Servidores Web de altas prestaciones
Grado en Ingeniería Informática
Universidad de Granada

SQL Injection

Víctor Rebollo Pérez

9 de junio de 2017

Índice

Índice de figuras

1. Introducción

Un ataque por inyección directa de comandos SQL es un método o técnica de infiltración en la cual el atacante se vale de una vulnerabilidad en el nivel de validación de las entradas de la base de datos para exponer datos ocultos, reemplazar datos, eliminarlos,etc

- SQLi es muy común en aplicaciones PHP y ASP y poco común en aplicaciones J2EE, ASP.NET
- Un ataque por SQLi es considerado por lo general de alto riesgo, aunque este depende de la habilidad y la imaginación del atacante como de las defensas en profundidad, como que la conexión al servidor de base de datos tenga bajo privilegio, etc.

SQLi además, es considerado uno del top 10 de mayores vulnerabilidades de aplicaciones Web por Open Web Application Security Project alcanzando el primer puesto

2. Tipos de ataque SQLi

Existen 3 tipos de clases de inyección SQL:

- **In-Band**: Es el tipo de ataque mas común. Ocurre cuando el atacante es capaz de usar el mismo canal de comunicación para atacar y para obtener los resultados del ataque

  Puede ser de dos tipos:

  - **Error-based SQLi**: En este tipo de ataques se trabaja sobre los mensajes de error de la base de datos para obtener información sobre la estructura de la base de datos
  - **Union-based SQLi**: En este tipo de ataque se aprovecha el operador unión para combinar los resultados de dos o más instrucciones SELECT en un único resultado que luego se devuelve como parte de la respuesta

- **Inferential SQLi (Blind SQLi)**: A diferencia del anterior este toma más tiempo para el atacante y es igual de peligroso que el anterior. En un ataque de tipo blind no se transmiten realmente datos a través de la aplicación web y el atacante no será capaz de ver el resultado de un ataque in-band Además un atacante es capaz de reconstruir la estructura de la base de datos únicamente enviando payloads, observando la respuesta de la aplicación y analizando el comportamiento del servidor de la base de datos

  Existen dos tipos:

  - **Boolean-based SQLi**: Se basa en el envío de una consulta SQL a la base de datos que obliga a la aplicación a devolver un resultado diferente dependiendo de si la consulta devuelve true o false.

Esto modificará el contenido de la respuesta HTTP o lo dejará igual, de este modo podemos inferir si el payload usado es verdadero o falso.

- **Time-based SQLi**: Se basa en el envío de una consulta SQL a la base de datos que obliga a la base de datos esperar una cantidad determinada de tiempo antes de responder. El tiempo de respuesta indicará al atacante si el resultado de la consulta es true o false.

- **Out-of-band SQLi**: El menos común de todos porque sucede cuando el atacante es incapaz de usar el mismo canal para lanzar el ataque y para obtener resultados.

3. Ejemplos práctico

Veamos un ejemplo práctico:
Supongamos ahora que tenemos un formulario que nos pide el usuario y el password. Además supongamos que tiene un sistema de seguridad nulo.

El código SQL de la consulta sería el siguiente:

```
var login;
var pass;
login = Request.form("loginForm");
pass = Request.form("passForm");
var ordenSQL = "SELECT \* FROM usuarios
WHERE login ='" + usuario +"' AND password ='" +
password + "';"
```

Si el usuario introduce el login Periquito y pass como contraseña la orden quedaría como la que sigue:

```
SELECT * FROM usuarios WHERE login = 'Periquito'
AND password = 'pass'; \\
```

Ahora llega lo interesante, ¿ y si añadimos una orden estática ?:

```
SELECT * FROM usuarios WHERE usuario = ''
OR 1 = 1 --Seguridad nula \\
```

Recordamos que – sirve en sql para marcar los comentarios luego todo lo que sigue no importará.

De esta forma podemos obtener todos los datos de los usuarios de una forma extremadamente sencilla.

