**. Rivals of Catan – ' חלק ג**

1. בעזרת F12 נכנסנו לDeveloper Console. מעבר על הקוד המקור של האתר גילה שיש כפתור חבוי שמיועד שלחזור סיסמאות:

<div class="button disabled" onclick="challenge\_me()">Passwords Recovery</div>   
שינוי הclass ל " button" גילה את הכפתור.

1. לחיצה על הכפתור Passwords Recovery העביר אותנו לעמוד ה- Challenge.  
   לפי ההוראות כתבנו תוכנית המקבלת קלט המכיל בכל שורה אלמנט מטרה בודד ומחשבת את סך העלות במשאבים הנדרשים לבניה של כל האלמנטים. כתבנו את התוכנית ב-C ותרגמנו אותה לאסמבלי בעזרת האתר [godbolt.org](https://godbolt.org/). מכיוון שהדרישה שנפח התוכנית תהיה קטנה מ2KB-, ערכנו את קוד האסמבלי והחלפנו קריאות מפורשות לפונקציות ספריה (strcomp, printf, scanf ) בקריאה עקיפה ע"י טעינה ידנית של msvcrt.dll וחיפוש כתובת הפונק' בזמן ריצה. השתמשנו בשיטה שהוצגה בתרגול: LoadLibrary, FindFunction , GetProcAddress. מצורף קוד של התוכנית challenge.S.  
   לאחר מכן קמפלנו את התוכנית, חילצנו את הקוד הבינארי מתוך קובץ ה-obj שנוצר ושירשנו אותו עם קובץ PE.bin ו- find\_function.bin בצורה הבאה:

gcc -c challenge.S -o challenge.o

objcopy -O binary challenge.o challenge.bin

type PE.bin challenge.bin find\_function.bin > challenge.exe

לבסוף בעזרת CFF Explorer פתחנו לעריכה challenge.exe שיצרנו וערכנו את ה- Section Header.  
עדכנו את הגדלים Virtual Size ו- Raw Size לגודל 39A שזה הגודל בבתים של הקוד הבינארי ( הגודל של challenge.bin + find\_function.bin )

1. הגענו למסך recovery והורדנו את הקוד, ביצענו ניתוח סטטי של הקוד.
   1. כדי לעבור את level 1 צריך לתת לתוכנית לפחות ארגומנט אחד.
   2. בשלב שני level 2:
      * הפונק מקצה על המחסנית מערך בגודל 9 בתים ומתחלת אותו בערכים הבאים:

char arr0[9] = {0x13, 0xE8, 0xAB, 0xD3, 0x38, 0xD5, 0xDB, 0x19, 0x1D}

* + - הארגומנט השני משמש כseed לפונקציית rand32 בעזרתה מאותחל מערך נוסף באורך 9 בתים (נקרא לו arr) , כל בית מוגרל בנפרד + מניפולציה כלשהי ( ???).
    - לאחר מכן יש צורך ב 2 קלטים y>0 ,x המשמשים offset מהמערך:

int \*ptr = (arr + y) & FFFFFFFC

int \*ptr2 = (arr + x) & FFFFFFFC

כלומר offset מיושר לכתובת הקרובה המתחלקת ב4.

* + - קריאה ל- \_printArrayמדפיסה את הערכים שבטווח (ptr, ptr2)
    - קלט מהמשתמש z בפורמט Hex, איתו מבצעים:

Int\* ptr3 = ptr

while(ptr3 < ptr2)

{

\*ptr3 = z ^ \*ptr3;

Ptr3++;

}

מה שקורה פה זה שהוא עובר על המחסנית באותן הכתובות שהוא הדפיס, ועושה xor על כל byte ביחד עם קלט מהמשתמש

* + - לבסוף מבצעים השוואה כל בית במערכים arr ו- arr0 והם צרכים להיות זהים כדי לעבור את השלב.

for(int i=0; i<9; i++)

{

If( arr0[i] != arr[i] )

return;

}

Puts("Level 2 Passed!");

לכן, נצטרך לגרום לכך ששתי המשתנים יהיו שווים, נעשה זאת באמצעות החלק שבו הוא עושה xor, ראשית נכניס לו ערכים שיאפשרו לנו לגשת למערך השני, שיהיו 0 ו 9 וכפי שנראה מההדפסה של המערך הם ידפיסו לנו את המערך ועוד כמה ביטים מלפניו(אלו ביטים מהמערך הראשון)

