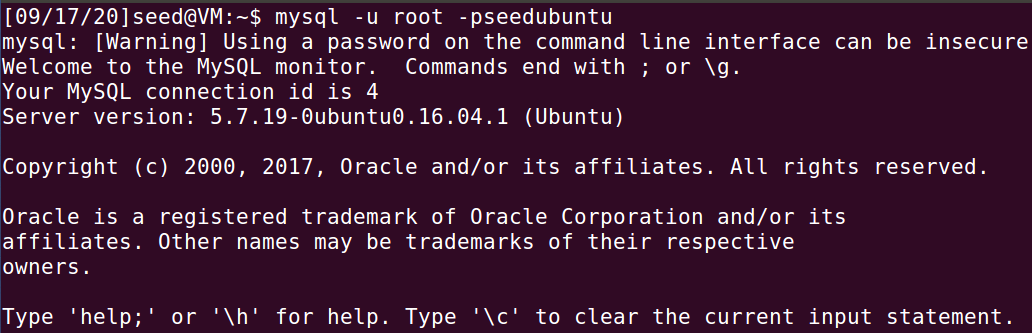
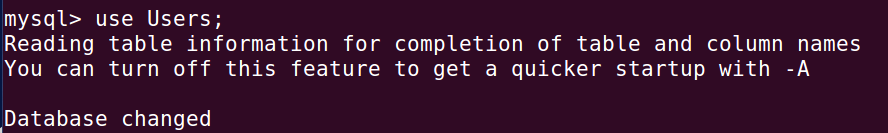
Task1：

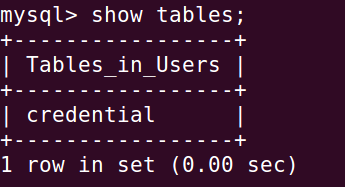
登入mysql模式



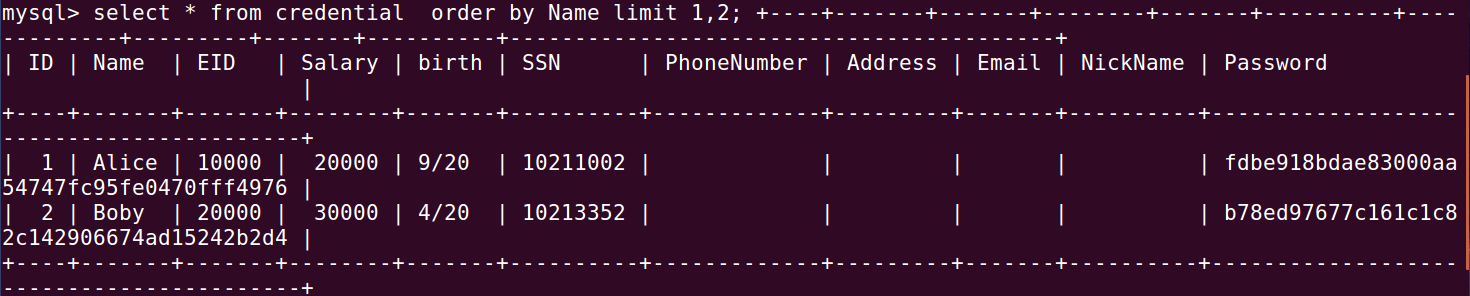
加载已经存在的数据

、

显示已有的数据表



用sql命令查询Alice数据如下：



Task2：

2.1

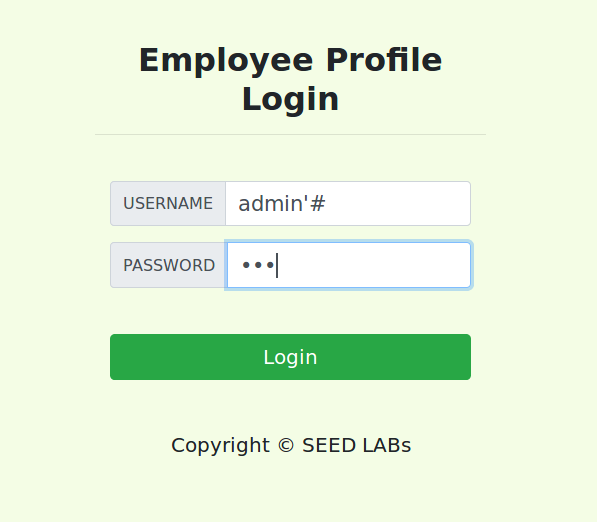
该网站用户名和密码信息是提交给unsafe\_home.php的，该文件中SQL查询语句即限制条件如下：



