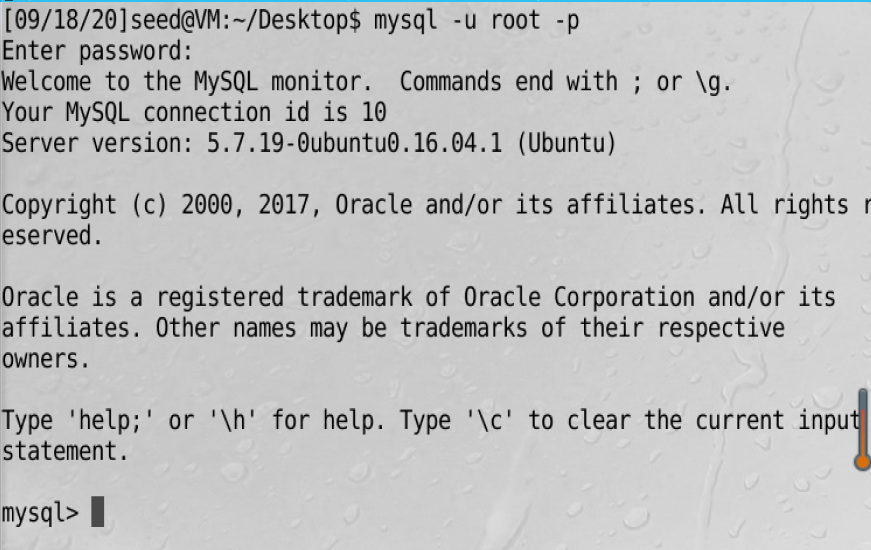
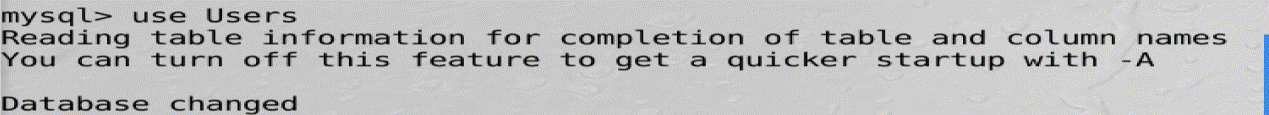
实验报告

