

Praktikum 3
IF2130 - Organisasi dan Arsitektur Komputer
"Father"
Buffer Overflow

Dipersiapkan oleh :
Asisten Lab Sistem Terdistribusi

Didukung Oleh :



Waktu Mulai :
Jumat, 26 November 2021, 17.59.59 WIB

Waktu Akhir :
Jumat, 3 Desember 2021, 17.59.59 WIB

I. Latar Belakang



Transmutasi adalah proses pengubahan sebuah substansi, yang berwujud maupun tidak berwujud kedalam sebuah bentuk ataupun keadaan yang baru. *Alchemist* merupakan sebutan bagi orang - orang yang ahli dalam menggunakan transmutasi. Mereka dapat melakukan transmutasi dengan bantuan sebuah pola yang disebut dengan *Transmutation Circles* untuk membuat apapun yang mereka inginkan dengan menukar sesuatu yang setara (*Law of Equivalent Exchange*). Hukum ini dapat diabaikan oleh pengguna dengan menggunakan *Philosopher's Stones*.

Sebagai seorang *Alchemist* terdapat larangan untuk melakukan transmutasi kepada manusia dan emas. Bagi mereka yang mencoba untuk mentransmutasi manusia akan kehilangan bagian tubuh mereka sebagai efek dari *Law of Equivalent Exchange* dan berujung menciptakan sebuah kegagalan dalam bentuk makhluk yang tidak manusiawi. Orang yang mencoba hal tersebut juga akan dihadapkan oleh *Shinri* (眞理), sebuah kebenaran akan dunia yang menyebabkan orang tersebut menerima informasi berlebihan akan kebenaran dan juga kemampuan untuk melakukan transmutasi tanpa *Transmutation Circle*.

Anda adalah seorang *alchemist* jenius. Di masa muda, Ayah Anda yang merupakan seorang *alchemist* terbaik di dunia menghilang tanpa alasan yang jelas bersama dengan ibu Anda dan meninggalkan Anda berdua saja dengan adik Anda. Sebelum ayah pergi, ia meninggalkan sebuah poster wasiat yang aneh.



Poster yang ditinggalkan oleh ayah Anda

Berusaha untuk membangkitkan kenangan yang tidak pernah ada, Anda ter dorong untuk membangkitkan karakter yang ada dalam poster tersebut. Ya, benar. Anda ~~sedang stress~~ mencoba untuk melakukan transmutasi manusia dan membangkitkan karakter kesayangan Anda. Ditemani dengan sang adik, Anda mulai menggambar lingkaran transmutasi dengan menggunakan Kapur B*gus Ajaib anti semut dan serangga 1 kotak isi 12 pcs.



Proses transmutasi yang dilakukan bersama dengan adik Anda

Seketika, cahaya yang sangat terang menyinari ruang bawah tanah Anda. Anda yang masih muda tidak paham dengan harga yang harus dibayarkan untuk melakukan transmutasi manusia. *Law of Equivalent Exchange* segera berlangsung dan membuat adik Anda berteriak

'AAAAAAAAAAAAAAAAAAAAAAA'

Seketika adik Anda lenyap. Meninggalkan Anda sendiri di ruang bawah tanah dengan sesosok monster aneh yang tidak berbahaya. Kebingungan dan ketakutan atas dampak yang terjadi, Anda mulai panik dan mencari adik Anda. Terdengar suara sayup dari adik Anda. Tak lama kemudian Anda mulai menyadari apa yang telah terjadi. Tubuh adik Anda menjadi 'tumbal' dari transmutasi yang Anda lakukan. Dengan sigap Anda mencari komputer terdekat dan menyegel jiwa adik Anda ke dalam komputer tersebut.

Tentu saja hal itu mudah bagi seorang *Alchemist* jenius seperti Anda. Setelah proses penyegelan berhasil, Anda dapat melihat adik Anda hidup di dalam komputer. Tidak sampai disitu, muncul pula masalah masalah lainnya. Komputer tempat jiwa adik Anda ternyata berisi virus (berasal dari situs download ilegal) yang berusaha menyerang jiwa adik Anda. Sekarang tugas Anda adalah menyelamatkan adik Anda dari virus - virus tersebut dan mengembalikan tubuhnya dengan merangkainya kembali dari satuan terkecil. Sebagai seorang *Alchemist* jenius tentu saja ini merupakan tugas yang mudah untuk dilakukan. Anda dapat melihat program hingga unsur - unsur terkecil dan memanipulasi alur program tersebut sesuai dengan keinginan Anda. Dengan menggunakan kemampuan ini Anda dapat memanipulasi alur virus dan menyelamatkan adik Anda.

Seberapa cepat Anda bisa menyelamatkannya ?

