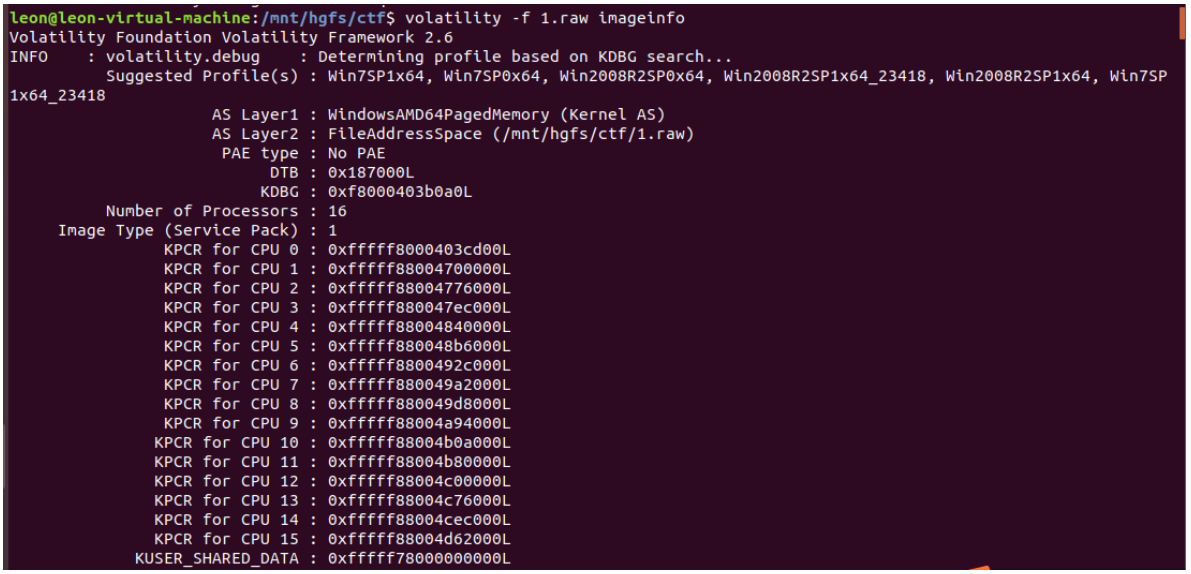
MISC

- Akira之瞳-1

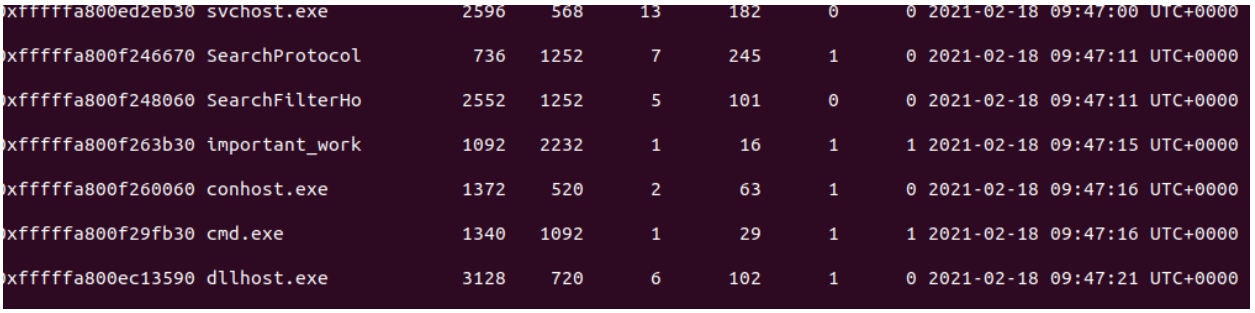
  打开附件得到raw文件，可知考点为内存取证。提取镜像文件



  查看进程

```
volatility -f 1.raw --profile=Win7SP1x64 pslist
```

在进程中看到题目相关的important work进程



尝试提取进程

```
volatility -f 1.raw --profile=Win7SP1x64  memdump -p 1092 -D ./
```

经历一系列的查看，筛选后，发现有work.zip，有一个压缩包。

尝试用foremost分离，得到一个zip文件。

里面是两张加密的png图片，旁边注释密码是login_password。

于是想到提取用户密码。

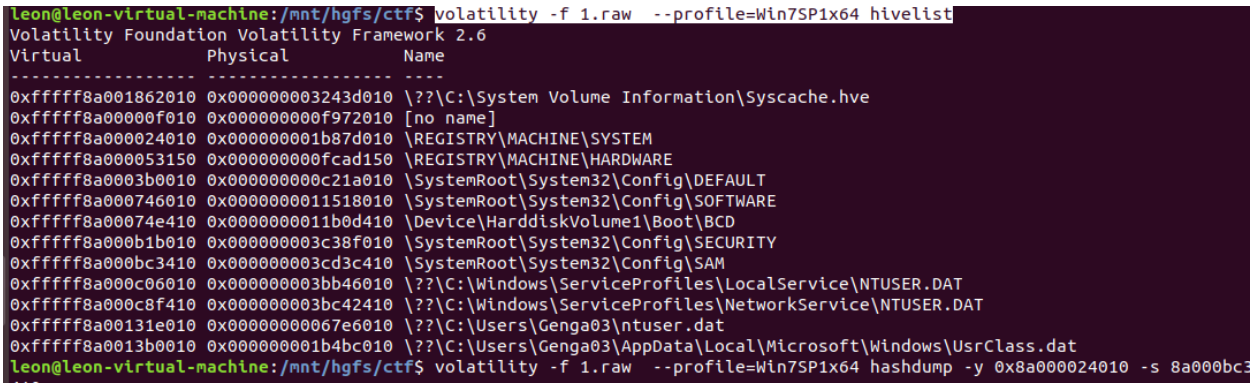
获取最后登录系统的用户