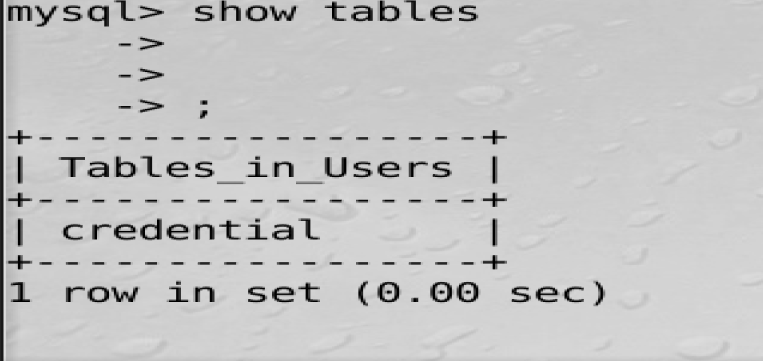
Task1

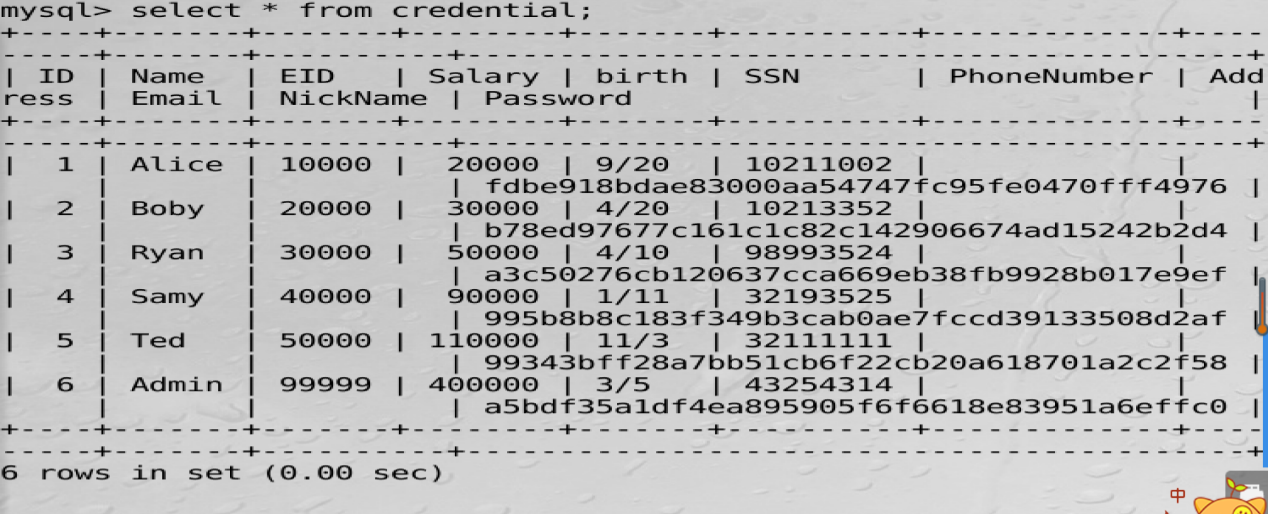


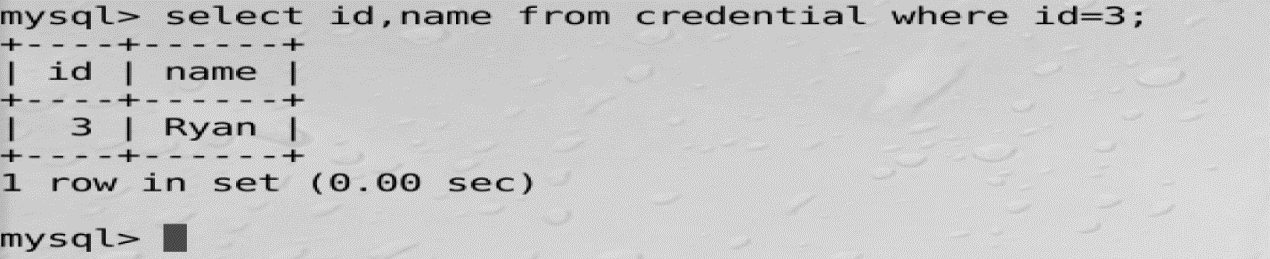
