תרגיל 4

הנחיות הגשה

1. העבודה היא ביחידים.
2. ההגשה היא עד ליום שישי, בתאריך 12.1.18 בשעה 16:30 (כניסת שבת).

רקע כללי

בתרגיל זה הינכם נקראים לדגל – Dr.Evil הטמין "פצצות בינאריות" (binary bombs) על המחשבים שלנו. נבחרתם להיות חלק מכוח המשימה שינטרל את הפצצות, הודות למומחיות שלכם באסמבלי x86-64, ייצוג מידע בזיכרון ופעולות אריתמטיות במחשב.
כל אחד מכם יקבל "פצצה בינארית" אותה צריך לנטרל (כל סטודנט את ה"פצצה" שלו). אם לא תעשו זאת, יקרה אחד מהשניים:

א. ציון לא טוב בתרגיל
ב. העולם יגיע לקיצו ו-Dr.Evil ינצח

אנחנו לא בטוחים מה יקרה מביניהם, אבל תבחרו באחד מהשניים בתור המוטיבציה שלכם לתרגיל ☺

מהי פצצה בינארית?

פצצה בינארית היא תוכנית שמורכבת ממספר שלבים (phases). בכל שלב צריך להכניס מחרוזת מסוימת דרך ה-stdin. אם הוכנסה המחרוזת הנכונה עבור שלב מסוים, השלב מנוטרל (defused), והפצצה עוברת לשלב הבא. אחרת, הפצצה מתפוצצת (explodes) – מדפיסה "BOOM!" והתוכנית מסתיימת.
כל שלב בודק היבט אחר של תוכניות בשפת מכונה:

1. שלב 1- השוואות
2. שלב 2- לולאות
3. שלב 3- conditionals/switches
4. שלב 4- קריאות רקורסיביות ושימוש במחסנית (stack)
5. מצביעים
6. רשימות מקושרות/מצביעים/stucts

ישנו גם "שלב סודי"...

הפצצה מנוטרלת לאחר שכל אחד מהשלבים נוטרל. כל סטודנט מקבל פצצה ייחודית, כך שיש פתרון (כלומר, רצף המחרוזות שצריך להכניס לתוכנית) שונה לכל פצצה. השלבים מסודרים לפי רמת קושי עולה – אמנם שלב 1 די קל, אבל זה לא אומר שאין עבודה בהמשך.

מכיוון ש-Dr.Evil הטמין כל כך הרבה פצצות, אנחנו נותנים לכל אחד מכם פצצה משלו. המשימה שלכם היא לנטרל את הפצצה לפני התאריך הנקוב בתחילת התרגיל.

בהצלחה וברוכים הבאים ל-Bomb Squad!

איך להשיג את הפצצה שלכם

כל פצצה היא Linux binary executable file שנוצר ע"י קימפול תוכנית בשפת C. כדי להשיג את הפצצה:

אפשרות 1 – מהאוניברסיטה

1. היכנסו לאחד ממחשבי האוניברסיטה (המחשבים במעבדות בבנין 604)
2. פתחו דפדפן אינטרנט והגיעו לכתובת http://u2.cs.biu.ac.il:15213/
3. מלאו את האי-מייל שלכם במחלקה (username@cs.biu.ac.il) ותעודת זהות
4. לחצו על "Submit"
5. יתקבל קובץ להורדה בשם bomb**k**.tar, כאשר k הוא מספר הפצצה שלכם – מספר ייחודי

את הקובץ bombk.tar צריך לשמור במשתמש שלכם בשרת u2. תוכלו לעשות זאת באופן הבא:

1. שילחו לעצמכם במייל את bombk.tar מהמחשב במעבדות שבו הורדתם את הקובץ
2. מהמחשב האישי שלכם העבירו את bombk.tar למשתמש שלכם ב-u2:
   scp bombk.tar username@u2.cs.biu.ac.il:~/any/folder/you/wish

אפשרות 2 – מהבית

1. הורידו את הקובץ get_bomb.zip מהאתר (דף "Exercises") וחלצו משם את הקובץ get_bomb.sh
2. פתחו טרמינל בתיקייה שבה חילצתם את הקובץ ולהריץ את הפקודות הבאות לפי הסדר:
   chmod +x get_bomb.sh
   ./get_bomb.sh
3. הכניסו את הפרטים שלכם כפי שהסקריפט מבקש. שימו לב - בשלב מסוים ייפתח חלון טרמינל חדש, שבו צריך להכניס את הסיסמא שאיתה אתם מתחברים ל-u2. אחרי ששמתם את הסיסמא שלכם ל-u2 ***בחלון החדש שנפתח***, חיזרו לחלון המקורי ולחצו enter
4. קובץ הפצצה יירד אל התיקייה הנוכחית שלכם (במחשב האישי שלכם – לא ב-u2), בשם bombk.tar (כאשר k הוא מספר הפצצה)
5. מהמחשב האישי שלכם העבירו את bombk.tar למשתמש שלכם ב-u2:
   scp bombk.tar username@u2.cs.biu.ac.il:~/any/folder/you/wish

(תודה לאמיר חוזז עבור הסקריפט)

תחילת עבודה על הפצצה

התחברו באמצעות ssh ל-u2 (ssh username@u2.cs.biu.ac.il) והגיעו לתיקייה בה שמור הקובץ bombk.tar.

פתחו את ה-tar. באמצעות הפקודה tar -xvf bombk.tar. הפקודה תיצור תיקייה בשם bombk.tar/. עם הקבצים הבאים:

1. README- זיהוי הפצצה והסטודנט האחראי עליה
2. bomb- קובץ הריצה (executable) של הפצצה
3. bomb.c- קובץ מקור (source file) עם ה-main routine של הפצצה

יתכן וכדי להריץ את הפצצה עליכם לשנות את הרשאות הקובץ bomb. עשו זאת באמצעות הפקודה chmod 755 bomb.