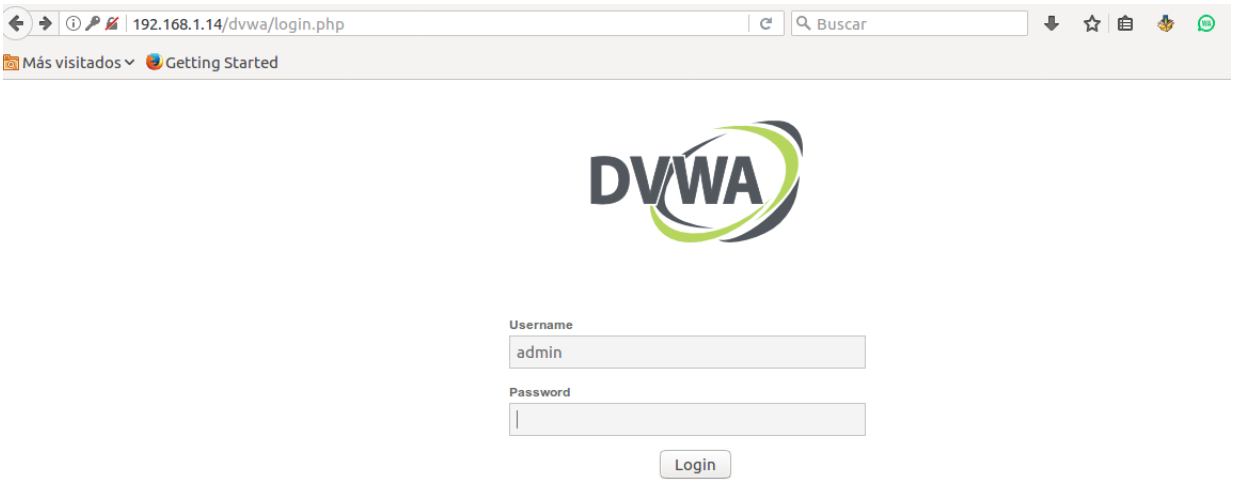

Figura 4.1: Login de DVWA

4. Damn Vulnerable Web Application

DVWA es una aplicación desarrollada en PHP y MySQL para el entrenamiento de explotación de vulnerabilidades web.

Algunas de estas vulnerabilidades son : ejecución de comando, cross-site scripting, RFI, Inyección SQL, etc.

DVWA está divido en tres niveles de seguridad: Low, medium y hight.
Cada uno respectivamente va aumentando su nivel de dificultad. En este caso solo nos centraremos en la dificultad Low

El proceso de instalación es muy sencillo. En la página oficial recomienda el uso de Fedora pero se puede usar en las máquinas Ubuntu empleadas en las clases prácticas de la asignatura.

Una vez configurada obtendremos algo similar a lo siguiente:
El login y el password son respectivamente admin y password.
Una vez dentro tendremos algo parecido a lo siguiente

