

राईकवाCTF

CHALLENGE NAME: [FIGHT FIGHT FIGHT]

DEV : [Pushkar]

CATEGORY: [Reverse
Engineering]

LEVEL: [Hard]



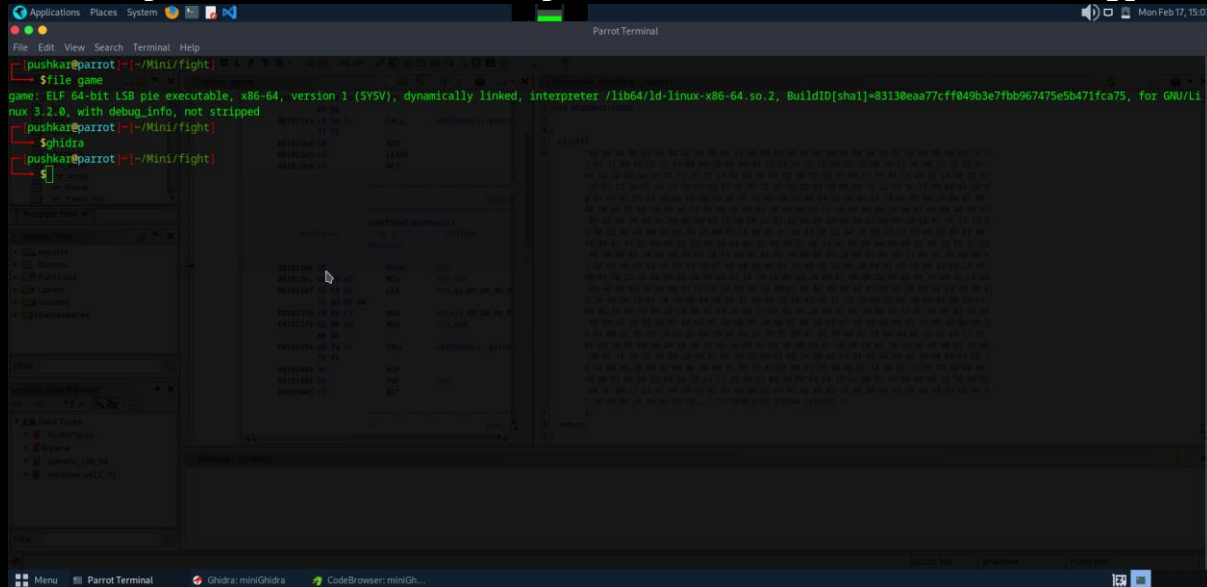
2025

Challenge Description:

Win this game, by hook or by crook.

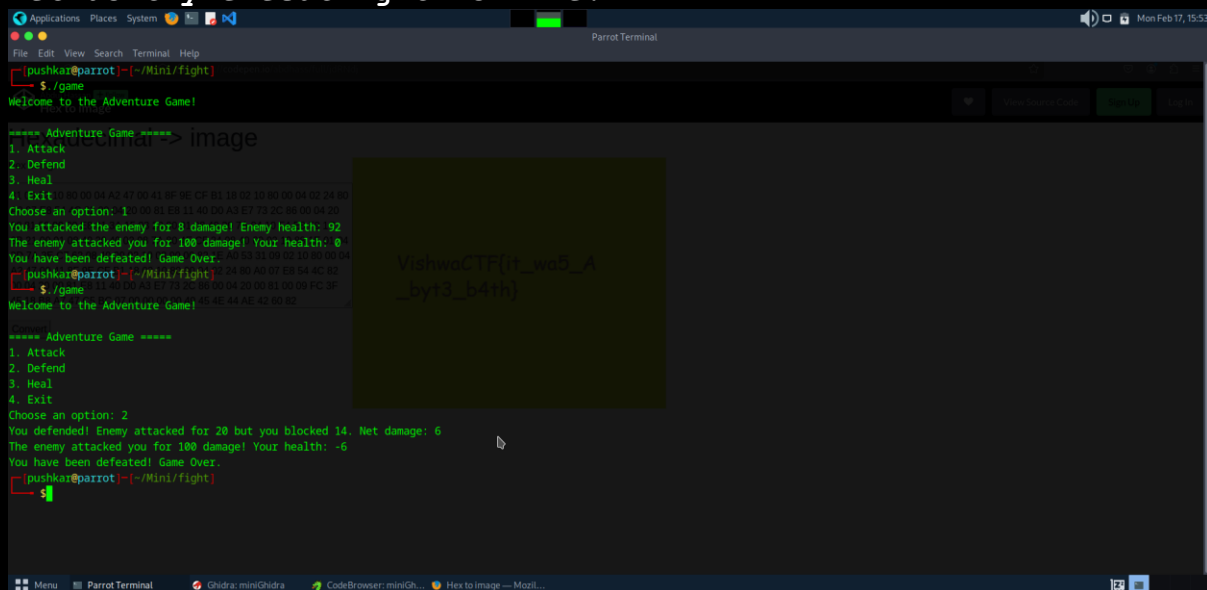
Solution:

