



TEL AVIV UNIVERSITY

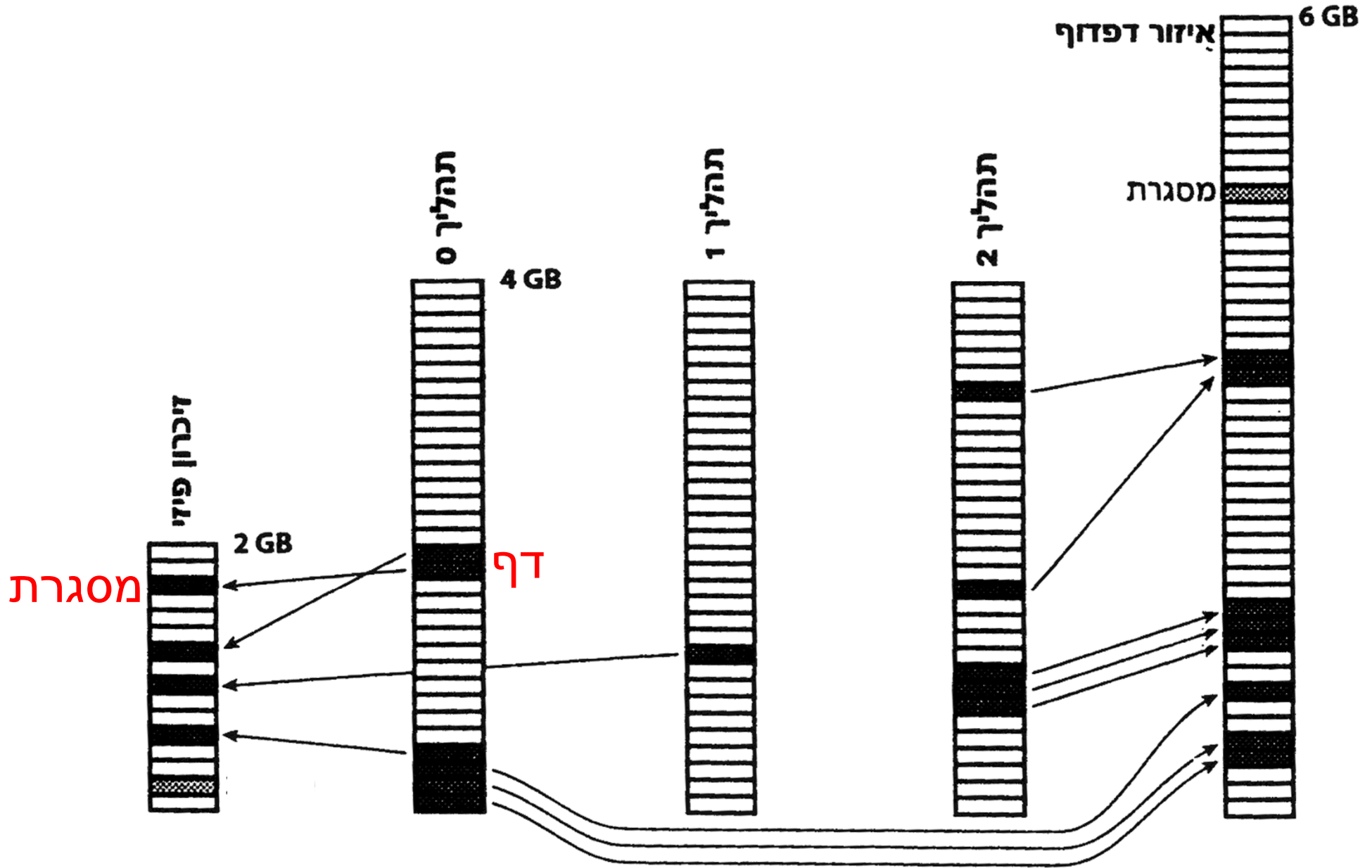
מערכות הפעלה

ערן טרומר
סמסטר א' תשע"ב

הרצאה 5

זיכרון וירטואלי (המשך)

זיכרון וירטואלי



הגנה על זיכרון

- ❖ כל הפקודות שעוסקות ב-TLB ובכתובת של טבלת הדפים מותרות רק במצב מיוחס
- ❖ אם טבלת הדפים הפוכה, מערכת ההפעלה מודיעה למעבד מה המזהה של התהליך שרץ כרגע על מנת להשתמש רק במיפויים שלו
- ❖ אי לכך, המעבד משתמש רק במיפויים שמערכת ההפעלה יצרה עבור התהליך שרץ כרגע; תהליך לא יכול להתייחס לכתובות פיזיות כלל, ולא יכול להתייחס למסגרות שאינן ממופות לדפים שלו

הגנה על זיכרון – סיביות גישה

❖ כל כניסה בטבלת הדפים כוללת מספר סיביות שמתארות הרשאות גישה

- האם מותרת גישה לדף במצב משתמש או רק במצב מיוחס?
- האם מותרות קריאה, כתיבה, ושימוש בתוכן כפקודות מכונה?