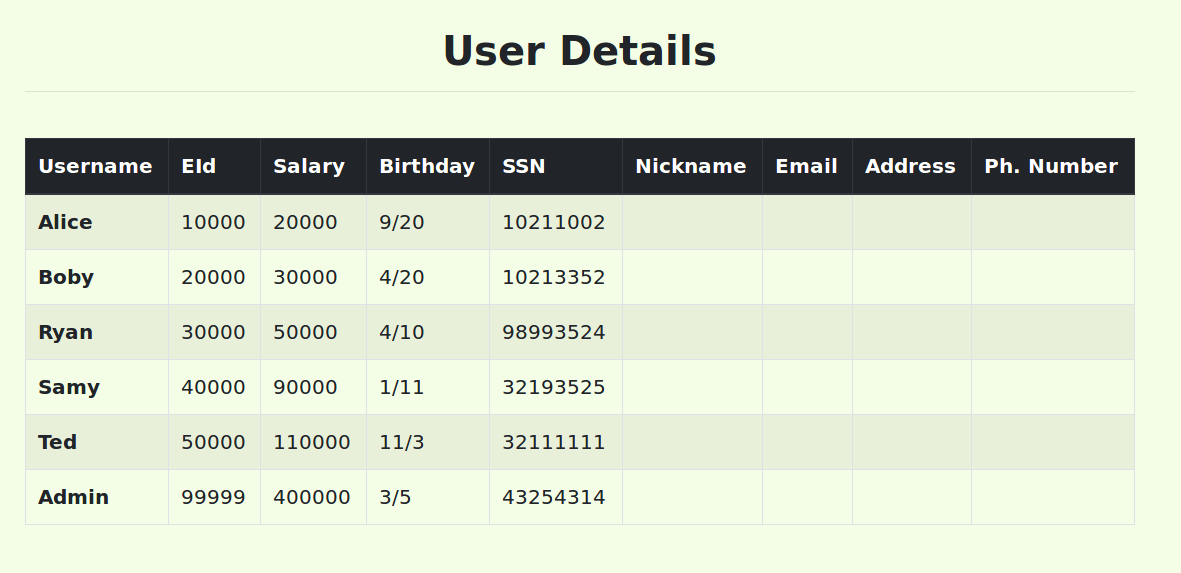
已知管理员id但未知密码的条件下可以在用户名处输入admin‘#使得限制条件改为

WHERE name= 'admin' #' and Password='$hashed\_pwd'

