תרגיל בית 9 – machine basics – רטוב – מסכם קורס
"חדר בריחה – הכספת"

מועד ההגשה:	יום ה', 14/07/2022, בשעה 23:55	
האחראי על התרגיל:	פבל ליפשיץ	pavel@ee.technion.ac.il

תרגיל הבית הזה – הינו חדר בריחה וירטואלי. החדר בריחה הזה, הוא כספת, המכילה 6 שלבים – דלתות ממולכדות. כל סטודנט יקבל קובץ binary **אישי** להרצה, עם 6 שלבים. בכל שלב מצופה מהסטודנט להקליד מחרוזת קלט. אם הקלדתם את המחרוזת הנכונה, פתחתם את הדלת ועברתם שלב. אחרת תתפוצץ פצצה ויודפס "BOOM!!!" והתוכנית תסתיים. כשתענו נכונה על כל 6 השלבים תקבלו את המפתח ליציאה מהחדר (ולציון של תרגיל הבית :).

שלב 1

הורידו את הכספת האישית שלכם. ישנם 2 קבצים:

- vault – קובץ הרצה
- vault.c – קובץ המקור עם פונקציית ה-main של הכספת

להורדת הכספת האישית, בצעו:

git clone https://github.com/pavel-acsl/hw5a.git

git checkout

כאשר XXXXXXXXX זה מספר תעודת הזהות שלכם כפי שהוא מופיע במערכת הmoodle-פחות 101357.

לדוגמא: אם מספר תעודת הזהות שלכם הוא 123456789 אז XXXXXXXXX הוא 123355432

שלב 2

המשימה שלכם היא לגלות את 6 המחרוזות לצורך פתיחת הכספת. היזהרו מהמלכודות.
תוכלו להשתמש במגוון כלים על מנת לגלות כיצד פותחים את הדלתות. תוכלו להיעזר ב*טיפים* בסוף המסמך.
הדרך הטובה ביותר היא להשתמש ב-debugger: GDB, וללכת צעד-צעד על ה- disassembly של קובץ ההרצה.

השלבים שווים 23 נקודות כל אחד. מה שאומר שהציון המירבי האפשרי הוא 138. כמו בחדר בריחה, ניתן לבקש רמז תמורת 15 נקודות. בפורום "בקשת רמזים" כתבו את השלב עליו אתם מבקשים רמז ואיזה רמז מבוקש.תתקבל תשובה בהודעה פרטית.

כל שלב נהיה קשה יותר בהדרגה, אז אל תחכו לרגע האחרון..

<u>כדי שלא תצטרכו להקליד מחדש כל סיסמא לדלת שכבר עברתם דרכה</u>, אתם יכולים להפעיל את הכספת עם ארגומנט של קובץ txt המכיל בכל שורה את הסיסמא לשלב שכבר גיליתם.

```
$ ./vault part_solution.txt
```

במידה ואתם מצליחים, מומלץ לבצע static analysis כפי שנלמד בתרגול. זוהי הכנה מעולה לחלק מהשאלות במבחן.

העזרו ב- single-step וב- break-points. הערך הלימודי כאן הוא שתלמדו היטב להשתמש ב-debugger. דבר שיהיה שימושי גם לשאר הלימודים והקריירה שלכם בכלל. זוהי מיומנות קריטית.

בסיום התרגיל עליכם להגיש את הקובץ solution.txt לכספת שלכם.

הוראות הגשה:

1. עברו היטב על הוראות ההגשה של תרגילי הבית המופיעים באתר טרם ההגשה!
2. יש להגיש לינק ל-repository המכיל את הקובץ solution.txt (שימו לב לשם הקבצים עם lower case). על ה-repository להיות בעל הרשאות public. בעת בדיקת התרגיל, אנו נבצע clone ל- repository שלכם, נריץ את הכספת האישית שלכם עם ארגומנט solution.txt ונבדוק את התוצאה.
במידה ועובדים בזוג, מספיק להגיש עבור אחד מאנשי/נשות הצוות.
בעת ההגשה, יש להגיש את שמות הסטודנטים המגישים **כאשר שם הסטודנט בעל תעודת הזהות שבו השתמשתם לצורך התרגיל יופיע בשורה הראשונה.**
שימו לב להגיש לפי הפורמט הבא:

```
https://github.com/your-username/repository-name

0123456789 student_1_mail@campus.technion.ac.il first_name_1 last_name_1

0123456789 student_2_mail@campus.technion.ac.il first_name_2 last_name_2
```

