מערכות הפעלה – יבש 4

יניב הולדר

207025297

yaniv.holder@outlook.com

אילון קורנבוים

315677880

eilon.ko@campus.technion.ac.il

שאלה 1

1.

	offset	index1	index2	index3
א.	12	9	9	2

נימוק:

גודל מרחב הזיכרון החדש הוא 2^{40} ביטים. כמות המסגרות במרחב הזיכרון היא גודל מרחב הזיכרון לחלק לגודל מסגרת כלומר $\frac{2^{12}}{2^{40}} = 2^{28}$. כמו כן, עבור כל מסגרת נצטרך עוד 12 ביטים לייצג דגלים והרשאות ולכן סך הכל כדי לייצג מספר מסגרת ודגלים (כלומר כניסה בטבלה) נצטרך 40 ביטים. מכיוון שמספר זה אינו חזקה שלמה של 2 נעגל למעלה ל 64 ביטים, כלומר כל כניסה בטבלה תהיה בגודל 64 ביט.

בנוסף, מראש אנו צריכים לשמור 12 ביט כדי לחשב את ה offset בתוך המסגרת בזיכרון הפיזי כדי להגיע לבית ספציפי.

אם במסגרת יש 4KB נוכל להכניס בתוכה $\frac{4KB}{8B} = 512$ כניסות (כלומר ייצוג למסגרת בזיכרון). כדי לייצג מספר כניסה מתוך 512 כניסות נצטרך $\log_2 512 = 9$ ביטים. בצורה דומה, כמו שהטבלה הובילה למסגרות בזיכרון נצטרך טבלה דומה שתוביל למסגרות הללו (שמובילות למסגרות בזיכרון) לצורך כך נשתמש בעוד 9 ביטים.

מכיוון שהשתמשנו כבר ב12 ביטים ל offset ועוד פעמיים 9 ביטים להליכה בטבלאות נשארו לנו 2 ביטים ל index3.

2. c. 5 בתים

נימוק:

אם לא היינו מעגלים למעלה היינו צריכים 40 ביט כדי לייצג כניסה בטבלת הדפים כפי שחישבנו בסעיף הקודם. לכן מספר הביטים המינימלי האפשרי הוא 40 ביטים, כלומר 5 בתים.

3. ראשית, כאשר אנו נשתמש ב40 ביט לייצוג כניסה בטבלה נוכל להכניס בכל טבלה $\left\lfloor\frac{4KB}{5B}\right\rfloor = 819$ כניסות.
התשובה הנכונה היא $f_1(vpn) = vpn\%819$.

נימוק:
אנו יודעים כי בטבלת הדפים קיימות 819 כניסות ולכן על מנת לבחור את הכניסה ברמה השלישית נשתמש בפעולת המודולו כדי לייצג מתוך הביטים התחתונים ביותר את מספר הכניסה מתוך 819 הכניסות שיש לנו בטבלה.

4. $f_2(vpn) = \left(\frac{vpn}{819}\right)\%819$

<u>נימוק:</u>
בדומה לסעיף הקודם נרצה לקבל מספר מתוך 819 הכניסות שיש לנו בטבלת הדפים, אך הפעם נשתמש בביטים עליונים יותר (פעולת החילוק תסיר את 9 הביטים התחתונים). נייצג מספר זה על ידי פעולת המודולו שתחלץ מספר כניסה בטבלה מתוך הביטים התחתונים מהמספר שנשאר אחרי פעולת החילוק.

5. $f_3 = \frac{\frac{vpn}{819}}{819}$

<u>נימוק:</u>
כעת, נשתמש בביטים העליונים ביותר ונקבל אותם על ידי פעולת החילוק פעמיים (כל פעם תסיר ביטים תחתונים שהשתמשנו בהם על מנת לחשב את הפונקציות הקודמות) מכיוון שהמספר שנישאר איתו קטן מ819 (0 או 1) נוכל בעזרתו לבחור כניסה מהטבלה ברמה הראשונה.

6. d. **טבלאות הדפים של תהליכי משתמש תופסות נפח קטן יותר בזיכרון.**

<u>נימוק:</u>

ההבדל בין המימושים השונים הוא בגודל הכניסה בטבלאות הדפים שונה (אצל שוקי – 5B ואצל עדן 8B) לכן במימוש של שוקי יש יותר כניסות בכל מסגרת (819 לעומת 512). גודל המסגרות והדפים נשארו כפי שהיו.