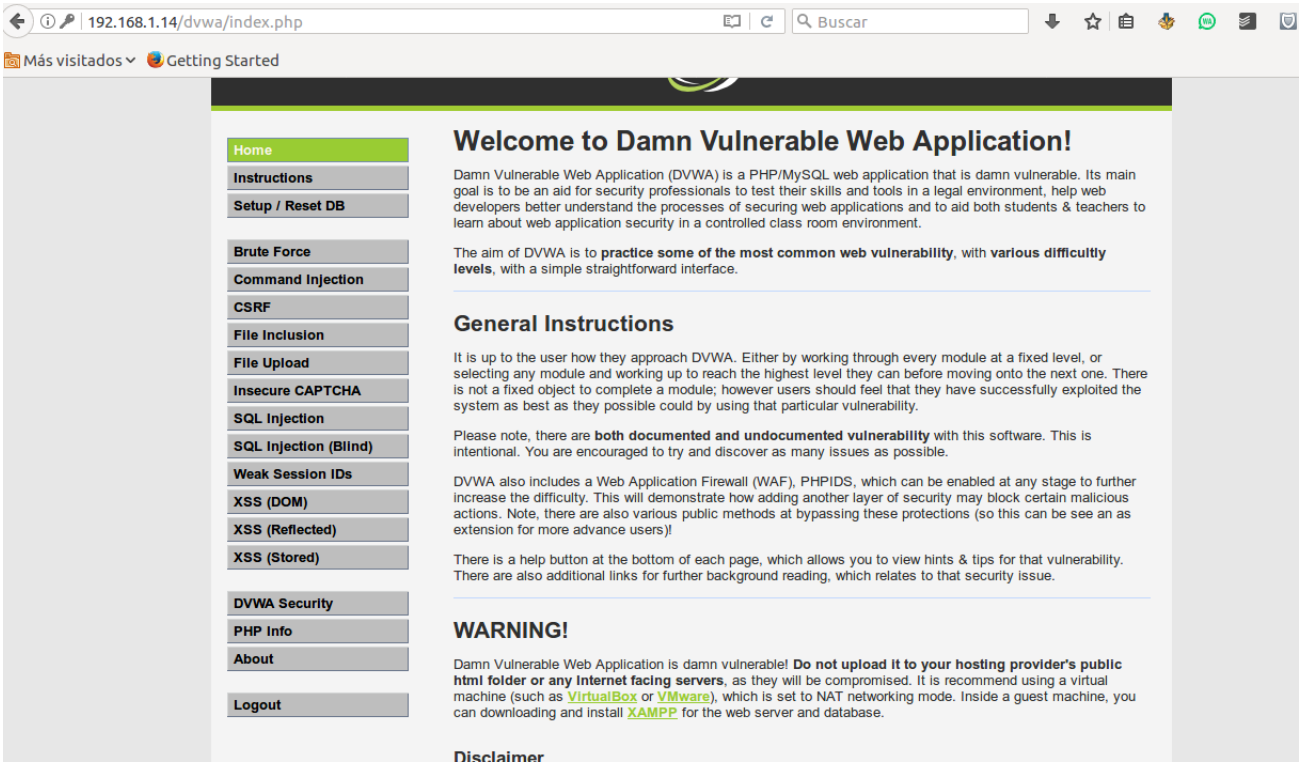

Figura 4.2: Inicio de DVWA

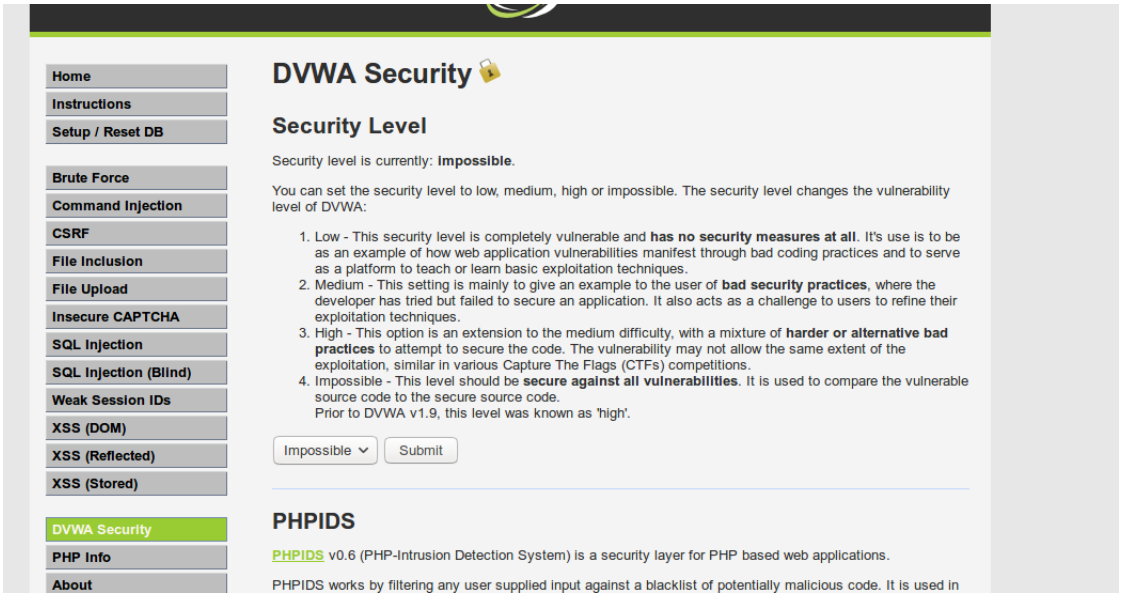

Figura 4.3: Niveles de seguridad

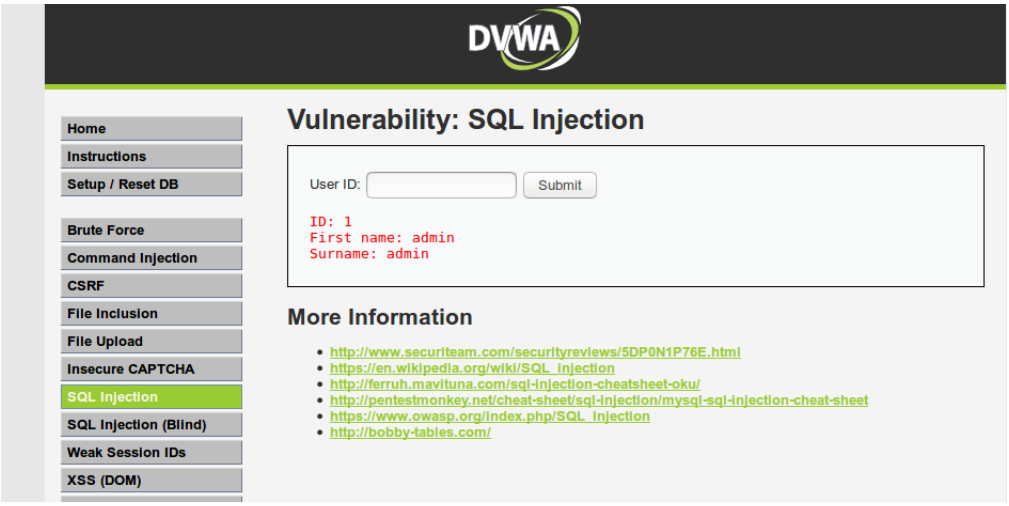

Figura 5.1: SQLi: ID=1

5. DVWA - SQLi

5.1. SQLi

Empecemos a realizar un ataque por inyección de comandos SQL a DVWA.

En este caso tenemos un formulario para escribir el ID de un usuario de la base de datos. Si escribimos 1 y aceptamos obtendremos lo siguiente:

Pero si escribimos una expresión estática correcta obtendremos todos los usuarios de la base de datos:

5.2. SQLi Blind

Probemos algo un poco más difícil. Un ataque SQL Blind.
Como podemos observar no nos da información de utilidad. Nuestro objetivo será ahora engañar al servidor para que de como true cualquier entrada.

Pensamos en una expresión estática como: 'OR 1=1 #. La probamos, obteniendo lo siguiente:
La ordén sería similar a lo siguiente

... WHERE User_ID =" OR 1=1 #;