Shape

Description automatically generated with medium confidence

לכן נרצה להכניס ערכים בצורה שתגרום למערך השני שכאן מתחיל ב81 (1d19db זה מהמערך הראשון, השאר מהשני) כך שכשנפעיל xor איתם עליו הערכים שנקבל זה ערכי המערך הראשון, נתחשב בזה שהכל בזה מיוצג בליטל אנדיאן ולכן נקבל

811D19DB xor 131d19db= 92000000

E1DF54FB xor 38d3abe8= d90cff13

874756BF xor 1d19dbd5 = 9a5e8d6a

ואחרי שנכניס את שלושת המספרים האלו נראה שבאמת נקבל את ההודעה שעברנו את שלב 2

3.3 כעת הגענו לשלב 3, נשים לב שאחרי שלב 2 התכנית חוזרת למיין ושם היא נגמרת, ואין לנו שום קפיצה לשלב שלוש, בעזרת חיפוש של הדפסה נמצא את שלב 3 במתודה בשם dummy , ולכן נרצה לגרום לתוכנית לקפוץ לשם, כפי שראינו, בשלב 2 התכנית איפשרה לנו להפעיל xor על המחסנית בטווח שאנחנו בוחרים כל עוד הערך החתון שלו מעל 0, לכן נוכל לגשת לכל ערך שנמצא במחסנית מעל המערכים שלנו, אם נחזור לפרוצדורה של שלב שתיים, בסוף הפרוצדורה קוראים לleave וret כדי לחזור למיין, מה שהפרוצדורות האלו עושות זה להעלות את המחסנית חזרה ולהוציא ממנה את הערך שלrip של מיין שנשמר לפני הקריאה לפונקציה, וכך הפונקציה חוזרת למיין, לכן אם נצליח לשנות את הערך הזה לכתובת של dummy נוכל לגרום לכך שבמקום לחזור למיין נחזור לכתובת של dummy , הכתובת של מיין בקוד היא 401a1d (השתמשנו בida לבדוק, קימפלנו את הקובץ עם gcc) ולכן נרצה למצוא אותה במחסנית, אז נריץ שוב את הקוד, הפעם במקום 9 נכניס ערך גדול, במקרה שלנו הכנסנו 49

Text

Description automatically generated

וכפי שניתן לראות הכתובת של מיין נמצאת בסוף ההדפסה של המחסנית בתחום שבחרנו, לכן נרצה להכניס ערכים כך שנעבור שוב את שלב שתיים, ונשנה את הכתובת חזרה ממיין לdummy לכן נכניס:

Text

Description automatically generated

ונקבל שעברנו את שלב 3 😊

3.4 הרצנו את התכנית, על ידי שימוש בדיבאגר הבנו שהתכנית מגיעה לנקודה בה יש משתנה שאינו מאותחל בה, שמאותחל במיין, ומשם שהבנו שבמקום לשנות את כתובת החזרה של המתודה של שלב 2 נשנה את הכתובת של מיין, וזה יחזיר אותנו להמשך מיין ובסוף מיין יעביר אותנו לdummy (חישבנו שוב הכל, הגדלנו את הטווח ל 0 עד 70 ועשינו שוב את הxor וקיבלנו הפעם 542 ) , אחר כך, בדקנו איפה מודפס שלב 4 וכשראינו שהוא מודפס בפונקציה בשם handler הבנו שצריך להפעיל חריגה, ראינו שבהמשך main מבוצע חישוב כלשהו שנשמר בdivider שאחר כך נחלק בו אחרי שלב 3, ולכן אם נאפס אותו תהיה חריגה מה שיפעיל את הhandler כאשר חלק החישב זה פקודת רנדום עם הסיד, לכן יצרנו קובץ הרצה חדש, שהפעם הוספנו לו שהוא יקפוץ לdummy בסוף main ואז השתמשנו הקובץ batch שרץ עם מספרים שונים כדי למצוא את הסיד שיאפס אותו, ולבסוף מצאנו שזה חמש.לכן נפעיל את התוכנית באופן דומה הפעם עם סיד 5 ונקבל שעברנו את שלב 4

Text

Description automatically generated