```
volatility -f 1.raw  --profile=Win7SP1x64 printkey -K "SAM\Domains\Account\Us
volatility -f 1.raw  --profile=Win7SP1x64 printkey -K "SOFTWARE\Microsoft\Wir
```

要从内存中获得密码哈希，先要获取到注册表中的system 的 virtual 地址，SAM 的 virtual 地址（百度的）。

查看注册表

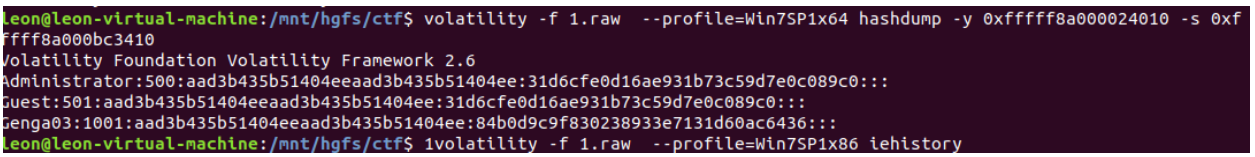
```
volatility -f 1.raw  --profile=Win7SP1x64 hivelist
```



使用命令

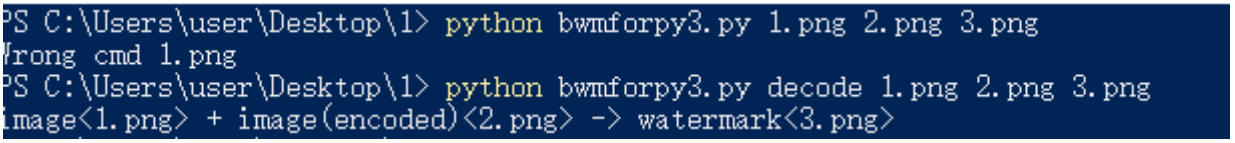
```
volatility -f 1.raw  --profile=Win7SP1x64 hashdump -y 0xfffff8a000024010 -s 0
```

