### הטכניון – מכון טכנולוגי לישראל

**ארגון ותכנות המחשב**

תרגיל 2 - חלק יבש

המתרגל האחראי על התרגיל: איתי אילת.

שאלותיכם במייל בעניינים מנהלתיים בלבד, יופנו רק אליו.

**כתבו בתיבת subject: יבש 2 את"ם.**

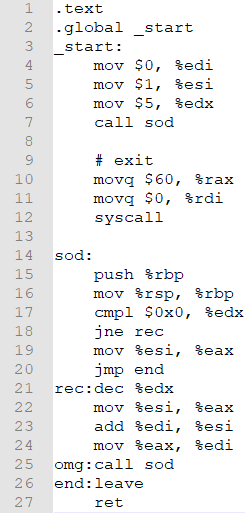
שאלות בעל-פה ייענו על ידי כל מתרגל.

הוראות הגשה:

* לכל שאלה יש לרשום את התשובה במקום המיועד לכך.
* יש לענות על גבי טופס התרגיל ולהגיש אותו באתר הקורס כקובץ PDF.
* על כל יום איחור או חלק ממנו, שאינו בתיאום עם המתרגל האחראי על התרגיל, יורדו 5 נקודות.
* הגשות באיחור יש לשלוח למייל של אחראי התרגיל בצירוף פרטים מלאים של המגישים (שם+ת.ז).
* שאלות הנוגעות לתרגיל יש לשאול דרך הפיאצה בלבד.
* ההגשה בזוגות.

# שאלה 1 (38 נק') – שגרות:

נתונה התוכנית הבאה בשפת אסמבלי:



1. האם הפונקציה עומדת בקונבנציה System v שנלמדה בתרגול? הקיפו בעיגול (או סמנו עם מרקר במחשב): (3 נקודות)

**כן / לא**

1. המירו את הפונקצייה לשפת c על ידי כך שתשלימו את המקומות החסרים בקוד (13 נקודות):

int sod (int num1, int num2, int index){

If (index != 0)

return sod(num2, num1 + num2, index-1)

return num2;

}

1. מה פלט הפונקציה (בהתאם לנתונים שמופיעים בתוכנית). (5 נקודות)

התכנית את מחשבת את מספר פיבונ'צי על פי מספר אינדקס שנתון ברגיסטר edx. במקרה זה יחושב מספר פיבונ'צי החמישי, שהוא 8. הפלט של הפוקנציה \_start היא ציאה מהתכנית עם קוד 0.

1. **הערה: אין צורך לדעת קידוד פקודות בשביל לפתור סעיף זה. אבל יש צורך להבין שפקודה שמורה** **בזיכרון כרצף ביטים בדיוק כמו במקטע הנתונים ואחר כך המעבד מפרש את קידוד הפקודה לפקודה.**

נתונה טבלה שמייצגת את קידוד הפקודות שנמצאות בשורות 22-24

|  |  |  |  |
| --- | --- | --- | --- |
| שורה | 22 | 23 | 24 |
| קידוד | 0xf089 | 0xfe01 | 0xc789 |

החליטו להכניס בין השורה 19 לשורה 20 את הפקודות הבאות:

19.1 mov $omg-2, %rsi

19.2 mov %rsi, 8(%rbp)

נתון שלאחר השינוי הכתובת של שורה 25 (omg: call sod) היא: 0x4005eb והערך שבכתובות זו הוא: 0xe8.

1. הקיפו בעיגול את התשובה הנכונה: (5 נקודות)
2. התוכנית כלל לא תיבנה.
3. התוכנית תסיים את ריצתה אך פלט השגרה עלול להיות שונה מהפלט של התוכנית המקורית
4. התוכנית לא תסיים את ריצתה לעולם.
5. התוכנית תסיים את ריצתה עקב שגיאה במהלך זמן ריצה.
6. מה יהיה הערך של אוגר eax בסוף ריצת השגרה לאחר השינוי? שימו לב במידה ובסעיף הקודם רשמתם שהתוכנית לא מסתיימת עקב לולאה אינסופית רשמו מה יהיה הערך של האוגר אחרי כמה איטרציות בלולאה. במידה ועניתם שהתוכנית תקרוס בזמן ריצה רשמו מה היה הערך של eax ברגע הקריסה. במידה ולדעתכם לא ניתן לדעת את הערך יש לרשום זאת! (5 נקודות)

