallo constituye una violación directa de las regulaciones de protección de datos (por ejemplo, GDPR, CCPA), podría provocar daños significativos a la reputación y proporciona un camino directo para el compromiso total del sistema.

Recomendaciones

Dada la criticidad del hallazgo, se propone el siguiente cronograma de respuesta:

Fase	Acción	Descripción Técnica
Táctica (Inmediata)	**Contención vía WAF**	Implementar filtros de entrada en el WAF para detectar y bloquear patrones de inyección (filtros de caracteres especiales) en el endpoint de búsqueda.
Estratégica (Corto Plazo)	**Refactorización de Código**	Eliminar la concatenación de cadenas en las consultas SQL y adoptar el uso de **Sentencias Preparadas**.
Preventiva (Largo Plazo)	**Seguridad en el SDLC**	Integrar herramientas de análisis estático (SAST) en el ciclo de desarrollo para identificar fallos de inyección automáticamente antes de pasar a producción.

Ejemplo de Corrección de Código (Causa Raíz)

A continuación, se muestra la diferencia técnica entre la implementación actual y la solución recomendada:

Estado Vulnerable (Concatenación Directa):

Python
```
# EL PELIGRO: La entrada del usuario se vuelve parte del comando SQL
query = "SELECT * FROM products WHERE name LIKE '%" + user_input + "%'"
cursor.execute(query)
```

Estado Remediado (Parametrización):

Python
```
# LA SOLUCIÓN: El motor de BD trata la entrada solo como texto, nunca como comando
query = "SELECT * FROM products WHERE name LIKE %s"
cursor.execute(query, ('%' + user_input + '%',))
```

7. Prevención a Largo Plazo

Para evitar la reincidencia de este tipo de fallos, se recomienda:

1. **Capacitación:** Programas de desarrollo seguro enfocados en el **OWASP Top 10** para el equipo de ingeniería.
2. **Pruebas Automatizadas:** Implementar escaneos **SAST** (durante el desarrollo) y **DAST** (en tiempo de ejecución) para atrapar vulnerabilidades de inyección antes de que lleguen a producción.

Conclusión

La vulnerabilidad de inyección SQL identificada representa un riesgo crítico para la seguridad e integridad de la plataforma DVWA y sus datos. La explotación es trivial y puede conducir a una brecha total de la base de datos de la aplicación. Se

recomienda que el equipo de desarrollo priorice e implemente la corrección mediante consultas parametrizadas como la solución permanente. Se aconseja realizar una prueba de seguimiento tras la remediación para confirmar que la vulnerabilidad se ha resuelto por completo.