❖ היכולת לאסור סוגי גישה מסוימים, יחד עם העובדה שהדפים הראשונים לעולם אינם ממופים, עוזרת לגלות שגיאות בתוכניות (התייחסות דרך מצביע null, שכתוב קוד, ...)

❖ היכולת לאסור גישה במצב משתמש:

- מאפשרת למערכת ההפעלה למפות את הקוד ומבני הנתונים שלה לכל התהליכים בלי לאפשר לתוכנית שהתהליך מריץ לגשת לזיכרון הזה
- מערכת ההפעלה ממופה בדרך כלל לכל התהליכים באותו מקום (לדוגמה, 3GB למשתמש, 1GB ללליבה)
- ממזער תקורה בעת קריאות מערכת ופסיקות

חריגי דף

(page fault / segmentation fault /
segmentation exception)

- ❖ המעבד מזהה גישות לדפים לא ממופים, לדפים שממופים למסגרת בדיסק, ולדפים שהרשאותיהם אינן מתירות את סוג הגישה או אינן מתירות גישה במצב משתמש
- ❖ המעבד מייצר חריג (exception) בשם חריג דף שגורם להפעלת שגרה של מערכת ההפעלה, בדומה לטיפול בפסיקות
- ❖ השגרה של מערכת ההפעלה מזהה את הסיבה לחריג ומגיבה
- ❖ אם צריך לקרוא מסגרת מהדיסק, מערכת ההפעלה משעה את התהליך, מביאה את המסגרת, מעדכנת את טבלת הדפים, וחוזרת לתהליך; הפקודה שגרמה לחריג תרוץ שוב
- ❖ רוב המקרים האחרים גורמים להעפת התהליך או הפעלת שגרה מיוחדת של התהליך (שגרה לטיפול באיתות, signal)

סיבות לחריגי דף והטיפול בהם

- ❖ דף שכלל אינו מוקצה לתהליך
- ❖ דף מוקצה בתהליך אך אינו ממופה למסגרת פיזית
- ❖ הרשאות לא מספיקות לסוג הגישה הנדרש (קריאה, כתיבה, הרצה)
- ❖ במצב מיוחד: דף של התהליך שאינו מוקצה, מוקצה אך אינו ממופה, או מוקצה וממופה אך ללא הרשאות לסוג הגישה (בדרך כלל החריג הוא תוצאה של גישה של מערכת ההפעלה לארגומנט של קריאת מערכת שכתובתו הועברה לקריאה)
- ❖ במצב מיוחד בגלל דף של מערכת ההפעלה: שגיאה בתהליך (ארגומנט לא נכון לקריאת מערכת) או תקלה במערכת ההפעלה

התייחסות לכתובות פיזיות

- ❖ מערכת ההפעלה צריכה להתייחס לכתובות פיזיות ולהקצות מערכים בזיכרון פיזי רצוף בשביל טבלאות דפים וזיכרון שגם בקרים ניגשים אליו, כמו חוצצים שבקר DMA מעביר מהם/אליהם נתונים
- ❖ דרך פשוטה להתייחס לכתובות פיזיות היא למפות את כל הזיכרון הפיזי או חלקו לזיכרון הוירטואלי של כל התהליכים
- ❖ דוגמה: מיפוי 2 GB של זיכרון פיזי לחצי העליון של מרחב כתובות של 32 סיביות: כתובת פיזית X ממופה לכתובת וירטואלית $X+2^{31}$
- ❖ רשומות מיפוי לדפים גדולים (למשל 4 MB) מייעלות את זה

פינוי מסגרות

- ❖ מערכת ההפעלה צריכה מאגר של מסגרות פנויות שניתן להקצות במהירות עבור תהליכים או עבור מערכת ההפעלה עצמה.
- ❖ מסגרות "נקיות" (שיש עותק עדכני שלהן בדיסק) ניתן לפנות במהירות
- ❖ מסגרות "מלוכלכות", שאין עותק עדכני שלהן בדיסק משום ששוננו מאז שנקראו או שמעולם לא נכתבו לדיסק, יש צורך לשמור בדיסק לפני פינויין
- ❖ כניסות בטבלת הדפים וה-TLB כוללות סיבית ניקיון/לכלוך