מספר פיבונ'צי המבוקש יחושב כרגיל, וeax יכיל את המספר 8 שהוא ערך החזרה של הפונקציה. בעת החזרה מן הפונקציה, בעקבות השינוי שתואר, הפונקציה תקפוץ לשורה 24 בקוד המקורי, ולאחר מכן השינוי בקוד יגרום להכנסת הערך 0x4005e9 לeax. בהמשך אותם שורות יקראו שוב ושוב, ואותו ערך ייכנס לאוגר eax.

1. (שימו לב השינוי בסעיף ד' לא תקף עבור סעיף זה) החליטו לשנות את הפקודה שבשורה 26 מleave לפקודה: add $24, %rsp.

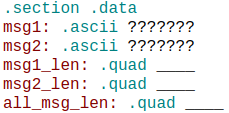
עבור אילו ערכי edx (הערך שedx מקבל בתוכנית הראשית. שורה 5) הפונקציה תחזיר את אותם ערכים בדיוק כמו השגרה המקורית? הסבירו (7 נקודות)

לבדוק

# שאלה 2 (30 נק') – קריאות מערכת:

ג'ואי מרגיש מתוסכל מכך שחבריו חושבים שהוא פחות חכם מהם. לכן, הוא מחליט להרשים אותם בעזרת כתיבת קוד אסמבלי.

1. לפניכם מקטע הנתונים שג'ואי כתב מבלי ערכי הנתונים עצמם:



ג'ואי לא יודע עדיין אילו מחרוזות הוא יכתוב. עליכם להשלים את המקומות הריקים שקשורים לאורכי המחרוזות כך שמשתנה msg1\_len יהיה האורך של msg1, בmsg2\_len יהיה האורך של msg2 ובמשתנה all\_msg\_len יהיה שווה לסכום אורכי המחרוזות msg1 וmsg2. שימו לב עליכם לעשות זאת בצורה כזו שהאורכים יהיו נכונים בעת ריצת התוכנית ללא קשר לאיזה מחרוזות ג'ואי ישים בmsg1 ובmsg2. (3 נקודות)

.section .data

msg1: .ascii ??????

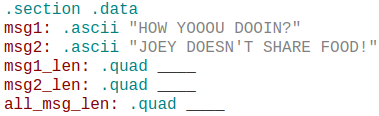
msg2: .ascii ??????

msg1\_len: .quad msg2 – msg1

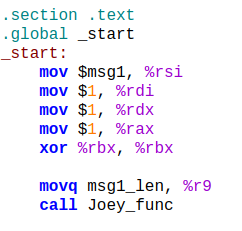
msg2\_len: .quad msg1\_len – msg2

all\_msg\_len: .quad msg1\_len – msg1

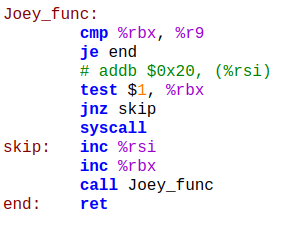
1. כעת נתון מקטע הנתונים שכולל את המחרוזות:



לפניכם נתונה התוכנית שג'ואי כתב:



ומוצגת כאן גם הפונקציה שכתב:



מה יודפס בסיום ריצת הקוד? (שימו לב השורה השלישית בפונקציה נמצאת בהערה ולא רלוונטית לסעיף). (5 נקודות)

השגרה תדפיס של אות במיקום זוגי(כולל 0) של המחרוזת msg1, כלומר יודפס:HWYOUDON

1. כעת מורידים את הסולמית שנמצאת בפונקציה (וכעת הפקודה חלק מהקוד) בנוסף מחליפים את השורה movq msg1\_len, %r9 בשורה: movq all\_msg\_len, %r9.

