Praktikum 3
IF2130 - Organisasi dan Arsitektur Komputer

"Disboard"

Buffer Overflow

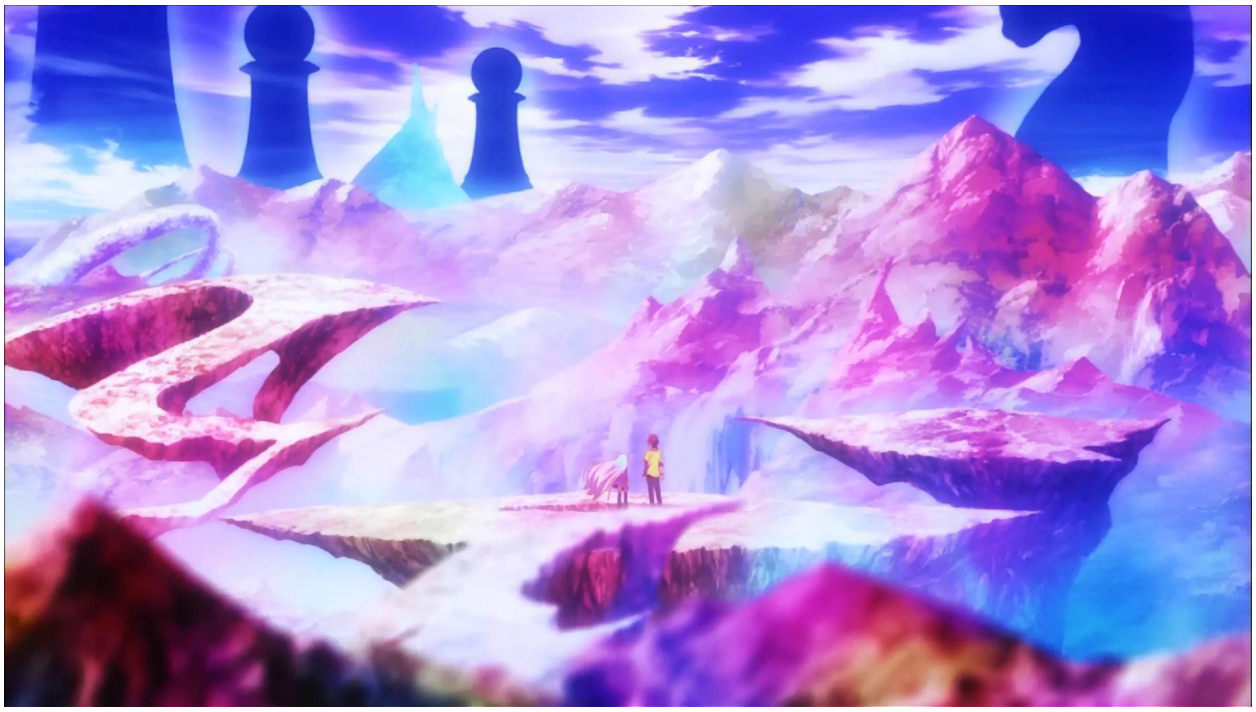
Dipersiapkan oleh :
Asisten Lab Sistem Terdistribusi

Didukung Oleh :

Waktu Mulai :
Jumat, 13 November 2020, 17.59.59 WIB

Waktu Akhir :
Jumat, 20 November 2020, 17.59.59 WIB

I. Latar Belakang



Anda adalah seorang pro gamer yang dilansir mampu menjadi juara dalam setiap permainan yang anda mainkan, apabila anda bersama dengan adik perempuan anda. Tidak ada yang tidak bisa kalian menangkan apabila anda bergerak sebagai sebuah kesatuan. Dikenal sebagai 「 」di jagat maya, anda sontak menjadi orang yang paling dicari-cari keberadaannya, namun sampai sekarang masih bisa kabur tanpa meninggalkan jejak kaki ataupun remah roti apapun.

Suatu malam, anda mendapatkan surel yang berisi tantangan untuk menyelesaikan suatu permainan. Permainan tersebut dinamai *Third Impact*, dan berasal dari seseorang yang asing dengan nama “St. Eve”. Sontak anda menerima tantangan tersebut dan berusaha untuk mengalahkan Tuan Eve.

「....」

Seminggu berlalu, anda telah mendapatkan gelar pemenang dari suatu pertandingan sengit untuk memutarbalikkan setiap tantangan yang diberikan. Sebuah perlawanan yang sangat sengit yang membuat anda bertanya.

“Sia teh... saha...?”「こいつは誰。。。強い」

Anda mendapat email dari St. Eve. Kemungkinan berisi ucapan selamat, atau bahkan bisa saja tantangan lain untuk menguji anda.

“Anda adalah seseorang yang pantas.”

“Dunia ini sudah menuju distopia yang hanya bermodal grafik yang bagus tapi dengan cerita yang ampas.”

“Apakah anda ingin sebuah dunia di mana segalanya ditentukan dengan permainan? Aku tau kamu akan suka, terlebih caramu memberi masukan perintah yang sangatlah unik.”

“Sebuah pertanyaan yang aneh”, pikirmu. Lagipula apakah benar ada suatu dunia yang hanya diatur dengan permainan? Namun ide tersebut cukup menarik minatmu, sehingga anda mengiyakan ajakan tersebut.