显示Users数据库



显示表格

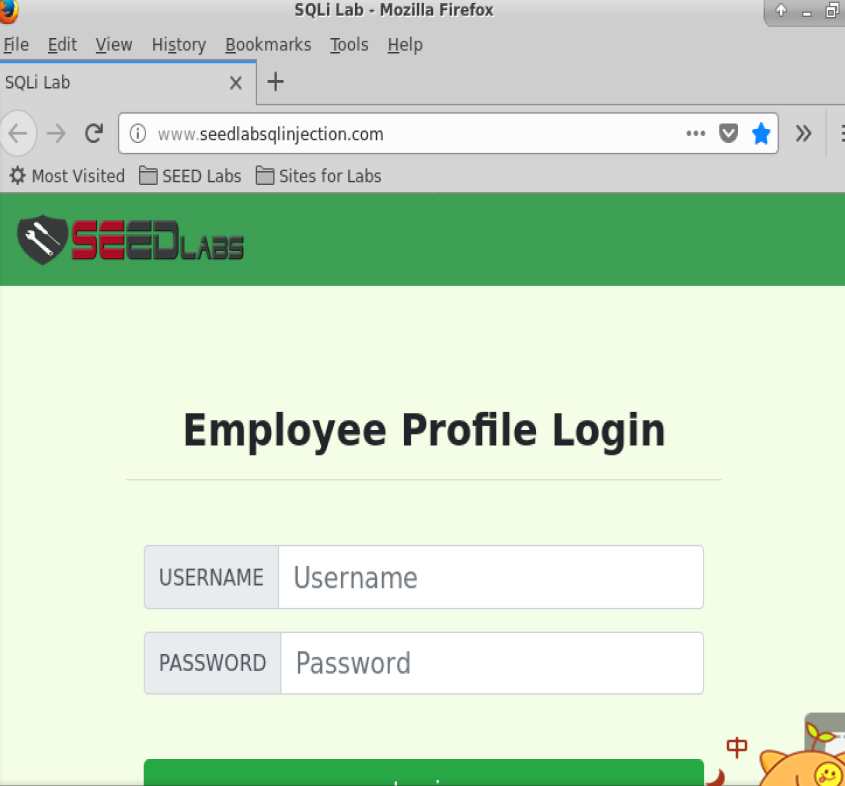


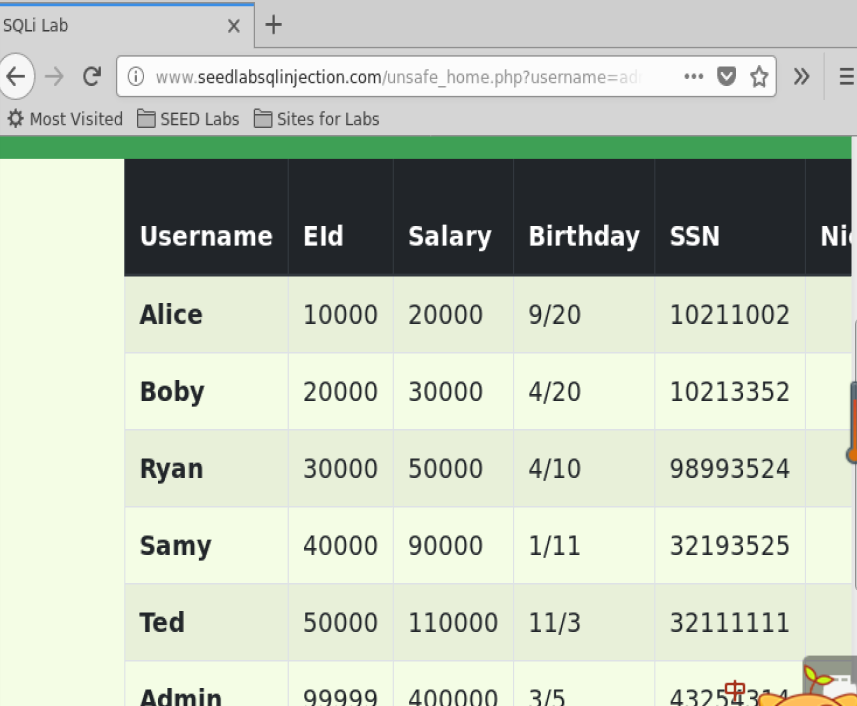


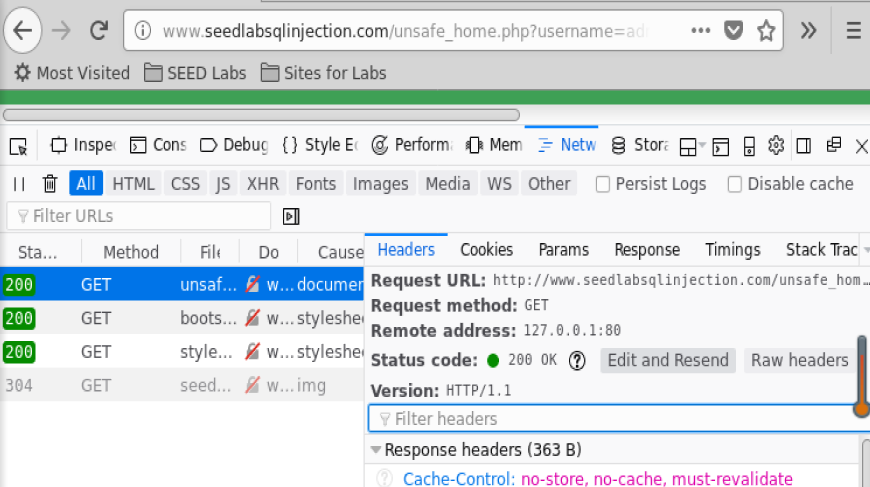