II. Deskripsi Tugas

Pada praktikum ini, kalian akan mengeksplorasi cara eksploitasi program dengan buffer overflow. Tugas kalian adalah untuk memasukkan input sedemikian rupa sehingga terjadi buffer overflow dan kalian dapat mengubah alur eksekusi program untuk mendapatkan jawaban tantangan. Seperti praktikum sebelumnya, dengan bantuan **gdb** atau tools sejenis kalian dapat memasang breakpoint, melihat perintah yang sedang dijalankan, disassembly suatu fungsi, isi memory (terutama *stack*), serta informasi lain yang dapat membantu keberjalanannya praktikum ini.

Apa itu buffer overflow? Buffer overflow adalah kondisi ketika program menulis data yang ukurannya lebih besar dibandingkan ukuran buffer yang disediakan, sehingga menimpa data yang bersebelahan. Kondisi ini dapat terjadi pada penggunaan fungsi input yang tidak memperhatikan ukuran input seperti **gets**. Sebagai contoh, perhatikan kode C berikut.

```
#include <stdio.h>

void vuln() {
    int val = 0;

    volatile int local = 0x12345678;
    char buff[4];

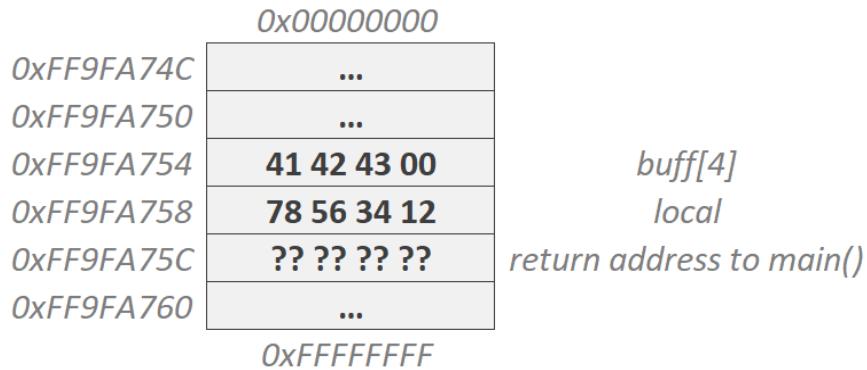
    printf("Local variable address %p\n", &local);
    printf("Buffer address %p\n", buff);
    printf("Your input : ");
    gets(buff);
    printf("Local variable value 0x%x\n", local);
}

int main(){
    vuln();
    return 0;
}
```

Melakukan compile program kemudian menjalankannya, didapatkan output berikut.

```
Local variable address 0xff9fa758
Buffer address 0xff9fa754
Your input :
```

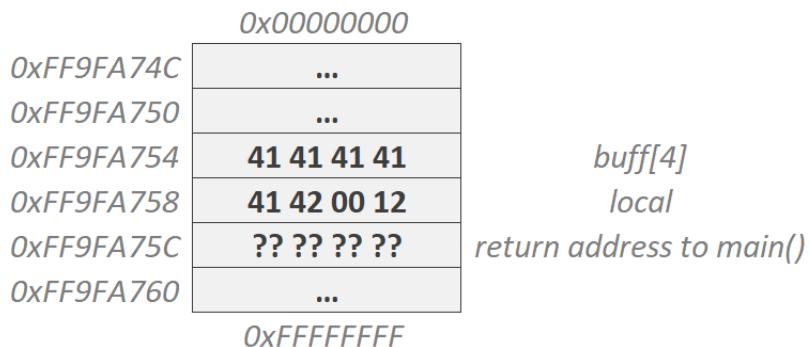
Perhatikan bahwa **buff[4]** memiliki address yang lebih kecil dibandingkan local. Jika pengguna menuliskan masukan **ABC** dan menekan enter, posisi kedua variabel pada stack dapat digambarkan sebagai berikut



Gambar diatas memperlihatkan kondisi stack setelah input ditulis ke dalam array **buff**. Sistem berbasis Linux menggunakan *little-endian* yang dapat terlihat pada susunan variabel lokal pada gambar. 0x41, 0x42, dan 0x43 merupakan karakter ABC dengan *encoding ASCII*.

Jika program diberikan input yang ukurannya lebih besar dari 3 karakter (3 byte karakter ditambah dengan 1 byte *null terminator*), maka dapat menimpa data yang telah ada di variabel **local**. Dengan input **AAAAAB**, didapatkan output sebagai berikut

```
Local variable address 0xff9fa758
Buffer address 0xff9fa754
Your input : AAAAAB
Local variable value 0x12004241
```



Dengan menggunakan fakta tersebut, fungsi **gets** dapat di-exploit dengan mengirimkan *payload* tertentu untuk memanipulasi eksekusi program. Variabel lokal dapat diubah dengan **menimpa nilai asli** (0x12345678) dengan input yang diterima oleh **gets**.