其中#作为注释符号将密码部分的限制条件注释掉且用户为admin导致攻击成功

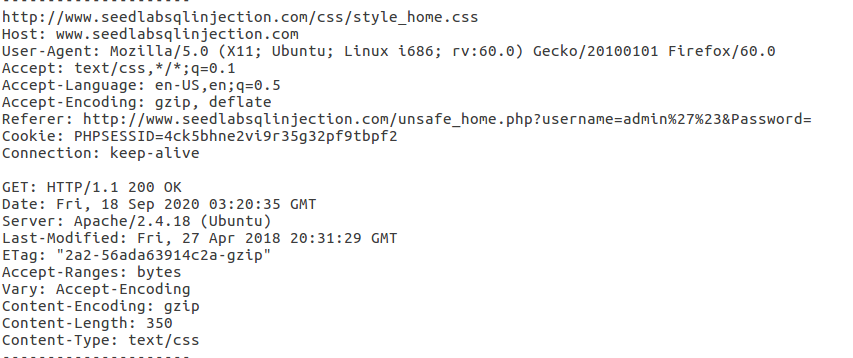


密码部分可随意输入或者不输入



进入数据界面表示攻击成功

2.2用HTTP Header Live捕获报文查看网页提交的表单转义情况



在terminal中用curl命令以报文中的格式键入用户名跟密码信息



用于查看SQL注入攻击网页提交的表单详细参数情况

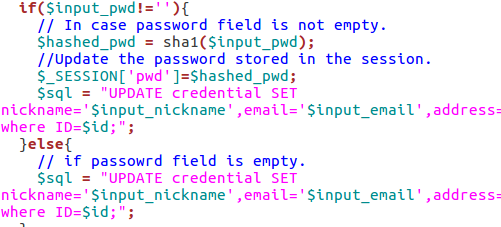


从中可以看到已经获取表中的所有信息，SQL注入成功

2.3

Task3：

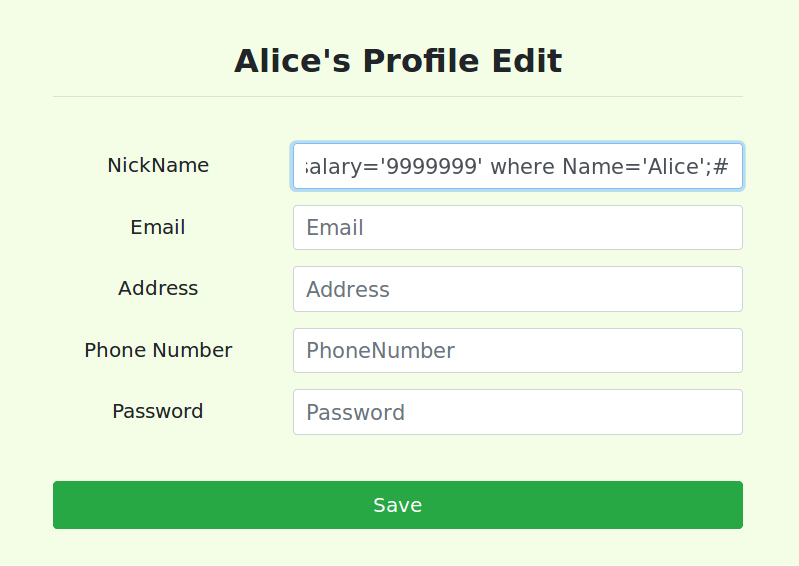
3.1 unsafe\_edit\_backend.php中Update指令部分代码如下图所示