**הערה: שינויים אלו ילוו אותנו גם בסעיפים הבאים (בסעיפים ד - ו השינויים בסעיף ג עדיין תקפים).**

מה יודפס כעת בסיום ריצת הקוד? (5 נקודות)

כעת השגרה תפעל באופן כמעט זהה של הדפסת המקומות הזוגיים, אך בנוסף תדפיס גם מיד לאחר מכן את msg2. מכיוון שאנו משתמשים בשורה שבהערה, ישתנה ערך הascii של כל אות שתודפס, ולכן יודפס: hwyoudonje@osG@hr@od

1. בזמן שג'ואי אכל בסלון סנדוויץ, חיית המחמד שלו (אפרוח) טיילה על המקלדת והוסיפה את הפקודה:

inc %r9. הפקודה נוספה שורה לפני הקריאה לפונקציה של ג'ואי בתוכנית הראשית.

מה יהיה פלט התוכנית כעת? (2 נקודות)

כעת תודפס אות נוספת למסך, שהיא 4 הביטים הראשונים של msg1\_len ועוד 32, ומכיוון שבmsg1\_len נמצא הערך 16, תודפס בנוסף האות 0.

סך הכל: hwyoudonje@osG@hr@od0

1. חברה טובה של ג'ואי פיבי אמרה לו ששימוש ברגיסטר r9 מביא מזל רע. ג'ואי נלחץ נורא והחליט שיש לבצע שינוי בקוד מבלי לשנות את תוצאות הפעולה של הפונקציה (כלומר הפלט צריך להיות זהה). כיוון ולא ידע איך לשנות את הקוד הוא החליט לבקש את עזרת חבריו.

בסעיף הזה יופיעו העצות של כל החברים. עליכם לרשום ליד כל עצה האם היא לדעתכם תעזור לג'ואי. נמקו **בקצרה(!)** (10 נקודות)

צ'נדלר מציע להחליף את השימוש בr9 בשימוש בr10.

יעזור, r9 לא משמש בקריאה לפונקציות או syscall, לכן יפעל בצורה זהה.

מוניקה מציעה להחליף את השימוש בr9 בשימוש בr11.

לא יעזור, r11 משמש לשמירת אוגר הדגלים בעת שימוש בsyscall, לכן לאחר הקריאה הראשונה הערך יידרס.

פיבי מציעה להחליף את השימוש בr9 בשימוש בrdi.

לא יעזור, rdi המועבר לsyscall הוא הfd אליו צריך להדפיס, ומכיוון שדרסנו את הערך הפונקציה לא תדפיס לפלט הסטדנרטי.

רייצ'ל מציעה להחליף את השימוש בr9 בשימוש בrcx.

לא יעזור, rcx משמש לשמירת rip על ידי הsyscall, כלומר הערך יידרס לאחר הקריאה הראשונה.

רוס מציע להחליף את השימוש בr9 בשימוש בrbp.

לא יעזור, בעת חזרה מפונקציה, המחסנית מקופלת, ושומרת את הערך הישן של rbp מחדש, ולכן הערך יידרס.

1. חבריו של ג'ואי מסבירים לו שהשימוש שלו ברקורסיה מיותר ובזבזני והוא יכול את אותו קוד בדיוק לכתוב בלולאות. ג'ואי מחליט לבצע את השינויים הבאים:

בתוכנית הראשית בשורה שלפני ביצוע הפקודה call ג'ואי מוסיף את הפקודה:

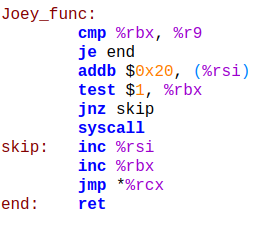
mov $Joey\_func, %rcx

ובתוך הפונקציה ג'ואי מוחק את השורה בה יש שימוש בפקודה call והחליף אותה בפקודה:

jmp \*%rcx

שימו לב שהתווית end נמצאת אחרי פקודה זו.

