

חקר מערכת הפעלה ורכיבי המחשב

+ כלים שימושיים מתוך SysinternalsSuite
המשקפים מבט עמוק אל מאחורי הקלעים של
מערכת ההפעלה ורכיבי המחשב

רון עובדיה

חקר מערכת ההפעלה ורכיבי המחשב

נכתב אפריל 2021

כתיבה + עריכה : רון עובדיה

- ❖ קרדיטים ולינקים שימושיים להרחבת ידע בעמודים האחרונים.
- ❖ קרדיט מיוחד לספר - מערכות הפעלה (המרכז לחינוך סייבר) – מאת ברק גונן.

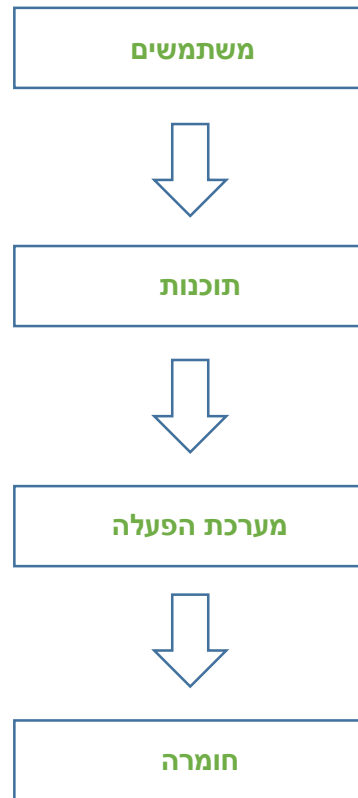
- בספר זה מוצגת מערכת ההפעלה על כל תפקידיה.
- מחקר זה נשען על מערכת הפעלה של Windows.
- חקר מעמיק על רכיבי המחשב והשימוש שמערכת ההפעלה עושה עימם.
- ארכיטקטורת המעבד בחקר זה נשענת על מעבידי אינטל.
- + הצגה והסבר מעמיק של מספר כלים מתוך SysinternalsSuite, כלים אלו נותנים דיאגנוזה בזמן אמת אודות חומרת ותוכנת המחשב.

תוכן עניינים :

מערכת הפעלה ותפקידה	• 5 – 4
ארכיטקטורת מערכת הפעלה	• 9 – 6
מעגלי הרשאה בארכיטקטורת מערכת הפעלה	• 12 – 10
סיום סיקור ארכיטקטורת מערכת הפעלה ותפקידיה	• 15 – 13
שגרת טיפול בפסיקה	• 16 – 15
ארכיטקטורת המעבד ושימוש בזיכרונות מטמון ו- אוגר	• 21 – 17
כיצד עובד מעבד 32 ביט	• 23 – 22
כיצד מעבדים עברו את מגבלת 4 גיגה	• 27 – 24
תפקידי המעבד תזמון המעבד החלפת תוכן במעבד	• 31 – 28
זיכרון	• 33 – 32
סוגי זיכרון שמערכת ההפעלה עושה איתם שימוש	• 38 – 34
כתובות פיזיות וכתובות ווירטואליות	• 40 – 39
תהליכים ותהליכונים	• 44 – 41
שיתוף זיכרון וסנכרון בין תהליכים	• 47 – 45
הצגה וניתוח של כלים מתוך Sysinternals Suite	• 65 – 48
קרדיטים ומקורות	• 69 – 66

מערכת הפעלה ומה תפקידה :

כדי שתוכנות, יישומים ומשתמשים יוכלו לעשות שימוש במשאבי המחשב, על המחשב תהייה מותקנת מערכת הפעלה, תפקידה של מערכת הפעלה הוא לקשר בין החומרה לתוכנות השונות ולמשתמש בצורה נגישה ונוחה.



כיום יש מגוון רחב של מערכות הפעלה חלקן למטרה מסוימת וחלקן לנוחות המשתמש. כל מערכת הפעלה עושה שימוש באלגוריתם בכדי לשלב תהליכים ותוכנות שרצות עליה (בהתאם למשתמש ולתפקיד הייעודי של מערכת ההפעלה). בין תפקידיה מערכת ההפעלה יודעת לנהל ציוד I/O כגון: מסך, רמקולים, כרטיס רשת. ותיתן למשתמש אפשרות נוחה כדי לנהל את ציוד I/O.

כדי להבין איך עובדת מערכת הפעלה צריך לסקור אותה לחתיכה וחתיכה וככול שנתקדם נצבור ידע הנותן הבנה על כל היבט הבונה את מערכת ההפעלה.

לפני שנתחיל סריקה לעומק של מערכת ההפעלה וכיצד היא עובדת נבין שיש כמה סוגים של מערכות הפעלה ייעודיות.

סוגי מערכות הפעלה נפוצות :

מערכת הפעלה שולחנית –

מערכת הפעלה זו פועלת בסביבה משתמש שולט במחשב אישי .
לדוגמה, Windows, Mac OS והפצות לינוקס שונות (אובונטו, Debian, ועוד ..).

מערכת ריבוי משימות / שיתוף זמן –

מערכת ההפעלה מטפלת בריבוי משימות באופן שהיא יכולה להתמודד עם פעולות מרובות , מבצעת מספר תוכניות בו זמנית, שיטת פעולה שבה משתמשים מרובים עם תוכניות שונות מתקשרים כמעט בו זמנית עם ה - CPU , מערכות הפעלה אלה פותחו בכדי לספק שימוש אינטראקטיבי במערכת מחשב.

מערכת הפעלה מרובת עיבודים –

המטרה העיקרית של שימוש במערכת הפעלה מרובת מעבדים היא לצרוך כוח מחשוב גבוה ולהגדיל את מהירות הביצוע של המערכת, מערכת הפעלה מרובת-עיבודים מסוגלת להריץ תוכניות רבות בו זמנית. כיום עובדים לרוב בשיטת "עיבוד אסימטרי" מעבד אחד מריץ את מערכת ההפעלה ותפקידו להקצות משימות לשאר המעבדים.

מערכת הפעלה בזמן אמת –

נעשה שימוש במערכות הפעלה בזמן אמת כאשר מגיע מידע גדול של נתונים וצריך לעבד אותו במהירות רבה, בדרך כלל יש שימוש במערכות אלו בסביבות רפואיות (מערכות הדמיה רפואית), מעקב אחר ניסויים מדעיים, מערכות תעבורה אווירית, מערכות בקרת פיקוד ...
למערכות אלו 2 סוגים עיקריים :

מערכת קשה – מידע המאוחסן לזמן קצר בלבד, המטרה להשיג תשובה תוך זמן קצר אחרת אין ערך למידע שעובר, אפשר לקחת דוגמא של מטוס שמבקש מסלול נחיתה דבר המחייב תגובה בזמן מאוד קצר.

מערכת רכה – מהמידע שעובר חייב לתת תשובה בזמן, אך אם היא לא מגיע בזמן זה לא נורא אפשר לשלוח שוב את המידע לעיבוד, בדרך כלל נעשה שימוש בעולם הסרטים ותעשיית הרובוטיקה.

מערכת הפעלה מבוזרת –

מעבדים מרכזיים מרובים משמשים מערכות מבוזרות לשרת מספר יישומים בזמן אמת ומספר משתמשים. בהתאם, משרות עיבוד נתונים מופצות בין המעבדים,
מערכת הפעלה מבוזרת מנהלת את המשאבים המשותפים למערכת המשמשים מספר תהליכים, את פעילות תזמון התהליכים (כיצד הקצאת תהליכים על מעבדים זמינים), את התקשורת והסנכרון בין תהליכים פועלים וכן הלאה.

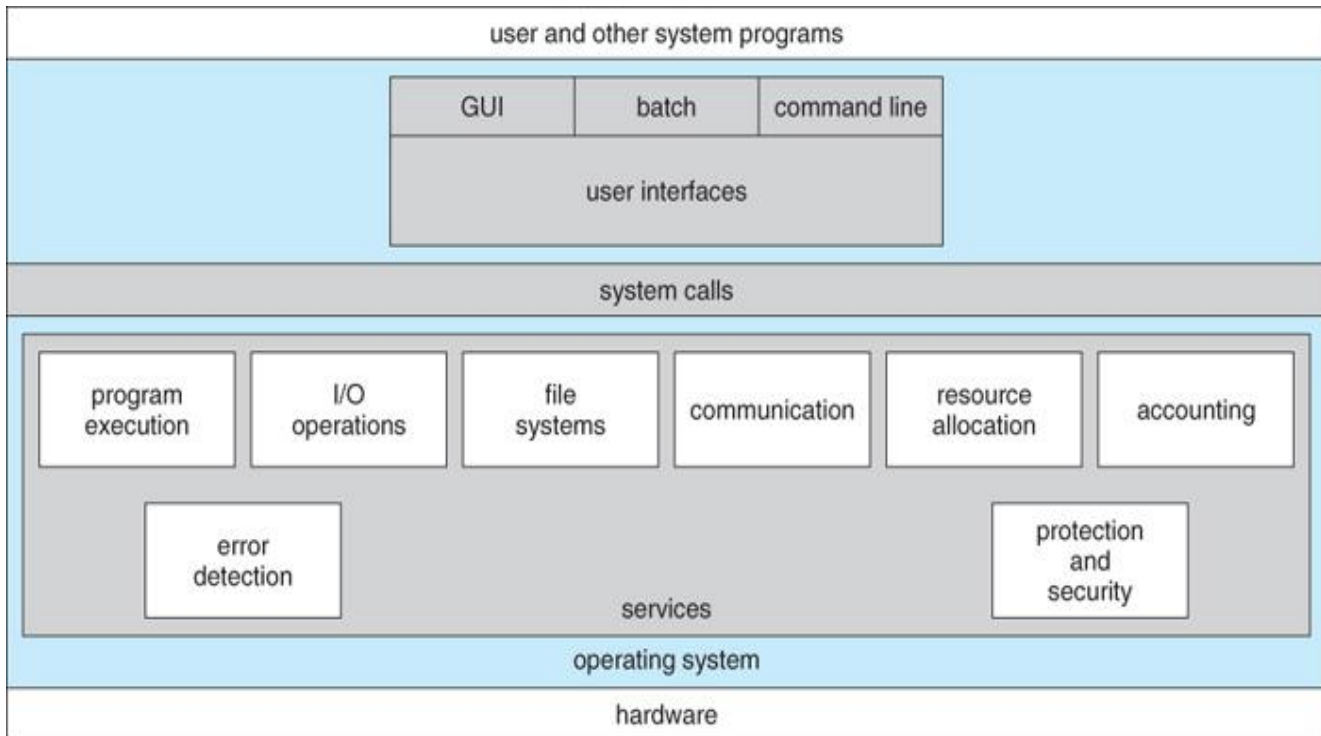
מערכת הפעלה ברשת –

מערכת הפעלה רשתית NOS היא מערכת הפעלה המנהלת משאבי רשת, למעשה מערכת הפעלה הכוללת פונקציות מיוחדות לחיבור מחשבים והתקנים לרשת מקומית .
לצורך המחשה WINDOWS SERVER נחשב למערכת הפעלה רשתית.

מערכת הפעלה ניידת –

מערכות הפעלה ניידות הן מערכות הפעלה המיועדות במיוחד להפעלת סמארטפונים, טאבלטים שעונים חכמים ...
כמה ממערכות ההפעלה הניידות המפורסמות ביותר הן Android -i , iOS, אך אחרות כוללות BlackBerry, watchOS. -i

ארכיטקטורת מערכת הפעלה :



נתחיל לפרק כל מסגרת ולהתייחס לאובייקטים השונים

– במסגרת העליינה user and other system programs

במסגרת זו מתייחסת לשימוש שעושה המשתמש הרגיל בארכיטקטורה של מערכת ההפעלה (האיור שלפנינו), והתוכנות השונות שכדי לפעיל אותן הן חייבות להיעזר בארכיטקטורה של מערכת ההפעלה.

– מסגרת ממשקי משתמש user interface

במסגרת זו יש התייחסות לאמצעי התקשורת בין המשתמש לבין מערכת ההפעלה, יכול תקשורת באמצעות הנפקת פקודות למערכת בהתאם לגרסת מערכת הפעלה.

ממשק GUI – למשל מערכות הפעלה כמו (windows, Ubuntu, macOS), הכוונה בממשק גרפי, נוכל לבצע פעולה ע"י לחיצה על אייקון בעכבר (ולא רק בפקודה) כגון: פתיחת תיקייה בשולחן העבודה.

– Batch

מתייחס לעיבוד באוטומט (לא הוזכר בסוגי במערכות הפעלה, מערכת אווזה כאמור כיום השימוש הנעשה במערכות הפעלה אלה הוא מועט), בקצרה - עיבוד אווזה הוא טכניקה שבה מערכת הפעלה אוספת את התוכניות והנתונים יחד באווצה לפני תחילת העיבוד ומגדירה עבודה רציפה של פקודות, תוכניות ונתונים כיחידה אחת.

– ממשקי command line

תקשורת בין המשתמש למערכת ההפעלה בעזרת (shell) – מסך פקודות, יכול להיקרא גם ממשק שורת פקודה, לדוגמא ב-windos יש לנו את ה – command prompt ו-power shell. בהפצות של Linux יש לנו את ה-command line כאשר שפת מערכת ההפעלה מתקשרת דרכה נקראת Bash (פקודות ב-windos שונות מהפקודות ב-Linux).

מסגרת System call – קריאה למערכת

קריאה/פנייה ישירה לליבת מערכת ההפעלה.

בכדי שנבין את הפעולה נסביר בקצרה על מעגלי אבטחה של מערכת הפעלה (הסבר מפורט בהמשך).

במרוצת השנים היבט אבטחה במערכת הפעלה שאנו משתמשים בה הלך והתפתח, עד לרעיון מעגלי אבטחה המטרה העיקרית להגן על משאבי המחשב שלא כל תהליך בין אם זדוני בין אם לא או תוכנה זדונית יוכלו לעשות כל שעולה על רוחם, מכן התחיל רעיון הרשאות במעגל אבטחה (כרגע נתמקד בשני מעגלים).

Userland – מתן הרשאות נמוכות לתוכנות ותהליכים שאינם דורשים משאבים ישירות מהחומרה אלא באופן עקיף (מתן הסבר מעמיק בהמשך).

Kernel – כינוי לליבת מערכת ההפעלה, תוכנות או תהליכים שמקבלים הרשאות גבוהות נכללים במעגל הזה וניתן להם גישה ישירה לחומרת המחשב.

וכאן נכנסת מסגרת System call - באמצעות פנייה זו תוכנות ותהליכים עם הרשאות נמוכות יכולות לבקש גישה לאזור ה - Kernel ולקבל גישה למשאבי החומרה.

ישנן כמה קטגוריות של System call :

- בקרה על תהליכים – קריאה ל - System call קושרת שליטה מלאה על התהליכים המבקשים שימוש בחומרת המחשב, בין אפשרויות השליטה הן: טעינה של תהליך, הרצת תהליך, יצירת תהליך, שחרור משאבים של תהליכים ועוד ...
- עבודה עם קבצים וספריות – יצירה, מחיקה, סגירה, ופתיחה מחדש (המטרה גישה ישירה לדיסק הקשיח עליו שמורים קבצים וספריות).
- ניהול התקני חומרה ותוכנה.
- תחזוקת המידע – קבלה/שינוי מידע על מאפיינים של תהליכים/קבצים.
- ניהול תקשורת – מכל סוג עם "העולם החיצוני למחשב" – הכוונה בין היתר לתוכנות שליטה מרחוק כגון עבודה עם פרוטוקול RDP , תהליכים המשותפים ל WEB ועוד ...

מסגרת Services –

מסגרת זו מתייחסת שירותים בשימוש מערכת ההפעלה נתחיל לפרק ולנתח את האובייקטים.

Program Execution – ביצוע תוכנית

מערכת ההפעלה חייבת להיות מסוגלת לטעון סוגים של תוכניות לזיכרון – RAM (נרחיב על רכיב זיכרון זה בהמשך), ומערכת ההפעלה צריכה להיות מסוגלת להפעיל את התוכנית ולהפסיקה באופן מקוון או לא מקוון. התוכנית/התהליך כולל ביצוע מלא של הקוד הכתוב.

בזמן טעינת התוכנית לזיכרון מתבצעים מאחורי הקלעים כמה שלבים נוספים:

- ביצוע של התוכנית.
- טיפול מלא בביצוע התוכנית (הכולל טיפול במצב של כשל ויציאה תקינה מהתוכנית).
- מספק מנגנון לסנכרון בין תהליכים.
- מספק מנגנון לתקשורת תהליכית.

I/O Operation – פעולת של קלט/פלט

מערכת ההפעלה אחראית על ניהול מכשירי קלט/פלט שונים כולל: עכברים, מקלדות, משטח מגע, כונני דיסק, מתאמי תצוגה, התקני USB, מסך ממופה סיביות, נורית, ממיר אנלוגי לדיגיטלי, מופעל / כיבוי מתג, חיבורי רשת, קלט / פלט שמע, מדפסות ועוד ...

File-System Manipulation – מניפולציה של מערכת הקבצים

בנוסף לאחסון נתונים מערכת ההפעלה אחראית גם על תחזוקת של מבני ספריות ותיקיות משנה, מיפוי שמות קבצים לבלוקים ספציפיים של אחסון נתונים, ומתן כלים לניווט ושימוש במערכת הקבצים הכוללים טבלאות לאיתור כתובת הקובץ בזיכרון (בהמשך נרחיב על כתובת מיקום קובץ בזיכרון),

כחלק מהפעילויות העיקריות של מערכת ההפעלה ביחס לניהול ואחסון קבצים הן –

- התוכנית דורשת לקרוא קובץ או לכתוב קובץ.
- מערכת ההפעלה נותנת את האישור לתוכנית הפעלה בקובץ.
- אישור זה משתנה מקריאה בלבד, קריאה-כתיבה, נדחתה וכן הלאה.
- מערכת ההפעלה מספקת ממשק למשתמש ליצירה / מחיקה של קבצים וספריות.
- מערכת ההפעלה מספקת ממשק ליצירת גיבוי של מערכת הקבצים.

Communications Inter-process – תקשורת בין תהליכים

או בקיצור (IPC) תקשורת בין תהליכים הרצים בין אם על אותו מעבד או תהליכים הרצים על מספר מעבדים נפרדים, התקשורת יכולה להתבצע בין אם בזיכרון המשותף לאותם תהליכים או בהעברת הודעות -

(העברת הודעות היא מערכת המשאירה לתהליך שסיים הודעה לתהליך שמתחיל, בדרך כלל על המשך ביצוע משימה משותפת או מידע עבור התהליך הקודם שתהליך חדש מבצע שימוש באותו מידע).

בנוסף מערך ממשק תכנות זה מאפשר למתכנת לתאם פעילויות בין תהליכי תוכנית שונים אשר יכולים לפעול במקביל במערכת הפעלה, זה מאפשר לתוכנית מסוימת לטפל בבקשות משתמשים רבות בו זמנית.

Resource Allocation – הקצאת משאבים

מערכת ההפעלה אחראית על הקצאת משאבי חומרת המחשב לתוכניות היישום, שיטת החלוקה יכולה להתבצע באמצעות מערכות גנריות ויכולה להתבצע באמצעות מערכות שתוכננו בקפידה רבה מאוד המותאמות אישית למשאב ולסביבת הפעלה מסוימת. מאחר שמערכת ההפעלה מחליטה על איזו תוכנית להשתמש במשאבי החומרה בזמן מסוים, לכן היא נקראת כמקצה משאבים.

Accounting – חשבונאות

מערכת ההפעלה מבצעת מעקב אחר של כמה אובייקטים :

- מעקב אחר פעילות המערכת ושימוש במשאבים.
- אחר חשבון של כל הפונקציות המתרחשות בכל פעם במערכת המחשב (פרטים כמו סוגי שגיאות).
- עוקב אחר המשתמשים שמשתמשים בכמה ובאיזה משאבי מחשב.

המטרה ליצור תיעוד (בין היתר של שגיאות שהתרחשו), ליצור רשומות סטטיסטיות בכל הנוגע לניצול משאבים, וליצור רשומות סטטיסטיות שניתן להשתמש בהם כדי לייעל את הביצועים העתידיים.

Error Detection – איתור שגיאות

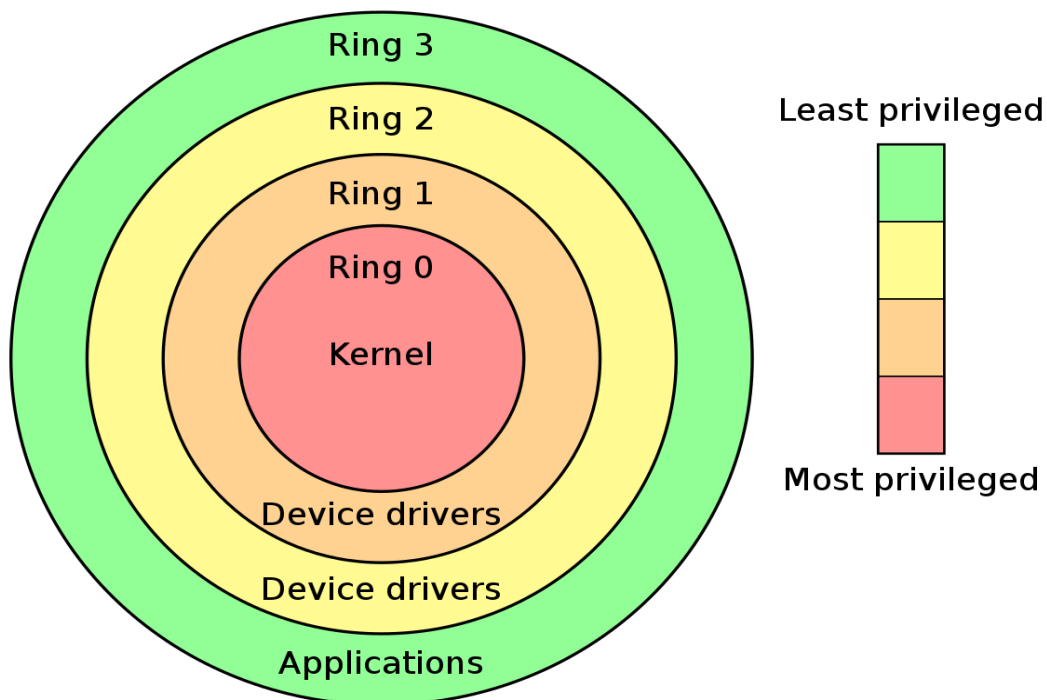
שגיאות יכולות להתרחש בכל זמן ובכל מקום, שגיאה עלולה להתרחש במעבד, בהתקני קלט / פלט או בחומרת הזיכרון יש לאתר הן שגיאות חומרה והן תוכנה ולטפל בהן כראוי, עם מינימום השלכות מזיקות. מערכת ההפעלה בודקת ללא הרף שגיאות אפשריות ונוקטת בפעולה מתאימה להבטחת מחשוב נכון ועקבי.

Protection and Security – הגנה ובטחון

במרוצת השנים בכדי ליישם הגנה על חומרת המחשב ועל מערכת ההפעלה עליה אנו עובדים, התפתח מערך הגנה בשם protection rings / protected mode – מעגלי הרשאה/אבטחה. הרעיון פותח בצמוד לפיתוח מעבדי intel x86. מעגלי הרשאה מספקות רמות שונות עבור גישה למשאבי המחשב שבניהם:

- 0 – מאפיין של Kernel – "ליבת מערכת ההפעלה" הרשאה/גישה ישירה למשאבי המחשב.
- 1 – מאפיין של Drivers – הרשאה לתהליכים או תוכנות למנהלי ההתקנים.
- 2 – מאפיין של I/O – הרשאה לתוכנות/תוכניות העושות שימוש בקלט/פלט.
- 3 – מאפיין של User – הרשאות נמוכות השייכות ליישומים שבדרך כלל משמשים את המשתמש.

❖ הערה – במידה ושיש תהליכים/תוכנות/יישומים אשר נדרשים לבצע שימוש במשאבים שאין להם הרשאה לעשות בהם שימוש, כאן תיכנס קריאה/בקשה למערכת ההפעלה הנקראת System call שבאמצעותה תאפשר את השימוש תחת ביקורת מערכת ההפעלה או ייתן שימוש באופן עקיף (יפתח אזור בזיכרון הנקרא - שיתוף מרחב זיכרון, שדרכו תינתן גישה למשאבים באופן מבוקר למשאב הנתון בלבד ולזמן הפעילות המוגדר מראש בלבד).



במערכות הפעלה של ימינו נהוג לעשות שימוש רק בשני מעגלים של מעלי הרשאה והן :

Userland – הכולל יישומים המשמשים את המשתמש ובעלי הרשאות נמוכות – כל יישום או הפעלת תהליך או תוכנה יזומה מצד משתמש לשימוש במשאב בעל הרשאה גבוהה יותר יעבור דרך System call.

Kernel – הכולל את "ליבת מערכת ההפעלה", מנהלי התקנים, ותוכניות עם הרשאות גישה של קלט/פלט.

מהות תכנון מערך "מעגלי הרשאה" –

המטרה העיקרית העומדת מאחורי רעיון מעגלי הרשאה היא למנוע מתהליך או יישום שתפקידו לרוץ במעגל הרשאה נמוך בלבד, להגיע למעגל הרשאה גבוה שיכול לסכן את חומרת המחשב או את מערכת ההפעלה בשוגג או בזדון.

נמנה מספר סיכונים שמעגלי הרשאה מנטרלים :

1. מניעת כתיבה לזיכרון של תהליך אחד לזיכרון של תהליך אחר (ללא בקרה), פעולה כזו יכולה לשבש את התהליך האחר שרץ, ולהשפיע על תקינות מערכת ההפעלה.
*הערה – קיימים מקרים בהם תהליך כותב לזיכרון של תהליך אחר אולם זה נעשה באופן מבוקר של מערכת ההפעלה שמספקת פקודות מתאימות.
2. מניעת כתיבה לחומרת המחשב באופן ישיר, במצב שכזה לתהליך הכותב לחומרת מחשב באופן ישיר (בשוגג או בזדון) יש יכולת להשתלט על משאב החומרה, למנוע מתהליכים אחרים לגשת למשאב ובכך לפגום בפעילות התקינה של המחשב.
3. מניעת כתיבה ישירה לדיסק הקשיח ללא הגבלה וללא פיקוח מערכת ההפעלה, מצב כזה עלול לגרום לשינוי של קוד תוכנה השומרה על דיסק (הן לאחת והן לרבים), וכתוצאה מכך תוכנות שהקוד שלהן השתנה יבצעו משימות שונות ממה שהן אמורות לבצע.
4. מניעת שינוי הרשאות של תהליכים - כלומר אם התאפשר שינוי של הרשאות תהליכים מצב שבו תהליך יוכל לגשת למקום בזיכרון שבו נשמר ההרשאות שלו ולשנות אותן, הוא יוכל בין היתר להעלות את ההרשאות שלו למעגל שהוא לא אמור לרוץ בו לדוגמא (מ 3 – Userland ל 0 – Kernel), תהליך זה יוכל לעשות ככול העולה על רוחו ללא זיהוי חריג וללא פיקוח של מערכת ההפעלה.
5. מניעת שינוי נתונים הנמצאים בטבלת פסיקות של מערכת ההפעלה – למערכת ההפעלה יש אוסף של פונקציות היכולות להתערב בפעילות משאבים חומריים הנקראים פסיקות (על "פסיקה" נרחיב בהמשך), ברגע שתהליך יוכל לשנות את הקוד של הפונקציות הללו, הוא יוכל לגרום למערכת ההפעלה לבצע באופן מוחלט הרצת קוד במעגל הרשאות ה – Kernel .

