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: 400000x0

File size: 1864

Memory size: 1864

Flags: Read, Execute

03:

Offset: 1e10x0

Memory location: 601e10x0

File size: 22c

Memory size: 230

Flags: Read, Write

1. בתחילת ריצת התוכנית תוכן הזיכרון ב-401234x0 הוא חלק מ-section 02. משום שה-offset שלו הוא 0, מדובר בבית שבמקום ה-1234 מתחילת התוכנית. מ-hexdump נסיק שזהו "c7"
2. 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}

1. מאותם השיקולים כמו בסעיף 3, נקבל כי

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

1. נתבונן ב-symtable ונראה כי check\_password מופיע במקום 4005f0x0 בזיכרון. מאותם השיקולים כמו בסעיף 3, נתבונן ב-disassembly של הקובץ החל משורה 0x5f, ונשלים את הפונקציה בהתאם:

int check\_passwoed(char\* s){

int i;

for (i=0;i<11;i++){

if s[i]!=foo[bar[i]]{

return 0;

}

}

return s[11]==0;

}

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

# 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. מהסתכלות ב-gdb נראה כי אחרי הכנסת הקלט הלא תקין, פקודת ret תקפוץ לכתובת 0x3435363738396162. זאת משום ש-main מעבירה ל-scanf את הכתובת הנוכחית של rsp, שנמצאת מתחת למיקום של כתובת החזרה במחסנית.

| 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 |
| 0x400835 | cf c3 | dec %rdi  ret |
| 0x401661 | 5e 41 57 c3 | pop %rsi  pop %r15  ret |