# BeeNote

Soal ini bertujuan sebagai tiebreaker dalam final. Untuk pengerjaan diharuskan menggunakan ubuntu 16.04, dikarenakan cara kerja malloc yang berbeda pada 16.04 dan versi ubuntu di atasnya (17.10 ~)

Diberikan sebuah **ELF 64 bit**, **source code**, dan **libc-2.23**:

```
tempest@tempestuous ~/projects/beefest/final/pwn/beenote  master  file beenote; checksec beenote
beenote: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-
inux 2.6.32, BuildID[sha1]=e1eda19102118a65400baf043f637a86ff454759, not stripped
[*] '/home/tempest/projects/beefest/final/pwn/beenote/beenote'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      No PIE (0x400000)
```

Saya di sini tidak akan melampirkan semua source codenya, hanya beberapa bagian yang menarik.

```
unsigned int index;
note *notes[16];

void create_note()
{
        int n;
        char buffer[16];
        if(index < 16)
        {
                note *p = (note *)malloc(sizeof(note));
                if(!p) exit(0);
                notes[index++] = p;
                p->print_note = print_note;
                p->title = (char *)malloc(16);
                if(!(p->title)) exit(0);
                printf("Title: ");
                read_bytes(p->title,16);
                printf("Size: ");
                n = read_num(buffer,16);
                p->content = (char *)malloc(n);
                if(!(p->content)) exit(0);
                printf("Content: ");
                read_bytes(p->content,n);
                puts("Success!");
        }
        else puts("You can't create notes anymore!");
}
```

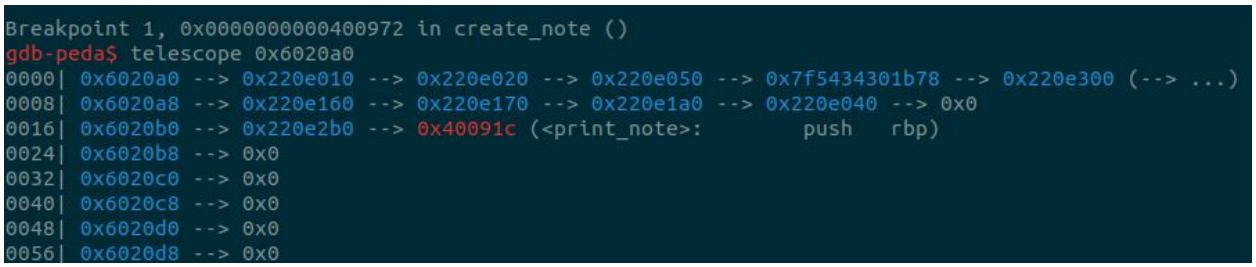
Ini merupakan bagian dari pembuatan notenya. Terlihat bahwa semuanya biasa saja.

Jika kita lihat pada bagian delete (clue sudah diberikan di deskripsi soal):