בנוסף לנושא אבטחה יש למערכת ההפעלה אמצעי הגנה ופתרונות המוצעים למשתמש כדי למנוע פגיעה במערכות משאבים ובקבצים רגישים, קיים על מערכת ההפעלה מערכת הגנה למניעת פגיעה במערכת ובמשאבים, באמצעות תהליכים פנימיים סוררים או גורמים זדוניים.

קיימים מערכות מאובטחות במיוחד שניתן להתקין על מערכת ההפעלה העשויות לתת ערך מוסף בכל הנוגע להגנה על מערכת ההפעלה ועל קבצים רגישים הקשורים למשתמש.

פתרונות	איומים נפוצים
שם משתמש סיסמא - לכל משתמש שילוב של שם משתמש וסיסמה מובחנים והם צריכים להזין אותו כהלכה לפני שיוכלו לגשת למערכת.	נגיף - נגיפים הם בדרך כלל קטעי קוד קטנים המוטמעים במערכת, מיועדים להשחתה/השמדה של נתונים עד לקריסת המערכת (נגיפים ניתנים לשכפל עצמם ולהתפשט ברשת פנימית).
זיהוי מאפיין משתמשים - זיהויים שונים של מאפייני משתמשים בהם ניתן להשתמש הם טביעת אצבע, רשתית עין ועוד.. המשתמש יכול לגשת למערכת רק אם יש התאמה.	סוס טרויאני - יכול לגשת בחשאי לפרטי הכניסה של מערכת, משתמש זדוני יכול לנצל את המידע לכניסה למערכת ולעשות ככול העולה על רוחו.
סיסמה חד פעמית - סיסמאות אלה מספקות אבטחה רבה למטרות אימות, ניתן ליצור סיסמה חד פעמית אך ורק לצורך כניסה בכל פעם שמשתמש רוצה להיכנס למערכת, לא ניתן להשתמש בו יותר מפעם אחת.	תולעת - בעלת יכולת להרוס מערכת על ידי שימוש במשאבים שלה לרמות קיצוניות, מצב זה יכול ליצור עותקים מרובים שתובעים את כל המשאבים ואינם מאפשרים לגישה אליהם לתהליכים אחרים, תולעת יכולה לנטרל רשת שלמה בצורה כזו.
מפתח סודי - התקן חומרה יכול ליצור מפתח סודי הקשור למזהה המשתמש להתחברות, מפתח זה יכול להשתנות בכל פעם.	מניעת שירות - התקפות מסוג זה אינן מאפשרות למשתמשים החוקיים לגשת למערכת, בנוסף קיימת אפשרות הצפה, להציף את המערכת בבקשות ולכן היא מוצפת ולא יכולה לעבוד כמו שצריך עבור משתמשים אחרים.
אימות דו שלבי – בנוסף לשם משתמש וסיסמה לצורך הזדהות, נוסף עוד שלב אימות המיושם בכמה דרכים: קוד המיוצר בכל פעם מחדש ונשלח למכשיר הנייד כמסרון, או באמצעות אפליקציות ייעודיות.	דלת סתרים - דלת מלכודת הינה פריצת אבטחה העשויה להיות במערכת ללא ידיעת המשתמשים, זה יכול להיות מנוצל כדי לפגוע בנתונים או בקבצים במערכת על ידי אנשים זדוניים.
פתרונות אלו הן הצעות התנהלות למשתמש בנוסף רצוי להתקין תוכנת אבטחה ייעודית לניטור ובלימה של נזקקות ומזיקים בזמן אמת בכדי למקסם את מערך האבטחה על מערכת ההפעלה.	

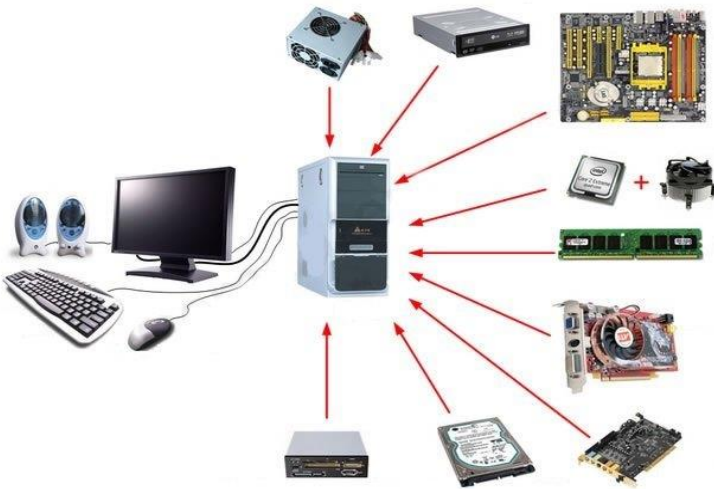
נעבור למסגרת המסומנת בתכלת Operating System –

המסגרת מייצגת את המערכת ההפעלה עצמה המשמשת גורם מתווך בין חומרת המחשב לבין תוכנות ומשתמשי המחשב. בנוסף מערכת ההפעלה הינה כלי לניהול החומרה, התוכנה, הדרייברים, והתהליכים הרצים על המחשב, במטרה לתת את הסביבה הנוחה ביותר למשתמש או לתפקיד ייעודי.

מסגרת תחתונה Hardware –

מסגרת זו מתייחסת לחומרת המחשב שעליו יושבת מערכת ההפעלה. חומרת המחשב כוללת בין היתר:

- לוח אם (MOTHERBOARD).
- מעבד (CPU).
- זיכרון קשיח (ROM).
- זיכרון גישה אקראית (RAM).
- כרטיס מסך (GPU).
- ספק כוח (PSU).
- כרטיס קול.
- כרטיס רשת.
- במחשב נייד נוסף סוללה.



נוסף לפירוק והסבר על האיור המתאר את ארכיטקטורת מערכת ההפעלה נושא שאינו מופיע במסגרות.

הנקרא – Drivers - התקן חומרה

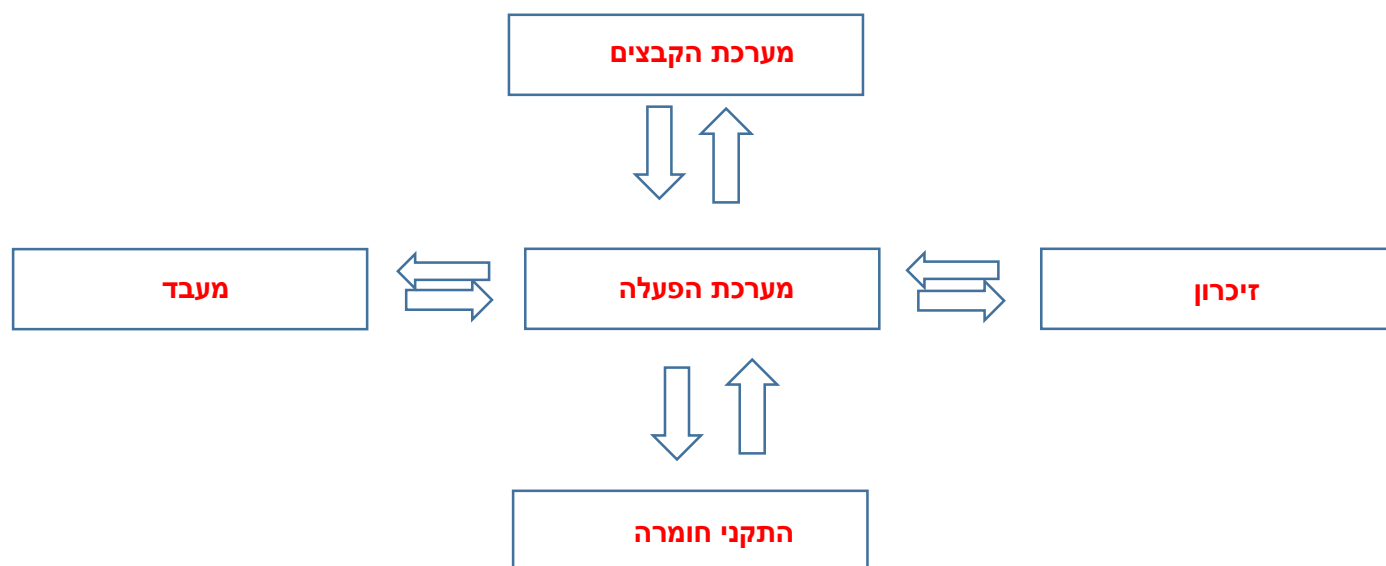
ברגע שאנו מוסיפים למחשב חומרה חדשה צריך להתקין תוכנה המנהלת את החומרה החדשה במערכת ההפעלה, התקן זה נקרא Driver.

התקני חומרה אלו שונים מתוכנות רגילות בכך שהם נכתבים באופן שונה על הדיסק, הם נכתבים באמצעות פונקציות מיוחדות של מערכת ההפעלה, ובכך רמת ההרשאות של מנהל ההתקן היא גבוהה.

התקנים של חומרה רצים מאחורי הקלעים ברגע הדלקת המחשב ועד ביצוע כיבוי, התקנים רצים מאחורי הקלעים מבלי שהמשתמש יצטרך לבצע פעולה מסוימת להפעלתם.

תפקידי מערכת ההפעלה :

מערכת ההפעלה היא הגורם המקשר בין התוכנות/היישומים שאנו מריצים לבין ארבעת האובייקטים הבאים:



התקני חומרה – התקנים אלו הנקראים Drivers התפקיד שלהם הוא להעביר מידע בין החומרה לבין מערכת ההפעלה (הלוח וחזר) מערכת ההפעלה אחראית על העברת המידע בין כל התקני החומרה, ללא קישור זה לא נוכל להשתמש (לדוגמא) במידע המגיע אלינו מכרטיס הרשת. מערכת ההפעלה אחראית על חלוקת זמן משאבים להתקנים, התקנים החולקים משאבים וקיים מצב שכמה התקנים יבקשו שירות ממערכת ההפעלה (System call) אזי מערכת ההפעלה תהייה אחראית על העדפה מי מבין ההתקנים יקבל זמן משאבים קודם לכן. (על חלוקת משאבים נרחיב בנושא המעבד).

מעבד – בכל פניה אל המעבד הוא יוכל לבצע רק פעולה אחת, שזה אומר המעבד יריץ תהליך אחד בכל פניה אליו, אולם המחשב מריץ הרבה יותר מתוכנית אחת או תהליך אחד, כאן מתערבת מערכת ההפעלה ע"י תכנון נכון של אלגוריתם ופונקציות מתאימות, תדע מערכת ההפעלה לתכנן את חלוקת משאבי המעבד בין התהליכים והתכניות שאנו מריצים על המחשב. כיום מחשבים מגיעים עם מספר מעבדים יתרון שמקצר את זמני ההמתנה בין התהליכים הזקוקים לזמן מעבד ומשפר את ביצועי המחשב.

זיכרון – כאמור בפרק על הזיכרון נדבר בהרחבה על כל סוגי הזיכרון שמערכת ההפעלה עושה בהם שימוש, רק נסביר שבכדי להריץ תהליך או תוכנה יש לטעון את הקוד שלה אל הזיכרון, הזיכרון ששומר באופן תמידי את הקבצים הוא הזיכרון הקשיח ע"י חלוקה לכתובות (נחריב על כתובות בהמשך), אולם הגישה לטעינה מהדיסק הקשיח אל המעבד לוקחת זמן יקר, מערכת ההפעלה צריכה לגשת לטבלאות המכוונות אותה אל אזור הכתובות בזיכרון השייך לקוד הנדרש (ללא טבלאות שמערכת ההפעלה נעזרת בהן מציאת קבצים בתוך הזיכרון ייקח זמן הנחשב לנצח, הטבלאות מייעלות את התהליך), זה אומנם אינו מספיק כדי לתת מענה בזמן קצר לכן מערכת ההפעלה תטען את התוכנה מהדיסק הקשיח אל זיכרון מהיר יותר (זיכרון RAM עליו נרחיב בהמשך), זיכרון RAM הוא קטן יותר ביחס לדיסק הקשיח (ROM) אבל מהיר בהרבה, לרוב אינו מכיל את כל כמות התהליכים הפועלים ברגע נתון וכאן מערכת ההפעלה אחראית על חלוקת משאבי המחשב בין התהליכים הפועלים, ועל חלוקת מקום בזיכרון לתהליכים הפועלים.

❖ דוגמא לתפקיד חלוקה ברגע שתהליך מסוים מסיים את זמן הריצה שלו או אינו דורש משאבי חומרה מערכת ההפעלה תטען אותו חזרה אל הדיסק הקשיח ובכך מפנה מקום בזיכרון RAM (זיכרון מהיר).

מערכת הקבצים – כדי שמערכת ההפעלה תמצא בקלות ויעילות קבצים השומרים בזיכרון (הן בדיסק הקשיח והן בזיכרון RAM) היא מנהלת מערך הנקרא מערכת הקבצים. מערכת זו אחראית על קטלוג שיאפשר מציאה קלה ונוחה של כל קובץ בכל אזור זיכרון שבו הוא שמור, מערכת הקבצים חייבת לדעת לעומק היכן נמצא טווח הכתובות שכל קובץ שמור, כלומר היכן טווח כתובות של קובץ מתחיל והיכן הוא נגמר, כדי שלמערכת ההפעלה תהייה גישה מהירה לטעון את הקובץ מטווח הכתובות שבו הוא שמור בזריזות ויעילות.

ללא מערכת הקבצים מערכת ההפעלה תראה את הדיסק הקשיח כגוף נתונים אחד גדול, מציאת של קובץ בצורה כזו תיקח זמן רב, ויכולה לגרום שגיאות רבות מערכת ההפעלה לא תדע היכן טווח כתובות של קובץ מתחיל והיכן הוא נגמר, בכך טעינת הקוד אל המעבד יבוצע באופן תקין ותגרום שגיאה בהפעלת הקוד.

מערכת הקבצים מצליחה לתת מענה אבטחה חשוב ע"י שיטת קטלוג שהיא מבצעת, בזמן שמערכת ההפעלה טוענת טווח כתובות של קובץ אל המעבד ונעזרת במערכת הקבצים למציאת הקובץ, אזי מערכת הקבצים לא תיתן לתהליך אחר לכתוב לאותו אזור כתובות שמערכת ההפעלה טענה ממנה הקובץ, ובכך מערכת הקבצים נותנת מענה שתהליכים בין אם בשוגג או בזדון לא יוכלו לכתוב לטווח כתובות של תהליך אחר ולשנות אותו או לפגום אותו, אלא לכל קובץ יש טווח כתובות מוגדר.

❖ הרחבת ידע – לפני שניגש לארכיטקטורת המעבד נסביר על רכיב אשר משפיע על עבודת המעבד, דרך מערכת ההפעלה.

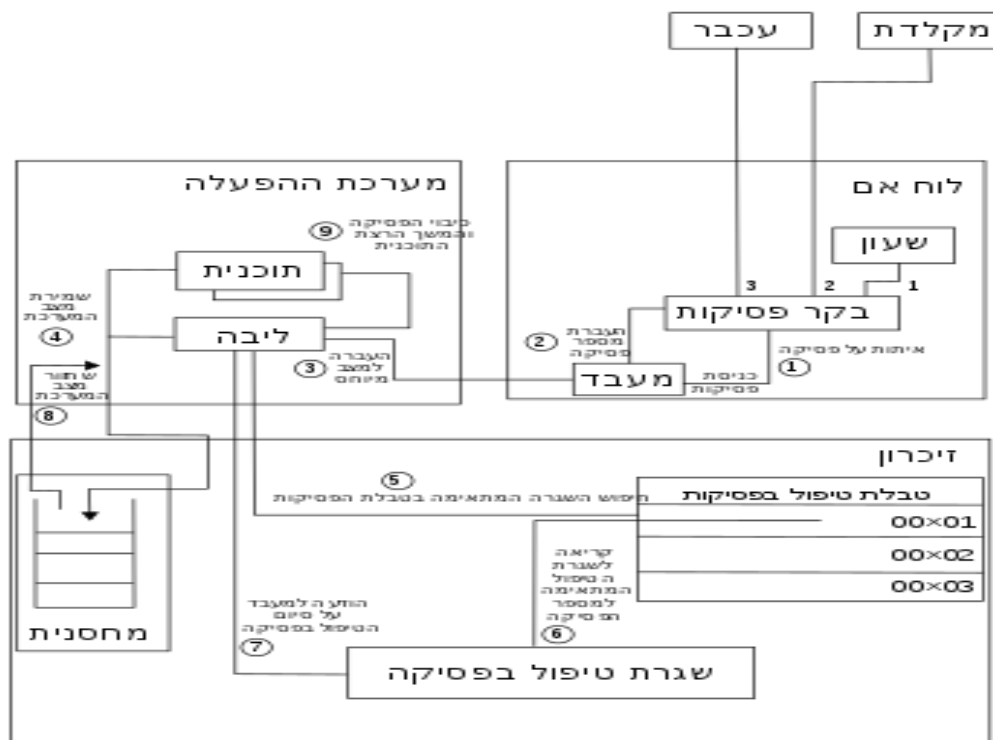
פסיקה Interrupt – על לוח האם יושב רכיב הנקרא בקר פסיקות, אופן הפעולה תהייה ע"י שליחת אות המגיע אל המעבד, מצד רכיב בקר הפסיקות או ממערכת ההפעלה, אות זה אומר למעבד לעצור את עבודתו על תהליך מסוים ולטפל בתהליך הדורש חשיבות דחופה יותר, בסיום הטיפול בפסיקה המעבד יחזור בדיוק לאותה נקודה שהוא הפסיק וימשיך משם לטיפול שותף של תהליכים ותוכנות.

סוגי פסיקות נפוצים הדורשים טיפול מיידי של המעבד :

- **אתחול:** מתרחשת על-ידי לחיצה על כפתור האתחול.
- **פקודה לא חוקית:** מתרחשת כאשר התוכנית המורצת מכילה פקודה שאינה שייכת לקבוצת הפקודות של שפת המכונה.
- **גישה לא מיושרת לזיכרון:** יכולה להתרחש בעת הבאת פקודה או בעת פקודת קריאה/כתיבה לזיכרון.
- **החטאת דף:** הכתובת האפקטיבית שייכת לדף שאינו אגור בזיכרון הראשי.
- **קריאה למערכת ההפעלה – (System Call):**
- **גלישה או חלוקה באפס.**
- **פסיקות של קלט/פלט:** פסיקות של התקנים חיצוניים למעבד כגון: מקלדת, עכבר, מודם, דיסק.
- **מלכודת :** התעוררות פסיקה בעקבות הדלקת דגל כלשהו.

טיפול בפסיקה מצד מערכת ההפעלה – מערכת ההפעלה מחזיקה טבלת פסיקות, הטבלה מחזיקה לרוב פקודות קפיצה לכתובות בזיכרון המכילות את אופן הטיפול בפסיקה, כחלק מהפקודות בטיפול פסיקה ישנם הוראות ישירות למעבד לעצור את עבודתו ולטפל באירוע, סט הפקודות מורות למעבד גם את אופן היציאה בעת סיום הטיפול בפסיקה, כלומר פקודה שאומרת למעבד לאיזה טווח כתובות הוא אמור לחזור ולהמשיך את עבודתו הרציפה.

כאשר מערכת הפעלה מקבלת התראה לפסיקה היא מפעילה מערך הנקרא "שגרת טיפול בפסיקה" כלומר מעיין מערך מנחה איזה פקודות להוציא מטבלת הפסיקות וכיצד לטפל באירוע, לרוב הטיפול מתחלק לשני רמות: ברמה הראשונה - הפסקת ריצת תהליך/תוכנית במעבד וטיפול בפסיקה. ברמה השנייה – הטיפול בפסיקה יתבצע לרוב מתוך תהליך נפרד או בזמן פנוי של המעבד.



כיצד מתבצעת שגרת טיפול בפסיקה.

ארכיטקטורת המעבד - CPU :

יחידת עיבוד מרכזית - Central Processing Unit

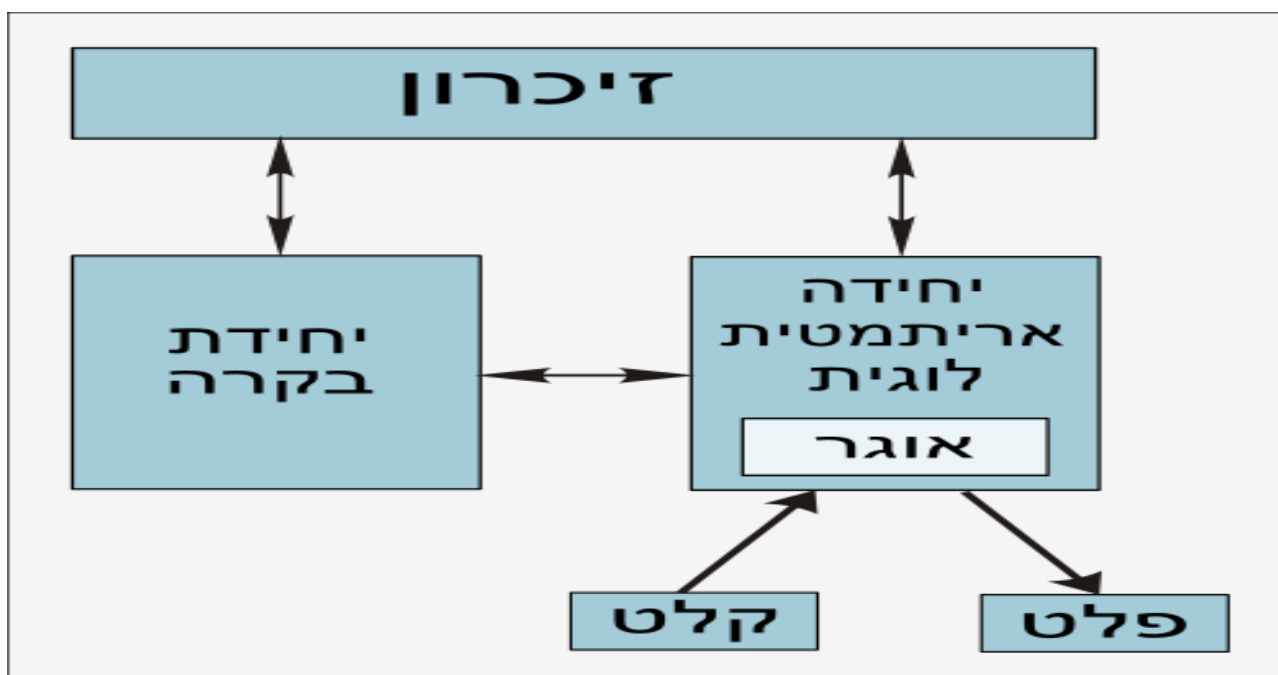
המעבד הוא חלק חומרתי במחשב אשר אחראי על ביצוע של פקודות אותן הוא מקבל מאזורי הזיכרון.

- בחלק זה נסקור כמה אזורי זיכרון חשובים שהמעבד וקבל ומעביר דרכן פקודות וחשובים.

המעבד יודע לבצע פקודות בשפת מכונה המכילה רצף של ביטים הבנויים מ 0 – 1 למרות שניתן הרגשה שזו שפה "דלה" זו אינה מהווה בעיה במגבלות ביצוע פקודה או תוכנית מחשב כלשהי, על המעבד צרוב סט פקודות המשמשות אותו לדוגמא :

העתקה של קובץ "X" לתוך קובץ "Y" – פקודה זו אצל יצרן א' יכול להיכתב בצורה של 10100000 ואצל יצרן ב' יכולה להיכתב בצורה שונה כגון – 10001100 .
בעבר כל יצרן היה עובד עם סט פקודות שונה כיום יש גישור והוא יבוא לידי ביטוי בכך שרוב המעבדים יעבדו עם סט אחד של פקודות התואם לכולם דבר המשפר את תקשורת המעבד עם שאר חלקי החומרה של המחשב.

ארכיטקטורת המעבד הומצאה עוד בשנות 1940 ועד היום מהווה קווים מנחים למעבדים של היום, אחת המודלים המובילים הוא ארכיטקטורת של פון נוימן למבנה המחשב, העיקרון הבולט במודל הינו הוספת רכיב זיכרון נוסף קטן אך מהיר בהרבה הנמצא פיזית בצמוד למעבד ומשמש אותו, המטרה לקצר את זמני חישוב שמעבד עושה, בצורת עבודה זו המעבד אינו ניגש ישירות לזיכרון גדול נפח ומחפש פקודות אלא הפקודה מגיע אל הזיכרון המהיר והמעבד שולף אותה משם.



ארכיטקטורת המחשב מאת פון נוימן.

- לפני שנמשיך לסקור הבדלים בין מעבדים של 32 ביט למעבדים של 64 ביט, נגדיר כמה מושגים שנעזר בהם בהמשך.

מושג שנתקלים בו המון בעולם המחשבים הנקרא - **Registers / רגיסטרים ובעברית אוגרים** – כל אוגר הוא יחידה אחת של זיכרון פנימי מהיר ביותר שנמצא לרוב בתוך יחידת העיבוד המרכזית של מחשב אשר מאפשר אחסון ערכים, בדרך כלל זמנית – זמנית מתייחס לזיכרון חשמלי העובד על מתח נמוך על עוד יש מתח הזיכרון שמור כאשר המתח נעצר הזיכרון נמחק, עבור פעולות בסיסיות שונות מסט הפקודות של המעבד כגון: חיבור, חיסור, והשוואה .

זיכרון מטמון / cache – זיכרון מהיר מאוד, היושב על המעבד, יש מעבדים בהם זיכרון המטמון מוגבל לפקודות בלבד, אחרים אשר המטמון משותף לפקודות ונתונים, ואחרים בהם יש מטמונות נפרדים לקוד ולנתונים. את זיכרון המטמון מסווגים בייצורו לכמה דרגות והן :

L0 - סוג המטמון המהיר ביותר, אך המוגבל ביותר בגודלו (נפח הזיכרון) הפועל בקצב השעון של המעבד עצמו.

L1 - סוג המטמון פחות מהיר אך נפח הזיכרון שלו גדול מ- L0 .

L2 - סוג המטמון מהיר פחות מ- L1 - אח הפחות גדול יותר (בחלק מיצרני המעבדים - L2 - בגלל הנפח שלו יכול להיות משותף למספר מעבדים).

- כדי למקסם את הבנתנו בנושא נוסיף כמה מושגים והרחבות ידע חשובים :

- כשאנו מדברים על זיכרון הצמוד למעבד כמו – **רגיסטרים** או **אוגרים** מדובר על חלק פיזי שיושב ממש על המעבד עצמו, עד היום הטכנולוגיה הנפוצה ביותר הייתה חלק סיליקון אשר מעביר מתח נמוך, אופן ייצורו נחשב ל"גידול" הסיליקון מיוצר במעבדה אשר שם בסוף התהליך הסיליקונים עוברים קטלוג לפי דולם וכמות המתח שהם מפיקים
 - הקטן ביותר יהיה האיכותי ביותר המהיר ביותר (וגם יותר יקר) יקוטלוג תחת - L0.
 - הבינוניים בגודלם יהיו בעל נפח גדול יותר ומהירות קטנה יותר יקוטלוג תחת - L1 .
 - הגדולים ביותר יהיו יותר גדולים מ- L1 - יקוטלוג תחת - L2 .

❖ הערה חלק מהיצרנים ימספרו את הקטלוג תחת "L" במספרים שונים אח הכוונה זהה.

❖ בנוסף בסוגי מעבדים חלק מהיצרנים יקראו לסיליקון זיכרון מטמון וחלק זיכרון אוגר שוב הכוונה זהה.