Perhatikan bahwa dengan menggunakan masukkan keyboard, nilai byte dari *payload* akan terbatas oleh karakter yang terdapat pada keyboard. Input menggunakan keyboard tidak dapat memasukkan byte 0x18 misalnya, dikarenakan 0x18 merupakan *invisible character* pada encoding ASCII. Untuk memasukkan byte yang merupakan *invisible character* dapat menggunakan *hex2raw.py* yang disediakan pada kit.

Pada tugas ini, terdapat 6 soal yang wajib dikerjakan serta 1 soal bonus. Pada setiap soal di platform, terdapat garis besar cara penggeraan yang dapat diikuti beserta hint jika terdapat masalah dalam penggeraan.

Langkah Umum Pengerjaan

1. Lakukan instalasi Operating System berbasis **Linux 32 bit**, di antaranya adalah Ubuntu, Fedora, atau distro Linux lainnya. OS dapat diunduh dari ubuntu.com maupun sumber lainnya. Anda dapat melakukan instalasi pada Virtual Machine ataupun Direct Installation. Dapat menggunakan WSL, tapi **HARUS WSL 2** karena WSL 1 tidak support program 32 bit.

Catatan: Jika belum terbiasa menggunakan Linux, disarankan menggunakan Ubuntu 20.04 Focal Fossa atau Ubuntu 18.04 Bionic Beaver.

2. Lakukan download kit di link [kit.zip](#). File kit yang anda download akan berupa file zip yang mengandung 3 file:
 - a. **main.c**: source code utama program secara garis besar.
 - b. **hex2raw.py**: script python3 untuk melakukan konversi teks hexadecimal (per 1 byte / 2 character hex) menjadi byte yang sesuai.
 - c. **input.txt**: contoh input untuk file **hex2raw.py**.
3. Download soal (**main**) pada platform di link <http://sister.suggoi.fun/challenges>. Login ke platform dapat dilakukan dengan username NIM dan password yang dikirimkan lewat email anda. Pastikan soal yang Anda download berada pada directory yang sama dengan kit.
4. Jalankan perintah **chmod +x main** pada terminal Anda di directory tempat Anda mengekstrak file hasil download. Hal ini dilakukan agar program dapat dijalankan.
5. Anda tidak disarankan untuk melakukan eksekusi file secara langsung untuk menebak karakter untuk buffer overflow. Anda dapat menggunakan perintah **gdb main** pada terminal untuk menjalankan gdb. Untuk memberikan input yang tidak termasuk dalam *printable character* (A-Z, a-z, 1-9 dan simbol), silahkan gunakan **hex2raw.py** dan pipe hasilnya ke **main**.

Sebagai contoh:

```
$ gdb main  
(gdb) run < <(python3 hex2raw.py)
```

6. Setelah input anda benar, anda akan mendapatkan jawaban pada komputer anda berupa string sebagai berikut:

```
This is the local flag for the Xth challenge, try sending the same payload  
to the server.
```

Gunakan input yang serupa pada binary yang disediakan pada server untuk mendapatkan jawaban sebenarnya. Server akan menjalankan binary yang sama.

Koneksi dapat dilakukan dengan perintah **nc** ke salah satu server berikut:

```
159.223.42.249:14045  
143.198.208.208:14045
```

Sebagai contoh:

```
$ nc 159.223.42.249 14045
```

7. Perhatikan contoh penggerjaan berikut dengan **hex2raw.py**:

Misalkan diperlukan 4 karakter untuk mengisi buffer sebelum target yang ingin diubah nilainya menjadi **0x12345678**. Maka, isi **input.txt** dengan nilai hex berikut.

```
61 62 63 64 78 56 34 12
```

Catatan: Setiap 2 angka dipisahkan dengan spasi berarti satu buah char. Isi file di atas akan diubah oleh **hex2raw.py** menjadi string **abcd** dan beberapa unprintable character. Jangan memasukkan **0A** karena **hex2raw.py** akan mengubah **0A** menjadi *newline* (LF) yang dianggap sebagai akhir input oleh program dan membuat input setelahnya tidak terbaca.