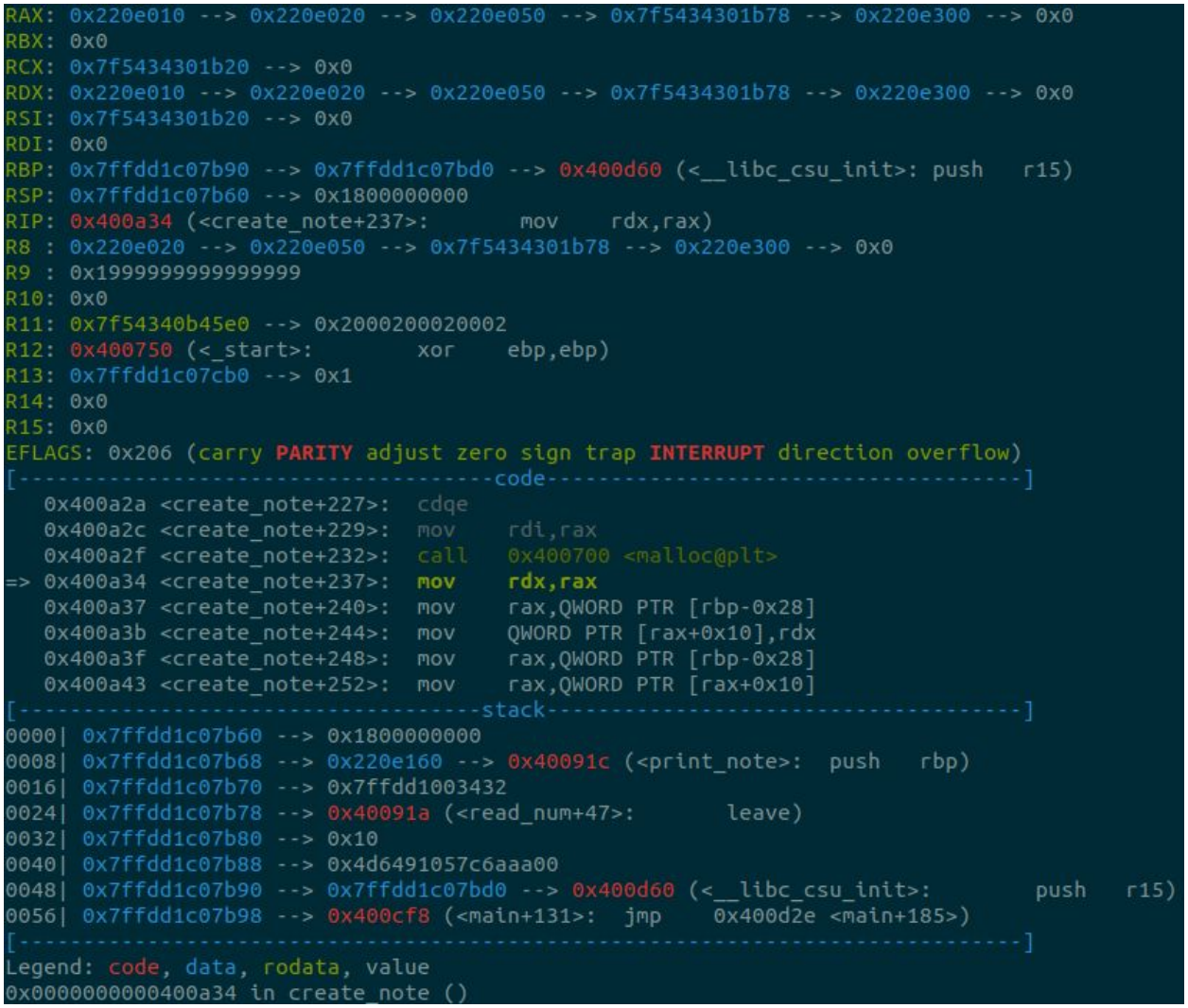
```
void view_note()
{
        int choice;
        char buffer[16];
        do
        {
                printf("Index of the note: ");
                choice = read_num(buffer,16)-1;
        }while(choice < 0 || choice > index-1);
        if(notes[choice])
        {
                note *p = notes[choice];
                p->print_note(p->title,p->content);
        }
        else puts("Note wasn't found.");
}

void delete_note()
{
        int choice;
        char buffer[16];
        do
        {
                printf("Index of the note: ");
                choice = read_num(buffer,16)-1;
        }while(choice < 0 || choice > index-1);
        if(notes[choice])
        {
                note *p = notes[choice];
                free(p->title);
                free(p->content);
                free(p);
                puts("Note deleted!");
        }
        else puts("Note wasn't found.");
}
```

Terlihat ada kelemahan dalam program, yaitu isi dari note dan pointer ke array of notes tidak di nol kan (Use After Free).
Target kita adalah memanfaatkan kelemahan ini supaya ***malloc*** memberikan kita ***struct note ** ketika kita meminta judul maupun isi dari note, kemudian mengganti dengan alamat system dan pointer ke string “/bin/sh”, dan “print” isi note.
User ID yang diberikan di awal-awal merupakan pointer ke stdin, sehingga kita tidak perlu membuat leak lagi (walaupun leak dapat dilakukan dengan mudah apabila kita mengerti cara

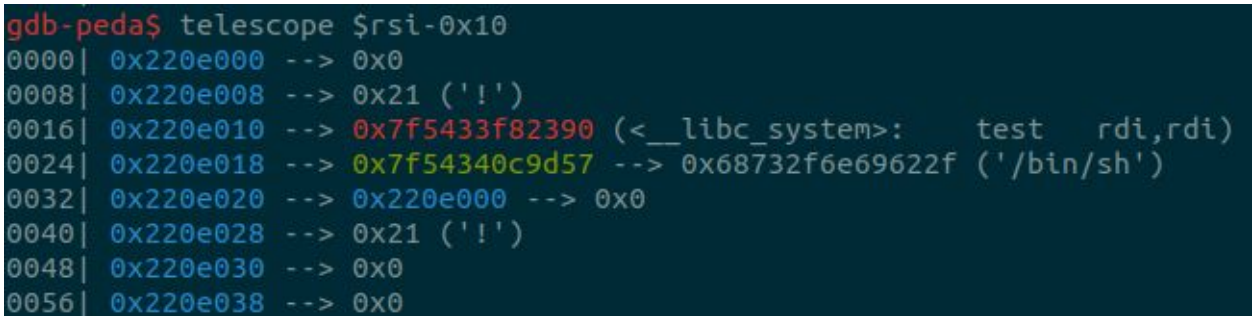
kerja ***malloc***). Selanjutnya kita tinggal menghitung offset dari stdin (lebih tepatnya ***_IO_2_1_stdin_***) ke ***system***. Setelah perhitungan dilakukan, kita tinggal mencari cara agar malloc memberikan kita ***struct note **** jika kita meminta ***char ****. Pertama-tama kita membuat 2 notes dengan ukuran content 256 (sizenya terserah, asalkan ukurannya di atas 127 dan di bawah 512 bytes). Selanjutnya, kita membuat sebuah note dengan ukuran content 16-32 bytes.