- **קצב שעון המעבד** – כיום שאנו ניגשים אל חומרת המעבד דבר ראשון נראה את שם יצרן המעבד את מספר הליבות שהמעבד מכיל ואחריו מספר עם אותיות Hz – (הרץ) המשמעות אירוע אחד לשנייה – זה מאפיין את זמן שעון המעבד עליו מדובר ושיטת חישוב הזמן, מעבדים כיום נמדדים ב GHz - (מיליארד גיגה הרץ לשנייה), כלומר מעבד נמדד במיליארד אירועים שהוא יודע לחשב לשנייה, ניתן דוגמא: מעבד 1.80 GHz יחשב לפחות חזק ממעבד 4.77 GHz . בנוסף מעבד נמדד בטווח תקן משוער זה אומר שמעבד אופטימלי יכול לרוץ בטווח מסוים כאשר האיכותיים שביניהם נמדדים בטווח הקצר ביותר לדוגמא: 1.60 GHz - 1.80 GHz .

❖ חשוב לציין שכיום בחלק מהמעבדים המודרניים בטכנולוגיה חדשה המחליפה את הסיליקון בלוח מתכתי של מילימטרים בודדים.

- כדי לעשות סדר קל מחשב ומערכת ההפעלה עושים שימוש באוגרים/ **Register** נציג מגוון סוגים של אוגרים נפוצים שמערכת ההפעלה והמחשב עושים בהם שימוש :

- **אוגר נתונים** - משמש לאחסון נתון מספרי שלם, לשם ביצוע פעולות אריתמטיות (פעולות חשבון המבוססות על 4 פעולות: חיבור, חיסור, כפל, חילוק).
- **אוגר כתובת** - מאחסן כתובת זיכרון.
- **אוגר רב-תכליתי** - מהווה שילוב של אוגר נתונים ואוגר כתובת.
- **אוגר נקודה צפה** – משמש בעיבוד נתונים בשיטת הייצוג של נקודה צפה .
- **אוגר קבוע** - מכיל ערך לקריאה בלבד .
- **אוגר וקטורי** - משמש לאחסון נתונים לעיבוד וקטורי בעזרת פקודות SIMD.
- **אוגרים מיוחדים** - כמו למשל האוגר המצביע למיקום בקוד, האוגר המתאר את מצב המערכת, המצביע לערך האחרון במחסנית הקריאות.
- **אוגר פקודות** – מאחסן את הפקודה המבוצעת באותו רגע.
- **אוגר אינדקס** – משמש לשינוי כתובת יחידת הנתונים עליה יש לבצע פעולה.
- **אוגר מותאם** - אלה הם אוגרים המותאמים למערכות מסוימות לשימושים ספציפיים אחרים.
- **אוגר חומרה** - אוגר היושב מחוץ למעבד המרכזי, על רכיב חומרה מסוים.
- **אוגר מונה (counter)** אוגר זה מכיל את הפקודה שצריכה להתבצע על ידי המעבד, כאשר המעבד עובר לפקודה זו הכתובת בתוך המונה תתקדם, דוגמה: קפיצה לכתובת מסוימת בעקבות פקודה כלשהי (לרוב בשפת אסמבלי), צבירת מספרים ועוד.
- **מצביע המחסנית (stack pointer)** מחסנית היא מקום מוגדר בזיכרון לאחסון זמני של נתונים, שמירת הנתונים במחסנית והוצאתם מתבצעת בשיטת "נכנס אחרון יוצא ראשון" שיטה שבה הנתון האחרון שנכנס למחסנית הוא גם הראשון שיוצא ממנה. דוגמה: מתבצעת תוכנית כלשהי כאשר המעבד מגיע לכתובת מסוימת הוא מגלה בה קריאה לפונקציה אחרת, לפני שהמעבד עובר לבצע את הפונקציה האחרת הוא שומר במחסנית את הכתובת של הפקודה שבאה אחרי הקריאה לפונקציה, על-מנת שיוכל לחזור לבצע את התוכנית שקראה לפונקציה, אחרי שהפונקציה תסתיים. בגלל תכונת ה LIFO-השימוש במחסנית לקפיצה לפונקציות וחזרה אל התוכנית הקוראת יכולה להתבצע בצורה רקורסיבית.
- **אוגר ההוראות (ip=instruction pointer)** האותות אשר מגיעים אל המעבד יכולים להיות מספרים או הוראות כלשהן, אם הגיעה הוראה אזי היא מופנית לאוגר ההוראות. יחידת הבקרה מפענחת את ההוראה בהתאם לקוד שלה (כפי שרשום באוגר) ומתבצעת.

- בדומה לשימוש בזיכרון אוגר שהמחשב ומערכת ההפעלה עושים בהם שימוש כך גם בזיכרון **מטמון / cache**, המחשב ומערכת ההפעלה משתמש בזיכרון זה למספר שימושים :

- **זיכרון מטמון של מעבד** – (שאותו סיקרו) זיכרון זה הוא זיכרון מחשב מהיר שקיבולתו קטנה, ותפקידו להאיץ את פעולות המעבד על ידי צמצום הגישה לזיכרון הראשי המאופיינת באיטיות ביחס למהירות המעבד.
- **זיכרון מטמון של דיסק קשיח** - מאיץ את זמן הגישה הממוצע לנתונים על הדיסק. בזיכרון המטמון מוחזקים נתונים מהדיסק הקשיח שמערכת ההפעלה ניגשה אליהם או צפויה לגשת אליהם.
- **זיכרון מטמון של דפדפן** - עיקרון דומה משמש גם את דפדפני האינטרנט, הדפדפן שומר במאגר מקומי נתונים, דפי אינטרנט ותוכן נוסף שהועבר מהאינטרנט, מכיוון שהגישה למאגר מקומי מהירה מן הגישה לשרת באינטרנט, ולעיתים אף חוסכת בעלות כספית.

❖ זיכרון מטמון (Cache) משתמשים תוכנות ושירותים רבים על מנת לשפר את מהירות עבודתם, לספק תשובות מהירות יותר של בקשות חוזרות, (שימוש בזיכרון מטמון נעשה אל מאחורי הקלעים ללא כל התערבות מצד המשתמש).

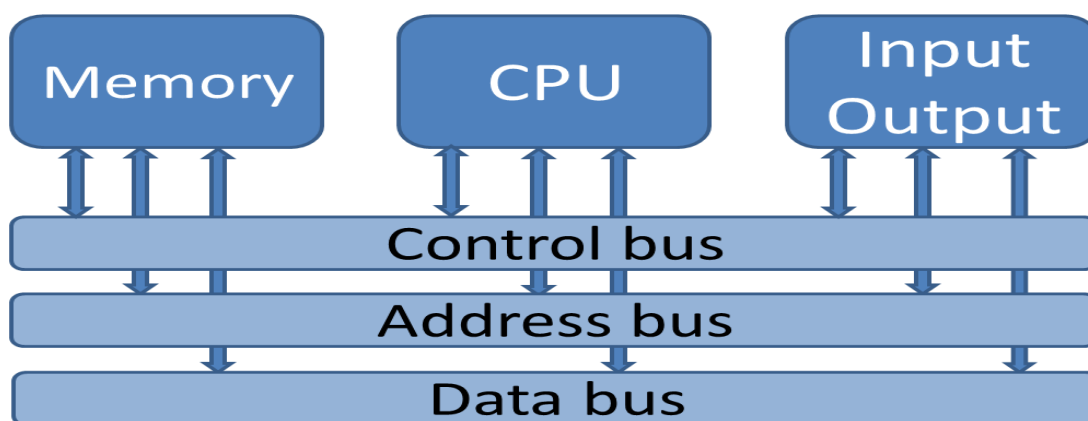
❖ הרחבת ידע – לעומר חקירת מעבד או זיכרון עושים שימוש במושג (bus) כדי להבין על מה מדובר חשוב לבין את המושג.

פסים (קווי נתונים) – Bus :

המעבד מבצע תקשורת תמידית עם סוגי זיכרונות ועם נתוני ציוד פלט/קלט, תקשורת זו מתבצעת באמצעות פסי תקשורת הנקראים (Bus).

פסי התקשורת הם למעשה אוסף של קווי חשמל המעבירים אותות של 0 ו-1 המתורגמים למתח חשמלי, כל פס תקשורת אחראי על מידע מסוג מסוים (לפי ייעוד הפעולה).

איור כיצד פסי התקשורת מעבירים מידע בין הזיכרון, המעבד, ונתוני פלט/קלט



על פי מעבד אינטל 8086, אצל יצרני מעבדים שונים ארכיטקטורת פסי תקשורת יכולה להיות שונה.

נתייחס לשלושת התפקידים של פסי התקשורת על פי האיור :

פס בקרה – Control bus : פס הבקרה מעביר 2 קווים, קו קריאה - (read) וקו כתיבה - (write) , זה קובע את כיוון העברת המידע, מכיוון שפסי תקשורת מעבירים אותו של 0 ו-1 .

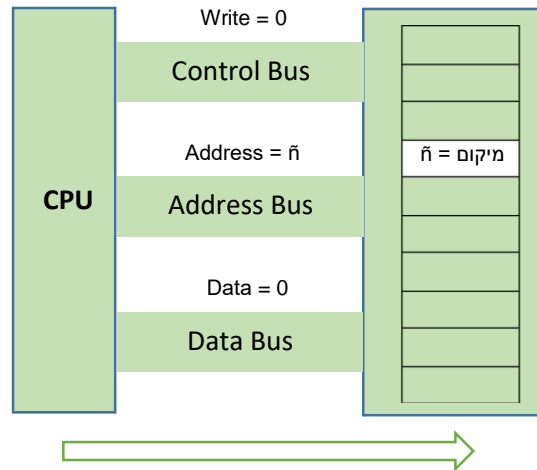
כאשר קווי read&write מעבירים אות של הספרה 1 – אין תקשורת בין הזיכרון למעבד.
כאשר קו read מעביר אות של הספרה 0 – המעבד קורא מהזיכרון.
כאשר קו write מעביר אות של הספרה 0 – המעבד כותב אל הזיכרון.

פס כתובות – Address bus : פס זה אחראי על מרחב הזיכרון, כלומר העברת טווח הכתובות או כתובת ספציפית בזיכרון אל המעבד.

פס נתונים – Data bus : פס זה אחראי על העברת הנתונים השונים המחשב, גודל הפס תלוי בסוג המעבד יכול להכיל – 64/32/16 קווים.
לפי סוג המעבד וכמות הקווים בפס הנתונים, המעבד יוכל לגשת לכל מרחב כמות הקווים או לפחות לפי דרישת הפעולה.

כיצד מידע נכתב בזיכרון בעזרת פסים (קווי נתונים) – Bus :

אופן כתיבה לזיכרון (מכיוון מהעבד אל הזיכרון)



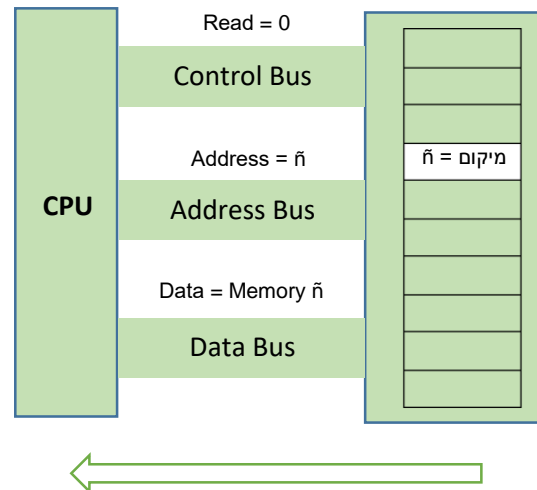
איור הממחיש כיצד מידע נכתב בזיכרון, ואיך הפסים פועלים.

Write = 0 - הקו פעיל מוכן לכתיבה בלבד.

Address - המעבד שם את הכתובת הרצויה בקו הכתובות.

Data = 0 - המעבד כותב את ערך המידע בקו.

אופן קריאה מהזיכרון (מכיוון הזיכרון אל המעבד)



איור הממחיש כיצד מידע נקרא מהזיכרון, ואיך הפסים פועלים.

Read = 0 - הקו פעיל מוכן לקריאה בלבד.

Address - המעבד קורא מהכתובת החוזרת הרצויה בקו הכתובות.

Data = Memory n - המעבד קורא את ערך המידע בקו, שהגיע מהזיכרון.

- בנושא חקר מעבדים נשמע לא מעט את המושג "סיביות / סיבית" כדי שנצליח להתקדם בהבנה כיצד מעבד עובד נסביר על המושג.

סיביות/סיבית – הכוונה לספרה בינארית בודדת – יחידת הנתונים הקטנה ביותר שבה משתמש המחשב. (סיביות – רצף של יחידות).

סיבית מכילה ערכים של 0 ו-1 בלבד, השימוש האלקטרוני בשיטה זו הוא:
0 = אין זרם
1 = יש זרם

באמצעות מחרוזת של סיביות נוצר במחשב ייצוג בינארי של אותיות (וסימנים אחרים), כאשר למטרה זו משמשת סיבית אחת, ניתן לייצג שתי אותיות שונות, וככל שמחרוזת הסיביות ארוכה יותר נוכל להגדיר יותר סימנים שונים, אך נשלם בצריכת מקום אחסון, רצף של סיביות המשמש להצגה של סימן אחד קרוי תו.

- יחידות מידה מבוססות סיביות :
בעולם המחשבים משתמשים במושג "בתים (Byte)" שבית אחד מורכב מ – 8 סיביות.

❖ לאחר שסיקרנו מושגים בסיסים בנושא המעבד נוכל להבין איך עובדים מעבדים של 32 ביט וכיצד נעשה שדרוגם למעבדים של 64 ביט.

כדי להבין כיצד בנוי מעבד של 32 ביט צריך להבין את העיקרון שעומד מולו

סיקרנו שפת מכונה מכילה ערכים של 0 ו-1 .
קריאה של שפת מכונה נקראת בסיס 2, כי הם נקראים בצימוד 01 או 10.
"משקל" המעבד נמדד לכמה כתובות בזיכרון הוא יכול לגשת, איך נעשה החישוב:
(על כתובות בזיכרון נרחיב בהמשך)

מעבד 32 ביט (בסיס 2) יכול לגשת ל 2 בחזקת 32, כתובות בזיכרון.
שזה מתורגם למשקל של – 4 גיגה (כתובות בזיכרון שהוא יכול לגשת אליהם בו זמנית)
או לייתר דיוק ל – 4,294,967,296.00

כמובן שכיום 4 גיגה RAM נשמע מעט מאוד וכמעט בלתי אפשרי לעבוד עם זיכרון בנפח כזה,
לכן יצרניות המעבדים אינטל פיתחה שיטה שבעזרתה ניתן לפרוץ את מגבלת 4 גיגה
ולאפשר תמיכה בעד 64 גיגה (גם במעבי 32 ביט).

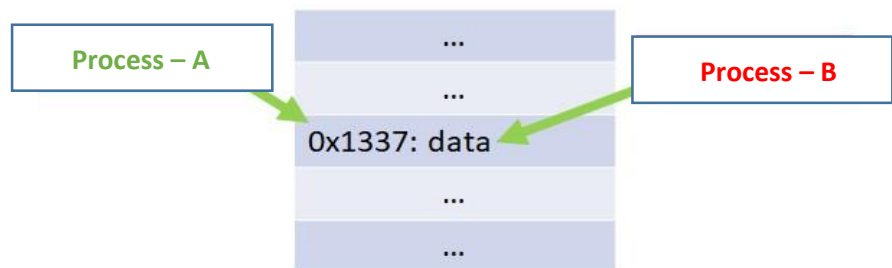
❖ הערה – חלק זה שייך לפרק הסוקר את הזיכרון לכן נרחיב רק על הטכניקה שבאמצעות המעבד יכול לגשת לכמות גדולה מ – 4 גיגה.

פריצת מגבלת 4 גיגה במעבדי אינטל –

כדי להבין כיצד מעבד יוכל לגשת ליותר כתובות יצרנית המעבדים אינטל הקימו ארכיטקטורה חדשה הנותנת למעבד את היכולת לגשת ליותר כתובות, הנקראת ארכיטקטורת x86 – עד היום מעבדים נשענים על אותה ארכיטקטורה.

עד לפני ארכיטקטורת x86 המעבד היה ניגש לכתובת פיזית לזיכרון – **Physical Memory** כלומר כל תהליך/תכניות חלקו אותו מרחב זיכרון פיזי, ברמה הפשוטה ביותר, המעבד ניגש לזיכרון הראשי ומבצע עליו קריאה/כתיבה.

במצב שכל התהליכים ניגשים לאותו זיכרון אחד פיזי, היינו מצפים מכל תהליך להשתמש באזור הזיכרון השייך לו בלבד, אך טכנולוגיה זו לא הייתה ניתנת לאכיפה ממשית, כלומר הייתה אפשרות לנצל את הטכנולוגיה בשיטה של זיכרון פיזי בלבד ולגרום לתהליכים לכתוב לזיכרון אחד למשנהו ובכך לדרוס את המידע הקיים של תהליך מסוים גם לתהליכי מערכת ההפעלה (מצב שגורם לקריסה).



בתמונה מתואר הדגמה כיצד תהליך דורס מידע של תהליך אחר, בכך שהוא כותב לאותה כתובת בזיכרון.

כדי למנוע מצב כזה אינטל פיתחה טכניקה לניהול זיכרון הנקראת **זיכרון ווירטואלי - Virtual Memory**

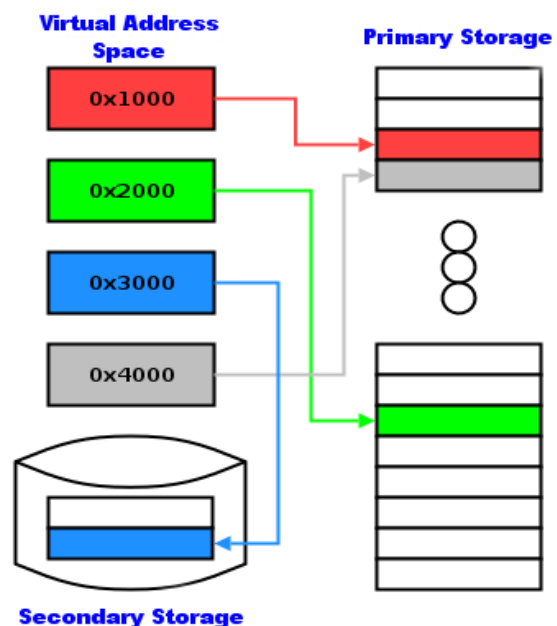
הרעיון העומד מאחורי טכניקת זיכרון ווירטואלי הוא די פשוט: כאשר תהליך ינסה לגשת לזיכרון בכתובת X (כתובת ווירטואלית), בפועל הוא ייגש לזיכרון בכתובת Y (כתובת פיזית), באמצעות יחידת ניהול זיכרון (MMU) האחראית על מיפוי מרחב הזיכרון הווירטואלי של כל תהליך לזיכרון פיזי, כך שכל תהליך נעזר בטבלת מיפוי כדי לתרגם כתובות זיכרון ווירטואליות לכתובות פיזיות.

בעזרת מיפויים אלו וע"י ניצול העובדה שאף תהליך לא משתמש בכל הזיכרון שהוא ביקש, בכל רגע נתון. המעבד יכול לגשת ליותר כתובות בזיכרון ובכך לעבור את מגבלת 4 גיגה.

דפדוף (Paging) x86 –

כתובת ווירטואלית מומרת לכתובת פיזית בכדי ליישם את האפשרות הזו צריך להיות מבנה נתונים אשר אחראי על התהליך, דפדוף הינה טכניקה לניהול זיכרון המאפשרת למערכת הפעלה העברת קטעי זיכרון בין זיכרון ראשי שהמעבד עושה עמו שימוש - (זיכרון מהיר) לבין זיכרון משני (הכוונה לזיכרון קשיח), העברת הנתונים מתבצעת במקטעי זיכרון בעלי גודל זהה המכונים דפים.

הדפדוף מתבצע באמצעות טבלאות המחזיקות ערכים בטבלת עמודים - (Page Table Entry – PTE) – מיכל את המיפוי בין כתובות ווירטואליות של דף לבין כתובות הפיזיות, קיים בנוסף מידע עזר על הדף כלומר מכמה סיביות המורכב הדף.



שיטת הדפדוף מתבצעת כאשר תהליך/תוכנית תנסה לגשת לדפים שאינם ממופים לזיכרון פיזי, בזמן הגישה לזיכרון המעבד נעזר באמצעות יחידת ניהול זיכרון - (Memory Management Unit – MMU) שהיא רכיב חומרה האחראי על ניהול הזיכרון ע"י המעבד, כיום יחידת ניהול זיכרון ממוקמת במעבד כיחידה אחת ואינה רכיבי פיזי נפרד.

תפקיד MMU - מחלק את הזיכרון הפיזי למקטעים, ל-MMU נעזר בטבלה המאפשרת להמיר כתובת ווירטואלית לפיזית בקלות, למפות בין כתובות ווירטואליות לכתובות פיזיות בזמן ריצה של תהליך/תוכנית, אם דף הזיכרון הנדרש לא נמצא כעת בזיכרון הראשי (זיכרון מהיר), היחידה מייצרת פסיקה הקראת – page fault (מידע עזר שזמין ב – PTE) ומערכת ההפעלה מטפלת בפסיקה על ידי טעינת הדף מהזיכרון המשני (זיכרון קשיח).

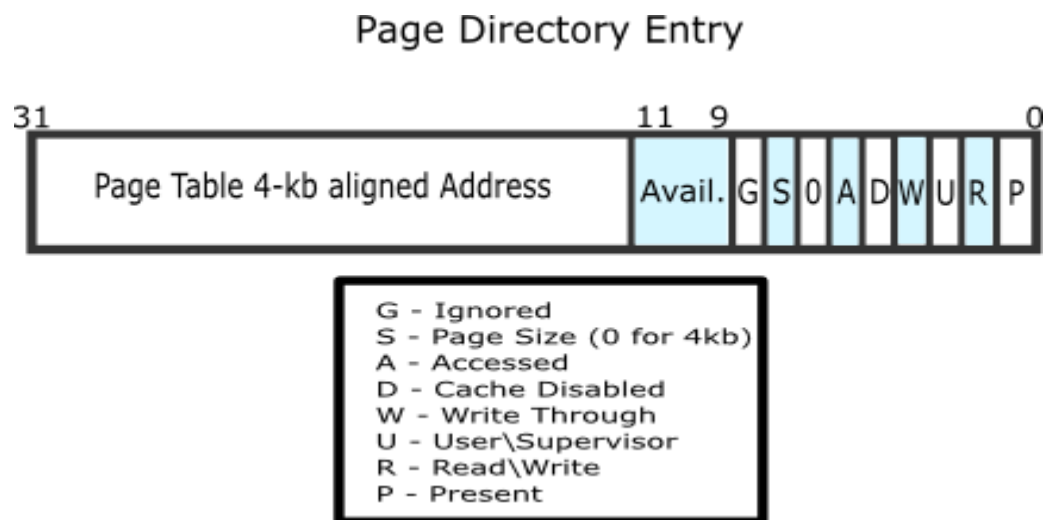
❖ עד לכאן ניתן להבין ששיטת הדפדוף באמצעות PTE אינה יעילה כלל, כאשר תהליך יפנה לזיכרון כדי לקבל כתובת פיזית הוא יקבל כתובת ווירטואלית ובצימוד טבלה האחראית על המיפוי שלו, מצב שגם הטבלה צורכת מקום בזיכרון ואנו עלולים להשתמש בכל הזיכרון שלנו רק עבור טבלאות מיפוי לכל תהליך, בנוסף ברגע שהמעבד יבצע Contact Switch (הסבר בהמשך) נצטרך לטעות מחדש מיפוי טבלאות אחר.

אינטל מצאו פתרון להקטנת משקל טבלאות המיפוי לתהליכים (PTE) באמצעות שיטה הנקראת –

Page Directory Entry – PDE

המטרה של PDE – לחלק את העמודים (PTE) לתיקיות כך שבכל תיקייה יהיו בדיוק 1024 עמודים, ונשתמש ב-1024 "תיקיות" בסה"כ. חישוב של 4KB לעמוד כפול 1024 עמודים, כפול 1024 תיקיות, שקול ל-4GB ממופים.

בשיטה זו נוכל למפות רק עמוד אחד בזיכרון השווה ל-4KB עבור מבנה התיקיות (PDE), שיטת החלוקה באופן זהו (PDE) חסכונית משמעותית משיטת החלוקה באופן (PTE).



בתרשים אופן המיפוי בשיטת (PDE) – לא ניכנס לעומק האובייקטים ואופן החלוקה, כי אנו עדיין סוקרים מעבדים בגבולות ה-4 גיגה.

❖ הערה (קרדיט): מי שרצה להרחיב ידע בכל הקשור לחקר עמוק בנושא, ממליץ לעבור על המאמר – (Intel Paging & Page Table Exploitation on Windows - מאמר מאת יובל עטיה) לינק למאמר - <https://www.digitalwhisper.co.il/files/Zines/0x61/DW97-3-PageTableExp.pdf>

כפי שהוצג בדפים הקודמים, המעבד עדיין יוכל לגשת רק ל - 4 גיגה הראשונים של הזיכרון הפיזי, כתוצאה מכך במידה ויש לנו תהליכים הרצים במקביל כולם בסופו של דבר ימופו לאותם 4 גיגה שחולקים בין היתר גם את מערכת ההפעלה.

❖ הערה: גם אם נוסיף פיזית למחשב עוד זיכרון RAM לא נוכל לנצל אותם מכיוון שהכתובת הפיזית הגבוהה ביותר שניתן למפות אליה בעזרת PTE של 32-bit היא 4 גיגה או ליתר דיוק - 4,294,967,296.00

עשור לאחר מכך מוציאה אינטל את מעבד ה-Pentium Pro -

ה-Pentium Pro – הינו מעבד 32 ביט אשר יכול לנצל מרחב כתובות פיזי של 36 ביט, המעבד החדש נתמך בשיטה הנקראת PAE. בבסיס בשיטה PAE מאפשר למעבד לנצל זיכרון פיזי של יותר מ- 32 ביט, וכך מאפשר לתהליכים לגשת לכתובות פיזיות גבוהות מ- 4 גיגה. כך במידה ולמחשב יש 8 גיגה של זיכרון RAM, והמעבד תומך PAE תהליך אחד במכונה יכול לנצל את הכתובות הפיזיות בטווח 0-4 גיגה ותהליך אחר יכול לנצל את הכתובות הפיזיות בטווח 4-8 גיגה וכך יתבצע ניצול טוב יותר של הזיכרון הפיזי הנגיש למחשב.

שיטת PAE - physical address extension – הרחבת כתובות פיזית – וכיצד היא עובדת

PAE – הינה תוכנה לניהול הזיכרון, עד כה כדי לפנות לכתובות מעבדי אינטל עבדו עם טבלאות עמודים אשר בעזרתן מתבצע התרגום מכתובת ווירטואלית לפיזית.

לראשונה אינטל הציגה את פתרון PTE – עבודה עם טבלאות המחולקות ל – 2 רמות (ראינו שזה אינו חסכוני עד ללא ריאלי).