得到密码



将密码解密后再sha256加密得到png的密码。

根据blind猜出是盲水印，提取盲水印就行了。

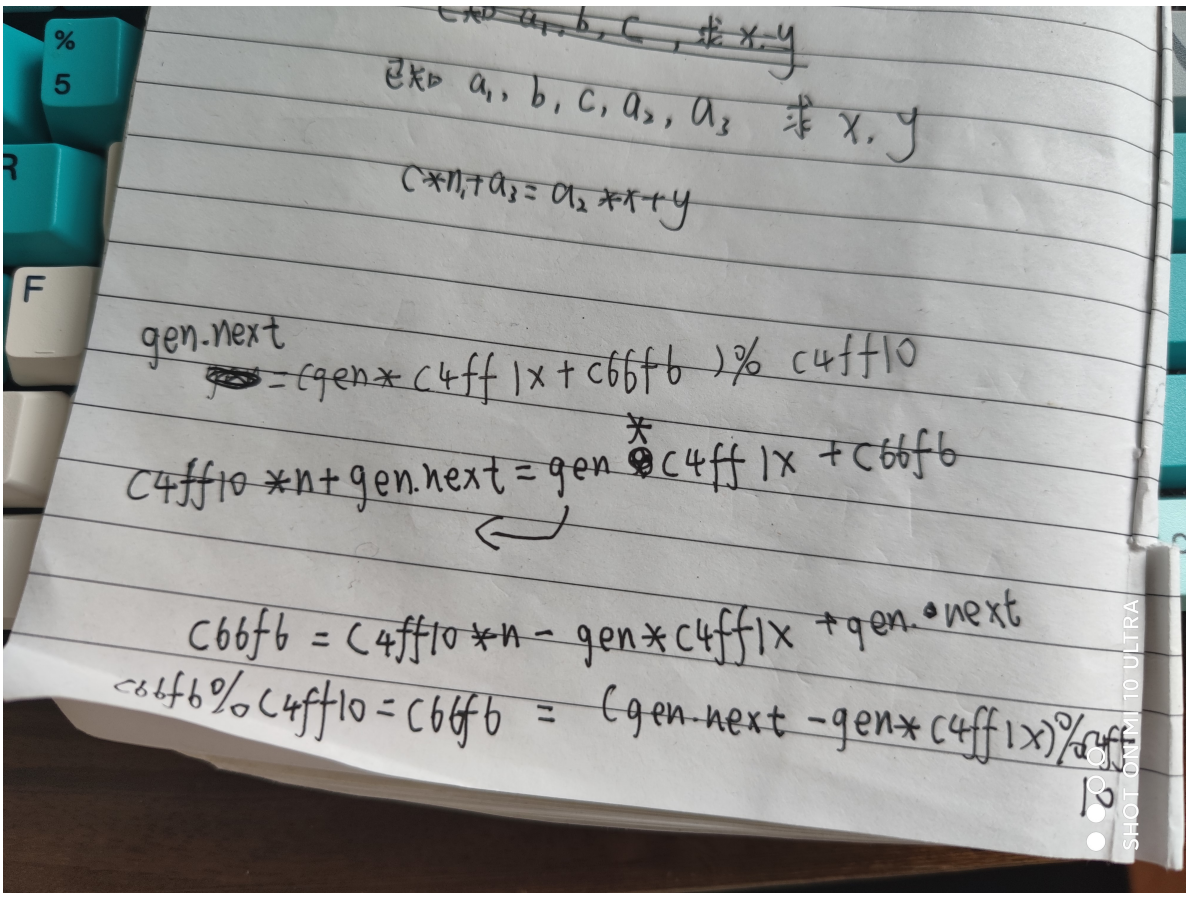


得到的图片能隐约看出flag。

flag：hgame{7he_f1ame_brin9s_me_end1ess_9rief}

crypto

- 夺宝大冒险1
  阅读任务文件，并连接nc，发现要得到flag，必须要完成三个test。首先test1比较简单，列在纸上计算一下就出来了。一下为计算过程：



  然后做到test2时发现问题了，能列出方程但是接不出来。后面经过不断求解后放弃了。然后百度了关键语句也就是next。发现了该题目的考点是LCG攻击。百度LCG攻击，得到了三种情况下的py攻击代码（对应的三个test）。然后考虑到题目的123都是随机的并且只在最后验证，所以考虑在exp中加入重复尝试解题。
  以下为exp:

```
from pwn import *
import libnum
def gcd(a,b):
    while a!=0:
        a,b = b%a,a
    return b
def findModReverse(a,m):
    if gcd(a,m)!=1:
        return None
    u1,u2,u3 = 1,0,a
```

```
    v1,v2,v3 = 0,1,m
    while v3!=0:
        q = u3//v3
        v1,v2,v3,u1,u2,u3 = (u1-q*v1),(u2-q*v2),(u3-q*v3),v1,v2,v3
    return u1%m
time=0
while True:
    try:
        p=remote('182.92.108.71',30641)
        s=p.recvline()
        m=int(s[s.find('(')+1:s.find(',')])
        n=int(s[s.find(' ')+1:s.find(')')])
        s0=int(p.recvline().strip('\n'))
        s1=int(p.recvline().strip('\n'))
        c=(s1-s0*m)%n
        p.sendline(str(c))
        n=int(p.recvline().strip('\n'))
        s0=int(p.recvline().strip('\n'))
        s1=int(p.recvline().strip('\n'))
        s2=int(p.recvline().strip('\n'))
        m=((s2-s1)*findModReverse((s1-s0),n))%n
        c=(s1-s0*m)%n
        p.sendline(str(m))
        p.sendline(str(c))
        list_s=[]
        for i in range(7):
            s=int(p.recvline().strip('\n'))
            list_s.append(s)
        t0=list_s[1]-list_s[0]
        t1=list_s[2]-list_s[1]
        t2=list_s[3]-list_s[2]
        t3=list_s[4]-list_s[3]
        t4=list_s[5]-list_s[4]
        t5=list_s[6]-list_s[5]
        x1=t2*t0-t1*t1