ניטרול הפצצה

כאמור, המשימה היא לנטרל את הפצצה ע"י הכנסת הקלטים הנכונים לכל 6 השלבים. כדי לעשות זאת, יש להתחבר למשתמש שלכם ב-u2 (באמצעות ssh) ולהריץ את הפצצה – אין אפשרות להריץ את הפצצה על המחשב האישי אלא רק על גבי ה-u2.

בכל פעם שהפצצה מתפוצצת, התוכנית שולחת עדכון למערכת ומאבדים ¼ נקודה מציון התרגיל.

שווי השלבים 1-4 הוא 10 נקודות כל אחד, ואילו שווי שלבים 5 ו-6 הוא 15 נקודות כל אחד. כלומר סה"כ שווי השלבים הוא 70 נקודות. לכן, אם רק ניסיתם לנטרל את הפצצה, כבר קיבלתם 30 נקודות במתנה!

רמת הקושי של השלבים עולה עם ההתקדמות בתרגיל, אבל כמובן שהמיומנות שלכם תגדל ככל שתתקדמו בשלבים. בכל זאת, השלבים האחרונים הופכים למאתגרים יותר ולכן הקפידו להתחיל את התרגיל בזמן.

הערות וטיפים

1. בכל הרצה של הפצצה, הפצצה תתחיל מהשלב הראשון. לכן, אם ניטרלתם את שלבים 1-3, למשל, כדי לנסות לנטרל את שלב 4 עליכם להכניס שוב את הקלטים הנכונים עבור שלבים 1-3
2. הפצצה מתעלמת משורות ריקות (blank lines). אם מריצים את הפצצה עם command line argument, למשל

```
linux> bomb psol.txt
```

הפצצה תקרא את שורות הקלט מ-psol.txt עד שתגיע ל-EOF, ואז את הקלטים הבאים תקרא מה-stdin. זה הודות לרחמים שגילה Dr.Evil, שרצה לאפשר למי שינסה לנטרל את הפצצות שלו להימנע מהקלדה חוזרת של פתרונות לשלבים שכבר ניטרלתם (ראו הערה 1)

3. הפקודה objdump –d על הפצצה שלכם תהיה מאוד שימושית כאן – אמנם ה-source code של הפצצה לא בידכם, אבל תוכלו לקבל את ה-disassembly. כדי לשמור את הפלט של objdump –d לתוך קובץ (שתוכלו למשל לכתוב עליו הערות), תוכלו להריץ את הפקודה הבאה: objdump –d bomb > mydisassembly.txt.
4. הפקודה objdump –t על הפצצה שלכם תדפיס את ה-symbol table, שם תוכלו לראות שמות של כל הפונקציות והמשתנים הגלובליים בתוכנית ושמות כל הפונקציות להן הפצצה קוראת והכתובות שלהם.
5. הפקודה strings על הפצצה שלכם תדפיס את ה-printable strings בפצצה.
6. את הפקודות objdump –d, objdump –t, strings אפשר להריץ גם על המחשב האישי שלכם
7. כדי להימנע מלפוצץ את הפצצה בטעות, השתמשו בדיבאגר כדי לקבוע breakpoints בתוכנית ולהתקדם צעד-צעד
8. בנוסף, דיבאגר יהיה כלי מאוד שימושי כאן – בדיקת ערכים ברגיסטרים ובזיכרון יתנו לכם מידע חיוני להבנת התוכנית (קראו עוד ב"שימוש ב-gdb")
9. יתכן ותצטרכו להיעזר בטבלת ASCII כדי להמיר ערכים ב-hex לתווים (טבלה בתרגול 1)
10. אם תצטרכו תיעוד של פקודות שונות, תוכלו להיעזר ב-**apropos** ו-**man** של פקודות. גם **man ascii** יוכל לעזור

הגשת התרגיל

לא צריך להגיש שום קובץ. ברגע שתנטרלו בהצלחה את הפצצה שלכם, התוכנית תעדכן את המערכת.

הערה חשובה

אם הייתה לכם שגיאה בעת הורדת הפצצה, תוכלו להוריד חדשה. אבל ברגע שהתחלתם לעבוד על פצצה מסוימת – על הפצצה הזו יתקבל ציון התרגיל.

שימוש ב-gdb

1. פקודות שימושיות בדיאבגר מופיעות בקובץ How_To_GDB.pdf (שמוכר כבר מתרגיל 3)
2. display $rdx תציג את תוכן הרגיסטר rdx אחרי ביצוע כל פקודה בתוכנית
3. stepi או si תקדם את התוכנית צעד אחד
4. כדי ש-gdb ידפיס את הפקודה הבאה בכל פעם: display /i $rip (מאוד שימושי!).
   הרגיסטר rip שומר את כתובת הפקודה הבאה בתוכנית שתתבצע.
5. x/4x 0xbfbff0d4c תדפיס 4 מילים (בפורמט hex) שמתחילות בכתובת 0xbfbff0d4c
6. x/s $rdx תדפיס את המחרוזת (רצף תווי ASCII שמסתיימים ב-null) שמתחילה בכתובת השמורה ב-rdx
7. כדי לבצע disassembly לפונקציה שלמה (למשל, main): disas main
8. כדי לקבל מידע על ה-frame הנוכחי במחסנית: info frame
9. כדי להדפיס מידע לגבי כל ה-active stack frames: backtrace

בהצלחה!