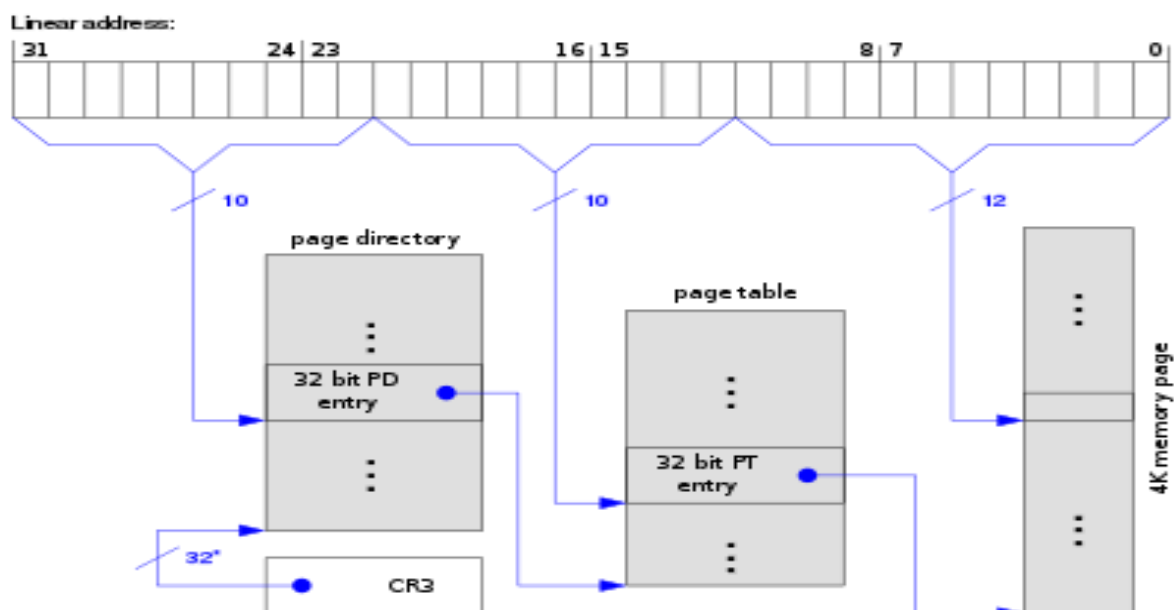
לאחר מכן אינטל נתנה פתרון באמצעות עבודה בשיטת PDE – בשיטה זו הטבלאות מחולקות ל – 3 רמות אשר מצמצמות את משקל טבלאות המיפוי באופן דרסטי, אך עדיין לא עברנו את מגבלת ה – 4 גיגה.

ולבסוף אינטל מצאה שיטה הנקראת PAE – בשיטה זו הטבלאות מחולקות ל – 4 רמות, בכך שהרחיבו יותר את המבנה ההיררכי של טבלת העמודים לארבע רמות זה מרחיב את שטח הכתובות הפיזיות שהמעבד ניגש אליהן, כלומר בהיררכית הטבלאות בשיטת PAE – המעבד ניגש למרחב גדול יותר של כתובות פיזיות בזיכרון מ – 4 גיגה ועד 64 גיגה.

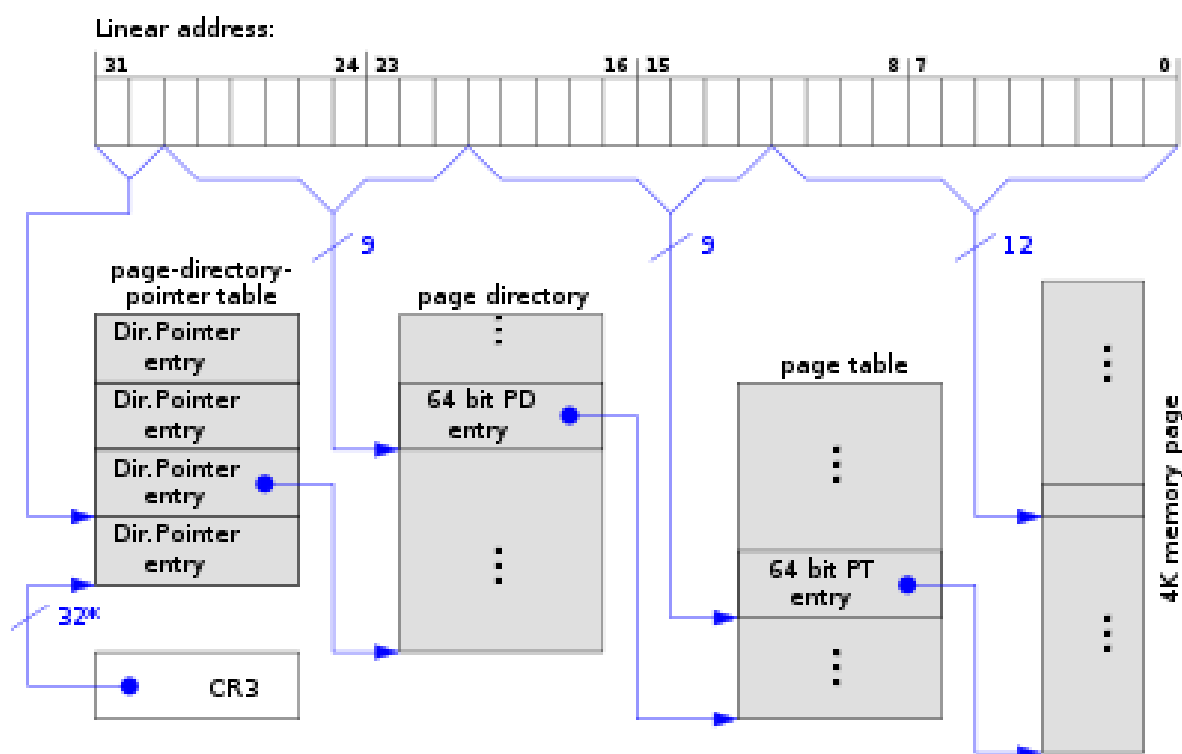
כיצד זה עובד - כאשר PAE דולק כל איבר מבני המיפוי מוכפל מ - 4 בתים ל - 8 בתים על מנת לתת גישה פוטנציאלית של עד 64 ביט (בסיס 2) יכול לגשת ל 2 בחזקת 64, כתובות בזיכרון. מכאן ארכיטקטורת המעבד X86 שונה ל X86-64.

כדי לתת המחשה כיצד הטבלאות נראות נציג איורים המראים את הטבלות, הערכים בתוכן, ואת חלוקת הביטים שהטבלאות עושות בהן שימוש.

מבני טבלת עמודים בשיטת PDE - (חלוקה ל 3 רמות)



מבני טבלת עמודים בשיטת PAE - (חלוקה ל 4 רמות)



❖ CR3 – המופיע באיורים הם אוגרים (Registers) של המעבד (CR3 מתאים למעבדי אינטל 32 ביט, במעבדים חדישים או מייצרן שונה השם או המספר המייצג יהיה שונה), אוגרים אלו מנהלים מספר פעולות הקשורות להפעלת המעבד עצמו.

מהפרק הקודם הבנו כיצד המעבד עובד ואת הארכיטקטורה שמאחוריו –

מעבד קורא שפת מכונה של 0 ו-1 . כל מעבד יש לו סט פקודות משלו (כיום נהוג שיהיה סט פקודות אחיד בין המעבדים השונים). למעבד יש זיכרון קטן שמחזיק את סט הפקודות. המעבד עושה פעולות חשיבה של חיבור חיסור כפל וחילוק (כך הוא פותר משוואות). עוד הבנו שמעבד ניגש לכמה זיכרונות בין היתר לזיכרון ראשי (זיכרון מהיר RAM) ו- זיכרון משני (ROM זיכרון קשיח).

תפקידי המעבד - CPU :

לפני שמעבד מתחיל לפעול מתבצעת טעינה של קוד מהזיכרון הקשיח אל זיכרון מהיר (למדנו) , לאחר הטעינה לזיכרון מהיר המעבד יכול לגשת אל הקוד ולהתחיל תהליך ביצוע הקוד.

למעבד יכול להיות 2 אפשרויות של סטים לצורך ביצוע קוד/פקודה (תלוי יצרן) :

אפשרות ראשונה :

- Fetch – הבאת פקודה מזיכרון מהיר אל המעבד / חישוב כתובת הפקודה הבאה לביצוע, וקריאת הפקודה הבאה לביצוע מהזיכרון המהיר.
- Decode – פענוח הפקודה, והבאת ערכי הרגיסטרים בהם הפקודה משתמשת. לדוגמא : מה סוג הפקודה ? , אם הפקודה כוללת גישה לכתובת בזיכרון, מהי הכתובת המבוקשת ?
- Execute – הרצה, ביצוע הפקודה – במקרה שהפקודה היא פקודת חישוב בשלב זה המעבד יבצע החישוב, במקרה והפקודה היא פעולת קריאה/כתיבה בזיכרון, כאן מחושבת הכתובת של הנתון בזיכרון, במקרה של פקודת הסתעפות (Branch/Jump) כאן מחושב קיום התנאי להסתעפות, ומתבצעת ההחלטה האם להסתעף, או לא.
- Memory – כתיבה/קריאה של נתון בזיכרון, במקרה שהפקודה כוללת כתיבה/קריאה בזיכרון.
- Write Back – כתיבת תוצאת החישוב משלב Executen או תוצאת הקריאה משלב ה Memory אל הזיכרון המיועד.

אפשרות שניה :

- חישוב הכתובת בזיכרון – בה נמצאת הפקודה באה.
- קריאה לפקודה הבאה – לביצוע מהזיכרון המיועד (לרוב מהזיכרון המהיר).
- פיענוח פקודה – בשלב זה המעבד יפענח מהי הפקודה שצריכה להתבצע, איזו יחידת ביצוע להפעיל ואילו משתנים מעורבים.
- חישוב – המעבד קורא ערכים מהזיכרון, (בסוגים שונים של מעבדים קריאה מהזיכרון תחשב לשלב נפרד), בהתאם לסוג פקודת הפעולה המעבד יפעיל חישוב מתאים לביצוע פעולות הרלוונטיות.
- כתיבה חזרה אל הזיכרון – הערך שהמעבד חישב (התוצאה) נכתב חזרה אל הזיכרון שבו הוא אמור להיות מאוחסן.

לאחר הרצת הפקודות בשפת מכונה מתבצע שוב מעבר לפקודה הבאה והשלבים חוזרים על עצמם שוב ושוב, האפשרויות להפסקת המעגל הוא ניתק מקור החשמל מהמעבד.

תזמון המעבד – CPU scheduling :

במצב אופטימלי המעבד שואף להיות תמיד בניצול וביעילות מקסימלית, זה נותן למחשב ולמשתמש זמן תגובה מהיר ויעיל, כדי שמעבד יהיה יעיל ומוצל, מערכת ההפעלה משתמשת במודול הנקרא מתזמן (Scheduler), מטרתו העיקרית של המתזמן הינה להגדיל את ביצועי המערכת באופן דרסטי בהתאם למערכת הקריטריונים שנבחרה.

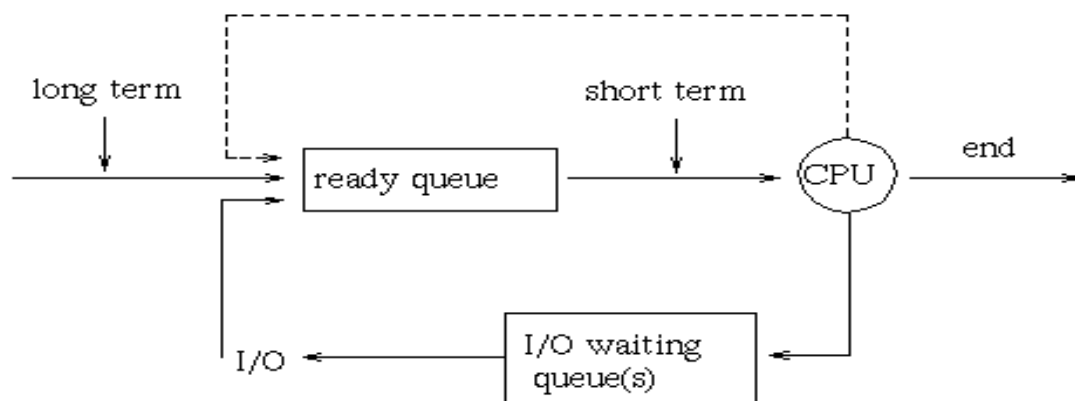
❖ הערה – הפצות שונות של מערכות הפעלה משתמשות בקריטריונים שונים בהתאם לדרישתם.

כיצד בנוי המתזמן (Scheduler) –

המתזמן הוא מודול הבנוי ע"י אלגוריתם שמערכת ההפעלה עושה בהם שימוש כדי לנהל תהליכים או כל בקשה (ישירה או עקיפה) אל משאב המעבד. כלומר מערכת ההפעלה חייבת לבחור איזה תהליך יורשה לגשת למשאב המעבד, מעיין סינון ומיון של התהליכים השונים.

- כדי להבין נגלוש מעט לתוכן הקשור לפרק הסוקר תהליכים.

כאשר תהליך מסוים הופעל מערכת ההפעלה משייכת אותו למעיין "תור" ששם הוא ממתין כדי לקבל גישה למשאבי המחשב, תור זה נקרא – **Job Queue**.
תור זה מכיל את כל התהליכים במערכת שנטענו לזיכרון המהיר (RAM) שמוכנים לרוץ אך עדיין לא קיבלו אישור גישה למשאבים.
לתהליכים אשר נטענו אל הזיכרון המהיר (RAM) ממתנים לריצה (אך עדיין לא קיבלו אישור גישה למשאבים) יש תור אחר הנקרא – **Ready Queue**.



כך נראה איור של תהליך המחכה לשימוש במשאב המעבד.

Job Queue – תור המייצג תהליכים אשר נטענו מהזיכרון, קיבלו הרשאה ומוכנים להשתמש במשאבים (כניסה לריצה – הסבר בפרק תהליכים).

Ready Queue – תור המייצג תהליכים שממתנים לקבל הרשאה לשימוש במעבדים, אחרי שהם כבר נטענו מהזיכרון.

I/O Waiting Queue – תור המייצג תהליכים של קלט/פלט, לאחר שהם מקבלים את השימוש במשאבים, הם חוזרים לתור של Ready.

כיצד המתזמן עובד (Scheduler) –

ברגע שתהליך מופעל, מערכת ההפעלה ע"י אלגוריתם מקנה לו מספר תורי הנקרא תזמון. מערכת ההפעלה מוכרחת לבחור איזה תהליכים נכנסים לריצה מתוך התורים כדי לא ליצור מצב של הרעבת תהליך (תהליך שמחכה זמן רב לשימוש במשאבים).

לצורך כך מערכת ההפעלה עובדת מול 2 סוגים של מתזמנים :

Short Term Scheduler – מתזמן מהיר, עובד מול תהליכים שקראים/נכתבים במהירות מהזיכרון, לרוב תהליכים שדורשים משאבים לזמנים קצרים מאוד.
Short Term – אחראי על התזמון בתור הנקרא **Ready Queue**, הנקרא גם CPU Scheduler.

Long Term Scheduler – מתזמן איטי יותר, עובד מול תהליכים הדורשים משאבי מעבד בדחיפות נמוכה, בטווחים של שניות/דקות, בנוסף מתזמן זה אחראי על כמות תהליכים שמערכת ההפעלה מוכנה להריץ במקביל (עובד לפי דחיפות תור Job Queue, ולפי מקום פנוי בתור Ready Queue).
Long term – הוא המתזמן של תור Job Queue מחליט אילו תהליכים נכנסים לתור Ready Queue.
מתזמן זה נקרא גם בשם JOB Scheduler.

❖ הערה – ההבדל העיקרי בין המתזמנים הוא תדירות שתהליכים ניגשים אליהם.

באופן כללי רוב התהליכים הדורשים משאבי מעבד יהיו תהליכי ציוד של קלט/פלט שדורשים טיפול תמידי, הם נכללים לתור הנקרא **Device queues** או **I/O** שם הם מחולקים ל 2 קטגוריות :

I/O Bound Process – תהליכי הזקוק באופן תמידי למידע של I/O, תהליך מהסוג הזה נמצא לרוב במצב של I/O Waiting Queue

CPU Bound Process – תהליך שדורש משאבי CPU באופן תמידי (צריך כל הזמן פעולות חישוב/פענוח CPU).

מטרות נוספות עומדות בפני מתזמן **Scheduler** :

1. הבטחת זמן תגובה לאירועים מסוימים - במערכות זמן אמת, על המערכת להבטיח זמן תגובה מינימלי לאירועים מסוימים.
2. הוגנות - חלוקת משאבים הוגנת בין התהליכים השונים.
3. זמן המתנה קצר לתהליכים הממתינים למעבד.

ביצוע החלפת קשר/תוכן במעבד CPU Context Switch -

החלפת קשר/תוכן היא יכולת מעבר בין הרצה של שני תהליכים באמצעות המעבד. בעזרת החלפת קשר/תוכן, מספר תהליכים מסוגלים לחלוק את אותו מעבד אליו הם מקושרים.

ההחלפה לתהליך אחד דורשת שמירה של המצב הקיים של התהליך הישן, וטעינה מצב של תהליך חדש כלומר - החלפת קשר/תוכן שומרת את המצב באמצעות האוגרים במעבד, ולאחר מכן מכניסה למעבד את נתוני ריצת תהליך חדש, בסיום ריצת התהליך החדש נתוני הריצה של התהליך הקודם חוזרים אל המעבד מהאוגרים להמשך טיפול.

CONTEXT SWITCHING



קיימים 3 תרחישים שבהם יש צורך להחלפת קשר/תוכן :

1. במערכת הפעלה הפועלת עם ריבוי משימות, כאן משתלב מתזמן המעבד (CPU Scheduler), כאמור לפי הסקירה שעשינו המתזמן קובע את סדר הרצת תהליכים ומקצה זמן מעבד לכל תהליך, קיימים מקרים כאשר תהליך מסיים את השימוש במשאב המעבד מופעלת פסיקה הגוררת החלפת קשר/תוכן.
2. בארכיטקטורות מסוימות למשל ארכיטקטורת X86 של אינטל (אותה סיקרנו) הן מונעות פסיקה מצד מערכת ההפעלה, משמעות הדבר היא שבמידה והמעבד צריך למשל לבצע קריאה/כתובה מהדיסק, הוא ישלח את בקשת הקריאה/כתיבה ויעבור בינתיים לבצע פעילות אחרת במקום להמתין למענה, עם סיום הקריאה מהדיסק תופעל פסיקה שתגרור החלפת הקשר, הפסיקה מהדיסק תטופל על ידי שגרת טיפול בפסיקה. (שגרת טיפול בפסיקה סיקרנו בפרק "תפקידי מערכת ההפעלה")
3. מצב נוסף שמצריך החלפת קשר/תוכן מעבר בין מצב משתמש (User Mode) למצב ליבה (Kernel Mode), תלוי מערכת הפעלה.

❖ הערה : לרוב ביצוע של החלפת קשר/תוכן מרוקנת את המעבד מכל תוכן הקיים בו לפני, כדי להתחיל תהליך אחר שלא קיים קשר ביניהם, התוכן שהיה במעבד לפני ההתרוקנות נשמר באוגר הצמוד אליו וחוזר אל המעבד בסיום התהליך החדש שיצא. (בעבר ביצוע Context Switch וריקון מעבד מתוכן היה באמצעות מטח חשמלי שהגיע אל המעבד, דבר שקיצר את תוחלת החיים של המעבד).

זיכרון Memory –

זיכרון פיזי הינה התקן חומרה המאפשר שמירה של ביטים, רצף של אחדות ואפסים. התקן החומרה הפיזי של הזיכרון משמש לאחסון נתונים והוראות של תוכניות, חלקם נשמרים בזמן ריצת התוכנית וחלקם נשמרים גם לאחר סיום ביצוע התוכנית. אנו משתמשים בהתקני זיכרון פיזיים ולא מסתפקים רק באוגרים או מטמונים כיוון שכמות המידע הנשמר בהם קטנה מאוד, וחלקם נמחק לאחר כיבוי המחשב. אנו משתמשים בנוסף בזיכרון ראשי (RAM) המשמש לאחסון נתונים ותוכנים אשר נמחקים בעת כיבוי המחשב, ובזיכרון משני (ROM) התקן זיכרון קבוע אשר הנתונים נשמרים בו גם לאחר כיבוי המחשב.

קיבולת בזיכרון נמדד לפי יחידות מידה :

מספר תאי הזיכרון	יחידת מידה
1,024 Byte – 1 KB	Kilobyte - KB
1,024 KB – 1 MB	Megabyte - MB
1,024 MB – 1 GB	Gigabyte - GB
1,024 GB – 1 TB	Terabyte - TB

לאחר שסיקרנו את הפרק על המעבד ראינו שללא שימוש בזיכרון המעבד אינו יכול כלל לתפקד, במיוחד שהפקודה הראשונה מסט הפקודות שלו היא Fetch – גישה אל הזיכרון והובאת הפקודה אל המעבד.

עוד סיקרנו שהמעבד עושה שימוש בזיכרון ראשי שהוא ה – RAM ובזיכרון משני שהוא ה – ROM. שימוש נוסף בסוגי זיכרונות – **אוגרים/ Register** ו - **מטמון / cache** כדי ליעל את תפקוד המעבד ולתת זמן תגובה מהיר מאוד.

גם מערכת הפעלה עושה שימוש קבוע בסוגי זיכרונות – **אוגרים/ Register** ו - **מטמון / cache**, ניהול ושימוש בסוגי זיכרונות אלו נותן למערכת ההפעלה זמני מהירות גבוהים ומאפשרים גישה נוחה למשתמש, ניקח דוגמא – מערכת ההפעלה עושה שימוש בזיכרון מטמון שמשוך לדפדפן בו אנו גולשים, זה נותן לנו את האפשרות שהדפדפן יזכור שם משתמש וסיסמה לאתר שאנו גולשים אליו.

מערכת ההפעלה מאפשרת לתהליכים ותוכנות שימוש בסוגי זיכרונות – **אוגרים/ Register** ו - **מטמון / cache**, כדי למקסם את חווית המשתמש לדוגמא – לתוכנות מסוימות יש אלגוריתם הנותן למידה על אופי המשתמש, התוכנה צריכה גישה כדי לשמור את הנתונים האלו ולעשות בהם שימוש בכל פעם שמשתמש מפעיל את אותה תוכנה.

לפני שניגש להיבט היררכי של סוגי זיכרונות במחשב חשוב שנדע את ההבדל בין זיכרון חשמלי – Volatile לבין זיכרון לא חשמלי – Non-Volatile :

זיכרון לא חשמלי – Non-Volatile : הכוונה לזיכרון המסוגל לשמור על הקבצים גם כאשר המתח החשמלי נפסק (כיבוי המחשב), זיכרון לא חשמלי הוא רכיב זיכרון המאפיין סוגי רכיבי זיכרון כמו – דיסק קשיח(ROM), USB, ודיס' אופטי.

זיכרון חשמלי – Volatile : הכוונה שברגע שהמתח החשמלי נפסק הזיכרון נמחק, אופייני לרכיבי זיכרונות בעלי נפח קטן אך נחשבים למהירים מאוד, כמו : Register, Cache, RAM . המטרה של זיכרון חשמלי לאכסן מידע שיש בו שימוש בזמן אמת לביצוע פעולות, לדוגמא הכי בסיסית תהייה הרצה של מערכת ההפעלה, אומנם היא אכן שמורה בזיכרון קשיח ולא חשמלי אך מערכת ההפעלה היא נחשב לתוכנה ובזמן שאנו מדליקים את המחשב ועולה מערכת ההפעלה זה בדיוק כמו הפעלה תוכנה, מערכת ההפעלה תיטען אל הזיכרון חשמלי כדי שתוכל לרוץ במהירות ותיתן למשתמש תפעול מהיר של אפליקציות ותהליכים הרצים עליה.

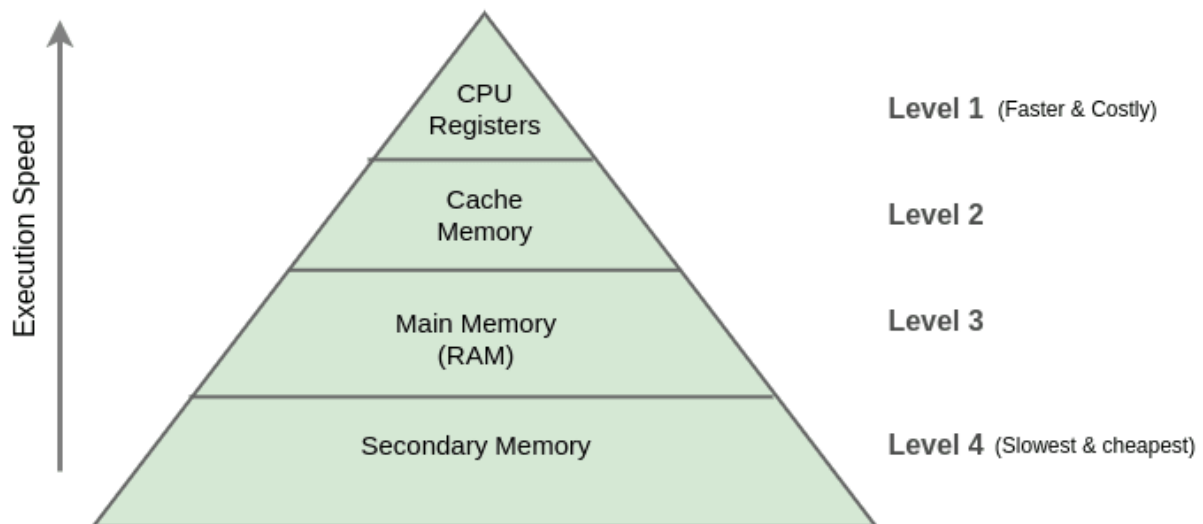
❖ הערה : זיכרונות כגון - **אוגרים/ Register** ו - **מטמון / cache** יכולים להישמר גם בזיכרון קשיח ולא חשמלי, על פי היעוד שאליו הם נועדו, שימוש של תוכנות/אפליקציות.

סוגי הזיכרון שמערכת ההפעלה עושה איתם שימוש במבנה היררכי –

תפקידה של מערכת ההפעלה הינו לנהל ביעילות את משאבי הזיכרון הקיימים פיזית במחשב, כי שסקרנו עד כה הסקנו שמידע שאנו זקוקים לו לעיתים קרובות ימצא בזיכרון מהיר יותר, ולהפך מידע שנשתמש בו לעיתים רחוקות יותר ישמר בזיכרון איטי יותר.

גישה אל מידע בזיכרון איטי יותר יכול לקחת זמן רב, ומכן מערכת ההפעלה מנסה לצמצם את הסיכוי שנצטרך מידע המאוחסן בזיכרון איטי (כדי שהמעבד לא ייגש לזיכרון הקשיח אלא רק לזיכרון מהיר וייתן תגובה מהירה), ככול שלמחשב יהיה רכיב זיכרון מהיר בעל משקל גדול יותר ככה כן למערכת ההפעלה תהייה עבודה קלה יותר, והסיכוי שמידע שאנו צריכים לעיתים דחופות ימצא בזיכרון מהיר (בלי שהמעבד יצטרך לגשת לזיכרון הקשיח).

לפנינו מוצג איור המקטלג את סוגי הזיכרון מהגדול יותר, זול יותר, ואיטי יותר אל זיכרון קטן יותר, יקר יותר, אך מהיר הרבה יותר. נפשט ונסביר על האיור מתחתית הפירמידה ועד ראש הפירמידה.



Secondary Memory – Level 4 – הכוונה בזיכרון משני לדיסק קשיח (ROM) – זיכרון איטי.

דיסק קשיח המכונה גם זיכרון משני, נחשב ל**זיכרון לא חשמלי – Non-Volatile**, שומר נתונים בלתי נדיף, דיסק קשיח משמש להכלה של נתונים בכמות גבוהה בהרבה מזיכרונות אחרים שיש במחשב, אך לעומת שאר הזיכרונות הדיסק הקשיח איטי יותר בפעולתו. נחשב לזיכרון הזול מבין שאר הזיכרונות בפירמידה.

דיסק קשיח יכול להיות גם מוטמע בתוך המחשב או חיצוני למחשב (רכיב המחובר חיצונית בכבל) או כדיסק אופטי.

נתייחס ל – 2 סוגים של דיסקים קשיחים ונבין את ההבדלים בניהם :
הראשון HDD.
השני SSD – הדור החדש של דיסקים קשיחים.