- We are given a file called "game". Let us check the file type of "game".



```
pushkatz@parrot: ~/Mini/fight
$ file game
game: ELF 64-bit LSB pie executable, x86_64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=83130aaa77c7ff849b3e7fbb967475e5b471fca75, for GNU/Linux 3.2.0, with debug_info, not stripped
pushkatz@parrot: ~/Mini/fight
$ ghidra
pushkatz@parrot: ~/Mini/fight
$
```

- This is an ELF file which is fortunately not stripped.
- Let us try executing this file.



```
pushkatz@parrot: ~/Mini/fight
$ ./game
Welcome to the Adventure Game!
===== Adventure Game =====
1. Attack
2. Defend
3. Heal
4. Exit
Choose an option: 1
You attacked the enemy for 8 damage! Enemy health: 92
The enemy attacked you for 100 damage! Your health: 0
You have been defeated! Game Over.
pushkatz@parrot: ~/Mini/fight
$ ./game
Welcome to the Adventure Game!
===== Adventure Game =====
1. Attack
2. Defend
3. Heal
4. Exit
Choose an option: 2
You defended! Enemy attacked for 20 but you blocked 14. Net damage: 6
The enemy attacked you for 100 damage! Your health: -6
You have been defeated! Game Over.
pushkatz@parrot: ~/Mini/fight
$
```

- Turns out, this is a game but we are losing no matter what.
- Let us understand how this game is working in order to play it better. Let us open this file in Ghidra.

