

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

```
leon@leon-virtual-machine:/mnt/hgfs/ctf$ volatility -f 1.raw imageinfo
Volatility Foundation Volatility Framework 2.6
INFO : volatility.debug : Determining profile based on KDBG search...
      Suggested Profile(s) : Win7SP1x64, Win7SP0x64, Win2008R2SP0x64, Win2008R2SP1x64_23418, Win2008R2SP1x64, Win7SP
1x64_23418
      AS Layer1 : WindowsAMD64PagedMemory (Kernel AS)
      AS Layer2 : FileAddressSpace (/mnt/hgfs/ctf/1.raw)
      PAE type : No PAE
      DTB : 0x187000L
      KDBG : 0xf8000403b0a0L
      Number of Processors : 16
      Image Type (Service Pack) : 1
      KPCR for CPU 0 : 0xffffffff8000403cd00L
      KPCR for CPU 1 : 0xffffffff80004700000L
      KPCR for CPU 2 : 0xffffffff80004776000L
      KPCR for CPU 3 : 0xffffffff800047ec000L
      KPCR for CPU 4 : 0xffffffff80004840000L
      KPCR for CPU 5 : 0xffffffff800048b6000L
      KPCR for CPU 6 : 0xffffffff8000492c000L
      KPCR for CPU 7 : 0xffffffff800049a2000L
      KPCR for CPU 8 : 0xffffffff800049d8000L
      KPCR for CPU 9 : 0xffffffff80004a94000L
      KPCR for CPU 10 : 0xffffffff80004b0a000L
      KPCR for CPU 11 : 0xffffffff80004b80000L
      KPCR for CPU 12 : 0xffffffff80004c00000L
      KPCR for CPU 13 : 0xffffffff80004c76000L
      KPCR for CPU 14 : 0xffffffff80004cec000L
      KPCR for CPU 15 : 0xffffffff80004d62000L
      KUSER_SHARED_DATA : 0xffffffff7800000000L
```

查看进程

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

在进程中看到题目相关的important work进程

xxxxffa800ed2eb30	svchost.exe	2596	568	13	182	0	0	2021-02-18 09:47:00 UTC+0000
xxxxffa800f246670	SearchProtocol	736	1252	7	245	1	0	2021-02-18 09:47:11 UTC+0000
xxxxffa800f248060	SearchFilterHo	2552	1252	5	101	0	0	2021-02-18 09:47:11 UTC+0000
xxxxffa800f263b30	important_work	1092	2232	1	16	1	1	2021-02-18 09:47:15 UTC+0000
xxxxffa800f260060	conhost.exe	1372	520	2	63	1	0	2021-02-18 09:47:16 UTC+0000
xxxxffa800f29fb30	cmd.exe	1340	1092	1	29	1	1	2021-02-18 09:47:16 UTC+0000
xxxxffa800ec13590	dllhost.exe	3128	720	6	102	1	0	2021-02-18 09:47:21 UTC+0000

尝试提取进程

```
1 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

```
1 volatility -f 1.raw --profile=Win7SP1x64 printkey -K "SAM\Domains\Account\Us
2 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

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

```
Leon@Leon-virtual-machine:/mnt/hgfs/ctf$ volatility -f 1.raw --profile=Win7SP1x64 hivelist
Volatility Foundation Volatility Framework 2.6
Virtual      Physical      Name
-----
0xfffff8a001862010 0x000000003243d010 \??\C:\System Volume Information\Syscache.hve
0xfffff8a00000f010 0x00000000f972010 [no name]
0xfffff8a000024010 0x000000001b87d010 \REGISTRY\MACHINE\SYSTEM
0xfffff8a000053150 0x00000000fcad150 \REGISTRY\MACHINE\HARDWARE
0xfffff8a00003b0010 0x000000000c21a010 \SystemRoot\System32\Config\DEFAULT
0xfffff8a000746010 0x0000000011518010 \SystemRoot\System32\Config\SOFTWARE
0xfffff8a00074e410 0x0000000011b0d410 \Device\HarddiskVolume1\Boot\BCD
0xfffff8a0000b1b010 0x000000003c38f010 \SystemRoot\System32\Config\SECURITY
0xfffff8a0000bc3410 0x000000003cd3c410 \SystemRoot\System32\Config\SAM
0xfffff8a000c06010 0x000000003bb46010 \??\C:\Windows\ServiceProfiles\LocalService\NTUSER.DAT
0xfffff8a0000c8f410 0x000000003bc42410 \??\C:\Windows\ServiceProfiles\NetworkService\NTUSER.DAT
0xfffff8a00131e010 0x00000000067e6010 \??\C:\Users\Genga03\ntuser.dat
0xfffff8a0013b0010 0x000000001b4bc010 \??\C:\Users\Genga03\AppData\Local\Microsoft\Windows\UsrClass.dat
Leon@Leon-virtual-machine:/mnt/hgfs/ctf$ volatility -f 1.raw --profile=Win7SP1x64 hashdump -y 0xfffff8a000024010 -s 8a000bc3
```