3. שאלות בנוגע לתרגיל יש להפנות לפורום התרגיל ב-moodle בלבד – ניתן לשלוח שאלות במייל **למתרגל האחראי על התרגיל בלבד**, ורק במידה והשאלה מכילה פתרון חלקי.
4. סיכום מפרט התרגיל:

סעיף	תיאור
נושא התרגיל	Machine Basics - debugging
תאריך ההגשה	יום ה', 14/07/2022 בשעה 23:55
המתרגל האחראי על התרגיל	פבל ליפשיץ pavel@ee.technion.ac.il
קבצי הקוד הנתונים	vault.c
קבצי הקלט והפלט הנתונים	
הקבצים שיש להגיש	solution.txt

בהצלחה!

<u>רמזים</u>

ישנם דרכים רבות "לפצח" את הכספת:

אפשר לנסות לפצח ב-brute force, בעזרת סקריפט, אך אינכם יודעים מה הם אורכי המחרוזות, ולא כמה זמן יקח לעבור על כל האפשרויות.. (ולכן זו לא הדרך עליה אנחנו ממליצים).

תוכלו למשל לנתח את הקובץ אף ללא הרצה שלו כלל.

היעזרו בכלים מההרצאות, התרגולים, והסדנאות –

- strings – כלי המדפיס את כל המחרוזות שישנם בקובץ בינארי
- objdump -t

ידפיס לכם את ה- symbol table של הקובץ הרצה. ה-symbol table מכיל את כל הפונקציות והמשתנים הגלובאלים בקובץ. את שמות הפונקציות והכתובות שלהם. *ניתן ללמוד רמז כלשהו משמות הפונקציות*

- objdump -d

יבצע disassembly לקובץ. תוכלו להסתכל על פונקציות ספציפיות. אבל הקריאה הזאת לא תתן לכם את כל התמונה: קריאות לפונקציית מערכת מופיעות בצורה מעט קריפטית (תמיד, זה לא חלק מהסוד של הכספת). לדוגמא, קריאת ל- sscanf יכולה להופיע כך:

```
8048c36:   e8 99 fc ff ff   call  80488d4 <_init+0x1a0>
```

כדי לגלות שמדובר ב-sscanf תצטרכו לעבוד עם gdb.

העזרו ב-man או ב-google כדי להבין מה הארגומנטים של פונקציות מערכת, לדוגמא: man sscanf.

תקציר פקודות שימושיות ב-gdb:

- disassemble – מציג assembly (disas != disa שימו לב)
- break (שם פונקציה או כתובת) – עוצר את התוכנית כשמגיעים לנקודה
  info b - מידע על כל ה-breakpoints המוגדרים
  disable # – מבטל breakpoint עם מספר #
- stepi – התקדם פקודה תוך כניסה לפונקציות
- nexti – התקדם פקודה ללא כניסה לפונקציה
- c – המשך עד לנקודת עצירה הבאה
- print [c ביטוי בשפת] -
  שימושי למשל כדי לבדוק משתנה מקומי או איזור זיכרון, שימו לב לבצע casting:
  לדוגמא, print *(long*)pointer_to_long_var
- x – ניתוח הזכרון
- info registers – מציג את ערכי הרגיסטרים
- set disassemble-next-line on
- show disassemble-next-line

להצגת פקודת ה-assembly הבאה אחרי כל step.

<u>שאלות נפוצות</u>

- לא מבין מה עושה קטע קוד גדול ב-assembly? GDB
- צריך לדעת מה יש בכתובת מסוימת בזכרון? GDB
- רוצה לדעת איך כמה רגיסטרים משתנים לאורך הזמן? GDB
- לא יודע איך בכלל להתחיל? רמזים בעמ' 3
- פקודות GDB שימושיות? Google: GDB cheat sheet
- מה פקודת Assembly מסוימת עושה? שקפי הרצאות
- שימוש ב-GDB? סדנא
- מבנה הרצה של תוכנית? שקפי הרצאות
- פקודות וכלים שימושיים? תרגולים וסדנאות