CS177 Computer Security Discussion
Spring 2020 - Week 8

Dongyu (Hector) Meng
May 18th
Today

- Administration changes for project 4
- Popular questions about `minecraft(_hello)`
- Hints and background knowledge for `lazy panel`
Some announcements

- Deadline for project 4 is midnight today (no change)
- Office hour 7-10pm today
- Office hour tomorrow will be canceled
- Quick walkthrough for the challenges of project 4 at the beginning of my office hour next week (May 26th, 1pm)
Questions about minecraft(_hello)_

- In case you’re still working on the first two challenges
- We got many private questions on piazza
- Being able to ask the right question is the first step towards a solution
- There’s still time, not much, but enough for the first two challenges
Environment setup

- Pwntools installation
  - Try use a linux machine if you can (it’s just easier...)
    - Working with vm is preferred (root, GUI)
    - If you’re on CSIL, use python virtual environment tools (@187)
- Mac user pwnlib.shellcraft not working
  - Use a linux machine
  - Generate the shellcode on a linux machine and load it in your script
  - Other tools / google for shellcode / DIY
    - It’s the most common shellcode anyways
Environment setup

- GDB problems
  - Pwntools attach doesn’t work
    - Make sure you have the latest version (@187)
    - If you’re on CSIL, need to set terminal context (@187)
    - Last resort: wait_for_debugger and manually attach gdb (@195)
  - “../sysdeps/unix/sysv/linux/read.c: No such file or directory”
    - GDB’s fine, it’s working
    - You’re stepping through libc function, not very helpful

- GDB’s fine, it’s working
- You’re stepping through libc function, not very helpful
“How do I know I successfully get a shell?”

[DEBUG] Received 0x10 bytes:
   b'Very good, bye.\n'
[
] Switching to interactive mode

$ ls

[DEBUG] Sent 0x3 bytes:
   b'ls\n'

[DEBUG] Received 0x8e bytes:
   b'Dockerfile\t config exploit.py minecraft_hello.bin minecraft_hello.c\n'   b'build_docker.sh core\t makefile minecraft_hello.bin.idb run.sh\n'   Dockerfile config exploit.py minecraft_hello.bin minecraft_hello.c build_docker.sh core makefile minecraft_hello.bin.idb run.sh

$ cat /flag
“I got a segfault. Any idea?”

- Yes and no. You can get a segfault for many reasons.
- But in general, this means you’re handling some pointer incorrectly.
"I got a segfault. Any idea?"

- For this assignment, most likely your esp or eip is invalid
- GDB is always a way out

- >>> dmesg | tail -1
- >>> [ 7027.856094] minecraft_hello[22542]: segfault at deadbeef ip 0000000008048726 sp 000000000deadbeef error 4 in minecraft_hello.bin[8048000+1000]
Diagnostic message

Minecraft_hello[22542]: filename[pid]

segfault at deadbeef

ip 00000000008048726

sp 00000000deadbeef

error 4

in minecraft_hello.bin[8048000+1000] your code? lib?
**What is the address of win function?**

- And “Can I infer the address of win function from the stack address?”
- Once again, we had no-pie enabled.
- That is to say...

<table>
<thead>
<tr>
<th>high address</th>
<th>command-line arguments and environment variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>stack</td>
<td>command-line arguments and environment variables</td>
</tr>
<tr>
<td>heap</td>
<td>initialized to zero by exec</td>
</tr>
<tr>
<td>uninitialized data (bss)</td>
<td>read from program file by exec</td>
</tr>
<tr>
<td>initialized data</td>
<td></td>
</tr>
<tr>
<td>text</td>
<td></td>
</tr>
</tbody>
</table>
“What is the address of win function?”

- Ghidra? IDA?
- Or if you prefer lightweight approaches:
  - objdump -d minecraft_hello.bin | grep win
The dynamic way and the static way

- As an exercise, we hope you understand the details so that you can learn something
- Static way:
  - Reverse the code, understand what ecx does, where the return address is, one-shot perfect overflow
- Dynamic way:
  - “I know I want to keep ecx intact, but where is the return addr?”
  - “Let’s debug and see which offset match the final eip!”
  - “I’ll just copy paste the secret from memory!”
Extra credit: Lazy Panel
Lazy Panel

@mdy-personal $ INSERT ~/repo/cs307n/asm14ts/4/lazy_panel

./lazy_panel.bin

Hi there, welcome to the cs177 admin panel!
Please login...
Username:
aloha
Now tell me your password:
password
Oops, wrong password!
Lazy Panel

