

Operating Systems – 234123

Homework Exercise 2 – Dry

Teaching Assistant in charge:

Alex Zhybirov

Assignment Subjects & Relevant Course material

Modules, Scheduling (Lectures 4--5, Tutorials 4--5)

Submission Format

1. Only typed submissions in PDF format will be accepted. Scanned handwritten submissions will not be graded.
2. The dry part submission must contain a single PDF file named with your student IDs –
123456789_300200100.pdf
3. The submission should contain the following:
 - a. The first page should contain the details about the submitters - Name, ID number and email address.
 - b. Your answers to the dry part questions.
4. Submission is done electronically via the course website, in the **HW2 Dry** submission box.

Grading

1. All question answers must be supplied with a full explanation. Most of the weight of your grade sits on your explanation and evident effort, and not on the absolute correctness of your answer.
2. Remember – your goal is to communicate. Full credit will be given only to correct solutions which are clearly described. Convolved and obtuse descriptions will receive low marks.

Questions & Answers

- The Q&A for the exercise will take place at a public forum Piazza **only**. Please **DO NOT** send questions to the private email addresses of the TAs.
- Critical updates about the HW will be published in **pinned** notes in the piazza forum. These notes are mandatory and it is your responsibility to be updated.

A number of guidelines to use the forum:

- Read previous Q&A carefully before asking the question; repeated questions will probably go without answers
- Be polite, remember that course staff does this as a service for the students
- You're not allowed to post any kind of solution and/or source code in the forum as a hint for other students; In case you feel that you have to discuss such a matter, please come to the reception hour
- When posting questions regarding **hw2-dry**, put them in the **hw2-dry** folder.

Late Days

- Only the TA in charge of the assignment can authorize postponements. In case you need a postponement, please fill out the attached form: <https://forms.gle/C4aU2Gv3QDA7xohS8>

מטלה זו מנוסחת בלשון זכר, אך מיועדת לשני המינים כאחד

חלק 1 - שאלות בנושא התרגיל הרטוב (30 נק')

מומלץ לקרוא את הסעיפים בחלק זה לפני העבודה על התרגיל הרטוב, ולענות עליהם בהדרגה תוך כדי פתרון התרגיל הרטוב.

1. מה עושה פקודת `yes` בלינוקס? מה הארגומנטים שהיא מקבלת? (3 נק')

היעזרו בman, ולאחר מכן השתמשו בפקודה `bash` שלכם כדי לבדוק.

2. מדוע השתמשנו בפקודת `yes` עם מחרוזת ריקה במהלך הפקודה הבאה? (3 נק')

```
>> yes '' | make oldconfig
```

נסו להריץ את הפקודה `make oldconfig` לבדה והסבירו מה הבעיה בכך.

3. מה משמעות הפרמטר `GRUB_TIMEOUT` בקובץ ההגדרות של GRUB? (3 נק')

```
GRUB_TIMEOUT=5
```

הסבירו מה היתרונות ומה החסרונות בהגדלת הפרמטר `GRUB_TIMEOUT`.

4. מדוע הפונקציה `run_init_process()` אשר נמצאת בקובץ `init/main.c` בקוד הגרעין קוראת לפונקציה `do_execve()` במקום לקרוא את המערכת `execve()`? (3 נק')

```
944 static int run_init_process(const char *init_filename)
945 {
946     argv_init[0] = init_filename;
947     return do_execve(getname_kernel(init_filename),
948                     (const char __user *const __user *)argv_init,
949                     (const char __user *const __user *)envp_init);
950 }
```

נסו להחליף את הפונקציות זו בזו ובדקו האם הגרעין מתקמפל.

5. מה עושה קריאת המערכת `syscall()`? כמה ארגומנטים היא מקבלת ומה תפקידם? באיזו ספרייה ממומשת קריאת המערכת `syscall()`? היעזרו בman בתשובתכם. (3 נק')

6. מה מדפיס הקוד הבא? כתבו קוד ברור יותר השקול לקוד שלפניכם. (3 נק')

```
int main() {
    syscall(62, syscall(39), 9);
    return 0;
}
```

רמז: התבוננו בקובץ `arch/x86/entry/syscalls/syscall_64.tbl` בקוד הגרעין.

7. ביצענו את השורה הבאה. ציינו 2 ערכים אפשריים של `s`, והסבירו מה משמעות כל ערך. (3 נק')

```
int s = capable(CAP_SYS_ADMIN);
```

8. נרצה להרחיב את קריאת המערכת `set_ban` כך שתוכל לחסום עוד קריאות מערכת. החתימה

החדשה:

```
long set_ban(int getpid_ban, int pipe_ban, int kill_ban,  
            int a, int b, int...)
```

כמה קריאות מערכת שונות סך הכל נוכל לחסום בדרך זו? אם קיימת הגבלה על מספר זה, הסבירו בפירוט ממה היא נובעת. (3 נק')

9. התבוננו בתוכנית הבדיקה `test1.cxx` שסופקה לכם והסבירו במילים פשוטות מה היא בודקת