מנגנונים לפינוי מסגרות

❖ המנגנון משתמש בשלושה מאגרים (רשימות) של מסגרות

1. מסגרות מלוכלכות מועמדות לניקוי

2. מסגרות נקיות מועמדות לפינוי

3. מסגרות פנויות וריקות

❖ תהליך מיוחד מתעורר מדי פעם, מוצא מסגרות ראויות לפינוי,

מוחק אותן מטבלת הדפים המתאימה, ומעביר לרשימה

המתאימה (1 או 2)

❖ תהליך אחר שגם הוא מתעורר מדי פעם כותב לדיסק מסגרות

מלוכלכות מועמדות לפינוי (1) ומעביר אותן לרשימת הנקיות (2)

❖ תהליך שלישי (פחות חיוני) מאפס מדי פעם מסגרות נקיות (2)

ומעביר אותן למאגר הפנויות (3)

הצלת מסגרות מועמדות לפינוי

- ❖ בזמן טיפול בחריג דף מערכת ההפעלה בודקת האם הדף (שאינו ממופה) שמור במסגרת באחד משני ממאגרי המועמדות לפינוי (1 או 2)
- ❖ במצב כזה מערכת ההפעלה מוציאה את המסגרת ממאגר המועמדות לפינוי ומשחזרת את המיפוי בטבלת הדפים, ללא גישה לדיסק
- ❖ זו הסיבה שכדאי להחזיק מאגר גדול של מועמדות נקיות אבל לא לפנות אותן

קביעת קצב הפינוי

- ❖ התהליכים המיוחדים ומפנים כמות מסגרות שנקבעת על פי מצב המערכת
- ❖ כאשר יש מעט מסגרות נקיות/פנויות, התהליכים יפנו מספר גדול כל אימת שהם מתעוררים
- ❖ כאשר יש הרבה מסגרות נקיות, התהליכים יפנו מעט או שלא יתעוררו כלל
- ❖ הפרמטרים שקובעים את קצב הפינוי תלויים בגודל הזיכרון הפיזי והם ניתנים לכיוון

מדיניות לבחירת קורבנות לפינוי

- ❖ מיטבי: מספר חריגי הדף ממוזער אם בכל חריג דף מפנים את המסגרת שהשימוש הבא בה רחוק ביותר
- ❖ לא מעשי כי מערכת ההפעלה אינה יודעת לאיזה מסגרות ייגשו בעתיד וגם משום שרצוי להחזיק מאגר של פנויות
- ❖ מיטבי במודל מתמטי שאינו מתיר הצצה לעתיד: least recently used (LRU), מפנים את המסגרת שהשימוש האחרון בה היה רחוק ביותר בעבר
- ❖ מדיניות LRU פועלת היטב בפועל משום שההיסטוריה חוזרת
 - שוב: מקומיות גישה
- ❖ קשה למימוש כי דרושות חותמות זמן שימוש ב-TLB ובטבלת הדפים וצריך למצוא את הישנה ביותר בכל טבלאות הדפים

קירובים ל-LRU

- ❖ מעבדים אינם תומכים בחותמות זמן שימוש מדויקות למסגרות
- ❖ מעבדים כן מממשים חותמת שימוש בת סיבית אחת ("accessed" bit) שמודלקת אוטומטית בכל שימוש; צריך לאפס אותה מפורשות
- ❖ כיבוי של הסיבית הזו מדי פעם בכל מיפוי ובחירה של מועמדות שהסיבית שלהן עדיין כבויה מבטיחה שלא נעשה בהן שימוש לאחרונה
- ❖ על ידי שמירת הסיבית הזו באוגר הזזה מדי פעם ניתן לקבל קירוב מדויק יותר ל-LRU
- ❖ ניתן לסמלץ סיבית כזו במעבד שאינו תומך בה בעזרת מנגנון חריגי הדף

מדיניות פינני דו-שלבית

- ❖ במערכות עם טבלת דפים לכל תהליך (היררכית למשל), קשה לסרוק את כל המסגרות על מנת לחפש כאלה שלא היו בשימוש לאחרונה
- ❖ מערכות הפעלה בוחרות מסגרות לפינני בשני שלבים: תהליך שמסגרות שלו יפונו, ואז מסגרות ספציפיות
- ❖ מדיניות דו-שלבית כזו גם מאפשרת לממש עקרונות של הגינות בין תהליכים, למשל לתגמל תהליכים שמשתמשים במעט מסגרות