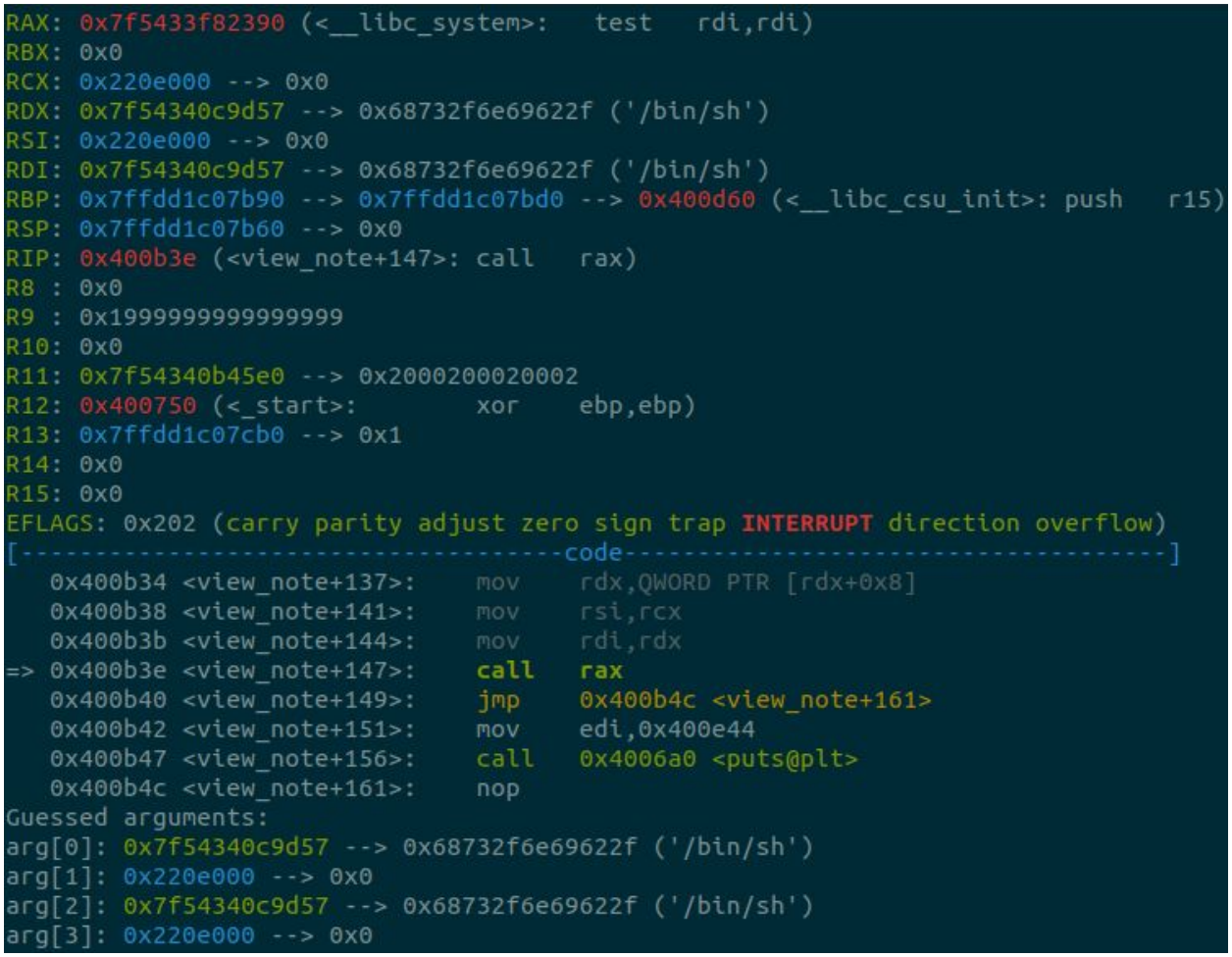

*0x6020a0* di sini merupakan alamat dari ***array of note pointers*** "notes". Tujuan dari 2 note pertama adalah untuk menaruh ***struct note **** ke *list of free memories*. Note 3 digunakan untuk mencegah supaya *list of free memories* ini tidak hilang (cara kerja malloc). Ketika kita membuat note yang keempat, isi dari note (content) akan diberikan ***struct note **** sebagai ***char **** (tipe data tidak begitu berpengaruh karena yang dilihat oleh malloc hanyalah alamat dan ukuran).




Terlihat bahwa return address dari malloc adalah sebuah pointer ke note (terlihat di screenshot sebelumnya). Kita tinggal mengganti function pointer dengan alamat dari system, serta judul dari note dengan pointer ke string “/bin/sh”.



Terakhir, kita “print” note pertama yang telah kita buat.




Flagnya adalah ***BeeFest{H4v3_y0u_h3ard_ab0ut_UAF_in_l0cal_CTF}***.