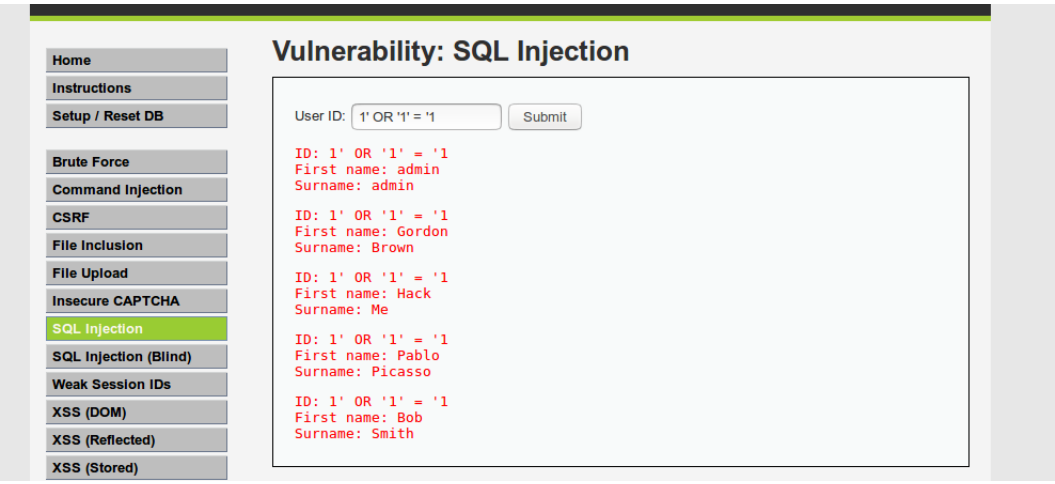

Figura 5.2: Ataque SQL

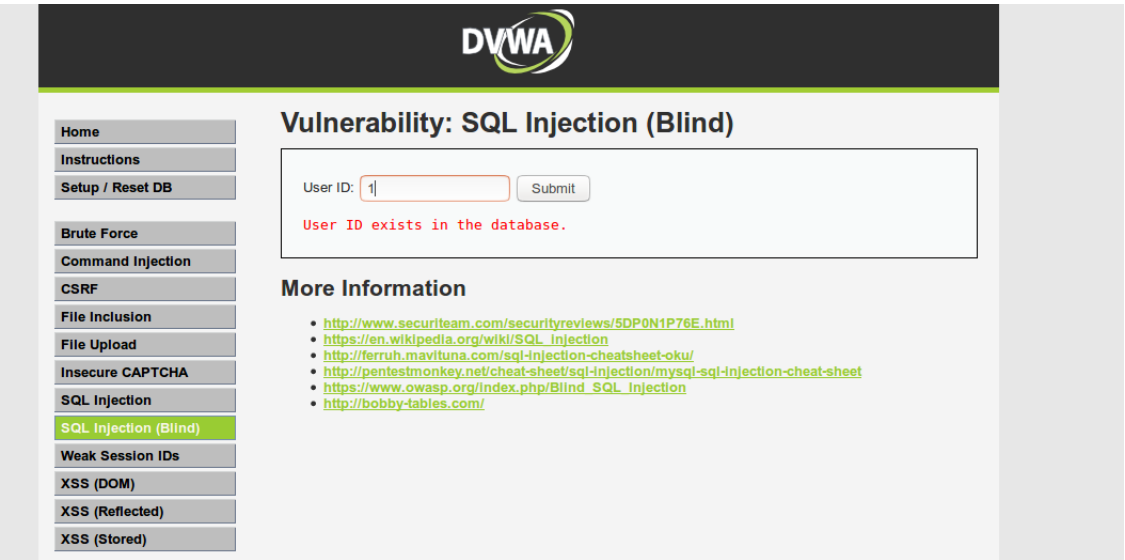

Figura 5.3: SQLi Blind - Funcionamiento

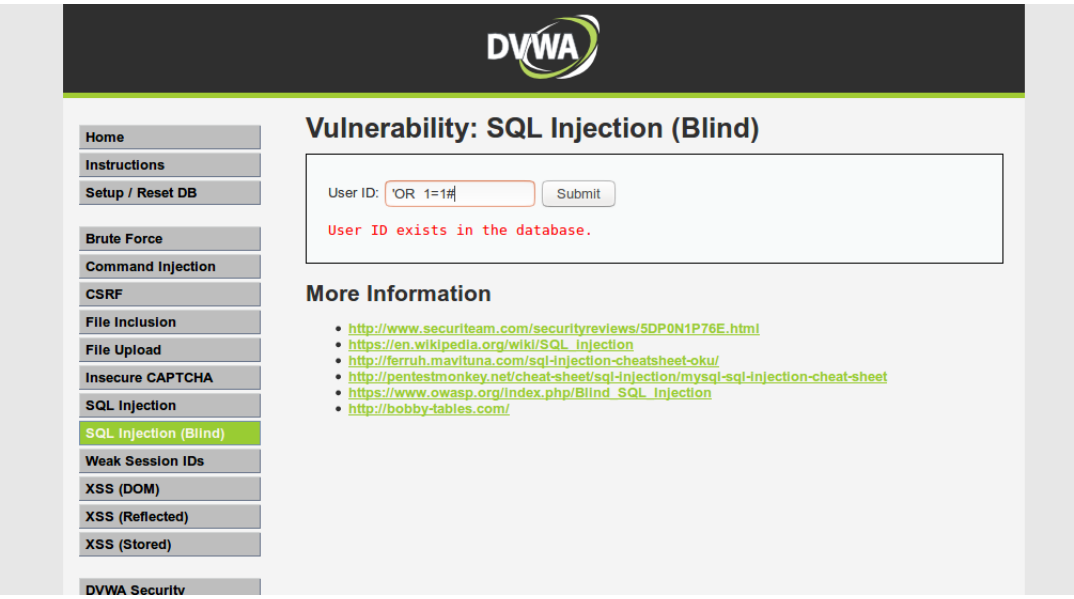

Figura 5.4: SQLi Blind inyección

6. SQLmap y BurpSuite

Llegados a este punto podemos introducir sqlmap y como automatizar los ataques de inyección SQL, para ello vamos a seguir los siguientes pasos mostrados en las capturas: Con las cookies obtenidas podemos empezar a usar sqlmap