HDD מבנה הדיסק הקשיח – דיסק קשיח מורכב ממספר דיסקות או אחת בודדת (תלוי בגודל הדיסק) אשר עשויות מאלומיניום או מזכוכית המצופות בחומר מגנטי. לצד כל דיסקה יש ראש כתיבה וקריאה המותקן על זרוע. בזמן פעולת הדיסק, הדיסקות מסתובבות יחד ותפקיד ראש הכתיבה/קריאה לנוע על הדיסקה הלוח וחזור בכדי שיוכל להגיע לכל נקודה בדיסקה.

כדי לבצע הכתיבה/קריאה בדיסקה, ראש הכתיבה/קריאה צורב בפעולה מגנטית את המידע על הדיסקה בסמן של 0 ו- 1 (שפת מכונה).



דיסק קשיח מסוג HDD שהמכסה שלו הוסר

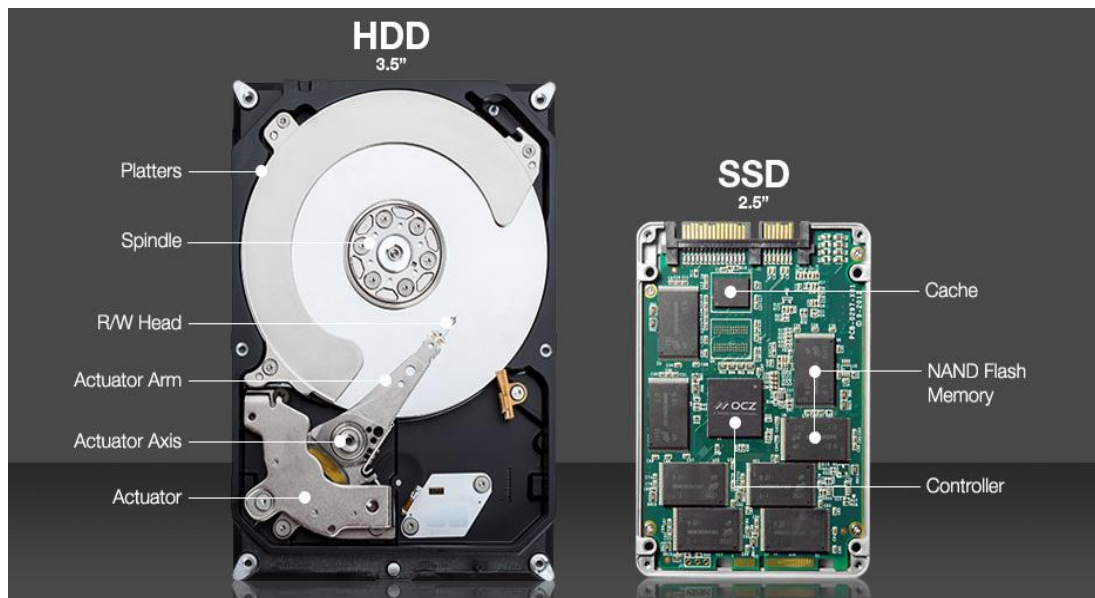
SSD מבנה וכיצד הוא עובד – SSD גם הוא נחשב לזיכרון בלתי נדיף המאפשר שמידע לא יימחק כאשר המתח החשמלי נפסק.

בניגוד לטכנולוגית דיסק קשיח HDD, ה-SSD מורכב בשונה לחלוטין, בטכנולוגית דיסק קשיח מסוג SSD לא יהיו דיסקות וראש כתיבה/קריאה, אלא לוח (כרטיס) אחד שעליו יהיו:

- **סוגי זיכרון Cache** - המשמש את ה-SSD.
- **controller** – המשמש לניהול הכרטיס, והזיכרון עצמו, שתפקידו לווסת ולפקח על התעבורה באפיק (bus) – סיקרנו קצת פני פרק המעבד.
- **NAD Flash Memory** - הזיכרון ב-SSD נצרב בטכנולוגיה הנקראת NAD Flash Memory - העיקרון מאחורי הטכנולוגיה היא שצריבת הזיכרון תתבצע באמצעות מטח חשמלי (המידע הופך צרוב ובלתי נדיף), יתרון בשיטה זו הינה כתיבה ומחיקה לתוך הזיכרון בכמות גדולה בכל פעם (מחיקה וכתיבה מתבצע בבלוקים בגדלים של עשרות ומאות KB's).

יתרונות של SSD מול HDD –

- SSD קטן בהרבה בגודלו הפיזי ושוקל פחות לעומת HDD.
- שיטת צריבת המידע על דיסק SSD מהירה יותר משיטת הצריבה של דיסק HDD.
- דיסק קשיח SSD שקט יותר בפעולה מאשר דיסק קשיח HDD.
- לדיסק קשיח SSD צריכת חשמל נמוכה ופיזור חום נמוך יותר מדיסק קשיח HDD.



SSD מול HDD (ללא מכסה)

כיצד מידע נשמר על דיסק קשיח –

בדיסק קשיח המידע נשמר בכפולות של 4KB – הנקרא (בלוק), גם אם קיים קובץ שאנו שומרים והוא קטן מ- 4KB הוא עדיין ישמר בבלוק שלם של 4KB.

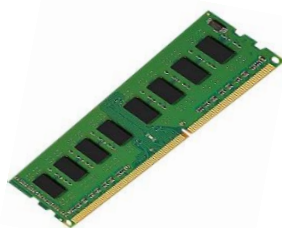
היתרון בשיטה זו היא יעול – מכיוון שזמן גישה של המעבד אל הדיסק הקשיח הוא ארוך, אז העדיפות להביא מהדיסק הקשיח אל הזיכרון מהיר בלוק שלם של המידע המבוקש, הזיכרון המהיר כבר יאתר את טווח הכתובות המבוקש מתוך אותו בלוק, (במקום שהמעבד יחפש את טווח הכתובות הספציפי מתוך כל נפח הדיסק הקשיח פעולה הנחשבת לזמן תגובה מאוד ארוך).

Level 3 – Main Memory – הכוונה בזיכרון ראשי, נקרא גם זיכרון מהיר (RAM).

זיכרון RAM נחשב לזיכרון חשמלי – Volatile, שומר מידע נדיף, כמו שסקרנו בחלקים הקודמים של המאמר ראינו שללא זיכרון RAM המחשב שאנו עובדים עליו יהיה איטי ונקבל זמני תגובה ארוכים, כיוון שהמעבד אינו מסוגל לשמור אצלו מידע רב בזמן עיבוד הנתונים, וגישה אל הזיכרון הקשיח אורכת זמן רב, פה נכנס השימוש של זיכרון RAM.

זיכרון זה שומר נתונים באופן זמני על מנת שיהיו מוכנים לשליפה מהירה עבור המעבד, המידע נשמר על פי עיקרון פיזיקלי של חשמל מגנטי, הזיכרון בנוי מקבוצה של טבעות בכל טבעת עוברים 3 – 4 כבלים (מזכיר מאוד מבנה פנימי של שנאי), הכתיבה והקריאה מהזיכרון מתבצע באמצעות מעבר זרם בכבלים העוברים בטבעות.

זיכרון RAM נראה בצורה של כרטיס, נפח הזיכרון הנפוץ בשוק מגיע ב 4 גיגה, 8 גיגה, ו 16 גיגה, כמובן שאפשר להרכיב עוד יותר (תלוי בלוח האם אם תומך או לא).



תמונות למחשה כיצד נראה כרטיס זיכרון RAM

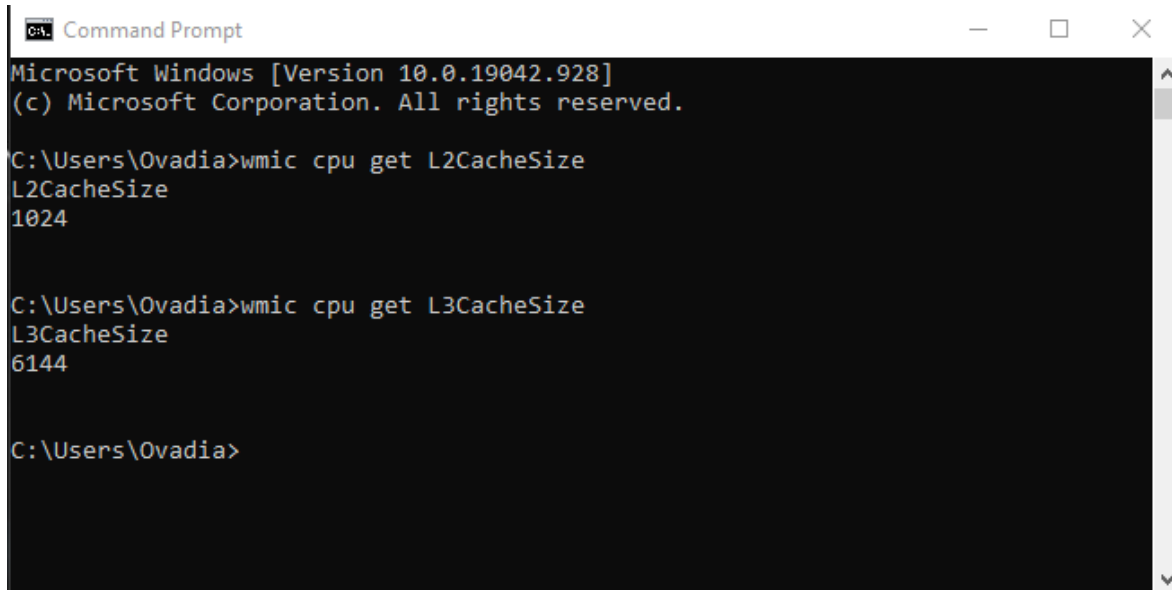
CPU Registers Level 1 - Cache Memory Level 2

את סוגי זיכרונות אלו סיקרנו לעומק בפרק של המעבד, סיקרנו שאלו רכיבי זיכרון מהירים ביותר (ויקרים יותר) לעומת שאר רכיבי הזיכרון במחשב, למדנו את תפקידם של סוגי זיכרון אלו, כיצד המעבד ומערכת ההפעלה עושים איתם שימוש, ושאלן תוכנות ויישומים נדרשים לעשות בהן שימוש, לצורך יעילות וזמני תגובה מהירים.

❖ הרחבת ידע כי למצוא את גודלם של זיכרונות Cache שיש לנו במחשב נריך פקודה באמצעות Command Prompt.

```
wmic cpu get L2CaheSize  
wmic cpu get L3CaheSize
```

כדי לאתר זיכרון L1 Cache פיזי, ניתן לבדוק מהו דגם המעבד שיש במחשב שבו אנו עובדים עליו יחד עם הביטוי L1 Cache



```
Command Prompt  
Microsoft Windows [Version 10.0.19042.928]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Users\Ovadia>wmic cpu get L2CacheSize  
L2CacheSize  
1024  
  
C:\Users\Ovadia>wmic cpu get L3CacheSize  
L3CacheSize  
6144  
  
C:\Users\Ovadia>
```

כתובות בזיכרון –

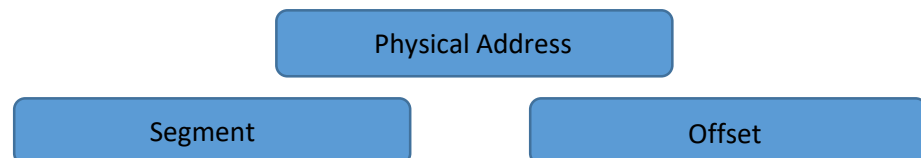
לכל תא בזיכרון (כל סוג של זיכרון) יש כתובת משלו, כתובת זו מייצגת את המיקום בזיכרון של תא בודד המשמש לאחסון מידע. למשל שאנו פותחים קובץ טקסט המכיל מספר מסוים של KB אותו קובץ מתפרש על מספר תאים בזיכרון, לכן יש לנו טווח כתובות לקובץ, כדי שמערכת ההפעלה תדע לשלוף את אותו הקובץ שבחרנו מהזיכרון, היא נעזרת בייצוג המיקום של הקובץ בזיכרון (טווח כתובות), בעזרת טכניקה זו נקבל תגובה מהירה והקובץ שבחרנו יעלה בכמה מאיות השנייה.

נסקור כיצד מורכב טווח כתובות בזיכרון וכיצד כתובת ווירטואלית מומרת לכתובת פיזית.

כתובת פיזית – Physical Address

כתובת פיזית נקראת גם כתובת מוחלטת, כתובת פיזית מורכבת מ – 20 סיביות. כאשר לתהליך/תוכנית יש דרישה לגשת לזיכרון הכתובת הראשונה תהייה בעלת 20 סיביות, וציון שאר הכתובות במהלך קריאה/כתיבה של התוכנית לזיכרון יהיו בעלות 16 סיביות, 4 סיביות ראשונות ישמשו כקידומת לכתובת הראשונה בעלת 20 הסיביות.

הסבר – כתובת בעלת 16 סיביות נקראת גם כתובת יחסית, כיוון שהיא יחסית לכתובת הראשונה שהיא בעלת 20 סיביות, ניתן לדמות את השיטה לקידומת של מספר טלפון. (לדוגמא בעבר 3 ספרות ראשונות של ניידים ייצגו את החברה ממנה הם מקבלים שירות).



Physical Address – כתובת המציינת באופן מולט תא ספציפי בזיכרון.

Segment – המעבד מחולק למקטעים הנקראים סגמנטים.

Offset – נקרא גם היסט מצביע על כך שהכתובת מבטאת מרחק בבתים מתחילת הסגמנט.

❖ כדי להמיר כתובת יחסית לכתובת פיזית, יש לדעת את הכתובת של תחילת הסגמנט וגם את ההיסט של תחילת הסגמנט.

בשילוב שלושת האובייקטים האלו ניתן להגדיר טווח כתובות לקובץ אחד.

כתובת ווירטואלית – Logical Address

כתובת לוגית נוצרת ע"י המעבד, כתובת לוגית נקראת כתובת ווירטואלית כיוון שהיא אינה קיימת פיזית, הכתובת הווירטואלית משמשת כפניית גישה למיקום בזיכרון הפיזי ע"י המעבד.

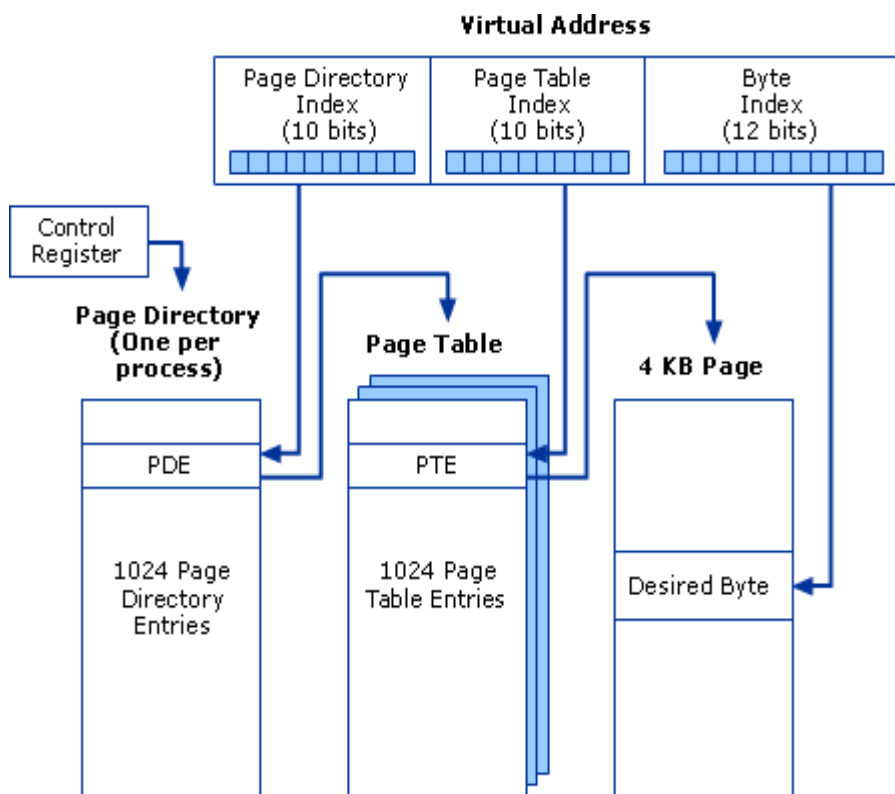
כדי להמיר כתובת ווירטואלית לכתובת פיזית המעבד נעזר ב - **Memory Management Unit – MMU** יחידה פיזית המשמשת להמרת כתובות ווירטואליות לפיזיות שסקרנו בפרק המעבד.

כתובות ווירטואליות עונות לנו על כמה צרכים :

- בעזרת כתובות ווירטואליות כל התהליכים הרצים בזמן אמת ירוצו בזיכרון RAM – מה שיביא לתגובה מהירה מאשר טעינה וריצה מזיכרון (ROM).
- לכל תהליך/תוכנית בעזרת כתובות ווירטואליות יש מרחב זיכרון משלו (אין מצב של דריסת מידע).
- ע"י כתובות ווירטואליות עיקרון מעגלי הרשאה (אותם סקרנו בחקר מערכת ההפעלה) באים לידי ביטוי, בכך שתהליכים/תהליכונים ותוכניות בעלי הרשאה נמוכה ירוצו במעגל הרשאה של - Userland בלבד, ואילו תהליכים/תהליכונים ותוכניות בעלי הרשאה גבוה יורשו לרוץ במגל הרשאה של - Kernel, כלומר כל אזור בזיכרון מוגדר לאיזה מעגל הרשאה הוא שייך. בעזרת טכניקה זו מנענו וירוסים ובאגים כמו "Meltdown" – גורם לזיכרון להפוך לגוש אחד ללא מקטעים ובכך יש גישה מלאה לאזור ה- Kernel, ואפשרות לדריסת מידע.

כיצד נראה מיפוי של כתובת ווירטואלית –

כדי שכתובת ווירטואלית תקבל מיקום פיזי (ממורת לכתובת פיזית), מערכת ההפעלה חייבת לדעת לאיזה Page Table בדיוק מתייחס ה- PTE שיש בכתובת הווירטואלית (סיקרנו בפרק המעבד). ובשביל זה יש לה טבלה נוספת Page Directory (PDE) סיקרנו את אופן השיטה בפרק על המעבד) - הקובעת לאיזה Page Table יש להגיע בכדי לפענח את הכתובת הווירטואלית.



האיור לפנינו ממחיש כיצד מתרחש תהליך המרה של כתובת ווירטואלית לכתובת פיזית.

רק נסביר שכתובת ווירטואלית בעלת 32 ביט מתחלקת ל 3 חלקים.

10 ביטים ראשונים – קובעים באילו Page Table נשתמש.

10 אחרים – קובעים אילו Page Table Entry ניקח.

12 ביטים אחרונים – קובעים את ה- Offset מתחילת ה- Page Frame

❖ הערה – בפרקים הקודמים הוזכרו מושגים כמו תוכניות ותהליכים, נסדר את הדברים ונגדיר את ההבדל ביניהם:

- תוכנית – Program: היא למעשה קובץ Executable שבוא הקוד הכתוב לפני ביצוע (סט הוראות שייעודם לביצוע פעולה ספציפית).
- תהליך – Process: הוא ביצוע של התוכנית (Program), הקוד הכולל טעינה מהזיכרון אל המעבד.

תהליכים ותהליכונים – Processes & Threads :

תהליכים ותהליכונים הם חלק מרכזי בהבנת אופן הפעולה של מערכת ההפעלה , כל תוכנה שאנו מפעילים והיא רצה במחשב שלנו נמצאת תחת "תהליכים" – Processes.

מערכות הפעלה מודרניות מאפשרות לנהל אוסף של תהליכונים (Threads) במסגרת ריצה של תהליך (Process) באותו מרחב כתובות (באותו טווח כתובות בזיכרון).

תהליכון Thread - תהליכון מייצג אוסף של פקודות המשמשות לביצוע פעולה אחת, לדוגמא נפעיל את התוכנה של דפדפן האינטרנט Chrome - בפעולה זו מערכת ההפעלה תפתח עבורנו מאחורי הקלעים "תהליכון ראשי" המיצג את דפדפן Chrome , אוסף הפקודות ש – Chrome מריץ הוא Thread של התוכנה Chrome, כפי שלמדנו עד כה כל הפקודות המרכיבות את תוכנת Chrome ייטענו אל הזיכרון המהיר (RAM) ומשם יעבוד למעבד לביצוע.

לאחר יצירת "תהליכון ראשון" ייבצרו לאותה תוכנה תהליכונים נוספים - מערכת ההפעלה מאפשרת לאותה תוכנה ליצור עוד תהליכים באותו מרחב זיכרון (הקשורים לאותה תוכנה) , ובכך מערכת ההפעלה מאפשרת למשתמש מהירות תגובה ורציפות פעולה כאשר תוכנה (במקרה של הדוגמא שלנו דפדפן Chrome) לבצע מספר משימות במקביל.

- ❖ בעבר למחשבים בעלי מעבד בודד לא הייתה אפשרות לבצע ריצה של כמה תהליכונים במקביל, כדי לקבל אפשרות ריצה של כמה תהליכונים במקביל הטמיעו בקוד אפשרות של קפיצה – קפיצה זו נקראת (Context Switch), כפי שסקרנו בפרק על המעבד, Context Switch מבצעת שמירה של המצב הקיים לאורג/למטמון במעבד, נטישה של התהליכון ומעבד לתהליכון אחר, בסיום הריצה של התהליכון החדש חוזר למעבד המידע הנשמר באורג/מטמון של התהליכון הקודם ומאותה נקודת הפסקה הקוד ממשיך לרוץ.
- פעולת Context Switch היא פעולה יקרה יחסית מבחינת זמן עיבוד ועל כן משתדלים מפתחי מערכות הפעלה לשפר את זמן ההחלפה ואף לשלב אלגוריתם ייעול לצמצום מספר ההחלפות.

כיום שיש למחשב מספר מעבדים אין שימוש ב – Context Switch מלא כדי לעבור מתהליכון אחד לאחר אלא מתבצעת פעולה הנקראת – Multi Threads .

ריבוי תהליכונים Multi Threads –

מערכת הפעלה מודרנית תומכות בריבוי תהליכונים, כיצד זה מתבצע – תהליך (Process) אחד מאפשר לכל התהליכונים (Threads) תחתיו גישה לאותו אזור בזיכרון שהתהליך (Process) רץ בו. מצב זה מאפשר שיתוף מידע פשוט ומהיר בין אותם תהליכונים.

בנוסף כאשר תהליכון אחד מקבל גישה ממערכת ההפעלה למשאב מסוים, שאר התהליכונים שרצים באותו טווח כתובות בזיכרון (תהליכונים משותפים) יורשו גם הם להשתמש באותו משאב במידה והם צריכים.

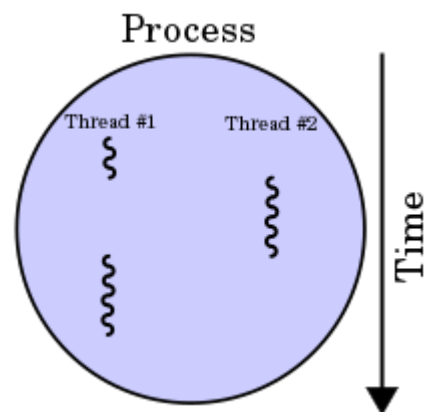
במצב של ריבוי תהליכונים, החלפת קשר (Context Switch) ע"י מערכת ההפעלה יבוצע באופן חלקתי, מכווין שריבוי תהליכונים הפועלים תחת תהליך (Process) מסוים הם משותפים, מערכת ההפעלה והזיכרון הצמוד למעבד (אוגר או מטמון) כבר יודעים כיצד להגיע לאזור הזיכרון של אותו תהליך שתחתיו רצים התהליכונים, ובכך המעבד לא צריך לנקות את כל הזיכרון אוגר/מטמון שלו כדי לתמרן בין אותם תהליכונים הרצים תחת תהליך אחד, מצב המייעל ומזרז את עבודת המעבד ומערכת ההפעלה.

כיצד מערכת ההפעלה מאפשרת עבודה של מספר תהליכונים (Threads) במקביל :

כל תהליכון (Thread) מעביר למעבד פקודות מכונה שונות, פה משתלב רכיב שסיקרנו בפרק המעבד הנקרא מתזמן (Scheduler), כפי שלמדנו המתזמן ע"י אלגוריתם שמערכת ההפעלה מנהל איזה תהליכון (Thread) יקבל זמן מעבד.

- ❖ כיום במערכות הפעלה מודרניות ומרובות מעבדים, משתמשות במתזמן (Scheduler) כדי לתמרן בין תהליכונים (Threads) אפילו אם הם שונים ולא קשורים לאותו תהליך (Process), מצב שכאשר תהליכון (Thread) המקבל קלט מהמשתמש יהיה חסר עבודה, יוענק משאב המעבד לתהליכון (Thread) אחר הזקוק למשאב כרגע.

איור הממחיש כיצד תחת תהליך אחד יש 2 תהליכונים, הראשון רץ עד לעצירה, אז השני נכנס לריצה ושהוא מסיים תהליכון ראשון חוזר מאותה נקודה שעצר ועד לסיום.



כדי לבחון כיצד נראה ריבוי תהליכונים לתהליך מומלץ להגיע לפרק המסביר על **Sysinternals Suite** להיכנס לכלי **Process Explorer** לעבור על המדריך, אפשר לראות כיצד תהליך מנותח לעומק ואת מספר התהליכונים הרצים תחתיו.

תהליכים Processes -

כפי שהוזכר תהליך (Process) הוא הביצוע של תוכנית (Program) במחשב המופעלת ומנוהלת ע"י מערכת ההפעלה, (תוכנית היא אוסף פקודות, תהליך הוא ביצוע/הפעלה של אותן פקודות).

לכל תהליך יש ייצוג במערכת ההפעלה אופן הגדרות וכללים לפני שהוא פועל, נמנה כמה הגדרות :

- העתקה ושליפת אוסף הפקודות של התוכנית והכנתם לריצה במעבד.
- מיפוי הזיכרון של תהליך במרחב הזיכרון הווירטואלי והמרה לזיכרון פיזי .
- רשימת משאבים זמינים שהוקצו לתהליך ע"י מערכת ההפעלה.
- הרשאות תהליך – מערכת ההפעלה צריכה לדעת מידע אודות מי הבעלים של התהליך, ובאיזה מעגל הרשאות תהליך רשאי לרוץ בו, בין אם ב – Kernel או ב – Userland.
- מערכת ההפעלה צריכה לדעת את מצב המעבד לפני הפעלת תהליך, כדי לאפשר למעבד לבצע החלפת קשר (Context Switch), שמירה של מצב קיים באוגרים, ומיפוי כתובות ווירטואליות לפיזיות.

עד כה סקרנו שתהליך (Process) מכיל את כל המשאבים המשותפים לתהליכונים (Threads) הרצים תחתיו, אפשר לומר שתהליך (Process) הוא מעין קונטיינר ואין לו אפשרות להריץ קוד בעצמו, הוא נעזר בתהליכון אחד (לפחות) כדי שירץ את הקוד.

לכל תהליך (Process) שנפתח מוקצה אוסף משאבים שאותם הוא חולק עם תהליכונים (Threads) תחתיו, נמנה מספר משאבים משותפים :

- **מזהה תהליך PID Process ID –** מערכת ההפעלה מעניקה לכל תהליך מספר מזהה ייחודי.
- **הרשאות אבטחה –** מערכת ההפעלה מנהלת רשימות המונות הרשאות אבטחה על מנת לגשת למשאבי החומרה, כל תהליך מקבל רמת הרשאה משלו (לפי ייעודו) אותה הוא חולק עם תהליכים תחתיו.
- **תהליכון (Thread) –** כדי שתהליך (Process) יוכל לרוץ הוא חייב לפחות תהליכון (Thread) אחד כדי שאותו תהליכון יריץ עבורו את הקוד, כאמור תהליך נחשב ל"קונטיינר" ואינו יכול להריץ קוד בעצמו.
- **הקצאת זיכרון –** מערכת ההפעלה מקצה זיכרון עבור כל תהליך (Process), תהליך חולק עם תהליכונים תחתיו את אותו מרחב זיכרון שהוקצה לו.
- **קוד הרצה הנטען אל זיכרון RAM –** כאשר תהליך (Process) מסוים הופעל כל אותם תהליכונים (Threads) הרצים תחתיו חולקים את אותו הקוד, הייעול במצב זה הוא שכל תהליכון יכול להשתמש בחלק אחר של הקוד ובכך המעבד לא צריך לטעון בכל פעם קוד אחר לצורך פעולה משותפת לתהליכונים הרצים תחת אותו התהליך.