- Also, every time the enemy attack function is called, it makes the player health 0 causing us losing the game no matter what we choose.
- Therefore, we must attack the enemy and make its health 0 before it attacks us.
- In order to this, we must be able to magically pause time and make enemy health 0.
- Let us try it with GDB debugger.

```

For help, type 'help'.
Type "apropos word" to search for commands related to "word"...
Reading symbols from game...
(gdb) info functions
All defined functions:
File game.c:
14: void attack(int *);
20: void defend(int *);
5: void displayMenu();
36: void enemyAttack(int *);
29: void heal(int *);
46: int main();
42: void whattext();

Non-debugging symbols:
0x0000000000000000 _init
0x0000000000000000 puts@plt
0x0000000000000000 printf@plt
0x0000000000000000 srand@plt
0x0000000000000000 time@plt
0x0000000000000000 __isoc99_scanf@plt
0x0000000000000000 rand@plt
0x0000000000000000 __cxa_finalize@plt
0x0000000000000000 _start
0x0000000000000000 deregister_tm_clones
0x0000000000000000 register_tm_clones
0x0000000000000000 __do_global_ctors_aux
0x0000000000000000 frame_dummy
0x0000000000000000 _fini
(gdb)

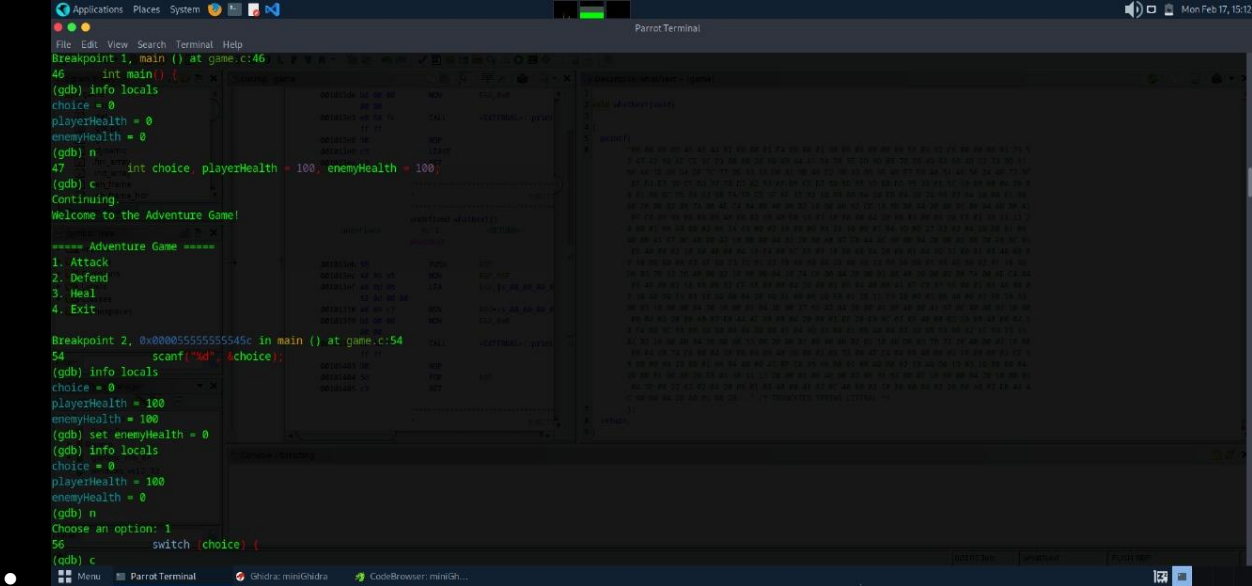
Dump of assembler code for function main:
0x0000000000000000: push %rbp
0x0000000000000001: <+0>: mov $0x0,%rbp
0x0000000000000002: <+1>: mov $0x10,%rsp
0x0000000000000003: <+2>: sub $0x10,%rsp
0x0000000000000004: <+3>: movl $0x0,%eax
0x0000000000000005: <+4>: movl $0x0,%ecx
0x0000000000000006: <+5>: movl $0x0,%edx
0x0000000000000007: <+6>: call 0x1050 <time@plt>
0x0000000000000008: <+7>: mov %eax,%edi
0x0000000000000009: <+8>: mov $0x3,%eax
0x000000000000000a: <+9>: call 0x1058 <srand@plt>
0x000000000000000b: <+10>: lea 0x10d4(%rip),%rax # 0x12198
0x000000000000000c: <+11>: mov %rax,%rdi
0x000000000000000d: <+12>: call 0x1030 <puts@plt>
0x000000000000000e: <+13>: mov $0x0,%eax
0x000000000000000f: <+14>: call 0x1189 <displayMenu>
0x0000000000000010: <+15>: lea -0x4(%rbp),%rax
0x0000000000000011: <+16>: mov %rax,%rdi
0x0000000000000012: <+17>: lea 0x10d3(%rip),%rax # 0x121b7
0x0000000000000013: <+18>: mov %rax,%rdi
0x0000000000000014: <+19>: mov $0x0,%eax
0x0000000000000015: <+20>: call 0x1070 <__isoc99_scanf@plt>
0x0000000000000016: <+21>: mov -0x4(%rbp),%eax
0x0000000000000017: <+22>: cmp $0x4,%eax
0x0000000000000018: <+23>: je 0x14c4 <main+190>
0x0000000000000019: <+24>: cmp $0x4,%eax
0x000000000000001a: <+25>: jg 0x14da <main+212>
0x000000000000001b: <+26>: cmp $0x3,%eax
0x000000000000001c: <+27>: je 0x14b6 <main+176>
0x000000000000001d: <+28>: cmp $0x3,%eax
0x000000000000001e: <+29>: jg 0x14da <main+212>
0x000000000000001f: <+30>: cmp $0x1,%eax
0x0000000000000020: <+31>:
--Type <RET> for more, q to quit, c to continue without paging--
  
```

- These are the functions available to us.
- Let us disassemble main.