(6 נק'):

```
int main() {  
    long x = get_ban('g');  
    cout << "getpid() should not be banned initially (0) -> " <<  
x << endl;  
    assert(x == 0);  
  
    x = set_ban(1,1,0);  
    cout << "set_ban(110) returned: " << x;  
    if(x == -1) cout << " (did you use sudo?)";  
    cout << endl;  
  
    assert(x == 0);  
    x = get_ban('g');  
    cout << "getpid() should be banned (1) -> " << x << endl;  
    assert(x == 1);  
  
    x = flip_ban_branch(1, 'k');  
    cout << "set_flip_branch returned: " << x << endl;  
    assert(x >= 0);  
  
    x = check_ban(pid_t(getppid()), 'k');  
    cout << "Our parent should be banned from using kill() (1) ->  
" << x << endl;  
    assert(x == 1);  
  
    x = getpid();  
    cout << "getpid() returned: " << x << endl;  
  
    ;cout << "===== SUCCESS =====" << endl  
    ;return 0  
}
```

חלק 2 - זימון תהליכים (50 נק')

במחשב עם מעבד בן 128 ליבות רץ תהליך אנטי וירוס ייעודי (בשאלה נקרא לו AV) שתפקידו לנהל רישום של כל התהליכים שרצו אי פעם במחשב. AV הינו תהליך משתמש רגיל עם שתי תכונות מיוחדות: (i) הנחמדות שלו היא 15-. (ii) מטעמי בטיחות הוא חייב להיות runnable כל הזמן. משום כך, גרעין הלינוקס של המחשב עודכן להתריע מיד אם AV מפסיק להיות runnable מאיזושהי סיבה, ע"י שליחת הודעה לאחראי הביטחון של החברה. בלתי אפשרי לחסום את ההתרעות של הגרעין ולזייף או לשנות את הרישום ש-AV מנהל. מובטח לנו שכאשר AV רץ הוא מודע לכל התהליכים הקיימים במערכת באותו הרגע, בכדי לנהל רישום מדויק.

שירה היא עובדת חברה בצוות ה-IT בעלת הרשאות אדמיניסטרטור על המחשב הנ"ל. היא גם סוכנת עוינת שרוצה להריץ קובץ זדוני בשם sauron.exe בסוד, מבלי שהרצה זו תופיע ברישומיו של AV או שתשלח הודעה לאחראי הביטחון. על המערכת להמשיך לפעול באופן תקין בתום ההרצה של הקובץ הזדוני. בשאלה זו תעזרו לשירה לבצע את זממה ולהריץ את sauron.exe על מחשב החברה. הערות:

1. שימו לב ששירה לא יודעת את שם התהליך AV או את ה-PID שלו.
2. לא תוכלו להשתמש במודולים כדי לפתור את השאלה.
3. מותר להשתמש בקריאת המערכת הבאה, או בכל קריאת מערכת אחרת שלמדתם עד כה בקורס:

```
// The process with ID 'pid' will run on only the CPU core with ID  
// 'core_id'.  
int sched_setaffinity(int pid, int core_id /*between 0 to 127*/);
```

(הערה - זוהי גרסה מופשטת של קריאת המערכת האמיתית)

1. עם איזה ערך נחמדות נריץ את sauron.exe כדי שלא יתגלה ע"י AV? (5 נק')

- a. nice > -20
- b. כל ערך nice יעבוד
- c. nice = -20
- d. שום ערך nice לא יעבוד
- e. nice = 0

נימוק:

2. באילו מהמקרים הבאים תהליך שרץ במדיניות זימון SCHED_FIFO מוותר על המעבד? (5 נק')

- a. התהליך יוצא לתור המתנה
- b. התהליך קורא getpid()
- c. לתהליך נגמר הקוונטום
- d. הגיע תהליך SCHED_FIFO נוסף עם עדיפות זהה
- e. אחד מבניו מסיים לרוץ

נימוק:

3. חסימת הליבות (10 נק')

שירה התחילה לרשום את קוד הפריצה אבל היא צריכה את עזרתכם כדי להשלים אותו.

השלימו את הקוד כך שיחסום את כלל הליבות במערכת – חוץ מליבה 0 (עליה שירה תריץ את sauron.exe).

```
sched_setaffinity(getpid(), 0);
struct sched_param param;
param.sched_priority = ____;
sched_setscheduler(getpid(), _____, &param);

pid_t p;
pid_t* children_pids = malloc(sizeof(pid_t)*(____));
for (int i=1;i<____;i++) {
    p = fork();
    if (p==0) {
        _____;
        _____;
    }
    children_pids[i-1] = p; // we save the pids of all children for later
}
```

4. הרצת sauron.exe ומחיקת עקבות (10 נק')

בזמן שהשלמתם את הקוד, שירה עברה על קוד האסמבלי של sauron.exe וגילתה שהוא לא יוצא להמתנות, ולא קורא ל(sched_yield(). השלימו את הקוד הבא כך שsauron.exe ירוץ מבלי להתגלות ע"י AV.