לצורך הבהרה הפונקציה נראת כך כעת:



כיצד שינוי זה ישפיע על אופן ריצת הפונקציה. מה יודפס אם נריץ את הפונקציה? (5 נקודות)

האות הראשונה תודפס בדיוק כמו קודם (h), אך לאחר הקריאה הראשונה לsyscall הערך שבrcx יידרס ויכיל את השורה הראשונה בskip. לכן כשתתבצע הקפיצה, זה יהיה לשורה הראשונה של skip, ובכך נכנס ללולאה אינסופית.

# שאלה 3 (32 נק') – רמות הרשאה ואוגר הדגלים:

1. הפקודה pushfq דוחפת את הערך של אוגר הדגלים למחסנית. והפקודה popfq מוציאה את אוגר הדגלים מהמחסנית. הסבירו כיצד באמצעות שילוב של שתי פקודות אלו ניתן להדליק את דגל SF. שימו לב במידה והדגל כבר דלוק יש להשאירו דלוק. אין לשנות את שאר הביטים בריגסטר הדגלים. בנוסף, אין לשנות אף רגיסטר שהוא לא rflags, rip, rsp (גם לא באופן זמני). (8 נקודות)

הערה: במידה ובדקתם את עצמכם באמצעות דיבגר וראיתם שנדלק גם דגל TF זה בסדר תלמדו בהמשך מדוע הוא נדלק תוך כדי דיבוג.

pushfq

pop %rsp

or $64,%rsp

push %rsp

popfq

כלומר אנו מדליקים את הביט ה-7 במידה והוא לא דלוק, ומחזירים את הערך לאוגר הדגלים.

1. הולי התחמנית רוצה לאפשר לעצמה גישה ישירה אל התקני הקלט פלט ללא צורך בקריאות מערכת. איזה שינוי **באוגר הדגלים** יכול לעזור להולי במטרתה? (8 נקודות)

הולי יכולה לשנות את הערך של IOPL להיות 3, ואז התנאי לביצוע פעולות קלט/פלט הוא CPL <= IOPL, ומכיוון שהוא בערך המקסימלי, תוכל לבצע פעולות קלט/פלט.

1. הולי מחליטה לנסות את התעלול מסעיף א' רק שבמקום לשנות את SF היא רוצה לשנות את IOPL. להפתעתה, היא לא מצליחה לשנות את הביטים הללו. הסבירו מה ההגיון בכך שהיא לא מצליחה לשנות את IOPL? התייחסולצורך בקריאות מערכת (6 נקודת)

אם היה ניתן לשנות את הערך, היה אפשר לבצע פעולות קלט/פלט ללא תלות במערכת ההפעלה, ולא היה צורך בקריאות מערכת כדי לבצע אותן. כלומר ניתן יהיה לבצע פעולות לא מורשות, ובכך לגרום נזק למחשב או לבצע פעולות זדוניות.

**הערה**: הסעיפים הבאים קשורים לפסיקות מומלץ לענות עליהם לאחר התרגול על פסיקות.

1. וולי החבר המבולבל של הולי מתלבט כיצד ניתן לחסום פסיקות תוכנה לכן הוא שואל את הולי. אילו מבין התשובות הבאות על הולי לענות לו? יש לסמן את האפשרות הנכונה. (5 נקודות)
2. כיבוי דגל IF באוגר הדגלים
3. הדלקת דגל IF באוגר הדגלים
4. שינוי CPL ל00
5. לא ניתן לחסום פסיקות תוכנה.
6. כעת נתון שוולי הצליח להגיע למצב שבו CPL שווה ל0. וולי מעוניין לחסום פסיקות חומרה שאינן מועברות דרך כניסת NMI. כיצד הוא יכול לעשות זאת? (5 נקודות)
7. כיבוי דגל IF באוגר הדגלים
8. הדלקת דגל IF באוגר הדגלים
9. עליו לחבר את הפסיקות לכניסת NMI ואז לכבות את דגל IF
10. לא ניתן לחסום פסיקות חומרה ולכן לא יצליח.