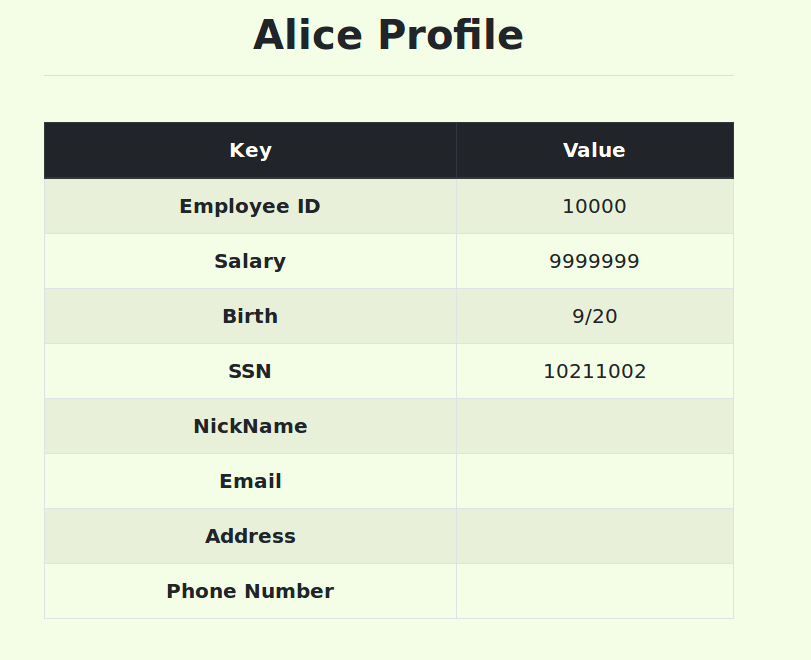
可以通过此部分进行sql注入攻击

首先登陆Alice账号进入编辑页面，在nickname字段中填入如下代码

',salary='999999' where Name='Alice'; #



保存后发现工资已经被修改

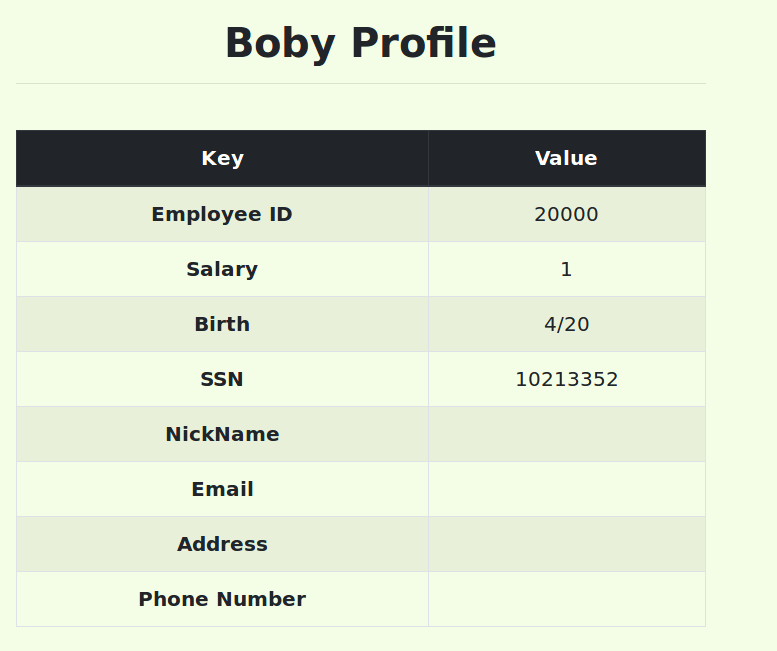


3.2

将nickname字段修改为', salary=1 where Name='Boby'#



保存后登入boby页面发现工资已被修改

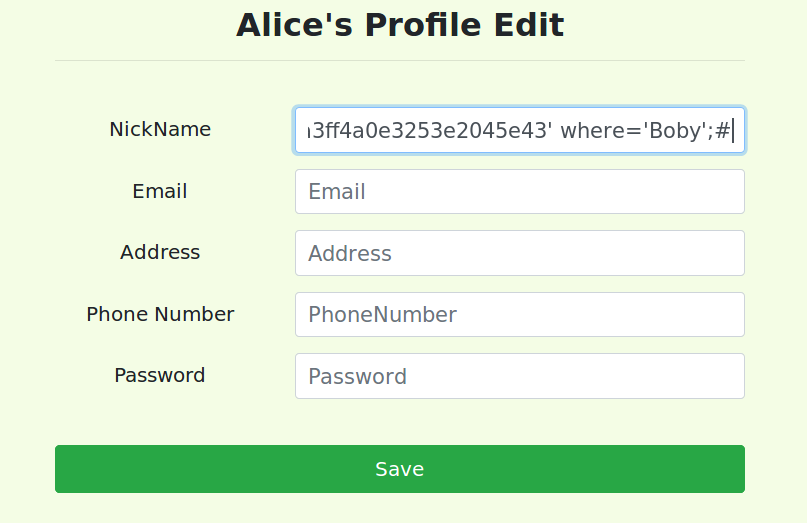


3.3

用命令行echo -n 9123 | openssl sha1将所设置的密码转换为SH1密码值



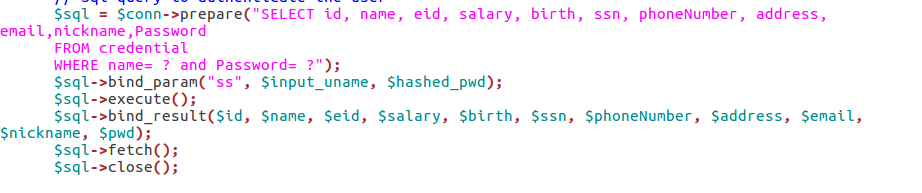
在alice的修改界面的nickname中输入代码



保存后发现使用0000可以登陆到boby账号

Task4

修改不安全的代码



再次尝试用task2中方法进行sql注入失败，证明防御开启成功