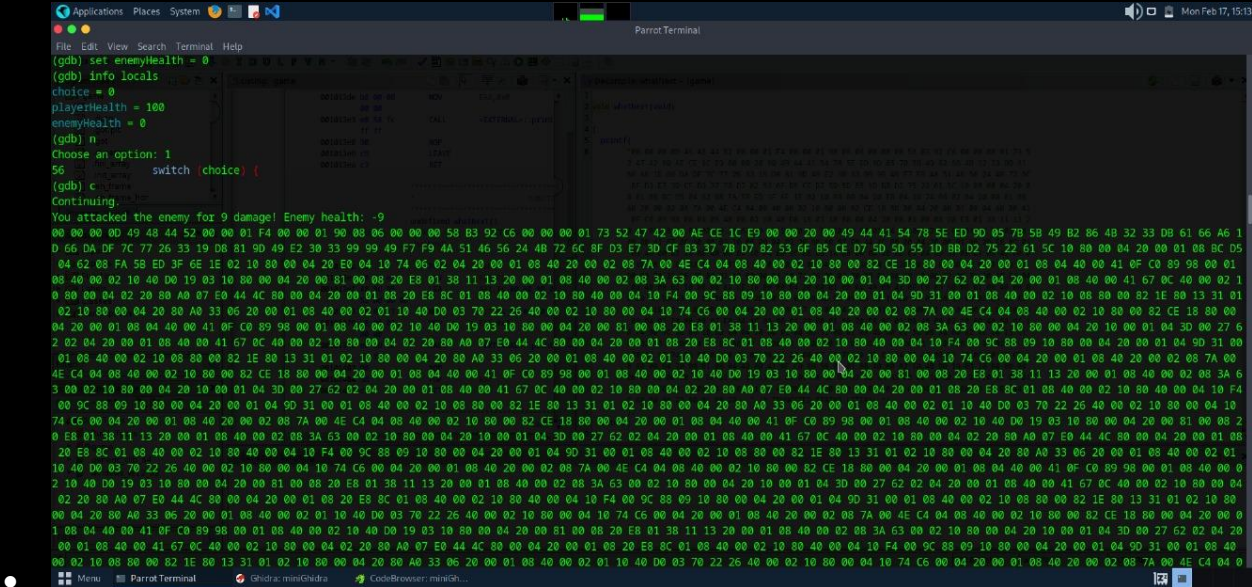
```

Here, we have a scanf@plt function. This must be the point where the game
asks us for our input like attack, defense, etc.
We must make the enemy health 0 before it attacks us.
Let us add breakpoints at required points and then run.
  
```

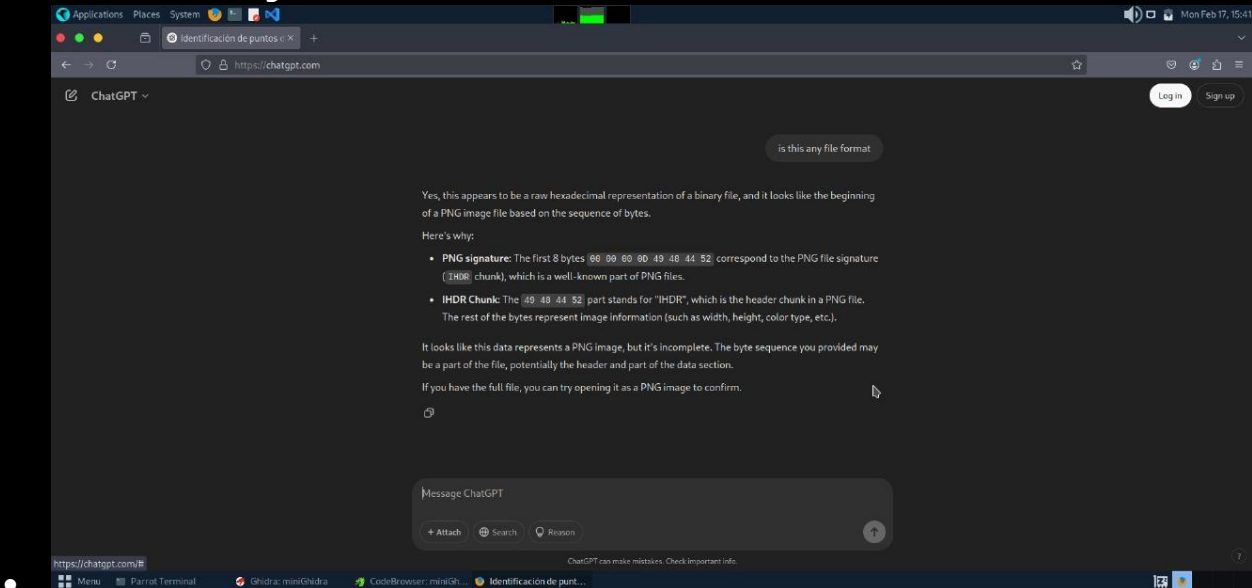
- Here, we have a scanf@plt function. This must be the point where the game asks us for our input like attack, defense, etc.
- We must make the enemy health 0 before it attacks us.
- Let us add breakpoints at required points and then run.