        x2=t3*t1-t2*t2
        x3=t4*t2-t3*t3
        x4=t5*t3-t4*t4
        x=[x1,x2,x3,x4]
        n=abs(reduce(libnum.gcd,x))
        p.sendline(str(n))
        if p.recvline()=='fail\n':
            time+=1
```

```python
                p.close()
            else:
                break
    except:
        time+=1
        p.close()
        continue
print("time:",time)
print("flag is:",p.recvline())
```

```
*] Closed connection to 182.92.108.71 port 30641
+] Opening connection to 182.92.108.71 on port 30641: Done
*] Closed connection to 182.92.108.71 port 30641
+] Opening connection to 182.92.108.71 on port 30641: Done
*] Closed connection to 182.92.108.71 port 30641
+] Opening connection to 182.92.108.71 on port 30641: Done
*] Closed connection to 182.92.108.71 port 30641
+] Opening connection to 182.92.108.71 on port 30641: Done
*] Closed connection to 182.92.108.71 port 30641
+] Opening connection to 182.92.108.71 on port 30641: Done
*] Closed connection to 182.92.108.71 port 30641
+] Opening connection to 182.92.108.71 on port 30641: Done
*] Closed connection to 182.92.108.71 port 30641
+] Opening connection to 182.92.108.71 on port 30641: Done
*] Closed connection to 182.92.108.71 port 30641
+] Opening connection to 182.92.108.71 on port 30641: Done
*] Closed connection to 182.92.108.71 port 30641
+] Opening connection to 182.92.108.71 on port 30641: Done
*] Closed connection to 182.92.108.71 port 30641
+] Opening connection to 182.92.108.71 on port 30641: Done
'time:', 331)
'flag is:', 'hgame{Cracking^prng_Linear)Congruential&Generators}\n')
*] Closed connection to 182.92.108.71 port 30641
eon@leon-virtual-machine:/mnt/hgfs/ctf$
```

得到flag。