Jalankan perintah berikut untuk memasukkan input dari **hex2raw.py**.

```
$ python3 hex2raw.py | ./main
```

Jika didapatkan output sebagai berikut:

```
This is the local flag for the Xth challenge, try sending the same payload  
to the server.
```

Kirimkan payload yang serupa ke server dengan cara berikut. Perhatikan bahwa untuk koneksi ke server, anda perlu menyediakan NIM.

```
$ python3 hex2raw.py <NIM ANDA> | nc <IP SERVER> <PORT SERVER>
```

Misalnya:

```
$ python3 hex2raw.py 18219300 | nc 159.223.42.249 14045
```

8. Setelah input Anda pada server (melalui **nc**) juga benar, anda akan mendapat jawaban berupa **flag** dengan format sebagai berikut:

```
Orkom3{XXXXXXXXXXXXXXXXXXXXXXXXXXXX}
```

Perhatikan bahwa **flag berbeda** untuk tiap orang dan untuk tiap soal.

Flag inilah yang harus Anda submit pada **platform** yang dapat diakses pada link <http://sister.suggoi.fun/challenges> untuk **mendapatkan nilai** untuk soal yang bersangkutan.

III. Pengumpulan dan Deliverables

1. Setiap soal yang telah berhasil dijawab akan memberikan skor sebesar 100 poin. Soal bonus akan memberikan skor dynamic, dimulai dari 100 poin tergantung jumlah peserta praktikum yang berhasil menjawab.
2. Setiap soal memiliki batasan **maksimal** 10 kali submit jawaban yang salah.
3. File **main** yang Anda gunakan **bukanlah** hasil kompilasi file **main.c** saja, sehingga jangan lakukan kompilasi terhadap file **main.c**.
4. Anda dapat melihat nilai yang tercatat pada server melalui scoreboard di link [http://sister.suggoi.fun\(scoreboard](http://sister.suggoi.fun(scoreboard)
5. **Mulai** Jumat, 26 November 2021, 17.59.59 WIB waktu server.
Deadline Jumat, 3 Desember 2021, 17.59.59 WIB waktu server.
Pengumpulan jawaban akan ditutup setelah waktu tersebut.
6. Dilarang melakukan serangan Denial of Service terhadap server.
7. **Dilarang keras** melakukan submisi dengan jawaban orang lain. Kami memiliki rekap semua submisi yang anda lakukan sehingga **segala bentuk kecurangan akan ditindaklanjuti dengan serius**.
8. Kami akan menindaklanjuti segala bentuk kecurangan yang terstruktur, masif, dan sistematis.
9. Diharapkan untuk mengerjakan sendiri terlebih dahulu sebelum mencari sumber inspirasi lain (Google, maupun teman anda yang sudah bisa). Percayalah jika menemukan sendiri jawabannya akan merasa bangga dan senang.
10. Dilarang melakukan kecurangan lain yang merugikan peserta mata kuliah IF2130.
11. Jika ada pertanyaan atau masalah penggerjaan harap langsung isi pertanyaan di [QnA](#).
(Sister Tech Support)

IV. Troubleshooting

4.1. No such file or directory

Jika OS 64 bit dan baru diinstal

1. Untuk menjalankan kode 32 bit dibutuhkan *library* versi 32 bit yang biasanya tidak otomatis diinstal. Jalankan kode berikut:

```
sudo dpkg --add-architecture i386  
sudo apt-get install libc6:i386 libstdc++6:i386
```

Jika masih gagal

1. Pastikan lokasi terminal di folder yang berisi file **main**, **hex2raw.py**, dan **input.txt**
2. Gunakan perintah **cd <folder>** untuk pindah ke folder tertentu atau lebih mudah klik kanan di file manager dan pilih **Open in Terminal**

4.2 Di GDB program *hang* saat dijalankan (*run*)

Jika saat perintah run diberikan program tidak bekerja dan kadang muncul tulisan error:

```
warning: Breakpoint address adjusted from 0xf7fd9be0 to 0xffffffffffff7fd9be0.  
warning: Breakpoint address adjusted from 0xf7fda195 to 0xffffffffffff7fda195.  
warning: Breakpoint address adjusted from 0xf7fdbd1c to 0xffffffffffff7fdbd1c.  
warning: Breakpoint address adjusted from 0xf7fdb924 to 0xffffffffffff7fdb924.
```

Kemungkinan besar disebabkan karena bug

<https://bugs.launchpad.net/ubuntu/+source/gdb/+bug/1848200>

Solusinya adalah menginstall GDB versi lebih lama dengan perintah berikut:

```
sudo apt install gdb=8.1-0ubuntu3
```

4.3 Address buffer berubah-ubah di local machine

Jika saat kalian menjalankan di komputer kalian mendapatkan bahwa address buffer berubah-ubah, hal itu disebabkan karena ASLR menyala. Untuk menonaktifkannya:

```
echo 0 | sudo tee /proc/sys/kernel/randomize_va_space
```

Perintah tersebut hanya akan menonaktifkannya sementara. Setelah restart, maka ASLR akan aktif kembali.