Task2

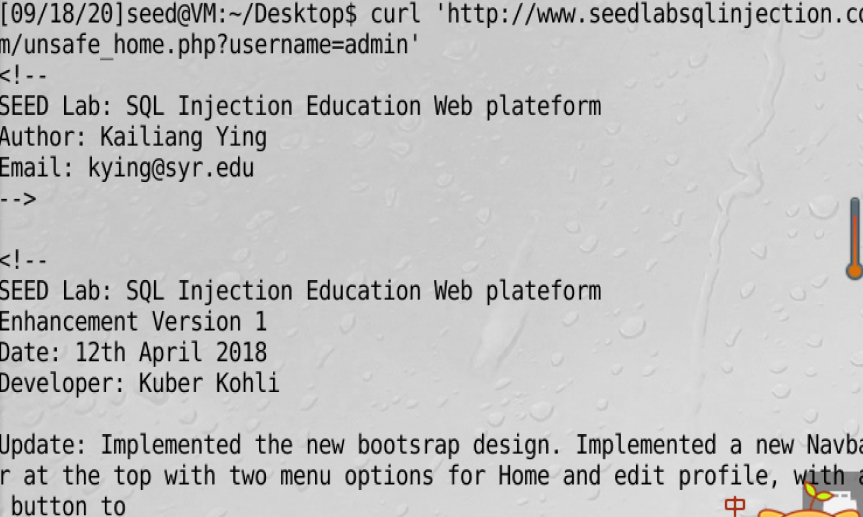
1. 登录目标网页，在不知道密码的情况下，在用户名区输入admin’#，因为’闭合前面的’，#把后面的都注释掉，所以不用验证密码