בחירת תהליך בפינוי מסגרות

- ❖ בלינוקס: מדיניות פשוטה, התהליך שגרם למספר חריגי הדף הקטן ביותר בזמן האחרון
- ❖ גרסאות ישנות יותר של לינוקס: התהליך עם מספר המסגרות הגדול ביותר
- ❖ חלונות: מדיניות `working sets`:
 - מערכת ההפעלה משערכת את מספר המסגרות שכל תהליך זקוק להן, כתלות בקצב חריגי הדף שלו ותוך כיבוד חסמים עליון ותחתון
 - התהליך שייבחר לפינוי מסגרות הוא התהליך שחורג ממספר המסגרות שהוא "זכאי" להן במידה הגדולה ביותר
 - שערך חוזר מדי פעם

אזורי דפדוף

❖ מסגרות שאין להן מקום בזיכרון הראשי מועברות לאזור דפדוף בדיסק

❖ אזור הדפדוף יכול להיות קובץ רגיל (טיפוסי בחלונות), מחיצה (טיפוסי בלינוקס), או אפילו מספר קבצים, מחיצות ודיסקים שלמים (אפשרי בלינוקס)

❖ בלינוקס ניתן להגדיר עדיפות לכל אזור דפדוף, ולהוסיף ולהסיר אזורים בזמן שהמערכת פועלת

```
$ /sbin/swapon -s
```

Filename	Type	Size	Used	Priority
/dev/sda1	partition	17832112	65536	-1
/tmp/swapfile	file	1048568	0	-2

מיפוי קבצים לזיכרון

- ❖ בכל מערכות ההפעלה ניתן להקצות בלוק של זיכרון וירטואלי שממופה לקובץ מסוים
- ❖ מאפשר לקרוא קובץ פשוט על ידי מיפוי וגישה לזיכרון הממופה
- ❖ במיפוי **משותף** כתיבה לזיכרון הממופה משנה את תוכן הקובץ (מאפשר להעביר מידע בין תהליכים)
- ❖ במיפוי **פרטי** כתיבה לזיכרון יוצרת עותק פרטי של המסגרת, ואם יהיה צריך לפנות אותה היא תפונה לאזור דפדוף, לא חזרה לקובץ הממופה
- ❖ מיפוי קבצים משמש לטעינה של קוד מכונה של תוכניות וספריות שגרות לזיכרון (DLL, Shared Objects)
- ❖ מיפוי משותף של תוכניות וספריות חוסך מסגרות פיזיות

דוגמה למיפוי זיכרון בלינוקס

```
$ cat /proc/27394/maps
```

```
00400000-00404000      r-xp 00000000 fd:01 12931  /usr/bin/xeyes
02719000-0275b000      rw-p 00000000 00:00 0      [heap]
3b6f40000-3b6f415000    r-xp 00000000 fd:01 6767  /lib64/libgcc.so.1
3b6fc0000-3b6fd39000    r-xp 00000000 fd:01 52431 /usr/lib64/libX11.so
3e5d20000-3e5d38f000    r-xp 00000000 fd:01 5914  /lib64/libc-2.14.so
3e5d58f000-3e5d593000    r--p 0018f000 fd:01 5914  /lib64/libc-2.14.so
3e5d593000-3e5d594000    rw-p 00193000 fd:01 5914  /lib64/libc-2.14.so
3e5d60000-3e5d683000    r-xp 00000000 fd:01 5921  /lib64/libm-2.14.so
7fff1c7d0000-7fff1c7f1000  rw-p 00000000 00:00 0      [stack]
7fff1c7ff000-7fff1c800000  r-xp 00000000 00:00 0      [vdso]
ffffffffffff600000-ffffffffffff601000  r-xp 00000000 00:00 0      [vsyscall]
```

address

perm offset

dev

inode

path

שקף בונוס: הדגמת קריאות מערכת

❖ מערכת ההפעלה מאפשרת לנטר את קריאות המערכת שמבצע תהליך

❖ בלינוקס, פקודת `strace`:

```
$ strace cat inputfile
execve("/usr/bin/cat", ["/usr/bin/cat", "inputfile"], ...) = 0
open("inputfile", O_RDONLY) = 3
read(3, "text-inside-inputfile\n", 4096) = 4
write(1, "text-inside-inputfile\n", 4) = 4
read(3, "", 4096) = 0
close(3) = 0
_exit(0) = ?
```

(במציאות, הפלט קצת יותר מבולגן בגלל טעינת ספריות קוד *.so)

❖ שימושי מאד כדי להבנת התנהגות ובעיות של תוכניות כ-"קופסה שחורה" ללא קוד מקור. נסו ותהנו!