Hi there, welcome to the cs177 admin panel!
Please login...
Username:
aloha
Now tell me your password:

Welcome to the panel! What can I do for you?
give me the flag
Nah, I'm too lazy... Maybe next time...
Lazy Panel: reversing

- You don’t have source code this time. Need to analyse the binary.
- What interesting (library) functions does the binary use?
- Use Ghidra decompiled code to have a quick grasp of program logic!
- The purpose of reversing is not to reconstruct the original program pitch perfect. It’s more about finding the critical information for you to proceed.
https://ghidra-sre.org/CheatSheet.html
local_98 = 0x6867666564636261;
local_90 = 0x706f6e6d6c6b6a69;
local_88 = 0x7877767574737271;
local_80 = 0x4645444342417a78;
local_78 = 0x2e2e2e4b4a494847;
local_70 = 0;
local_c = 0;
while (local_c < 0x10) {
    iVar1 = rand();
    local_2c = iVar1 % 0x28;
    local_20[local_c] = *(char *)((long)&local_98 + (long)(iVar1 % 0x28));
    local_c = local_c + 1;
}
local_20[0x10] = '\0';
puts("Hi there, welcome to the cs177 admin panel!");
puts("Please login...");
puts("Username:");
pnl_in(local_18,0x14);
pnl_in(local_28,0x14);
iVar1 = strncmp(local_28,local_20,0x10);
if (iVar1 == 0) {
    pnl_out("Welcome to the panel! What can I do for you?\n",0x2d);
    pnl_in(local_108,400);
    puts("Nah, I\'m too lazy... Maybe next time...");
} else {
    puts("Oops, wrong password!");
}
rand

- The \texttt{rand()} function returns a \texttt{pseudo-random integer} in the range 0 to \texttt{RAND\_MAX} inclusive (i.e., the mathematical range [0, \texttt{RAND\_MAX}]).
- Read the \texttt{document}
- Learn about what \texttt{pseudo-random integer} means
strncmp

- int strncmp(const char *s1, const char *s2, size_t n);
- The strncmp() function compares the two strings s1 and s2.
- It returns an integer less than, equal to, or greater than zero if s1 is found, respectively, to be less than, to match, or be greater than s2.
- It compares the first (at most) n bytes of s1 and s2.
- No, “\x00” is not equal to anything!
Static + Dynamic

- They actually help each other.
- After you get the essence of what the program is trying to do, debug wisely to get the information you need!
local_98 = 0x6867666564636261;
local_90 = 0x706f6e6d6c6b6a69;
local_88 = 0x7877767574737271;
local_80 = 0x4645444342417a79;
local_78 = 0x2e2e2e4b4a494847;
local_70 = 0;
local_c = 0;
while (local_c < 0x10) {
    iVar1 = rand();
    local_2c = iVar1 % 0x28;
    local_20[local_c] = *(char*)((long)&local_98 + (long)(iVar1 % 0x28));
    local_c = local_c + 1;
}
local_20[0x10] = '\0';
puts("Hi there, welcome to the cs177 admin panel!");
puts("Please login...");
puts("Username:");
panel_in(local_18, 0x14);
puts("Now tell me your password:");
panel_in(local_28, 0x14);
iVar1 = strncmp(local_28, local_20, 0x10);
if (iVar1 == 0) {
    panel_out("Welcome to the panel! What can I do for you?\n", 0x2d);
    panel_in(local_108, 400);
    puts("Nah, I\'m too lazy... Maybe next time...");
} else {
    puts("Oops, wrong password!");
}
libc

- libc has all the code you need to create a shell.
- C standard library and more
- `system("/bin/sh");`
- Need to know the address of `system`
- To know that, you first need to find the address of libc
GOT and PLT

- Global Offset Table and Procedure Linkage Table
- When the program is executed, addresses of library functions it uses will be loaded to GOT
- A table of function pointers
- Your program call the dummy functions in PLT to jump to those library functions
- .got.plt and .plt in Ghidra (Program Trees window)
Use leaked address

- To get to the function you care about, which is `system`
- ASLR only relocates an executable entirely
- The relative address between functions within library is not messed up!
- Find out which libc version it is and where each function is [here](#)
- Or load your libc into pwntools (ELF) and use it for analysis
ROP tools

- angrop
- pwnlib.rop
Thanks!

Good luck if you’re not done yet!

Individual office hour starts at 7:00pm.