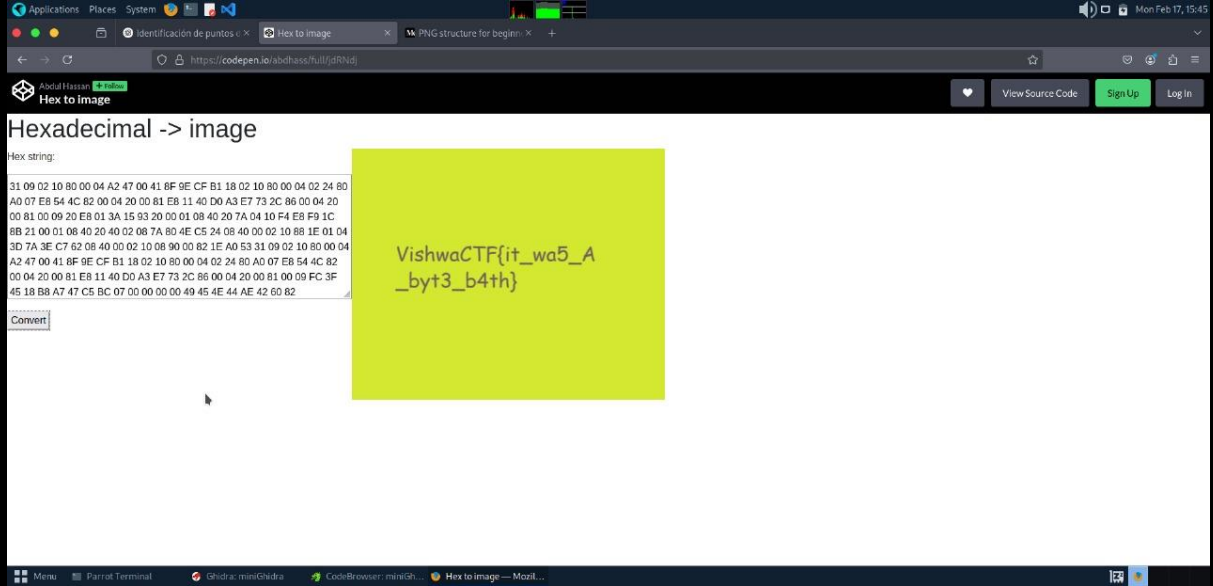
- Now that we have made the enemy health 0, let us attack now because that is the only time we are checking the enemy health value to be 0.



- We get a big hex dump with a message that we won.
- If we put some part of it on ChatGPT, it tells us this is a PNG dump with some missing headers.



- 89 50 4E 47 0D 0A 1A 0A is the missing header.
- Now let us put it at the start of the hex dump and try to convert it.



- Voila! We get back our flag.

- **Flag:** VishwaCTF{it_wa5_A_byt3_b4th}