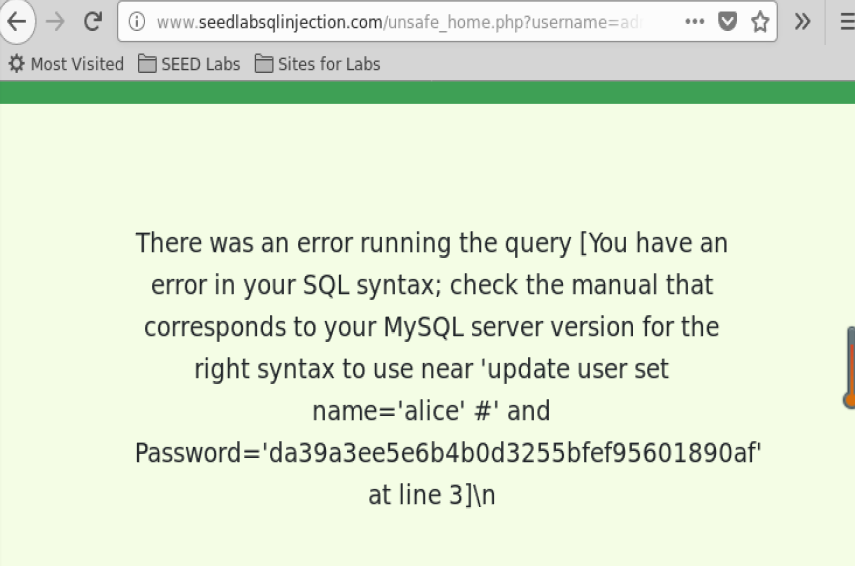




获取请求登录的报文

2.用curl注入sql



1. admin'; update user set name='test' #加入用户test，可以发现注入失败

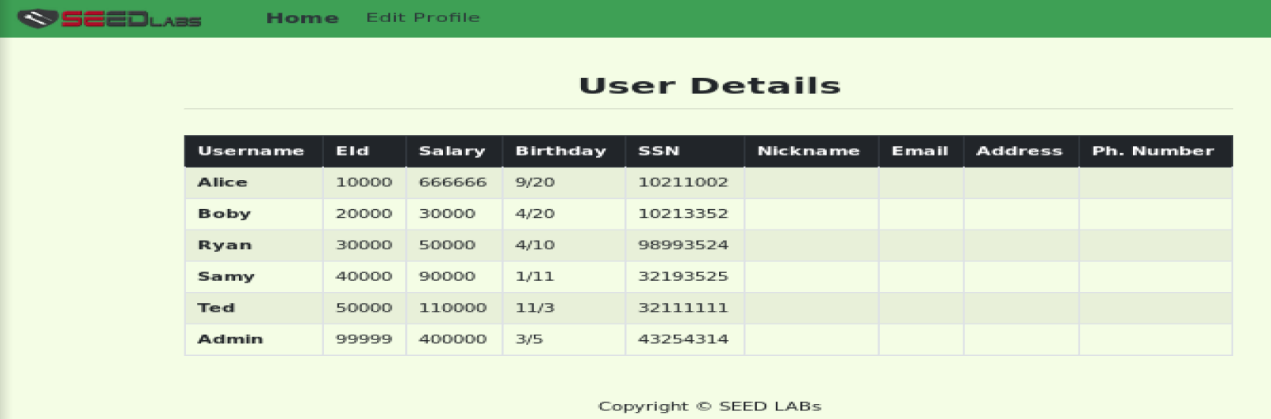
Task3

1. 编辑Alice的profile，在nickname一栏中输入', salary=666666 where EID=10000;#

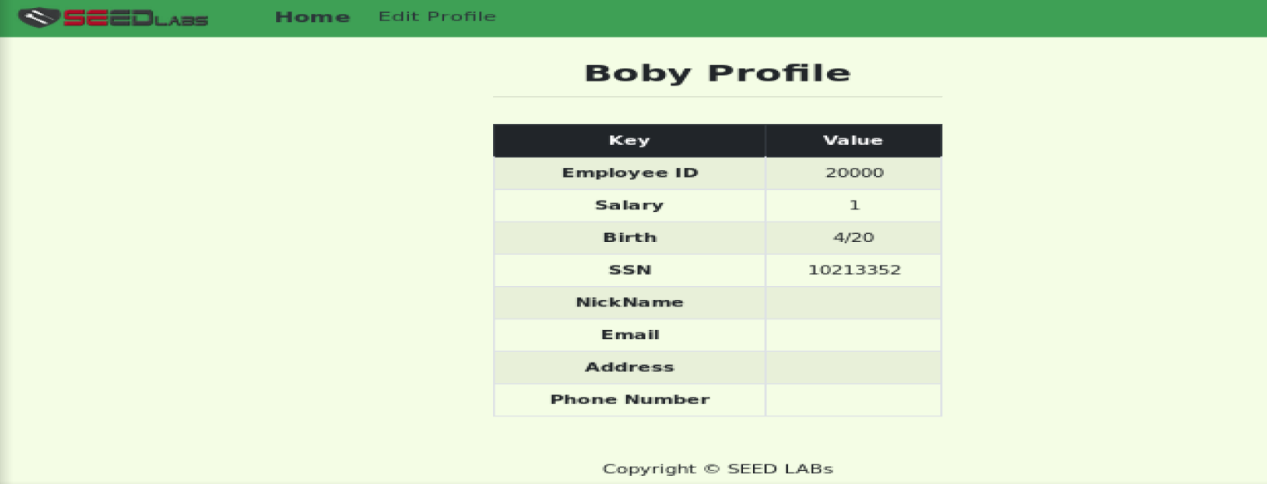
返回profile界面，可以看见工资已被修改为666666