Referencias

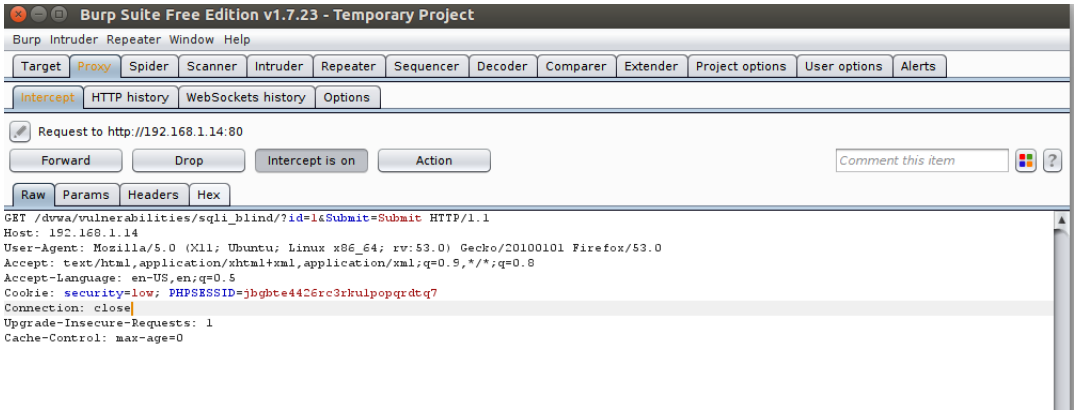

Figura 6.1: Usamos el software Burp-Suite para obtener las cookies necesarias para el ataque