הבדלים בין תהליכים Processes לבין תהליכונים Threads

<u>Threads</u>	<u>Processes</u>
משתתף במשאבים שהוקצו לתהליך (Process)	מקבל משאבים ישירות ממערכת ההפעלה
משתף משאבים עם תהליכונים (Threads) אחרים תחת אותו התהליך (Process)	לא משתף משאבים עם תהליכים אחרים
בעל יכולת להריץ קוד	לא מסוגל להריץ קוד, חייב שיהיה לו תהליכון (Thread) שירץ עבורו את הקוד
חייב שיהיה לו תהליך (Process) שייצור אותו	מסוגל להכיל כמה תהליכונים (Threads)

ריבוי תהליכים Multi Processes –

כאמור כפי שראינו תהליך (Process) אחד מכיל מספר תהליכונים (Threads) משותפים, רצים וחולקים ביניהם את המשאבים המוקצים לתהליך (Process), מצב יעיל אך אינו מספיק, כדי לקבל תגובה והספק גבוה מערכת ההפעלה משתמשת "בעיבוד רב" – ריבוי תהליכים. ריבוי תהליכים מאפשר למערכת ההפעלה בעזרת מתזמן (Scheduler) להריץ מספר תהליכים במקביל כאשר כל תהליך ירוץ על מעבד או ליבה נפרד.

לרוב מערכת ההפעלה מעדיפה להשתמש בריבוי תהליכים וליצור לכל תוכנה תהליך, מאשר ריבוי תהליכונים. להבדיל מריבוי תהליכונים (Multi Threads) יש לריבוי תהליכים (Multi Processes) מספר יתרונות מוחלטים:

- במצב עבודה של ריבוי תהליכים (Multi Processes) כל תהליך עובד במרחב זיכרון נפרד (מאשר מצב עבודה בריבוי תהליכונים שכל תהליכון עובד במרחב זיכרון משותף לתהליך), במצב שכל תהליך עובד במרחב זיכרון נפרד, גם אם תהליך מסוים קורס אפשר לסיים אותו ויתר התהליכים ימשיכו לעבוד ללא כל הפרעה.
- כיום לרוב המחשבים המודרניים יש מספר מעבדים, והם עובדים ביעילות, כאשר כל מעבד יריץ תהליך (Process) יהיו משויכים לו מספר תהליכונים (Threads), ומעבד אחר יריץ תהליך (Process) שונה יהיו משויכים לו מספר תהליכונים (Threads), מערכת ההפעלה מנצלת כל מעבד לעבודה על תהליך עם ריבוי תהליכונים משלו, ובכך היא מאפשרת לנצל בצורה טובה יותר את ריבוי המעבדים במחשב.
*לרוב מערכת ההפעלה לא תפרוס תהליך בעל ריבוי תהליכונים על כמה מעבדים, פעולה זו תהייה לא יעילה מכיוון שהמעבד יאלץ להחליף את טבלאות מיפוי הזיכרון שלו מספר פעמים (כדי למצוא את טווחי הכתובות בזיכרון).
- יתרון אבטחתי – לכל תהליך מערכת ההפעלה מגדירה מראש לאיזה מעגל הרשאה הוא שייך ובאיזה משאבים הוא יכול להשתמש, אם מערכת ההפעלה הייתה מעדיפה לעבוד לרוב במצב עבודה של ריבוי תהליכונים (Multi Threads) אז כאמור תהליכונים היו יורשים את ההרשאות שהוקצו לתהליך תחתיו הם רצים, מצב שיגרור סיכון מועט במערך מעגלי ההרשאה, כמובן שבמצב עבודה כזה קוד זדוני יוכל בקלות להשתלב במערכת ולרשת הרשאות מתהליך שהוא אינו אמור לקבל.

זיכרון משותף –

עד כה למדנו שכל תהליך (Process) רץ במרחב זיכרון (טווח כתובות בזיכרון) השמור רק לו, זיכרון משותף הוא זיכרון המשותף בין שני תהליכים או יותר. אין הפרעה לשתף זיכרון על תוכניות שעשויות לרוץ על מעבד יחיד או על מספר מעבדים, בין היתר שיטה זו מאפשרת תקשורת בין תהליכים ומניעת עותקים מיותרים.

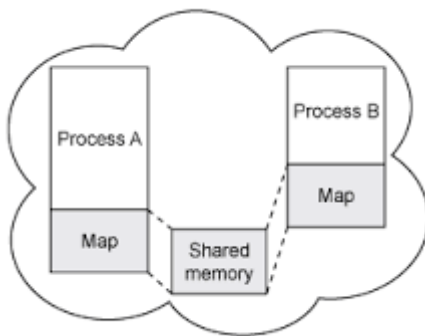
זיכרון משותף מעביר תקשורת נחוצה בין בתהליכים, כדי להבין ניתן דוגמא – ניקח את התוכנה **OneDrive** של **Microsoft** בגרסה העסקית של התוכנה ניתנת אפשרות לעבוד על קובץ מכמה תחנות שונות בזמן אמת, זה אומר שהתוכנה צריכה לקבל קלט מכמה תחנות בזמן אמת, להוציא פלט (הדפסה על המסך) מכמה תחנות בזמן אמת, לבצע שמירה אוטומטית, ושכל זה יסונכרן במאיות השנייה בכמה תחנות שונות.

לרוב זיכרון משותף יתבצע באמצעות **תקשורת בין תהליכים – IPC Inter-Process Communication**.
IPC – הינו אוסף של מנגנונים ושיטות כדי להעביר נתונים בין תהליכים הפתוחים תחת אותו תהליך אב, או בין תהליכים לגמרי שונים.

יכולת התקשורת בין התהליכים יכולה להתבצע על תחנה אחת או על מספר תחנות שונות המחוברים באותה הרשת.

IPC – תקשורת בין תהליכים אחראית לביצוע מספר פעולות:

- סנכרון.
- זיכרון משותף.
- העברת מסרים.
- הפעלת פרוצדורות מרחוק.



תמונת המחשה כיצד שני תהליכים משתמשים בזיכרון משותף

כיצד זיכרון משותף מתבצע ברמת החומרה:

ברמת החומרה זיכרון משותף מתייחס לטווח כתובות גדול (הנקרא גם בלוק) של זיכרון RAM אליו ניתן לגשת באמצעות יחידת עיבוד מרכזית (מעבד) או במערכת מחשב מרובת מעבדים (מחשבים בעלי מספר מעבדים).

- גישה אחידה לזיכרון – **UMA – Uniform Memory Access**: המעבד או כל המעבדים (תלוי חומרת מחשב) חולקים את הזיכרון הפיזי/ווירטואלי בצורה אחידה.
- גישה לא אחידה לזיכרון – **NUMA – Non Uniform Memory Access**: זמן הגישה אל הזיכרון תלוי במיקום הזיכרון ביחס למעבד.
- ארכיטקטורת זיכרון מטמון בלבד – **COMA – Cache Only Memory Architecture**: הזיכרונות המקומיים עבור המעבדים בכל צומת משתמשים כמטמון במקום לגשת לזיכרון הראשי (RAM) לצורך מהירות ותגובה מהירה (בזמן סיום שימוש בתוכנה או ביצוע שמירה מצד המשתמש המידע מועתק לזיכרון לא מחיק).

סנכרון בין תהליכים –

עד כה למדנו שתהליכים שונות רצים עבור תהליך אחד משתפים בניהם מידע, זיכרון, ומשאבים, גם למדנו שתהליכים שאינם קשורים אחד לשני יכולים לשתף בניהם מידע, זיכרון, ומשאבים.

כדי שמערכת ההפעלה תצליח לנהל מערך סנכרון ושיתוף מידע בין תהליכים הקשורים לאותו תהליך ובין תהליכים השונים זה מזה, למערכת ההפעלה יש כלי אחראי ליישום וניהול של סנכרון ושיתוף מידע/זיכרון/משאבים והוא נקרא **מנהל האובייקטים – Object Manager**

מנהל האובייקטים – Object Manager – אחראי על ניהול משאבי המחשב, משאבים במערכת ההפעלה מוצגים **כאובייקטים – Objects** אובייקטים יכולים להיות חומרה פיזית, מרחב זיכרון מוגדר, קבצים, תיקיות, ערכי רישום, וגם תהליכים פועלים שלמים (המריצים תהליכים). מנהל האובייקטים אחראי על ניהול משאבים משותפים ולכן גם תתי מערכות הרצות המערכת ההפעלה העוסקות גם הן בצורך במשאבים צריכות עבור דרך מנהל האובייקטים.

מנהל האובייקטים עוקב אחר המשאבים המוקצים יש לו יכולת לנהל על סוג של משאב, מסוגל לשומר תיעוד של האובייקטים הנמצאים בשימוש בזמן אמת באמצעות ספירת הפניות למשאבים כמו גם את פרטי בעלות תהליכים (מי אחראי על התהליך), בנוסף מנהל האובייקטים בודק את ההרשאות של כל בקשה אליו – אם תהליך יגיש לאובייקט מסוים אך אין לו את ההרשאה לכך התהליך לא יקבל גישה למשאב.

כל בקשה של תהליכים הנפתחת אל מול מנהל האובייקטים בכדי לקבל גישה לאובייקט מסוים נקראת **Handle** – התרגום החופשי אינו משקף את התפקיד של Handle.

Handle – מסוגל להחזיק כמה פניות לאובייקט או מספר אובייקטים לפניה, מסוגל להחזיק במרחב זיכרון לביצוע פעולה, בשיטה זו תהליכים ותהליכים יכולים לפנות אל Handle אחד או לכמה Handles כדי לבצע סנכרון בניהם באופן מלא ומאובטח. זה אומר שכאשר תהליך/תהליך צריכים לבצע סנכרון בניהם הם יכולים לגשת לאותו Handle או Handles, שימוש ב Handle חוסך למעבד זמן גישה לטבלאות מיפוי והחלפת Cache.

עד כה סיקרנו את הצד של מערכת ההפעלה כיצד היא משתפת סנכרון בין תהליכים/תהליכים כדי למקסם הבנה בנושא נתייחס לצד התוכנה, כיצד היא פונה אל מערכת ההפעלה ומבקשת לבצע שימוש במשאבים לצורך סנכרון בין תהליכים של אותה התוכנה או בין היתר סנכרון בין תהליכים שונים.

בכדי למקסם הבנה נדרשת הבנה קונקרטית של מושגי מערכת ההפעלה לתכנון/פיתוח יישומים חכמים, לפי המינוח של מערכת ההפעלה מתכנת/מפתח אשר יבצע בקוד אפשרות של סנכרון בין תהליך/תהליכים ישתמש בשני טכניקות הנקראים **Mutex** ו- **Semaphore** שהם משאבי ליבה (Kernel) המספקים שירותי סנכרון.

Mutex ו- **Semaphore** – הם סוגי "מנעולים" עליהם לבחור איזה תהליך/תהליך יקבל את השימוש במשאב ברגע זה (מצב זה חוסך למערכת ההפעלה ביצוע של Context Switch של המעבד), **Mutex** ו- **Semaphore** – נחשבים למשאבים של מערכת ההפעלה, והכרחי להבטיח כי "מנעולים" אלו יהיו מוכרים ע"י תהליכים (Threads) ותהליכים (Processes).

Mutex ו- **Semaphore** – מסוגלים לספק לתהליך / תהליך אובייקטים או Handles שהם צריכים כדי להסתכנן בניהם.

הבדלים הין מנגנון Mutex לבין מנגנון Semaphore -

Mutex – נחשב למנגנון נעילה המשמש לסנכרון גישה למשאב, רק בקשה / משימה אחת (יכולה להכיל שרשור של משימות אבל של אותה משימה ראשית) יכולה לרכוש גישה אל ה – Mutex, משתמע מכך שישנה בעלות של תהליך הקשור ל – Mutex ורק הבעלים יכול לשחרר את הנעילה. הכוונה בכך באשר תהליך משתמש במנעול זה עד שהוא לא מסיים את השימוש ומשחרר את המנעול אף אחד אחר אינו ראשי להשתמש בו.

Semaphore – מנעול העובד בצורה שונה, מעיין מנגנון "איתות" המאפשר לתהליכונים או לתהליכים לאותת לתהליכונים או לתהליכים אחרים כאשר אירוע מתרחש. עובד ברמה מספרית, עם יצירת המנגנון מוגדר מספר מקסימלי כלשהו, תפיסת המשאב מורידה את ערכו ב – 1, שחרור המשאב מעלה את ערכו ב 1, אם ערכו של משאב Semaphore שווה לאפס אז לא ניתן לתפוס בעלות על המשאב (כך תהליכים ותהליכונים מודעים לכך שהמשאב תפוס).

Sysinternals Suite

Sysinternals – מאגר המונה מספר גדול של כלים עבור Windows
את SysinternalsSuite פיתחו מארק רוסינוביץ' ו- וברייס קוגסוול, בשנת 1996.
בתאריך 18 ביולי 2006, נרכשה ע"י Microsoft.
מאגר הכלים חינמי וניתן להוריד אותו ללא כל התקנה, מתאים לכל גרסה של מערכת הפעלה Windows.

לינק להורדה - <https://docs.microsoft.com/en-us/sysinternals/downloads/sysinternals-suite>

אוסף כלים אלו פותחו כדי לתת מעקב אחר תפעול מערכת ההפעלה ואחר תפקוד רכיבי המחשב,
בין היתר תפקידים של כלים אלו כולל: ניטור, ניתוח, ופתרון בעיות.
המערכת מכילה מגוון רחב של כלים המחולקים לקטגוריות:

- ניהול קבצים והדיסק הקשיח (File and Disk Utilities)
- ניהול תהליכים (Process)
- מידע על המערכת (System Information)
- כלי אבטחה (Security)
- רשתות (Networking)
- תוספות (Miscellaneous)

במדריך יוצגו מספר כלים, לא נסקור את כל הכלים המצויים ב- SysinternalsSuite, הסבר מפורט כיצד
הכלי עובד מה מטרותיו וצילומי מסך.

הכלים שיוצגו הם:

- Process Explorer
- Autoruns
- RAMMap
- VMMap
- CPUStress
- Cacheset
- WinObj
- TCPView
- Desktops

Process Explorer

מטרת כלי זה לתת דיאגנוזה בזמן אמת אודות תהליכים ותהליכונים הרצים במערכת ההפעלה, לרוב משתמשי Windows יצא להשתמש בכלי הנקרא Task Manager בין אם להרוג תהליך תקוע או לצפות בניצול משאבי המחשב, ועוד ..
כלי Process Explorer הוא מעיין הרחבה מעמיקה לכלי Task Manager.
Process Explorer - נותן מידע אודות תהליכים ותהליכונים הרצים במערכת ההפעלה בזמן אמת, מספק מידע אודות תהליכונים (כמה תהליכונים רצים תחת אותה תהליך).
באמצעות Process Explorer ניתן לצפות בשמות ב- Mutex שתהליכים מסוימים עושים אתו שימוש. נוכל לראות איזה תוכנות עושות שימוש בקבצי DLL.
תהליך ע"י לחיצה על Properties של תהליך נוכל לראות ניצול הוא עושה במשאבי המחשב.

כדי להגיע אליו בחבילת SysinternalsSuite לוחצים על - procexp64.exe

procexp.exe	11/09/2020 15:06	Application	2,733 KB
procexp64.exe	11/09/2020 15:01	Application	1,456 KB

כלי Process Explorer נותן לנו תצוגה מעמיקה של תהליכים הפתוחים על המחשב, על המשאבים שתהליך צורך מהמחשב בזמן אמת –

Process	CPU	Private Bytes	Working Set
RuntimeBroker.exe	< 0.01	12,812 K	41,520 K
SettingSyncHost.exe		4,684 K	6,552 K
YourPhone.exe	Susp...	24,976 K	23,188 K
RuntimeBroker.exe		2,748 K	17,428 K
RuntimeBroker.exe		3,892 K	18,420 K
TextInputHost.exe	0.03	13,776 K	52,900 K
RuntimeBroker.exe		2,688 K	13,788 K
Cortana.exe	Susp...	28,016 K	77,628 K
RuntimeBroker.exe		2,748 K	15,912 K
transport_proxy.exe		2,344 K	8,528 K
ApplicationFrameHost.exe		25,676 K	34,844 K
WinStore.App.exe	Susp...	51,640 K	3,892 K
RuntimeBroker.exe		5,640 K	21,884 K
SystemSettings.exe	Susp...	24,812 K	3,020 K
UserOOBEBroker.exe		2,020 K	9,440 K
WWAHost.exe	Susp...	90,744 K	3,472 K
dllhost.exe	< 0.01	5,972 K	13,976 K
ShellExperienceHost.exe	Susp...	14,328 K	62,948 K
RuntimeBroker.exe		4,828 K	17,376 K
Video.UI.exe	Susp...	19,908 K	60,256 K
RuntimeBroker.exe		1,704 K	8,084 K
backgroundTaskHost.exe	Susp...	15,424 K	28,668 K
RuntimeBroker.exe		3,240 K	21,068 K
dllhost.exe		3,068 K	9,732 K
smartscreen.exe	0.25	8,340 K	23,672 K
MicTray64.exe		2,324 K	3,000 K
taskhostw.exe	< 0.01	7,520 K	18,104 K
sihost.exe			

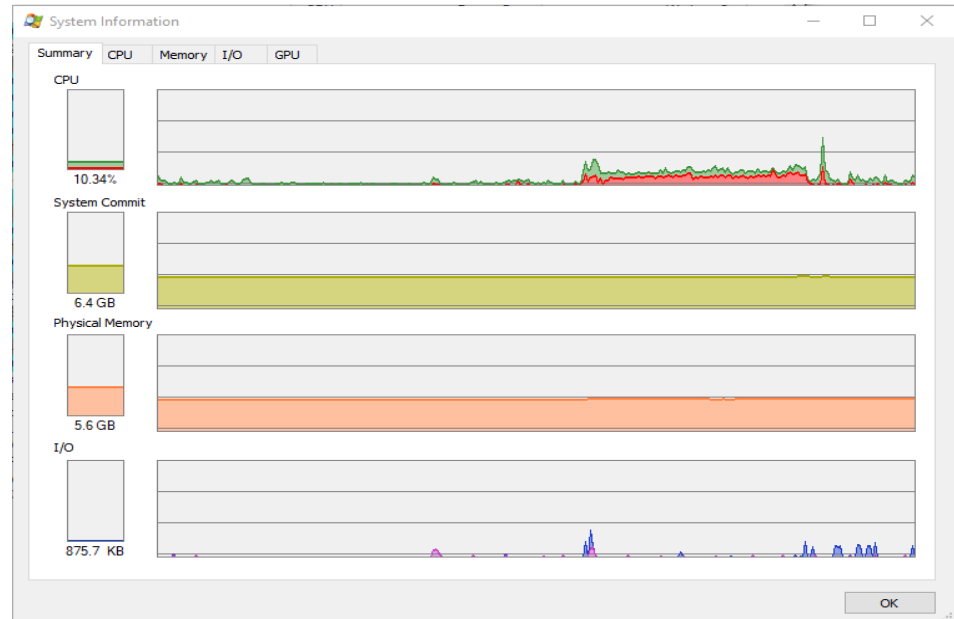
Type	Name
ALPC Port	\RPC Control\OLE0A3777420907AEF67A69DAD2BB25
Desktop	\Default
Directory	\KnownDlls
Directory	\Sessions\1\BaseNamedObjects
Event	\KernelObjects\MaximumCommitCondition
File	C:\Windows\System32

CPU Usage: 20.87% Commit Charge: 35.45% Processes: 266 Physical Usage: 35.98%



ע"י לחיצה על כפתור System Information יפתח חלון המציג סיכום של ניצול משאבי המחשב הכולל תצוגה של:

- CPU - ניצול מעבד.
- MEMORY – ניצול זיכרון.
- I/O – קלט/פלט – את כמות הנתונים שעוברים.
- GPU – מציג את השימוש בזיכרון המערכת השמור לגרפיקה.



ע"י לחיצה על כפתור Users -

נקבל מידע אודות איזה USER עובד על המחשב, ואם נלחץ על החץ יפתחו לנו אפשרויות הכוללות Connect, disconnect, logoff, Remote Control, & Send Message.

ע"י לחיצה על כפתור Process - ונסמן תהליך רצוי

נקבל אפשרויות שליטה של הריגת התהליך, הריגת כל עץ התהליך, הפעלת תהליך מחדש, עצירת תהליך. בנוסף יש אפשרות לייצא קובץ DUMP עבור אותו תהליך מסומן או עבור כל התהליכים הרצים.

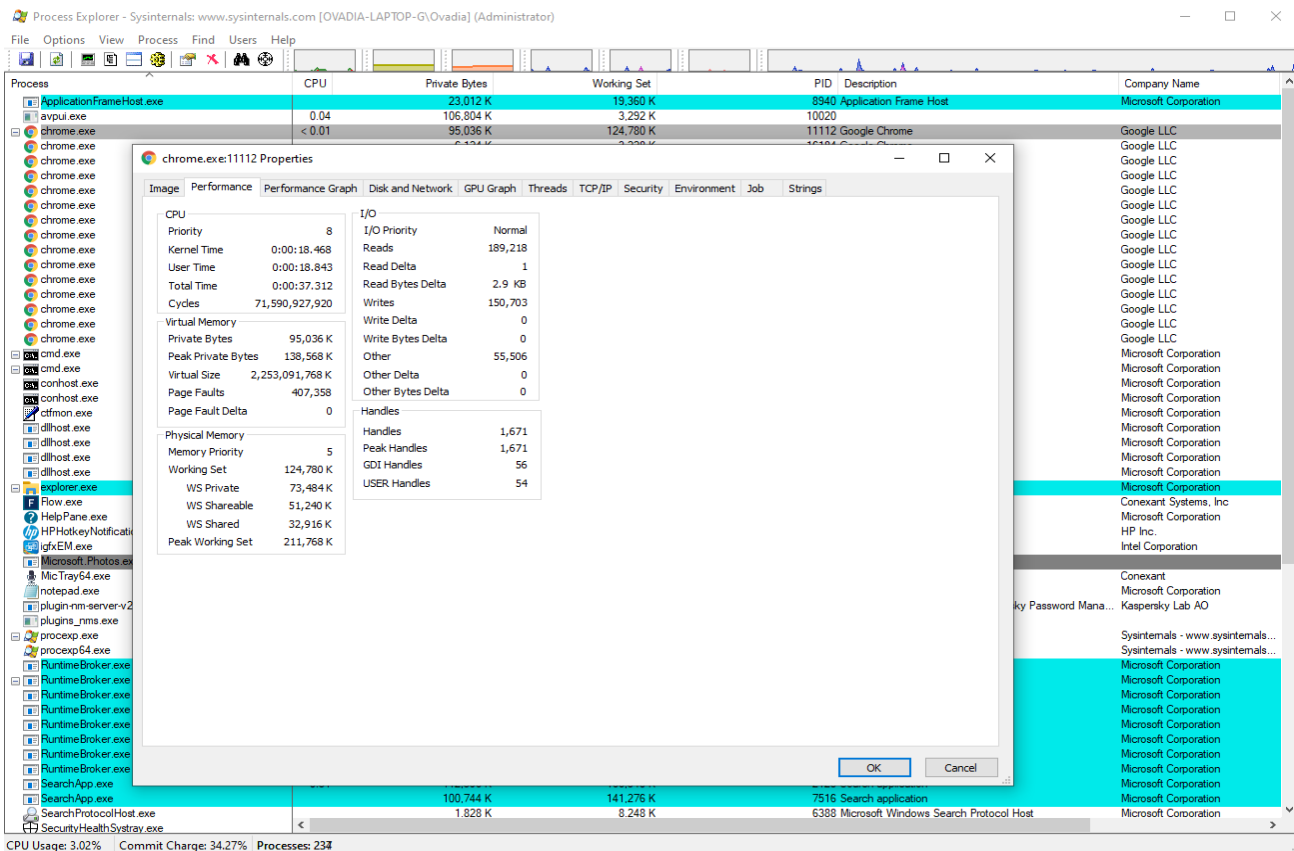
הכלי Process Explorer מציג את התהליכים הרצים במחשב בזמן אמת בתוך טבלה המחולקת למספר עמודות:

- Process – שם התהליך שרץ בזמן אמת.
- CPU – זמן מעבד שהתהליך צורך.
- Private Bytes - מונה בתים פרטיים המציין את כמות הזיכרון הכוללת שהתהליך הקצה.
- Working Set - מונה בתים פרטיים המציין את כמות הזיכרון הכוללת שהתהליך העבד עד כה.
- PID – מספר מזהה של כל תהליך.
- Description – תיאור התהליך בשם.
- Company Name – שם חברת הייצור של התהליך.
- Virus Total – בדך כלל עמודה ריקה, כאשר יבוצע סריקה לתהליך יופיע מידע בטבלה.

הכלי Process Explorer מסמן תהליכים לפי צבעים הלן:

- ירוק** – תהליך חדש שנפתח.
- אדום** – תהליך שנסגר ועומד להיעלם.
- סגול בהיר** – תהליכים שרצים באותו USER שבו מופעל Process Explorer.
- ורוד** – תהליכים המשרתים את מערכת ההפעלה.
- אפור** – תהליכים הנמצאים במצב השהייה.
- סגול** – בצע הרצה דחוסים, (קבצים שמהותם מתבררת רק בזמן ריצה), יכולים לאפיין ווירוסים כן כנסת עמודת Virus Total – שעליה נרחיב.

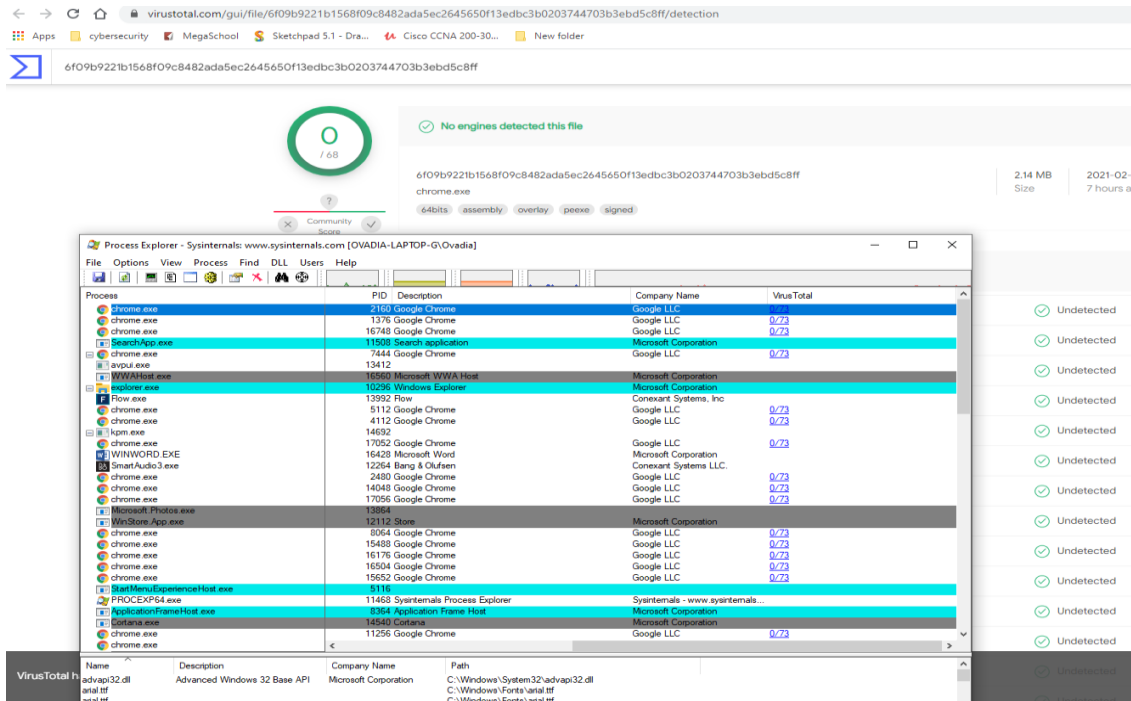
כדי לקבל מידע נוסף עבור תהליך ספציפי יש ללחוץ קליק ימני עבור אותו תהליך רצוי ללחוץ Properties יפתח לנו חלון שנותן מידע מורחב ומעמיק עבור התהליך הרצוי –



נבחר לדוגמא את התהליך Chrome ונעבור על הלשוניות בעלות מידע חיוני.