לכן, טבלאות הדפים של שוקי יתפסו נפח קטן יותר בזיכרון של המשתמש. למשל, עבור תהליך שמשתמש שמחזיק 819 דפים אצל שוקי נצטרך להחזיק רק מסגרת אחת בשלב השלישי (4KB) ואילו עבור עדן נצטרך להחזיר 2 מסגרות בשלב השלישי (8KB). ככל שמספר הדפים של המשתמש יהיה גדול יותר המימוש של שוקי יתפוס פחות זיכרון פיסי.

שאלה 2

חלק 1

1. *Strace* הוא כלי בלינוקס המשמש לאבחון ודיבוג תוכניות. הכלי מקבל פקודה ומריץ אותה כך שהוא עוקב ומדפיס את קריאות המערכת שמבוצעות וכמו כן את הסיגנלים שהתקבלו לאורך כל הריצה. עבור כל קריאת מערכת הוא מדפיס את הארגומנטים שלה ואת ערך החזרה שלה.
2.

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char** argv){

    int x = atoi(argv[1]);
    char* arr = (char*)malloc(x);
    return 0;
}
```

```
student@ubuntu18:~/Desktop/ATAM_HW4/wet$ gcc -std=c99 debugee_simple.c -o debugee_simple
student@ubuntu18:~/Desktop/ATAM_HW4/wet$ strace ./debugee_simple 5
execve("./debugee_simple", ["./debugee_simple", "5"], 0x7ffc131718c8 /* 50 vars */) = 0
brk(NULL)                               = 0x55be1b37c000
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=104452, ...}) = 0
mmap(NULL, 104452, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f52e5a0d000
close(3)                                = 0
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\260\34\2\0\0\0\0\0"..., 832) = 832
fstat(3, {st_mode=S_IFREG|0755, st_size=2030544, ...}) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f52e5a0b000
mmap(NULL, 4131552, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f52e540f000
mprotect(0x7f52e55f6000, 2097152, PROT_NONE) = 0
mmap(0x7f52e57f6000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1e7000) = 0x7f52e57f600
0
mmap(0x7f52e57fc000, 15072, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f52e57fc000
close(3)                                = 0
arch_prctl(ARCH_SET_FS, 0x7f52e5a0c4c0) = 0
mprotect(0x7f52e57f6000, 16384, PROT_READ) = 0
mprotect(0x55be1ac8f000, 4096, PROT_READ) = 0
mprotect(0x7f52e5a27000, 4096, PROT_READ) = 0
munmap(0x7f52e5a0d000, 104452)          = 0
brk(NULL)                               = 0x55be1b37c000
brk(0x55be1b39d000)                     = 0x55be1b39d000
exit_group(0)                           = ?
+++ exited with 0 +++
student@ubuntu18:~/Desktop/ATAM_HW4/wet$
```

.3

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/syscall.h>

int main(int argc, char** argv){

    int x = atoi(argv[1]);
    getpid();
    char* arr = (char*)malloc(x);
    return 0;
}
```

```
student@ubuntu18:~/Desktop/ATAM_HW4/wet$ gcc -std=c99 debugee_simple.c -o debugee_simple
student@ubuntu18:~/Desktop/ATAM_HW4/wet$ strace ./debugee_simple 5
execve("./debugee_simple", ["./debugee_simple", "5"], 0x7ffe8f585ed8 /* 50 vars */) = 0
brk(NULL)                               = 0x5633ec2c9000
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=104452, ...}) = 0
mmap(NULL, 104452, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f4fa5564000
close(3)                                = 0
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\260\34\2\0\0\0\0\0"..., 832) = 832
fstat(3, {st_mode=S_IFREG|0755, st_size=2030544, ...}) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f4fa5562000
mmap(NULL, 4131552, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f4fa4f66000
mprotect(0x7f4fa514d000, 2097152, PROT_NONE) = 0
mmap(0x7f4fa534d000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1e7000) = 0x7f4fa534d00
0
mmap(0x7f4fa5353000, 15072, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f4fa5353000
close(3)                                = 0
arch_prctl(ARCH_SET_FS, 0x7f4fa55634c0) = 0
mprotect(0x7f4fa534d000, 16384, PROT_READ) = 0
mprotect(0x5633eaaad000, 4096, PROT_READ) = 0
mprotect(0x7f4fa557e000, 4096, PROT_READ) = 0
munmap(0x7f4fa5564000, 104452)          = 0
getpid()                                = 48933
brk(NULL)                               = 0x5633ec2c9000
brk(0x5633ec2ea000)                     = 0x5633ec2ea000
exit_group(0)                           = ?
+++ exited with 0 +++
```

הוספו קריאת מערכת ()getpid לפני ביצוע ה malloc מכיוון שאנו יודעים שקריאת מערכת זו אינה נקראת במהלך ה malloc ולכן היא נותנת לנו מידע לגבי מתי מתחיל הmalloc.