זכרו – עליכם להשאיר את המערכת במצב תקין, ולהשאיר מינימום עקבות במערכת. לרשותכם כל המשתנים ששמרתם עד כה בתרגיל (לדוגמא children_pids).

מכאן והלאה יהיו יותר שורות ריקות מהנדרש.

```
pid_t sauron_pid = fork();
if (sauron_pid==0) {
    _____
    _____
    _____
}
else {
    _____
    _____
}
return 0; // sauron executed on the system undetected
```

5. טיפול בהמתנות (20 נק')

זמן קצר לאחר שסיימתם להריץ טסטים על הקוד שלכם, המפעילים של שירה מסרו לכם גרסא חדשה של sauron.exe. שירה בדקה את הגרסא ולצערכם הרב היא מכילה יציאות להמתנה. כעת בכל הסימולציות של ששירה מריצה AV מצליח לרשום את sauron.exe. למה הקוד שכתבתם הפסיק לעבוד?

שעת השין מתקרבת, ועליכם לסיים את המלאכה. ממשו מחדש את הרצת sauron.exe, הפעם בצורה שמונעת מAV לרוץ על מעבד במקרה בו sauron.exe יוצא להמתנה.

```
pid_t sauron_pid = fork();
if (sauron_pid==0) {
    _____
    _____
    _____
}
else { // should hold the core while sauron.exe is not actively running
    _____
    _____
    _____
}
return 0; // sauron executed on the system undetected
```

שירה מורידה את הקוד לUSB רגע אחרי שסיימתם לכתוב אותו. היא נוסעת להריץ אותו על מחשב החברה מרובה הליבות בתקווה שהוא יבצע את המשימה בסודיות הנדרשת. לכם נשאר לחכות ולקוות שהפעם הבאה שתראו אותה לא תהיה בחדשות, אלא בקריביים.

העשרה

לפתרון שלכם חסרים כמה פרטים קטנים (מחוץ לסקופ הקורס), שאם תשלימו אותם תוכלו לנסות את הקוד הנ"ל על המחשב/מכונה וירטואלית.

1. בפועל, לינוקס לא מאפשרת לתהליכי RT להשתלט לגמרי על ליבות (כדי למנוע מצב בו המערכת נתקעת). לינוקס מאפשרת לתהליכי RT מקסימום של 95% מזמן המעבד עליו הם רצים. ניתן לשנות את הפרמטר הזה, ואז הקוד שכתבתם יעבוד כמצופה. הפרמטר נמצא ב"/proc/sys/kernel/sched_rt_runtime_us", ומכיל את הזמן במיקרו שניות שתהליך RT יקבל, מתוך 1,000,000. שנו אותו ל-1 כדי לאפשר לתהליכי RT שליטה מוחלטת על הליבה שלהם. לא לשכוח לשנות בחזרה!

2. השתמשו בפונקציה sysconf(_SC_NPROCESSORS_ONLN) כדי לקבל את כמות המעבדים הפעילים על המחשב שלכם.

3. קראו באינטרנט/man על הפונקציה sched_setaffinity, ההגדרה שלה במציאות שונה מהגרסא המופשטת שהשתמשתם בה בשאלה.

4. שימו לב שהרצה לא מושכלת של הקוד עלולה לגרום למחשב שלכם להתקע (למשל, אם sauron.exe הוא לולאה אינסופית).

חלק 3 - מודולים (20 נק' +10 בonus)

1. ממשו דרייבר להתקן פיקטיבי, בעל המפרט הבא:

(א) בפתיחת (open) ההתקן יודפס "DHW2 DEV READY\n"

(11) בעת קריאה מההתקן, יוחזר 'f'.

(||) בעת כתיבה להתקן, נדווח שלא נותר מקום בהתקן.

(IV) במידה ולהתקן 8 minor יוחזר בקריאה '8' במקום 'f'

הערות:

- ממשו רק את `f_ops` + הפונקציות הנלוות לו (`open, read, write, ...`), ניתן להשמיט חלקים כמו `include, module_init` וכד'.
- במהלך המימוש תדרשו לכתוב מידע לחוצצים במרחב המשתמש, ממרחב הגרעין. דבר זה אינו טריוויאלי כפי שלמד בהמשך הקורס, וניתן לעשות זאת ע"י שימוש בפונקציה הבאה:
- ```
void put_user(char data, char* usr_ptr);
```

This image shows a blank sheet of white paper with horizontal ruling lines. The lines are evenly spaced and extend across the width of the page. There are no margins, text, or other markings on the paper.

## **בנוס (10 נק')**

2. נניח שבחלק הקודם יכולתם להשתמש במודולים בפתרון שלכם.

הסבירו כיצד הייתם משתמשים במודולים כדי להריץ את sauron.exe מבלי להתפס (לא חייב לכתוב קוד).  
ניתן להניח שאין הרבה תהליכים בעלי נחמדות 15- על מחשב היעד, ולשנות את דרך פעולתם זמנית לא יזיק למערכת.

---

---

---

---

---

---

בשמחה תמיד,  
סגל מערכות הפעלה