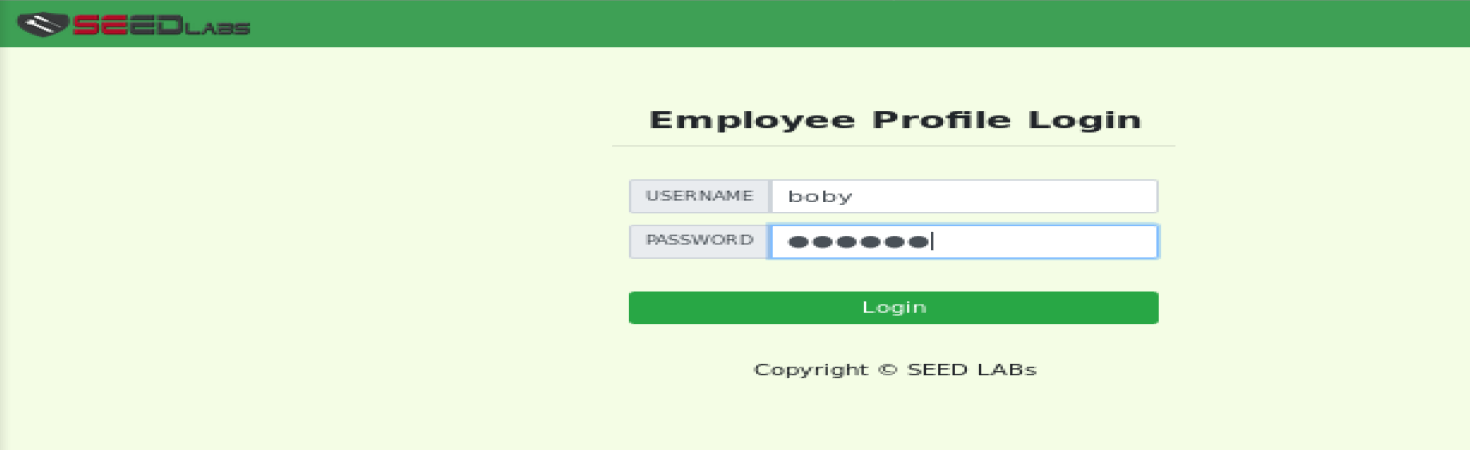
1. 首先登陆admin账号查看boby的EID=20000



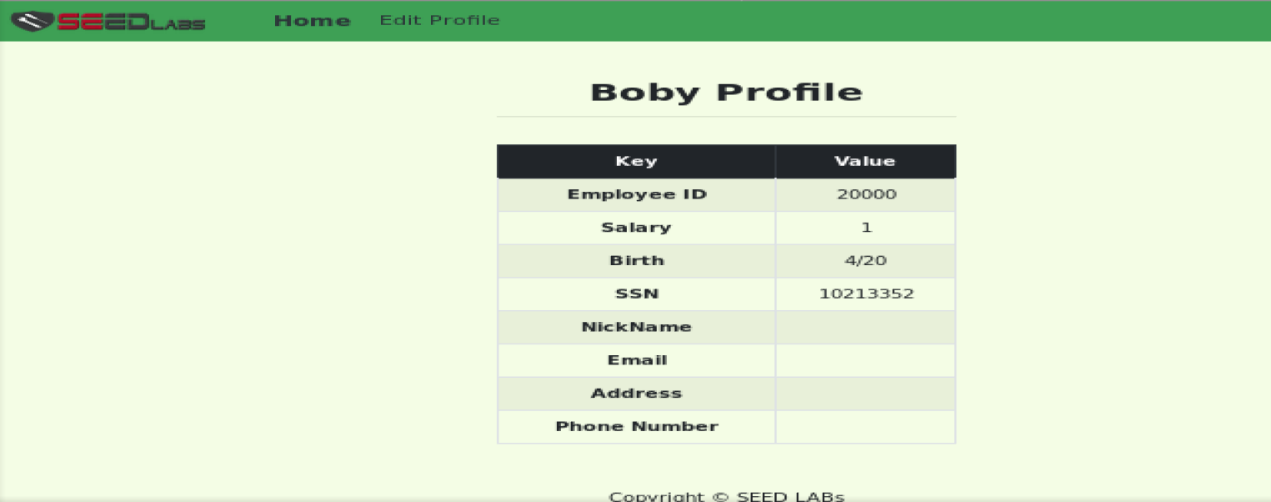
在alice的账号上编辑profile，在nickname一栏中输入’,salary=1 where EID=20000;#，再登录boby的账号，可以看见工资已被修改为1



1. 登录alice的账号，在编辑profile的界面里，在nickname那一栏中输入',Password=sha1(123456) where EID=20000;# 再在登录界面尝试登录boby的账号，密码一栏输入123456