使用命令

```
1 volatility -f 1.raw --profile=Win7SP1x64 hashdump -y 0xfffff8a000024010 -s 8a000bc3
```

得到密码

```
Leon@Leon-virtual-machine:/mnt/hgfs/ctf$ volatility -f 1.raw --profile=Win7SP1x64 hashdump -y 0xfffff8a000024010 -s 0xfffff8a0000bc3410
Volatility Foundation Volatility Framework 2.6
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Genga03:1001:aad3b435b51404eeaad3b435b51404ee:84b0d9c9f830238933e7131d60ac6436:::
Leon@Leon-virtual-machine:/mnt/hgfs/ctf$ 1volatility -f 1.raw --profile=Win7SP1x86 iehistory
```

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。

```
PS C:\Users\user\Desktop\1> python bwmforpy3.py 1.png 2.png 3.png
Wrong cmd 1.png
PS C:\Users\user\Desktop\1> python bwmforpy3.py decode 1.png 2.png 3.png
image<1.png> + image(encoded)<2.png> -> watermark<3.png>
```

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：

Handwritten notes showing the derivation of the LCG state 'gen' from 'gen.next'.

$$\begin{aligned} \text{gen.next} &= (\text{gen} * c4ff1x + c66fb) \% c4ff10 \\ c4ff10 * n + \text{gen.next} &= \text{gen} * c4ff1x + c66fb \\ c66fb &= c4ff10 * n - \text{gen} * c4ff1x + \text{gen.next} \\ c66fb \% c4ff10 &= c66fb = (\text{gen.next} - \text{gen} * c4ff1x) \% c4ff10 \end{aligned}$$

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:

```
1 from pwn import *
2 import libnum
3 def gcd(a,b):
4     while a!=0:
5         a,b = b%a,a
6     return b
7 def findModReverse(a,m):
8     if gcd(a,m)!=1:
9         return None
10    u1,u2,u3 = 1,0,a
```

```

11     v1,v2,v3 = 0,1,m
12     while v3!=0:
13         q = u3//v3
14         v1,v2,v3,u1,u2,u3 = (u1-q*v1),(u2-q*v2),(u3-q*v3),v1,v2,v3
15     return u1%m
16 time=0
17 while True:
18     try:
19         p=remote('182.92.108.71',30641)
20         s=p.recvline()
21         m=int(s[s.find('(')+1:s.find(',')])
22         n=int(s[s.find(' ')+1:s.find(')')])
23         s0=int(p.recvline().strip('\n'))
24         s1=int(p.recvline().strip('\n'))
25         c=(s1-s0*m)%n
26         p.sendline(str(c))
27         n=int(p.recvline().strip('\n'))
28         s0=int(p.recvline().strip('\n'))
29         s1=int(p.recvline().strip('\n'))
30         s2=int(p.recvline().strip('\n'))
31         m=((s2-s1)*findModReverse((s1-s0),n))%n
32         c=(s1-s0*m)%n
33         p.sendline(str(m))
34         p.sendline(str(c))
35         list_s=[]
36         for i in range(7):
37             s=int(p.recvline().strip('\n'))
38             list_s.append(s)
39             t0=list_s[1]-list_s[0]
40             t1=list_s[2]-list_s[1]
41             t2=list_s[3]-list_s[2]
42             t3=list_s[4]-list_s[3]
43             t4=list_s[5]-list_s[4]
44             t5=list_s[6]-list_s[5]
45             x1=t2*t0-t1*t1
46             x2=t3*t1-t2*t2
47             x3=t4*t2-t3*t3
48             x4=t5*t3-t4*t4
49             x=[x1,x2,x3,x4]
50             n=abs(reduce(libnum.gcd,x))
51             p.sendline(str(n))
52             if p.recvline()=='fail\n':
53                 time+=1

```

```
54         p.close()
55     else:
56         break
57 except:
58     time+=1
59     p.close()
60     continue
61 print("time:",time)
62 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
time:', 331)
'flag is:', 'hgame{Cracking^prng_Linear)Congruential&Generators}\n')
*] Closed connection to 182.92.108.71 port 30641
leon@leon-virtual-machine:/mnt/hgfs/ctf$
```

得到flag。