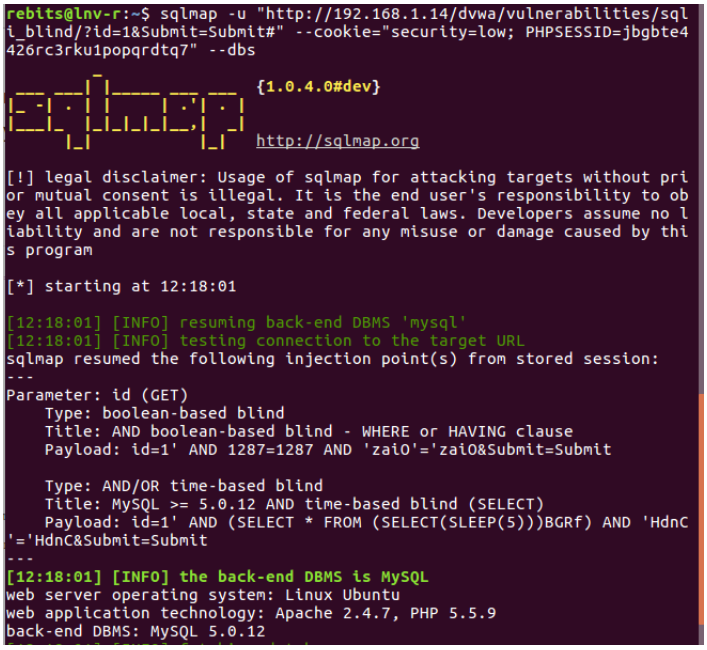

Figura 6.2: Usamos el parámetro –dbs para obtener la lista de bases de datos

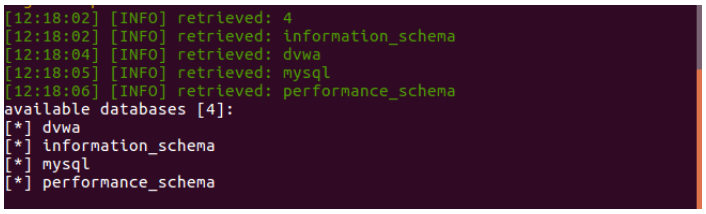

Figura 6.3: Bases de datos del sistema

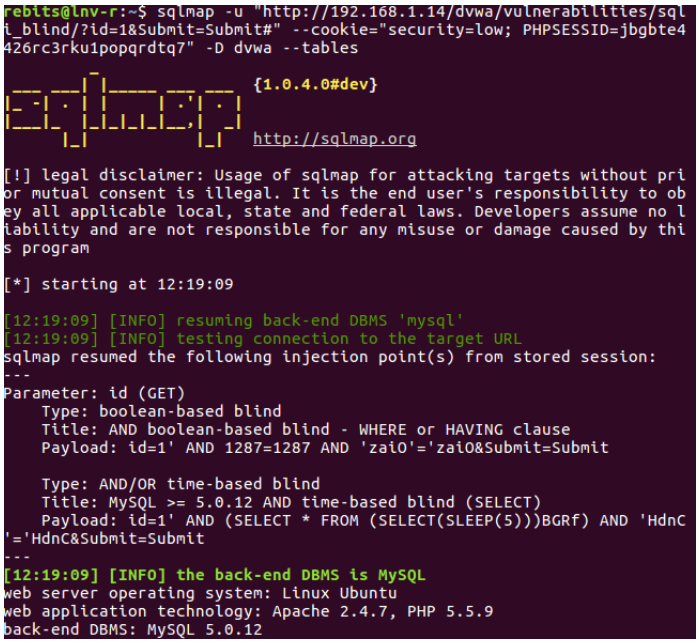

Figura 6.4: Usamos los parámetros -D dvwa –tables para mostrar las tablas de la base de datos dvwa

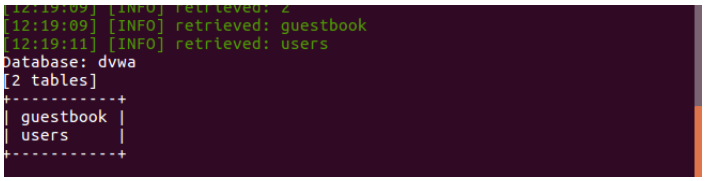

Figura 6.5: Tablas del la base de datos dvwa

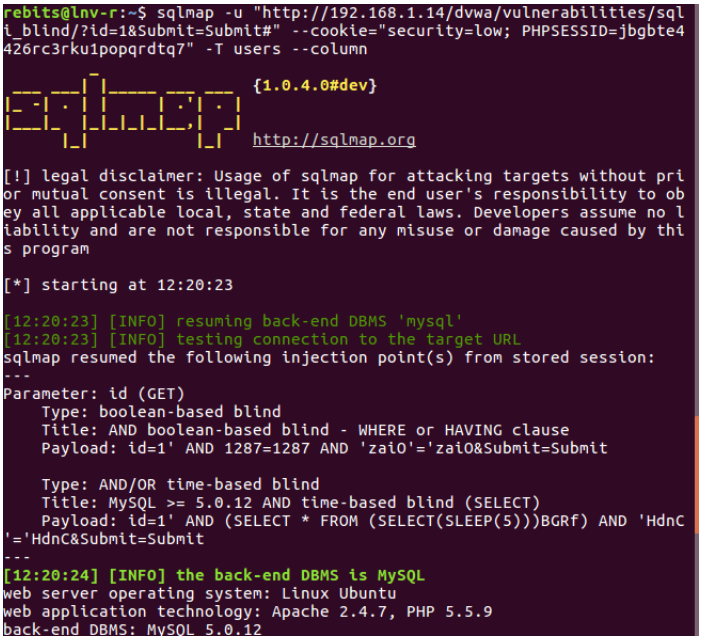

Figura 6.6: Usamos el parámetro -T users –column para obtener las columnas de la tablas users

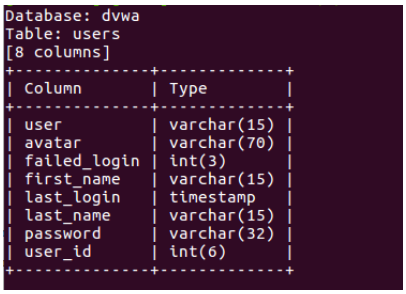

Figura 6.7: Columnas de la tabla users

Figura 6.8: Por último empleamos esta combinación de parámetros para obtener los usuarios y contraseñas

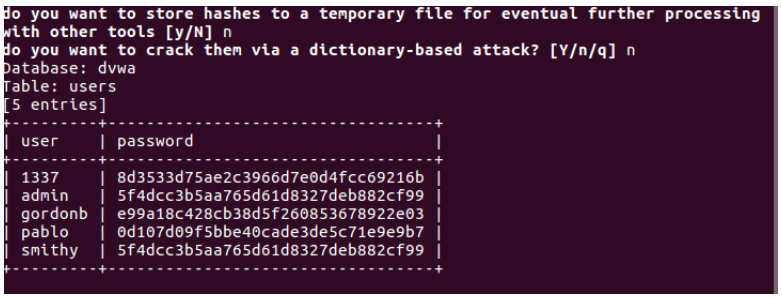

Figura 6.9: Usuarios y contraseñas