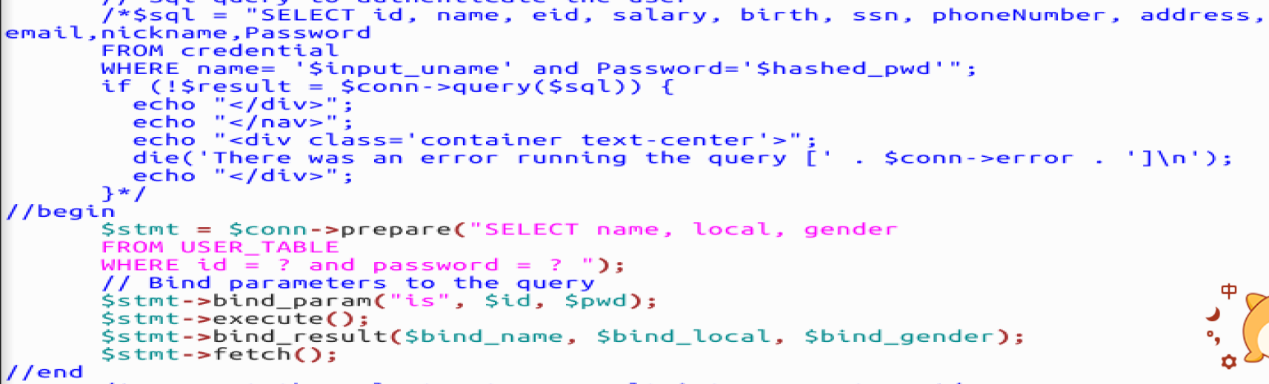


登录成功，密码修改成功

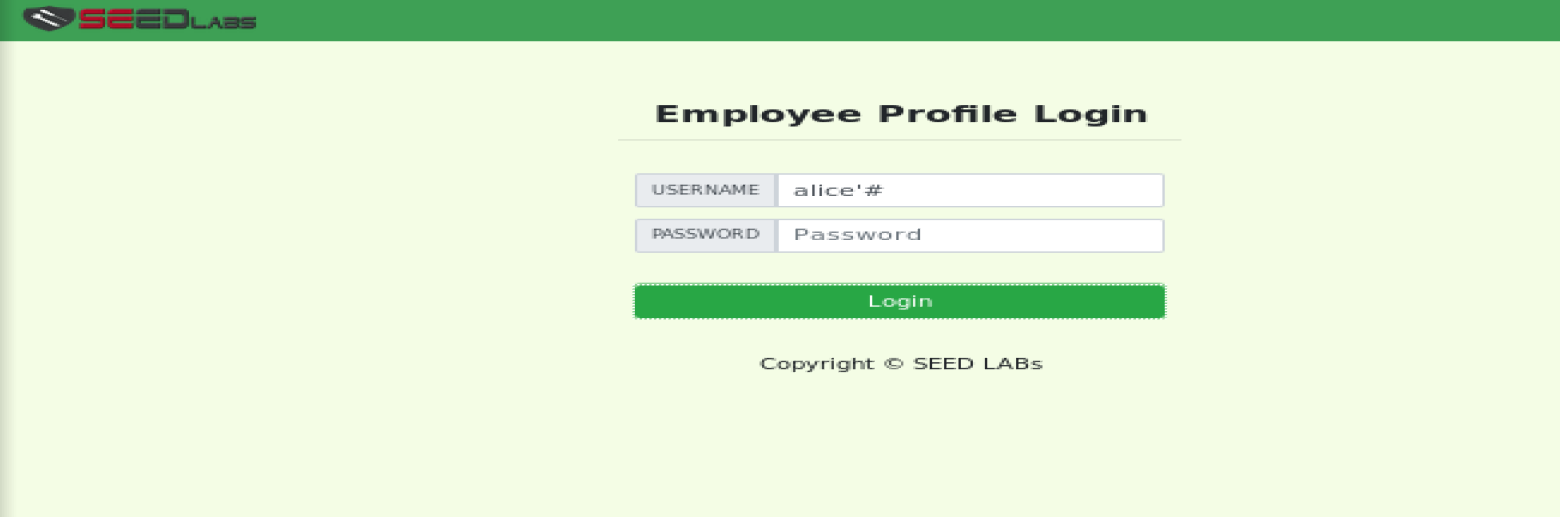


Task4

1. 修改/var/www/SQLinjection目录下的unsafe\_home.php文件，注释掉原来的不安全代码，插入新的安全代码



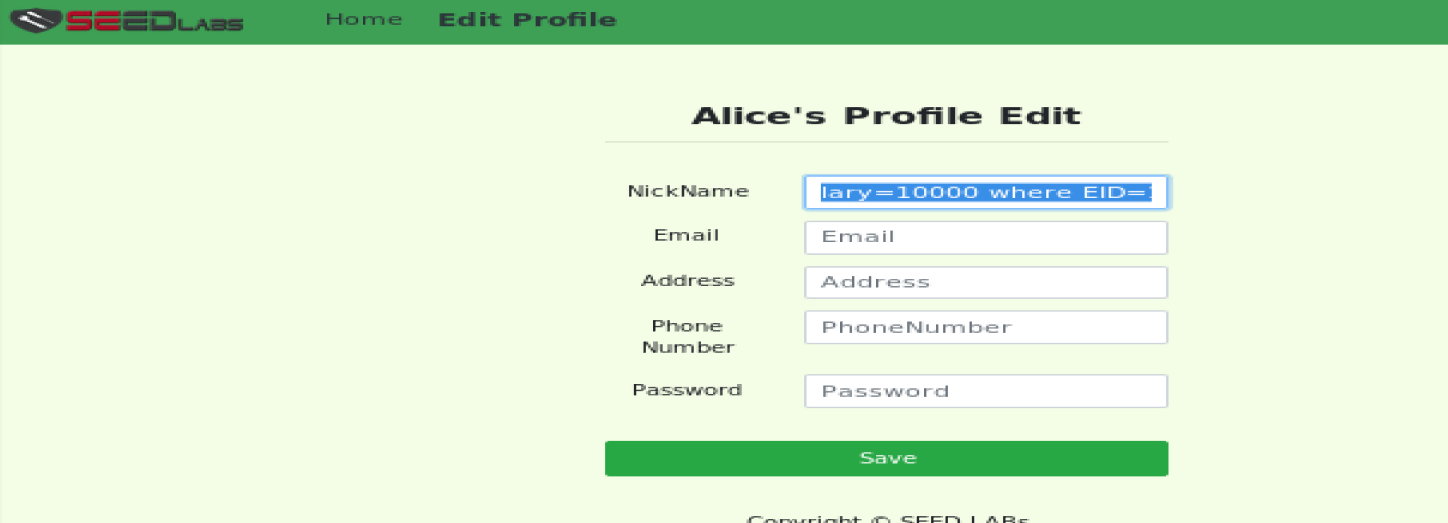