חלק 2:

1. שתי הקריאות הן:
   1. brk()
   2. mmap()

בעזרת הקוד המצורף קיבלנו את הפלט שמוכיח

את תשובתנו:

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/syscall.h>

int main(int argc, char** argv){

    int x = atoi(argv[1]);
    int y = atoi(argv[2]);
    getpid();
    char* arr = (char*)malloc(x);
    getpid();
    char* arr2 = (char*)malloc(y);
    return 0;
}
```

```
student@ubuntu18:~/Desktop/ATAM_HW4/wet$ strace ./debugee_simple 10 10000000
execve("./debugee_simple", ["./debugee_simple", "10", "10000000"], 0x7fff0f2dcde0 /* 50 vars */) = 0
brk(NULL)                               = 0x55756f75f000
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=104452, ...}) = 0
mmap(NULL, 104452, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f6d1cbd0000
close(3)                                = 0
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\260\34\2\0\0\0\0\0"..., 832) = 832
fstat(3, {st_mode=S_IFREG|0755, st_size=2030544, ...}) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f6d1cbce000
mmap(NULL, 4131552, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f6d1c5d2000
mprotect(0x7f6d1c7b9000, 2097152, PROT_NONE) = 0
mmap(0x7f6d1c9b9000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1e7000) = 0x7f6d1c9b900
0
mmap(0x7f6d1c9bf000, 15072, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f6d1c9bf000
close(3)                                = 0
arch_prctl(ARCH_SET_FS, 0x7f6d1cbcf4c0) = 0
mprotect(0x7f6d1c9b9000, 16384, PROT_READ) = 0
mprotect(0x55756e4e3000, 4096, PROT_READ) = 0
mprotect(0x7f6d1cbea000, 4096, PROT_READ) = 0
munmap(0x7f6d1cbd0000, 104452)          = 0
getpid()                                = 49715
brk(NULL)                               = 0x55756f75f000
brk(0x55756f780000)                     = 0x55756f780000
getpid()                                = 49715
mmap(NULL, 10002432, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f6d1bc48000
exit_group(0)                           = ?
+++ exited with 0 +++
```

2. ה threshold ש malloc משתמש בו הוא 131072B כלומר בערך 131KB. נוכיח זאת על ידי 2 קריאות ל malloc פעם אחת עם מספר קצת יותר קטן ופעם שנייה עם המספר ה threshold. בנוסף, הוספנו לתוכנית קריאת malloc ראשונה כדי לפתור את הבעיה שהוסברה בסעיף

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/syscall.h>

int main(int argc, char** argv){

    malloc(10); // to ignore first malloc affects
    int x = atoi(argv[1]);
    int y = atoi(argv[2]);
    getpid();
    char* arr = (char*)malloc(x);
    getpid();
    char* arr2 = (char*)malloc(y);
    return 0;
}

```

```
student@ubuntu18:~/Desktop/ATAM_HW4/wet$ strace ./debugee_simple 131072 131073
execve("./debugee_simple", ["./debugee_simple", "131072", "131073"], 0x7ffc9d744840 /* 50 vars */) = 0
brk(NULL)                               = 0x555a7150f000
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=104452, ...}) = 0
mmap(NULL, 104452, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f807b1fd000
close(3)                                = 0
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\260\34\2\0\0\0\0\0"..., 832) = 832
fstat(3, {st_mode=S_IFREG|0755, st_size=2030544, ...}) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f807b1fb000
mmap(NULL, 4131552, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f807abff000
mprotect(0x7f807ade6000, 2097152, PROT_NONE) = 0
mmap(0x7f807afe6000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1e7000) = 0x7f807afe600
0
mmap(0x7f807afec000, 15072, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f807afec000
close(3)                                = 0
arch_prctl(ARCH_SET_FS, 0x7f807b1fc4c0) = 0
mprotect(0x7f807afe6000, 16384, PROT_READ) = 0
mprotect(0x555a71282000, 4096, PROT_READ) = 0
mprotect(0x7f807b217000, 4096, PROT_READ) = 0
munmap(0x7f807b1fd000, 104452)          = 0
getpid()                                = 50361
brk(NULL)                               = 0x555a7150f000
brk(0x555a71530000)                     = 0x555a71530000
getpid()                                = 50361
mmap(NULL, 135168, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f807b1da000
exit_group(0)                           = ?
+++ exited with 0 +++
```