Perlahan, seluruh hal yang ada di sekitarmu mulai berguncang. Satu demi satu buku dan koleksi game PS5 dari lemarimu jatuh akibat gempa ini. Perlahan hal aneh mulai terjadi, mulai dari kabel-kabel yang mulai terdisintegrasi, *glitch* di mana-mana, dan ruangan anda menjadi sebuah ruang *void*. Lalu cahaya putih mulai bersinar sungguh berkilau.

Saat anda membuka mata, dunia sudah berubah. Segalanya menjadi sangat... berwarna. Setelah itu anda sadar kalau anda sedang jatuh terjun bebas dari langit, dan ada seorang bocah yang terlihat seperti terbang bersama anda.

“Selamat datang, di duniaku! Segala hal yang ada disini ditentukan dari permainan. Aksi dan **input** anda adalah penentu hidup mati anda.”

Aksi terjun bebas anda berlanjut sambil diceramahi peraturan yang berlaku dalam dunia ini, termasuk peraturan umum untuk setiap permainan yang ada. Seketika orang itu hilang, dan anda terdiam sejenak, mencoba menganalisis sekitar, untuk sadar kalau anda masih terjun bebas.

Tentu saja anda ingin segalanya berjalan dengan baik.

Walaupun anda melakukan *crash landing*, tidak ada hal yang menyakiti anda dan adik anda. Mungkin adalah sebuah berkah dari St. Eve. “Siapa pula bocah yang tadi itu...”, gumam anda selagi merapikan diri sendiri.

Dan disinilah anda, di dunia yang segalanya ditentukan oleh permainan.

Dapatkah anda memberi aksi yang sesuai untuk mengeksploitasi dunia ini?

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 keberjalanan praktikum ini. Petunjuk menggunakan gdb dapat dilihat pada petunjuk praktikum 2 (s.id/third-impact).

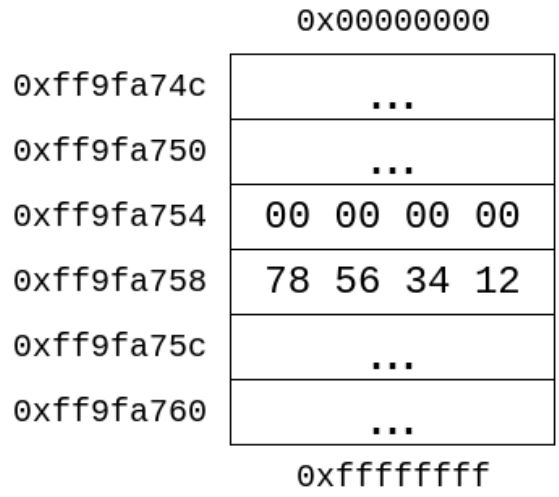
Apa itu buffer overflow? Buffer overflow adalah kondisi ketika program menulis data yang ukurannya lebih besar dibandingkan ukuran buffer yang disediakan, sehingga menimpa data yang bersebelahan. Kondisi ini dapat terjadi pada penggunaan fungsi input yang tidak memperhatikan ukuran input seperti **gets**.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

```c
#include <stdio.h>

void vuln() {
   int val = 0;

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

   fprintf(stdout, "Local variable address %p\n", &local);
   fprintf(stdout, "Buffer address %p\n", buff);
   fprintf(stdout, "Your input : ");
   gets(buff);
   fprintf(stdout, "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. Sehingga posisi kedua variabel pada stack dapat digambarkan sebagai berikut.



Jika program diberikan input yang ukurannya lebih besar dari 4 byte (4 karakter), maka dapat menimpa data yang telah ada di variabel **local**. Dengan input **AAAAB**, didapatkan output sebagai berikut.

```
Local variable address 0xff9fa758
Buffer address 0xff9fa754
Your input : AAAAB
Local variable value 0x12340042
```

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

Langkah Umum Pengerjaan

1. Lakukan instalasi Operating System berbasis Linux, di antaranya adalah Ubuntu, Fedora, atau distro Linux lainnya. OS dapat diunduh dari ftp://ftp.itb.ac.id/ maupun sumber lainnya. Anda dapat melakukan instalasi pada Virtual Machine ataupun Direct Installation.

   Catatan: Jika belum terbiasa menggunakan Linux, disarankan menggunakan Ubuntu 18.04 Bionic Beaver karena juga digunakan di server.

2. Lakukan download kit di link https://s.id/kit_prak3. 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://52.187.112.61/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:

52.187.112.61 11337
18.138.35.94 11337

Sebagai contoh:

```
$ nc 52.187.112.61 11337
```

7. Perhatikan contoh pengerjaan 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 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 18218027 | nc 52.187.112.61 11337
   ```

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://52.187.112.61/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://52.187.112.61/scoreboard.

5. **Mulai** Jumat, 13 November 2020, 17.59.59 WIB waktu server.
   Deadline Jumat, 20 November 2020, 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 pengerjaan 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 0xfffffffff7fd9be0.
warning: Breakpoint address adjusted from 0xf7fda195 to 0xfffffffff7fda195.
warning: Breakpoint address adjusted from 0xf7fdbd1c to 0xfffffffff7fdbd1c.
warning: Breakpoint address adjusted from 0xf7fdb924 to 0xfffffffff7fdb924.
```

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 dan 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.