进入登录页面，尝试重新采用task2中的方式攻击

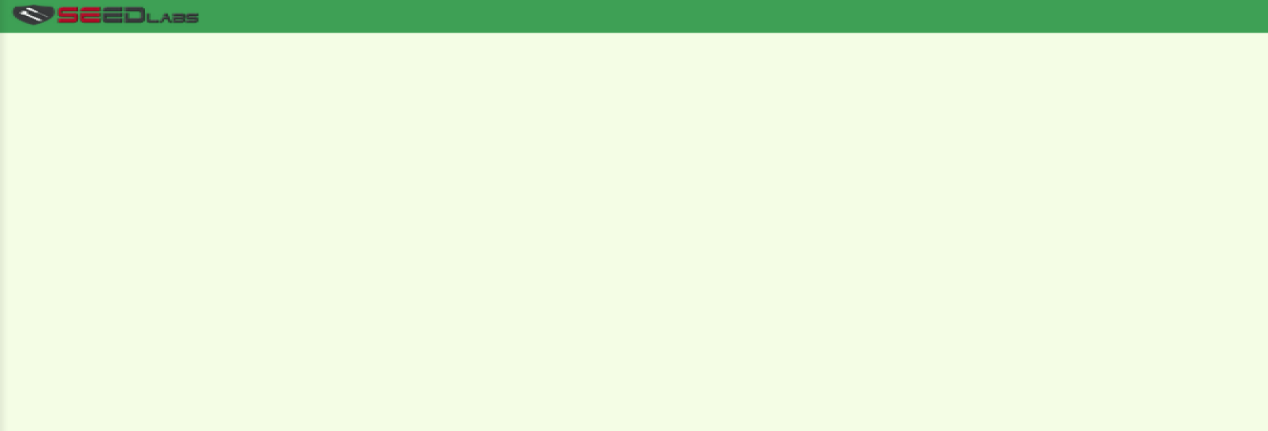


攻击失效



尝试task3中的攻击，再登录alice账号后再修改代码，然后编辑alice的profile，在nickname中输入', salary=10000 where EID=10000;# 尝试把alice的工资修改为10000





发现攻击失败，在重新将代码还会不安全代码后刷新页面，会发现修改成功，攻击成功