- תחת לשונית Image – נראה את שם התהליך ואת הנתבי שהקובץ יושב בו
- תחת לשונית Performance – נוכל לראות כיצד מנצל התהליך את משאבי המחשב, בכמה CPU משתמש התהליך, נוכל לראות כמה מידע I/O התהליך צורך (מהחומרה הרלוונטית בדוגמא שלנו Chrome צורך מידע I/O מרכיב כרטיס הרשת המחשב), בנוסף לפני מה שלמדנו בפרק על הזיכרון בלשונית Performance נוכל לראות כמה זיכרון ווירטואלי צורך התהליך ולכמה זיכרון פיזי ממור הזיכרון הווירטואלי שהתוכנה עושה עמו שימוש.
- תחת לשונית Treads – כפי שאני מכירים תהליכונים רצים תחת תהליך אב, נוכל לראות שתחת התהליך – **Chrome רצים 26 תהליכונים (Treads)** שרובם חולקים/מסנכרנים מידע בניהם.
- תחת לשונית Security – נוכל לראות לאיזה קבוצות התהליך שייך ותחת איזה דגלים הוא מסומן.

בכלי Process Explorer יש יכולת איתור תהליך שיכול להיחשב לזדוני באמצעות Virus Total אתר אליו הכלי מקושר, ברגע שיש חשד שתהליך זדוני אפשר לסרוק את התהליך ע"י לחיצה על התהליך הרצוי בקליק ימני על העכבר – Check Virus Total
 הכלי יסרוק את התהליך הרצוי ויצג את הנתונים בעמודת Virus Total -



- אחרי תוצאות הבדיקה אם נלחץ שני קליקים שמאליים על תוצאות הבדיקה, הכלי יעביר אותו לאתר Virus Total, באתר נוכל לראות מידע נוסף עבור סריקות אבטחה.

במידה והתחברתם לכלי Process Explorer יש אפשרות להחליפו במקום כלי Task Manager
 לוחצים על Options ואז על Replace Task Manager
 לאחר מכן בכל פעם שנלחץ על Alt+Ctrl+Del נגיע ישירות אל Process Explorer.

במידה ונרצה לבטל את הפעולה יש לצאת מהכלי להיכנס מחדש אבל הפעם מהכלי שלא מסתיים ב-64

procexp.exe	11/09/2020 15:06	Application	2,733 KB
procexp64.exe	11/09/2020 15:01	Application	1,456 KB

לוחצים שוב על Options ואז על Replace Task Manager ו- Task Manager יחזור שוב.

Autoruns

כלי שירות זה בעת הפעלתו ייתן מידע מקיף ביותר עבור כל התוכניות ושירותים אשר מוגדרים מראש לפעול אוטומטית במהלך אתחול מערכת ההפעלה או כניסה למשתמש.

כלי זה מסוגל לאתר תוכנית זדונית המותקנת על מערכת ההפעלה, כיוון שגם תוכנית זדונית המותקנת צריכה להיות מסוגלת לעלות מחדש על מערכת ההפעלה בעת אתחול. Autoruns בין אם יזהה את התוכנית הזדונית או בין אם המשתמש יבחין בה, בעזרת כלי זה מאפשר לראות את מיקום התוכנית היכן היא מותקנת, את הנתבי שלה.

בעת הפעלת הכלי נגיע לכרטיסיית Everything – בכרטיסייה זו נראה כל שירות ותוכנית אשר מוגדרים לעלות אוטומטית בעת אתחול מערכת ההפעלה או כניסה למשתמש, נראה את שם התוכנית, תיאור התוכנית, מיקום ונתבי התוכנית.

כלי זה גם הוא מקושר לאתר Virus Total המציג מידע האם תוכנה מסוימת נחשבת לזדונית. במידה ונסמן תוכנית מסוימת נוכל לראות פרטים עבור אודותיה בחלונית בסוף הצג כגון: תיאור, משקל, גרסה, נתבי, ומתי הותקנה התוכנית לראשונה.

Autoruns - Sysinternals: www.sysinternals.com

File Entry Options Help

Filter:

AppInit KnownDLLs Winlogon Winsock Providers Print Monitors LSA Providers Network Providers WMI Office

Everything Logon Explorer Internet Explorer Scheduled Tasks Services Drivers Codecs Boot Execute Image Hijacks

Autorun Entry	Description	Publisher	Image Path
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\AlternateShell			
<input checked="" type="checkbox"/> cmd.exe	Windows Command Processor	(Verified) Microsoft Windows	c:\windows\system32\cmd.exe
HKLM\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Run			
<input checked="" type="checkbox"/> Dropbox	Dropbox	(Verified) Dropbox, Inc	c:\program files (x86)\dropbox\clie
<input checked="" type="checkbox"/> KeePass 2 PreLoad	KeePass	(Verified) Open Source Developer, D...	c:\program files (x86)\keepass pa
HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run			
<input checked="" type="checkbox"/> GoogleDriveSync	Google Drive Sync	(Verified) Google LLC	c:\program files\google\drive\goo
<input checked="" type="checkbox"/> kpm.exe	Kaspersky Password Manager	(Verified) Kaspersky Lab JSC	c:\program files (x86)\kaspersky la
<input checked="" type="checkbox"/> OneDrive	Microsoft OneDrive	(Verified) Microsoft Corporation	c:\users\ovadia\appdata\local\m
C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup			
<input checked="" type="checkbox"/> AnyDesk.lnk	AnyDesk	(Verified) philandro Software GmbH	c:\program files (x86)\anydesk\an
<input checked="" type="checkbox"/> VPN.ht.lnk	VPN.ht	(Verified) VPN.ht Limited	c:\program files (x86)\vpn.ht\vpnh
HKLM\SOFTWARE\Microsoft\Active Setup\Installed Components			
<input checked="" type="checkbox"/> Google Chrome	Google Chrome Installer	(Verified) Google LLC	c:\program files (x86)\google\chro
<input checked="" type="checkbox"/> Microsoft Edge	Microsoft Edge Installer	(Verified) Microsoft Corporation	c:\program files (x86)\microsoft\ec
<input checked="" type="checkbox"/> n/a	Microsoft .NET IE SECURITY REGIS...	(Verified) Microsoft Corporation	c:\windows\system32\mscories.dl
HKLM\SOFTWARE\Wow6432Node\Microsoft\Active Setup\Installed Components			
<input checked="" type="checkbox"/> n/a	Microsoft .NET IE SECURITY REGIS...	(Verified) Microsoft Corporation	c:\windows\syswow64\mscories.c
HKLM\SOFTWARE\Classes\Protocols\Filter			
<input checked="" type="checkbox"/> text/xml	Microsoft Office XML MIME Filter	(Verified) Microsoft Corporation	c:\program files\common files\mic
HKLM\Software\Classes\ShellEx\ContextMenuHandlers			
<input checked="" type="checkbox"/> ANotepad++64	ShellHandler for Notepad++ (64 bit)	(Verified) Notepad++	c:\program files (x86)\notepad++\
<input checked="" type="checkbox"/> DropboxExt	Dropbox Shell Extension	(Verified) Dropbox, Inc	c:\program files (x86)\dropbox\clie
<input checked="" type="checkbox"/> GDContextMenu	Google Drive shell extension	(Verified) Google LLC	c:\program files\google\drive\con
<input checked="" type="checkbox"/> Kaspersky Anti-Virus 21.3	Shell Extension	(Verified) Kaspersky Lab JSC	c:\program files (x86)\kaspersky la
<input checked="" type="checkbox"/> MEGA (Context menu)			File not found: C:\Users\Dvir Roza
<input checked="" type="checkbox"/> TeraCopy		(Verified) Code Sector	c:\program files\teracopy\teracop
<input checked="" type="checkbox"/> WinRAR	WinRAR shell extension	(Verified) win.rar GmbH	c:\program files\winrar\warext.dll
HKLM\Software\Classes\Drive\ShellEx\ContextMenuHandlers			
<input checked="" type="checkbox"/> Kaspersky Anti-Virus 21.3	Shell Extension	(Verified) Kaspersky Lab JSC	c:\program files (x86)\kaspersky la
<input checked="" type="checkbox"/> MEGA (Context menu)			File not found: C:\Users\Dvir Roza
<input checked="" type="checkbox"/> TeraCopy		(Verified) Code Sector	c:\program files\teracopy\teracop
HKLM\Software\Classes\AllFileSystemObjects\ShellEx\ContextMenuHandlers			
<input checked="" type="checkbox"/> MEGA (Context menu)			File not found: C:\Users\Dvir Roza
HKLM\Software\Classes\Directory\ShellEx\ContextMenuHandlers			
<input checked="" type="checkbox"/> DropboxExt	Dropbox Shell Extension	(Verified) Dropbox, Inc	c:\program files (x86)\dropbox\clie
<input checked="" type="checkbox"/> GDContextMenu	Google Drive shell extension	(Verified) Google LLC	c:\program files\google\drive\con
<input checked="" type="checkbox"/> Kaspersky Anti-Virus 21.3	Shell Extension	(Verified) Kaspersky Lab JSC	c:\program files (x86)\kaspersky la
<input checked="" type="checkbox"/> MEGA (Context menu)			File not found: C:\Users\Dvir Roza

keepass.exe Size: 3,064 K
KeePass Time: 09/01/2021 13:43
Dominik Reichl Version: 2.47.0.0
"C:\Program Files (x86)\KeePass Password Safe 2\KeePass.exe" --preload

Ready. Signed Windows Entries Hidden.

נסקר כרטיסיות רלוונטיות למשתמש ואת התוכן שהן מציגות –

Everything – מציגה את כל התוכניות והשירותים אשר מוגדרת לעלות אוטומטית בעת אתחול מערכת ההפעלה או כניסה למשתמש.

Logon – מציגה את כל תוכנית אשר מוגדרת לעלות אוטומטית בעת כניסה למשתמש ספציפי, כאמור משתמשים יכולים לעבוד עם תוכנות או שירותים שונים, בכרטיסיה זו נוכל לראות איזה תוכנות יעלו אוטומטית אצל אותו משתמש ספציפי הנכנס לתחנה.

Explorer – כרטיסיה זו מתייחסת לסייר של Windows, מציגה מידע אותו תוספים מסוימים כגון:

- מודולים של קבצי DLL המשמשים כתוספים עבור דפדפני אינטרנט.
- סרגלי כלים של סייר, הכוונה תוספים שדפדפנים עושים איתם שימוש ועזרים של תוכנות העושות שימוש צד שלישי בדפדפנים.
- סיומות מעטפת - הכוונה לתוספים בודדים עבור סייר Windows - למשל יכולת להציג תצוגה מקדימה של קובץ PDF על הדפדפן.

Internet Explorer – כרטיסיה מציגה מידע אודות עזרים שהדפדפן עושה איתם שימוש, ותוכנות צד שלישי העושות שימוש עם הדפדפן – דוגמאות להמחשה: אם בתחנה מותקן אנטי-וירוס המיועד להגן מפני אתרים זדוניים אזי הוא התערב בדפדפן ונוכל לראות זאת בכרטיסיה הנ"ל.

Task Scheduler – כרטיסיה מציגה מידע עבור משימות מתוזמנות שהוגדרו להתחיל בעת אתחול מערכת ההפעלה או כניסות למשתמשים.

בעזרת משימות (Task) אפשר להגדיר אוטומציה לתהליך או לשירות שיפעל בעת אתחול או כניסה, בגלל שאוטומציה פועלת מאחורי הקלעים תוכנית או תהליך זדוני יכול לנצל זאת ולהשתמש באוטומציה כדי לשרוד אתחול של מערכת ההפעלה.

❖ הרחבת ידע – במערכת ההפעלה של Windows יש כלי מובנה – Task Scheduler שדרכו המשתמש יכול לבנות Task (הפעלה באוטומציה), מומלץ לא להשתמש בכלי זה ללא ידע מקדים כדי לא לפגום באוטומציה שמערכת ההפעלה נשענת עליה.

Services – כרטיסיה זו מציגה מידע אודות כל שירותי Windows שהוגדרו לפעול באופן אוטומטי.

Drivers – כרטיסיה זו מציגה מידע על מנהלי התקנים המאפשרים לחומרת המחשב לתקשר עם מערכת ההפעלה, בכרטיסיה מוצגים כל מנהלי ההתקנים הרשומים בתחנה, (למעט התקנים מושבתיים).

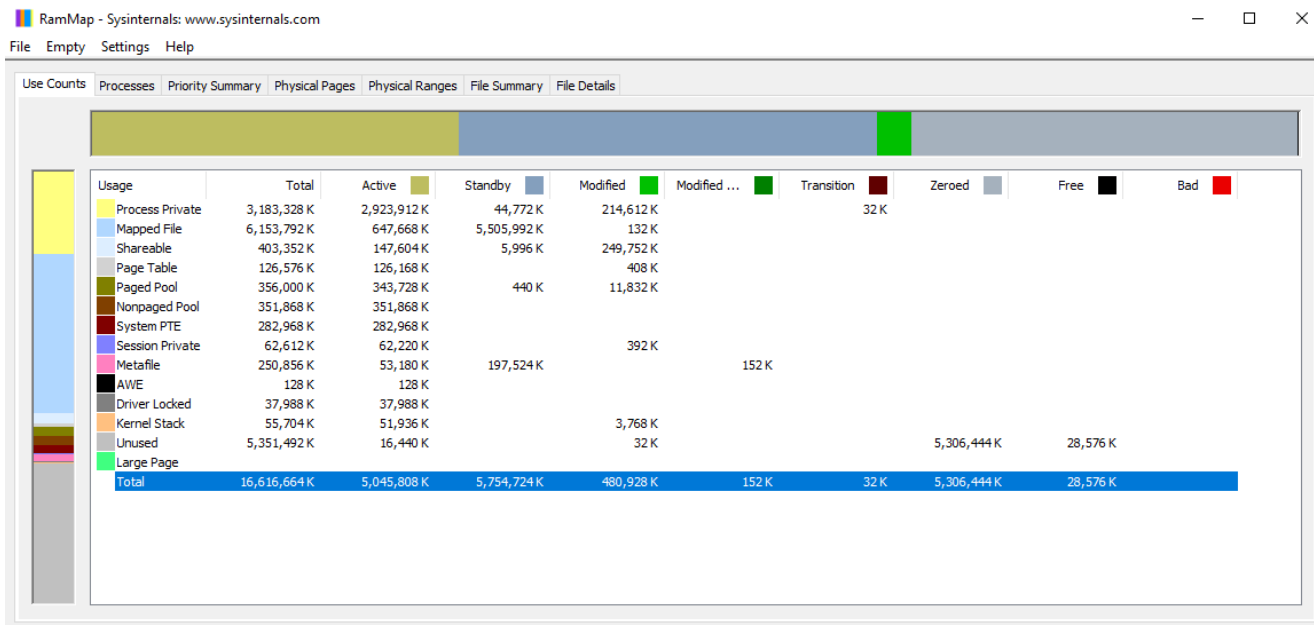
בנוסף נוכל לסמן כל תוכנה או שירות שנבחר ללחוץ קליל שמאלי ועל Properties יפתח לנו חלון הנותן מידע רחב על אותו האובייקט שבחרנו.

לסיכום: Autoruns מאפשרת לזהות אוטומציה של תוכנות ושירותים אשר הוגדרו מראש לעלות בעת אתחול מערכת ההפעלה או כניסה למשתמש, בין היתר נעשה שימוש ב – Autoruns כדי לזהות תוכנות זדוניות שהוגדרו לשרוד אתחול באמצעות שימוש באוטומציה, ואת מיקום התוכנה הזדונית.

Autoruns מקושרת עם אתר Virus Total ובכך נותן למשתמש מידע רחב עבור תוכנות, באמצעות שיתוף פעולה עם Virus Total ניתן לסמן כל תוכנה או שירות להגיע לעמודה של Virus Total להקליק פעמיים וזה יעביר אותנו לאתר של Virus Total וכבר יוצג לנו סריקות נוספות שעשו על התוכנית או שירות שבחרנו.

RAMMap

כלי זה מיועד לתת דיאגנוזה מלאה ומקיפה בזמן אמת אודות כל הזיכרון (RAM) במחשב.



כאשר נפעיל את הכלי נפתחת טבלה אשר מסדרת את מיפוי הזיכרון באופן מלא, נתחיל להתייחס לטבלה ולאובייקטים בתצוגה.

תחילה נסקור את החלק העליון בטבלה –

- **Active** – דפים פיזיים בזיכרון RAM שהם בשימוש פעיל (ייצוג של תהליכים פעילים).
- **Standby** – דפים פיזיים בזיכרון RAM שאינם בשימוש פעיל, ייצוג של דפים בהמתנה בדרך כלל משמשים כ"מטמון", כאשר מערכת ההפעלה יודעת שיש מידע שהיא עשויה להשתמש בו לעיתים דחופות אבל לא באופן רציף הוא יאוחסן במרחב הזה (Standby) וכאשר מערכת ההפעלה תעשה שימוש במידע היא תעביר את הדפים הרלוונטיים מ – Standby אל – Active.
- **Modified** – עמודה זו מייצגת דפים שהוחלפו ויש לטעון אותם לדיסק הקשיח לפני שימוש חוזר בהם.
- **Modified no write** – דפים שסומנו ע"י מערכת ההפעלה כלא כתובים לדיסק.
- **Transition** – מייצג דפים הנמצאים במעבר שבין אחת מהקטגוריות.
- **Zeroed** – מייצג דפים שאופסו ומוכנים לשימוש חוזר.
- **Free** – מייצג דפים שיש בהן מידע שמערכת ההפעלה מסמנת כ"מלוכלכים" ויש לאפסם.
- **Bad** – דפים פיזיים שמערכת ההפעלה מסמנת כלא טובים.

כעת נסקור את החלק השמאלי בטבלה –

Process Private – מייצג זיכרון המוקצה לתהליך יחיד בלבד.

Mapped File – מייצג זיכרון העובר מיפוי לזיכרון ווירטואלי.

Shared Memory – מייצג זיכרון המשותף בין מספר תהליכים/תהליכונים.

Page Table – מיצג את השימוש בטבלאות עמודים PTE.

Paged Pool – זיכרון מאוגד ליבה שניתן לדפדף לדיסק.

Non-paged Pool – זיכרון מאוגד ליבה שלא ניתן לדפדף לדיסק.

System PTE – ייצוג של מערכות רשומות בטבלת העמודים מאפשרות מיפוי של כתובות זיכרון וירטואלי לכתובות זיכרון פיזיות.

Session Private – זיכרון פרטי עבור מושב מחובר מסוים.

Metafile – מייצג זיכרון שהוא חלק מזיכרון מטמון במערכת ההפעלה.

AWE – מייצג הרחבת כתובות בזיכרון, מתן אפשרות ליישומים למפות תצוגות שונות של זיכרון פיזי אל מרחב הכתובת שעליה היא רצה.

Driver Locked – מייצג דפים שנעולים בזיכרון RAM ע"י מנהל התקן מסוים.

Kernel Stack – ייצוג של כמות השטח עבור דפים השייכים לתהליכוני ליבת המערכת.

כאשר נעבור ללשונית Processes נוכל לראות מידע אודות תהליכים הרצים בזמן אמת על מערכת ההפעלה, נוכל לראות את השימוש בדפים שתהליכים אלו עושים, ואת PID (מספר מזהה) של כל תהליך.

RamMap - Sysinternals: www.sysinternals.com

File Empty Settings Help

Use CountsProcessesPriority SummaryPhysical PagesPhysical RangesFile SummaryFile Details

Process	Session	PID	Private	Standby	Modified	Page Table	Total
System	-1	4	0 K	0 K	0 K	92 K	92 K
Registry	-1	124	8,900 K	0 K	0 K	276 K	9,176 K
smss.exe	-1	572	216 K	0 K	0 K	148 K	364 K
smartscreen.exe	5	5000	4,720 K	0 K	0 K	560 K	5,280 K
backgroundT...	5	14324	0 K	0 K	0 K	36 K	36 K
vpnht-service.	0	11880	0 K	0 K	0 K	28 K	28 K
csrss.exe	0	728	1,168 K	0 K	0 K	240 K	1,408 K
plugin-nm-serv	5	16592	2,200 K	0 K	0 K	340 K	2,540 K
svchost.exe	0	11552	0 K	0 K	0 K	32 K	32 K
avpui.exe	5	10244	2,024 K	0 K	82,316 K	1,100 K	85,440 K
chrome.exe	5	6464	10,576 K	0 K	0 K	984 K	11,560 K
dllhost.exe	5	6488	0 K	0 K	0 K	32 K	32 K
smss.exe	5	15796	0 K	0 K	0 K	32 K	32 K
dwm.exe	5	11152	34,600 K	0 K	8 K	816 K	35,424 K
kpm.exe	4	6152	0 K	0 K	0 K	32 K	32 K
HPSUPD-Win...	3	14528	0 K	0 K	0 K	36 K	36 K
dllhost.exe	5	13968	0 K	0 K	0 K	32 K	32 K
svchost.exe	0	10496	812 K	0 K	0 K	240 K	1,052 K
w3wp.exe	0	10304	14,072 K	0 K	0 K	952 K	15,024 K
SecurityHealth	5	5572	840 K	0 K	0 K	304 K	1,144 K
HPSUPD-Win...	2	2960	0 K	0 K	0 K	36 K	36 K
StartMenuEx...	5	14072	21,988 K	4 K	0 K	1,052 K	23,044 K
vpnht-service.	0	12588	4,812 K	0 K	0 K	348 K	5,160 K
igfxEM.exe	5	7144	4,104 K	0 K	0 K	596 K	4,700 K

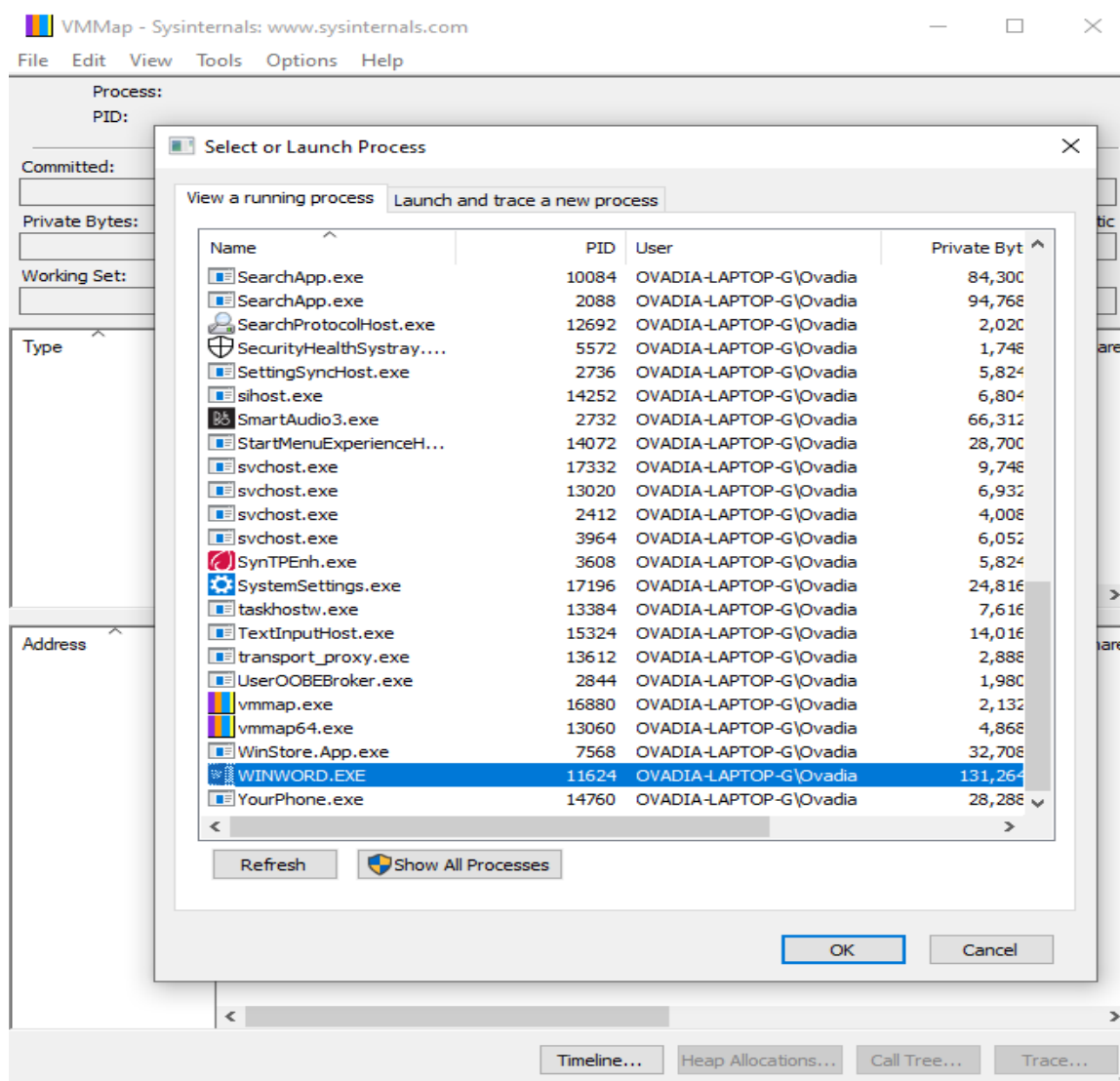
VMMMap

כלי זה נועד בעיקר לשימוש של מפתחי תוכנה אך גם המשתמש הרגיל יכול לעשות שימוש עם כלי זה, VMMMap הוא כלי ניתוח זיכרון הפיזי והן ווירטואלי, כלי אידיאלי כדי להבין ולייעל את השימוש במשאבי הזיכרון של יישומים הרצים.

