Danny North

12/5/15

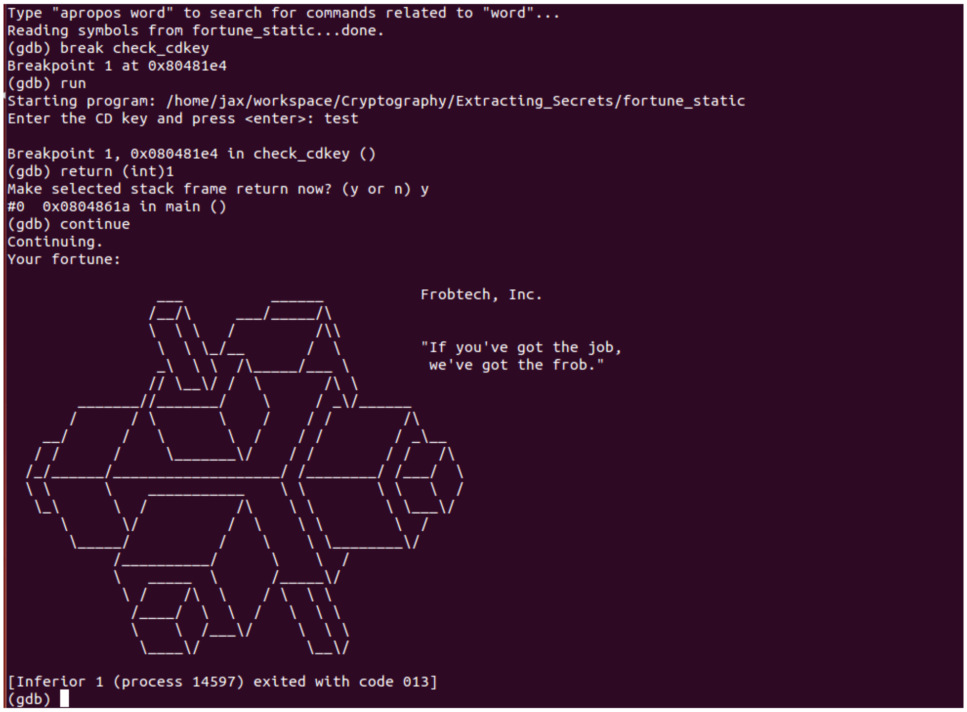
CS 465

Fred Clift

Project 11 - Extracting Secrets

1. **How did you use the debugger to bypass the password mechanism? What variables were modified? Please include a screenshot of the debugger in the report.**

   In GDB, I disassembled the main function to see what the possible functions were that I could access. Seeing the check_cdkey() function, I created a breakpoint inside of it and put some arbitrary CD key (the cd key didn't matter because I was going to overwrite what I return anyways!). With the breakpoint in the check_cdkey() function, and being able to disassemble within it now, I was able to see that many locations returned a false value (0) while one location returned a true value (1). I used the command "return (int)1" to get past the cdkey checker and receive a fortune.



```
Type "apropos word" to search for commands related to "word"...
Reading symbols from fortune_static...done.
(gdb) break check_cdkey
Breakpoint 1 at 0x80481e4
(gdb) run
Starting program: /home/jax/workspace/Cryptography/Extracting_Secrets/fortune_static
Enter the CD key and press <enter>: test

Breakpoint 1, 0x080481e4 in check_cdkey ()
(gdb) return (int)1
Make selected stack frame return now? (y or n) y
#0  0x0804861a in main ()
(gdb) continue
Continuing.
Your fortune:

                 ___          ______           Frobtech, Inc.
                /__/\     ___/_____/\
                \  \ \   /         /\\
                 \  \ \_/__       /  \         "If you've got the job,
                 _\  \ \  /\_____/___ \         we've got the frob."
                // \__\/ /  \       /\ \
        _______//_______/    \     / _\/______
       /      / \       \    /    / /        /\
    __/      /   \       \  /    / /        / _\__
   / /      /     \_______\/    / /        / /   /\
  /_/______/___________________/ /________/ /___/  \
  \ \      \    ___________    \ \        \ \   \  /
   \_\      \  /          /\    \ \        \ \___\/
      \      \/          /  \    \ \        \  /
       \_____/          /    \    \ \________\/
            /__________/      \    \  /
            \   _____  \      /_____\/
             \ /    /\  \    / \  \ \
              /____/  \  \  /   \  \ \
              \    \  /___\/     \  \ \
               \____\/            \__\/

[Inferior 1 (process 14597) exited with code 013]
(gdb) 
```

2. How did you edit the program to bypass the cdkey mechanism?

I was able to edit the program in a simple text editor, Sublime Text 2. When opening up the fortune_static executable in Sublime Text 2, all of the hex bytes are shown. By disassembling the main, I was able to see where the function "get_quotes_file" was. I also noticed that the first function call in main was printf. By finding the hex of where that function call was and changing the hex, I was able to make the first thing that happens the get_quotes_file() function instead.

So by changing the one byte of the call from:

0x080485f1 <+17>: e8 3a 35 00 00 call   0x804bb30 <printf>

To this:

0x080485f1 <+17>: e8 4a 00 00 00 call   0x8048640 <main+96>

I was able to get the program to jump to where I wanted it to jump to get a fortune every time. (This took a little bit of "frobbing".... At first I was guessing memory locations randomly but eventually found the pattern of where I wanted to go. The biggest clue came from this line:

0x08048640 <+96>: e8 4b fc ff ff call   0x8048290 <get_quotes_file>

3. How did you obtain all the fortunes from the encrypted file?

Now that I have a program that ends up giving me all of the plaintext fortunes from the get_quotes_file() function, I created a simple script that ran the program multiples of times and pulled all distinct plaintext fortunes and wrote them to a file. Here's the script:

```python
import subprocess

def writeToFile(string):
    f = open('fortunes.txt', 'a')
    f.write(string)

def main():
    fortunes = []
    while True:
        process = subprocess.Popen(['./fortune_static', '-a'], stdout=subprocess.PIPE)
        out, err = process.communicate()
        split1 = out.split("Your fortune:\n\n")[1]
        fortune = split1.split("\n\n")[0]
        if fortune not in fortunes:
            fortunes.append(fortune)
            writeToFile(fortune + "\n\n")
            print fortune
            print "\n\n Fortunes: " + str(len(fortunes))


if __name__== '__main__':
    main()
```