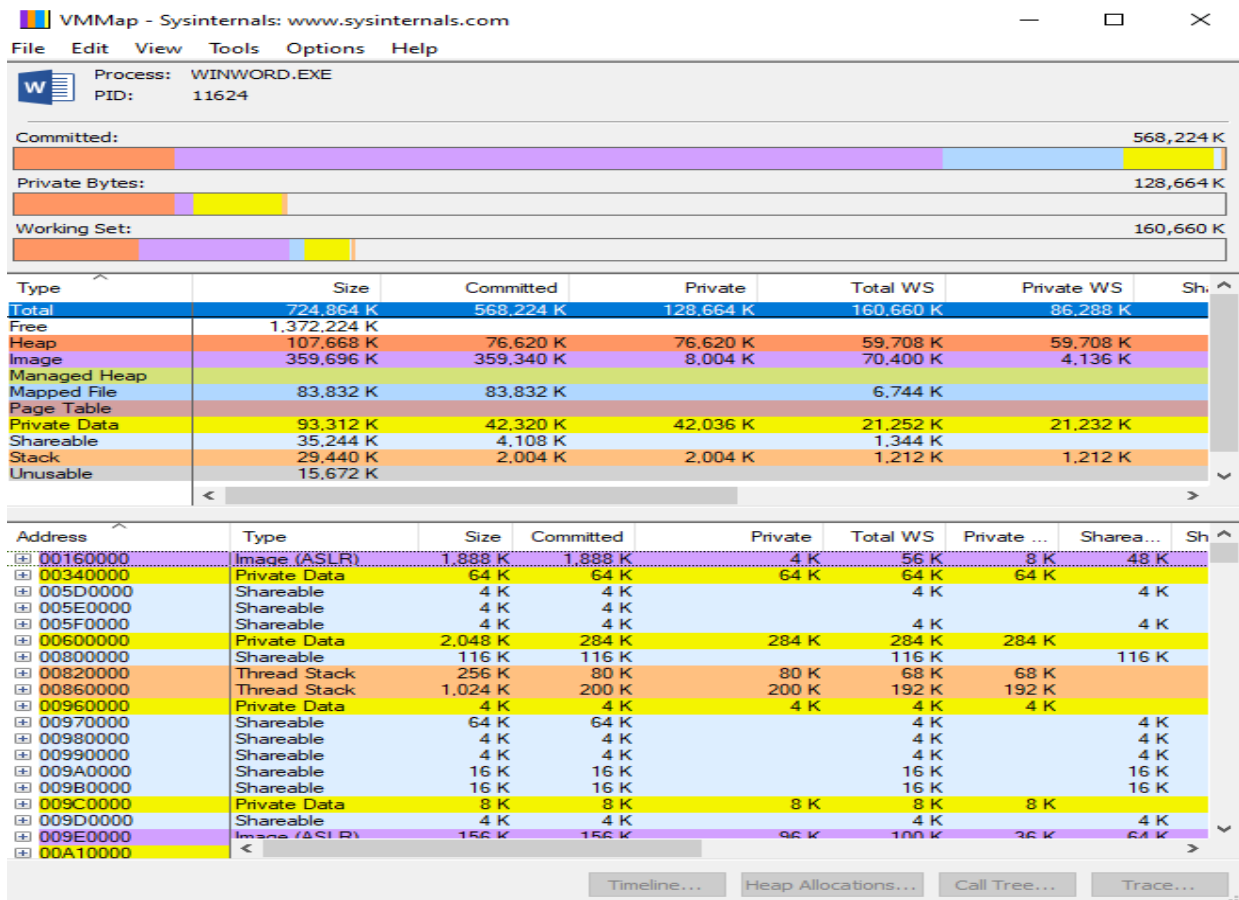
בשונה מהתוכנה RAMMap שהוצגה קודם אשר מנתחת את כל הזיכרון RAM במחשב, VMMMap מתייחס לניתוח זיכרון של תהליך ספציפי שכבר רץ על מערכת ההפעלה בזמן אמת.

VMMMap – מציג פירוט של סוגי הזיכרון הווירטואליים המחוברים של התהליך וכן את כמות הזיכרון הפיזי (סט העבודה) שהקצתה מערכת ההפעלה לסוגים אלה.

כאשר התוכנה VMMMap עולה, נפתח מולנו מסך עם אפשרות לבחור איזו תוכנה שנרצה מתוך התוכנות הרצות על מערכת ההפעלה בזמן אמת, ולהריץ דיאגנוזה אודות הזיכרון שלה, תמונה להמחשה:



לדוגמה נבחר את Word הנקרא WINWORD.EXE לאחר שבחרנו יפתח עבורנו חלון המציג ניתוח מעמיק אודות השימוש שתוכנה עושה עם משאבי הזיכרון במחשב, תמונה להמחשה:



נתחיל לעבור על האובייקטים בתמונה שמוצגת מולנו ונסקור את המידע ש – VMMap מציע לנו על התוכנה Word שבחרנו כאשר היא רצה בזמן אמת על מערכת ההפעלה:

Committed – מייצג את כל מרחב הזיכרון הווירטואלי עבור התוכנה, לזיכרון שהוא Committed קיים גיבוי פיזי לדיסק קשיח, הגיוני לתוכנה Word שעושה שמירה אוטומטית כל כמה זמן לדיסק קשיח.

Private Bytes – מייצג את מרחב הזיכרון שהתוכנה ביקשה ממערכת ההפעלה להקצות לה ממשאב הזיכרון, והיא נמוכה מזיכרון Committed כיוון שהיא לא מחשבת את מיפוי הזיכרון הווירטואלי לפיזי שמערכת ההפעלה אחראית עליו.

Working Set – מייצג את כמות הזיכרון שהתוכנה עושה בו שימוש בפועל.

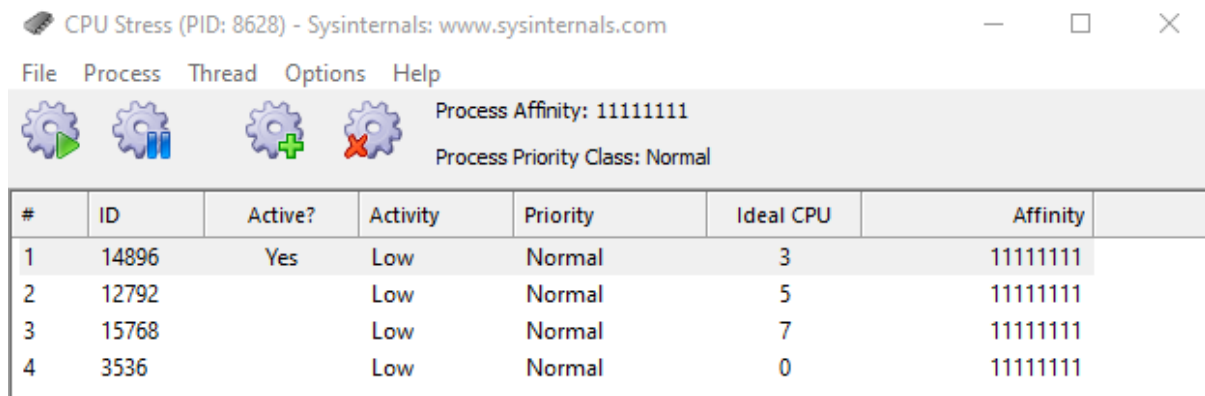
נעבור לסקור את האובייקטים בטבלה הקטנה:

- Heap** – מייצג את כמות הזיכרון המוקצה לתהליך באופן דינמי במהלך ריצה.
- Image** – מייצג את כמות הזיכרון המוקצה לקוד התוכנה.
- Mapped File** – מייצג את כמות הזיכרון המוקצה לדפים בזיכרון שעברו מיפוי.
- Page Table** – מייצג את כמות הזיכרון עבור PTE.
- Private Data** – מייצג את כמות הזיכרון המוקצה עבור משתנים של התוכנית שאינם במחסנית או ב Heap.
- Shareable** – מייצג את כמות הזיכרון המשותף בין תהליך או תהליכונים.
- Stack** – מייצג את מחסנית הזיכרון, מרחב הזיכרון המשמש לשמירת נתונים.
- Unusable** – מייצג את כמות הזיכרון המוקצה לתוכנה אך לא בשימוש.

CPUStress

כלי הנותן מידע אודות חומרת המעבדים שיש במחשב, כלי זה יכול להשהות את עבודת המעבד ולהמשיכה, לעצור את עבודת המעבד באופן מוחלט, ולהפעיל מעבד שנכנס להשהיה או הפסקת עבודה (במחשבים מודרניים מצב שלא קורה).

תמונה למחשה מחשב בעל 4 במעבדים :



#	ID	Active?	Activity	Priority	Ideal CPU	Affinity
1	14896	Yes	Low	Normal	3	11111111
2	12792		Low	Normal	5	11111111
3	15768		Low	Normal	7	11111111
4	3536		Low	Normal	0	11111111

נעבור על הפלט הרלוונטי שהטבלה מציגה :

ID – לכל מעבד מערכת ההפעלה מקצה מספר מזהה.

Active – מסמן איזה מעבד נמצע בפעולה בזמן אמת.

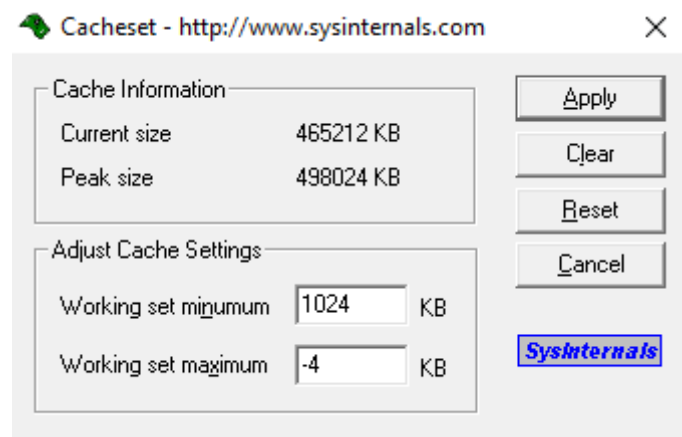
Activity – מציין באיזה דרגת קושי המעבד עובד (רמת העומס עליו) – נמוך, בינוי, גבוה.

Priority – מציין האם יש מעבד שנמצע בעדיפות מסוימת, מצב רגיל מעבדים הם על Normal.

❖ במצב נתון זה רואים שמעבד אחד עובד, אין עליו עומס, אין מעבד מסוים בעדיפות.

CPUStress

תוכנה זו מתוך חבילת Sysinternals מאפשר לנו לנהל את זיכרון Cache שיש לנו במחשב, נציג תמונה למחשה:



CPUStress מציג את הגודל הנוכחי של זיכרון מטמון המתעדכן פעמיים בשנייה, ומאפשר להגדיר מינימום ומקסימום סט קבוצות עבודה.

Apply – יעדכן את זיכרון Cache במידה ויש מידע חדש שאמור להיטען לזיכרון Cache.

ע"י לחיצה על Clear נוכל לאפס את הזיכרון Cache .

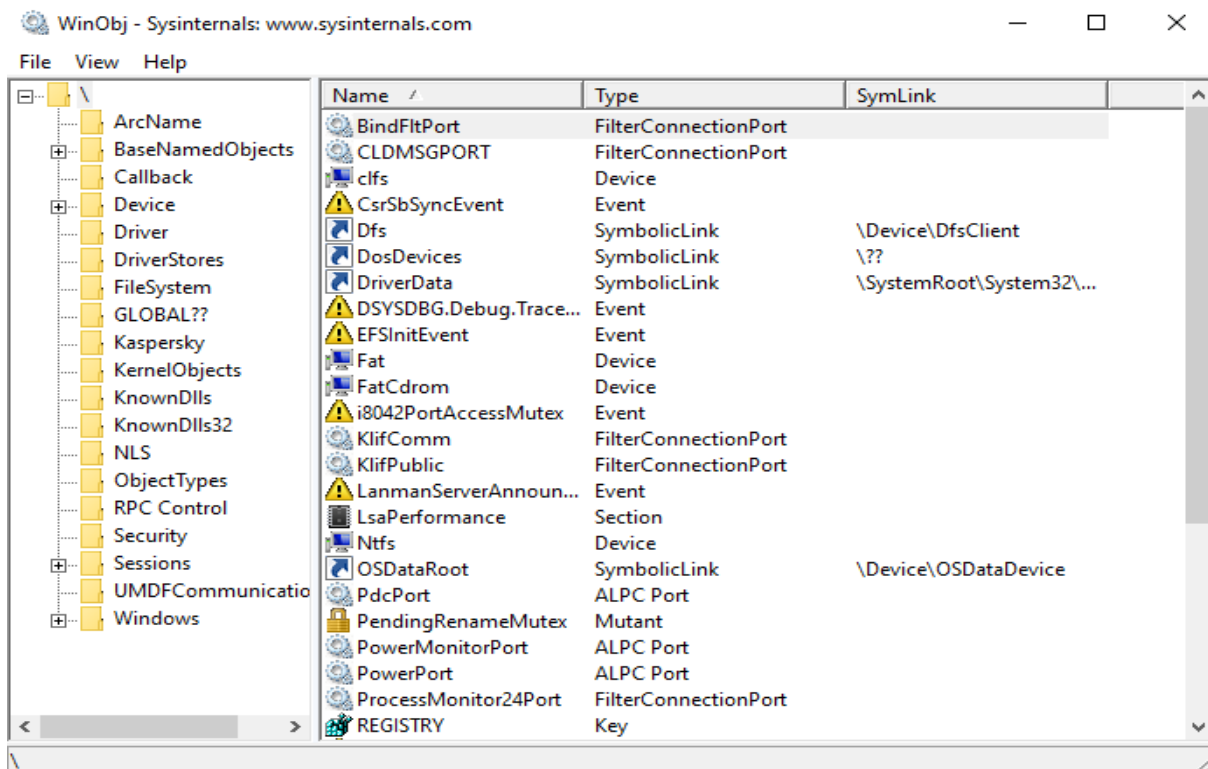
Reset – יאפס ויטען מחדש את הזיכרון, רק אם מערכת ההפעלה תאשר פעולה זו.

❖ זיכרון Cache יכול להשפיע על העבודה של מערכת ההפעלה אל מול המשתמש ולכן לא מומלץ "לשחק" עם כלי זה ללא ידיעה על ההשפעה של הפעולה שלו !

WinObj

למדנו שמערכת ההפעלה Windows מחזיקה תת מערכת הנקראת "מנהל אובייקטים" האחרית על מעקב רציף של משאבים במערכת, משאבים יכולים להיות: קבצים, ערכי רישום, התקני חומרה, תהליכים/תהליכונים פועלים עוד..

מידע זה אינו נחשף אל המשתמש אלא עובר אל מחורי הקלעים, אך מפתחי תוכנה ומנהלי צוות יעשו שימוש במידע הקיים בכלי זה, בדרך כלל כדי לגלות מידע על משאבים שהתוכנה WinObj מספקת, חלק מהסיבות להשתמש במידע זה הן סיבות אבטחה של מערכות.



תצוגת הכלי מתחלקת ל – 2 חלונות:

השמאלי – מציג עץ היררכי של מרחבי השמות במנהל האובייקטים.

הימני – פירוט של תוכן שמות האובייקטים, אם נלחץ פעמיים על אובייקט מסוים יפתח לנו חלון עם מידע נוסף עבור אותו אובייקט.

TCPView

כאשר כלי זה מופעל מתוך חבילת Sysinternals יפתח חלון שיציג רשימה מפורטת של כל נקודות הקצה של TCP ו- UDP בתחנה, כולל כתובות מקומיות ומרוחקות, ומצבי חיבור TCP.

TCPView מציגה את שם הבעלים של תהליכים (כלומר מי הבעלים של תהליכים פתוחים שעושים שימוש בפרוטוקולים TCP – UDP).

TCPView - Sysinternals: www.sysinternals.com

File Options Process View Help

Process	PID	Protocol	Local Address	Local Port	Remote Address	Remote Port	State
[System Proc...	0	TCP	ovadia-laptop-g5.h...	3705	ec2-52-31-179-16...	https	TIME_WAIT
[System Proc...	0	TCP	ovadia-laptop-g5.h...	3728	206.19.49.191	https	TIME_WAIT
[System Proc...	0	TCP	ovadia-laptop-g5.h...	3731	104.21.2.111	https	TIME_WAIT
[System Proc...	0	TCP	ovadia-laptop-g5.h...	3742	35.208.175.104	https	TIME_WAIT
[System Proc...	0	TCP	ovadia-laptop-g5.h...	3754	172.217.171.225	https	TIME_WAIT
[System Proc...	0	TCP	ovadia-laptop-g5.h...	3755	172.217.171.194	https	TIME_WAIT
[System Proc...	0	TCP	ovadia-laptop-g5.h...	3775	69.16.175.42	https	TIME_WAIT
[System Proc...	0	TCP	ovadia-laptop-g5.h...	3796	52.208.57.208	https	TIME_WAIT
[System Proc...	0	TCP	ovadia-laptop-g5.h...	3717	163.171.129.149	https	TIME_WAIT
[System Proc...	0	TCP	ovadia-laptop-g5.h...	3812	54.179.69.1	https	TIME_WAIT
[System Proc...	0	TCP	ovadia-laptop-g5.h...	3816	52.196.92.15	https	TIME_WAIT
[System Proc...	0	TCP	ovadia-laptop-g5.h...	3752	157.240.195.35	https	TIME_WAIT
[System Proc...	0	TCP	ovadia-laptop-g5.h...	3826	54.179.245.209	https	TIME_WAIT
AnyDesk.exe	4760	TCP	ovadia-laptop-g5.h...	nfa	relay-bf15d983.net...	http	ESTABLISHED
AnyDesk.exe	4760	TCP	Ovadia-Laptop-G5	7070	Ovadia-Laptop-G5	0	LISTENING
AnyDesk.exe	4760	UDP	Ovadia-Laptop-G5	50001	*	*	
avp.exe	4852	TCP	Ovadia-Laptop-G5	2787	localhost	2788	ESTABLISHED
avp.exe	4852	TCP	Ovadia-Laptop-G5	2788	localhost	2787	ESTABLISHED
avp.exe	4852	TCP	ovadia-laptop-g5.h...	2789	62.67.238.208	https	ESTABLISHED
avp.exe	4852	TCP	ovadia-laptop-g5.h...	3666	dc1.ksn.kaspersk...	https	ESTABLISHED
avp.exe	4852	TCP	Ovadia-Laptop-G5	5373	Ovadia-Laptop-G5	0	LISTENING
avp.exe	4852	TCP	Ovadia-Laptop-G5	5373	localhost	2501	ESTABLISHED
avp.exe	4852	TCP	Ovadia-Laptop-G5	5373	localhost	2511	ESTABLISHED
avp.exe	4852	TCP	Ovadia-Laptop-G5	5373	localhost	3221	ESTABLISHED
avp.exe	4852	TCP	Ovadia-Laptop-G5	5373	localhost	3675	ESTABLISHED
avp.exe	4852	TCP	Ovadia-Laptop-G5	5373	localhost	3682	ESTABLISHED
avp.exe	4852	TCP	Ovadia-Laptop-G5	5375	localhost	5376	ESTABLISHED
avp.exe	4852	TCP	Ovadia-Laptop-G5	5376	localhost	5375	ESTABLISHED
avp.exe	4852	TCP	Ovadia-Laptop-G5	50883	localhost	50884	ESTABLISHED
avp.exe	4852	TCP	Ovadia-Laptop-G5	50884	localhost	50883	ESTABLISHED
avp.exe	4852	TCP	Ovadia-Laptop-G5	5373	localhost	3228	ESTABLISHED
avp.exe	4852	TCP	ovadia-laptop-g5.h...	3744	72.246.151.16	http	ESTABLISHED
avp.exe	4852	TCP	Ovadia-Laptop-G5	5373	localhost	2864	ESTABLISHED
avp.exe	4852	TCP	Ovadia-Laptop-G5	5373	localhost	2960	ESTABLISHED
avp.exe	4852	TCP	ovadia-laptop-g5.h...	3791	93.184.220.29	http	ESTABLISHED
avp.exe	4852	TCP	Ovadia-Laptop-G5	5373	localhost	3702	ESTABLISHED
avp.exe	4852	TCP	Ovadia-Laptop-G5	5373	localhost	3045	ESTABLISHED
avp.exe	4852	TCP	ovadia-laptop-g5.h...	3795	93.184.220.29	http	ESTABLISHED

Endpoints: 274 Established: 108 Listening: 62 Time Wait: 14 Close Wait: 14

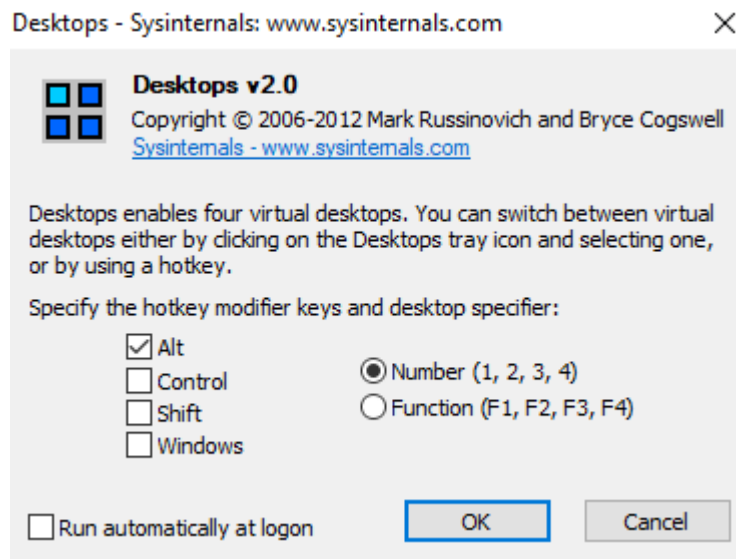
כפי שרואים בתמונה TCPView מציג פעילות שתהליכים עושים עם TCP - UDP.

- Process – מציג את כל התהליכים שמשתמשים בפרוטוקולים TCP - UDP.
- PID – מציג מספר מזהה של כל תהליך ברשימה.
- Protocol – באיזה פרוטוקול התהליך שברשימה משתמש או TCP או UDP.
- Local Address – מחליף את הכתובת המקומית בשם של התחנה.
- Local port – מציג באיזה פורט מקומי התהליך משתמש.
- Remote Address – מציג לאיזו כתובת מרוחקת התהליך פונה.
- Remote port – מציג לאיזה פורט מרוחק התהליך פונה.
- State – באיזה מצב התהליך נמצא.

Desktops

Desktops נחשב לכלי בקטגוריית "תוספות" מתוך חבילת Sysinternals, כלי מעניין הנותן למשתמש לפתוח עד 4 Desktops שונים ובכל Desktop המשתמש יכול לעשות ככול העולה על רוחו.

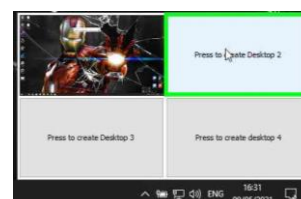
ברגע שנפעיל את הכלי יפתח חלון הגדרות כיצד נוכל לתמרן בין ה – Desktops ותצוגת המספור שלהם.



לאחר מכך נוכל לנוע בין ה – Desktops השונים, נראה את הסמן כחול המונה ריבועים כחולים בשורת התחל –



נלחץ עליו ויפתח לנו ארבעת ה – Desktops ומשם נוכל לעבור מאחד לשני.



קרדיטים ומקורות

❖ קרדיטים לספרים – כדי למקסם ידע על מערכות הפעלה ועל רכיבי המחשב מומלץ לכל מי שקרה את העבודה לעבור על ספרים שבהם נעזרתי:

- ברק גונן - ספר על מערכות הפעלה (המרכז לחינוך סייבר)
https://data.cyber.org.il/os/os_book.pdf
- Intel Paging & Page Table Exploitation on Windows מאמר מאת יובל עטיה
<https://www.digitalwhisper.co.il/files/Zines/0x61/DW97-3-PageTableExp.pdf>
- ארגון המחשב ושפת הסף
http://www.lamed-oti.com/school/gvahim_assembly_book.pdf
- סיכום קורס מערכות הפעלה
<http://danzig.jct.ac.il/os/OS-hebrew-summary.pdf>
- חקר זיכרון מאת יניב מרקס ואפיק קסטיאל
<https://www.digitalwhisper.co.il/files/Zines/0x2D/DW45-1-Memory.pdf>
- מבוא להנדסת מחשבים
<https://school.kotar.cet.ac.il/KotarApp/Viewer.aspx?nBookID=94321270&nTocEntryID=94323941#89.2739.5.default>

(כולל לינקים לתמונות)

<https://he.wikipedia.org/wiki/Sysinternals>

<https://docs.microsoft.com/en-us/sysinternals/downloads/system-information>

<https://www.pcworld.com/article/3181348/how-to-use-process-explorer-microsofts-free-supercharged-task-manager-alternative.html>

<https://www.guru99.com/operating-system-tutorial.html>

https://he.wikipedia.org/wiki/%D7%A2%D7%99%D7%91%D7%95%D7%93_%D7%91%D7%90%D7%A6%D7%95%D7%95%D7%94

https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/images/Chapter2/2_01_OS_Services.jpg

<https://www.tutorialspoint.com/file-system-manipulation>

https://www.tutorialspoint.com/operating_system/os_services.htm

<https://translate.google.co.il/?hi=iw&sl=en&ti=iw&text=Protection%20and%20Security%20of%20Preventing%20harm%20to%20the%20system%20and%20to%20resources%2C%20either%20through%20internal%20ways%20or%20malicious%20outsiders.%20Authentication%2C%20ownership%2C%20and%20restricted%20access%20are%20obvious%20parts%20of%20this%20system.%20Highly%20secure%20systems%20may%20log%20all%20process%20activity%20down%20to%20an%20excruciating%20detail%2C%20and%20security%20regulation%20dictate%20the%20storage%20of%20those%20records%20on%20permanent%20non-erasable%20medium%20for%20an%20extended%20time%20in%20a%20secure%20manner%20off>

sites%20to20facilities.%20%A0%A0Protection%20and%20Security%20requires%20that%20computer%20resources%20such%20as%20CPU%2C%20software%2C%20memory%20etc.%20%A0are%20protected.%20This%20extends%20to%20the%20operating%20system%20as%20well%20as%20the%20data%20in%20the%20system.%20This%20can%20be%20done%20by%20enforcing%20integrity%2C%20confidentiality%20and%20availability%20in%20the%20operating%20system.%20The%20system%20must%20be%20protected%20against%20unauthorized%20access%2C%20viruses%2C%20worms%20etc.%A0AThreats%20to%20Protection%20and%20Security%20AA%20threat%20is%20a%20program%20that%20is%20malicious%20in%20nature%20and%20is%20able%20to%20do%20harmful%20effects%20for%20the%20system.%20Some%20of%20the%20common%20threats%20that%20occur%20in%20a%20system%20to%20users%20are%20E%28B892%A0AVirus%A0Viruses%20are%20generally%20small%20snippets%20of%20code%20embedded%20in%20a%20system.%20They%20are%20very%20dangerous%20and%20can%20corrupt%20files%2C%20destroy%20data%2C%20crash%20systems%20etc.%20They%20can%20also%20spread%20further%20by%20replicating%20themselves%20as%20they%20are%20required.%A0AA%20Trojan%20Horse%20AA%20trojan%20horse%20can%20set itself up%20to%20access%20the%20login%20details%20of%20a%20system.%20Then%20a%20malicious%20user%20can%20use%20these%20to%20enter%20the%20system%20as%20a%20harmless%20being%20and%20wreak%20havoc.%A0AA%20Trap%20Door%20AA%20trap%20door%20is%20a%20security%20breach%20that%20may%20be%20present%20in%20a%20system%20without%20the%20knowledge%20of%20the%20users.%20It%20can%20be%20exploited%20by%20a%20hacker%20to%20data%20or%20files%20in%20a%20system%20by%20a%20malicious%20person.%A0AA%20Vorm%20AA%20worm%20can%20destroy%20a%20system%20by%20using%20its%20resources%20to%20extreme%20levels.%20It%20can%20generate%20multiple%20copies%20which%20can%20allow%20it%20to%20destroy%20resources%20and%20don%27t%20allow%20any%20other%20processes%20to%20access%20them.%20AA%20worm%20can%20 shut down%20a%20whole%20network%20in%20this%20way.%A0AA%20Denial%20of%20Service%20AA%20these%20types%20of%20attacks%20don%20not show%20the%20legitimate%20users%20to%20access%20a%20system.%20It%20overwhelms%20the%20system%20with%20requests%20so%20that%20its%20overwhelmed%20and%20cannot%20work%20properly%20for%20other%20user.%A0AA%20Protection%20and%20Security%20Methods%20AA%20the%20different methods%20that%20may%20provide%20protect%20and%20Security%20for%20different%20computer%20systems%20are%20E%28B892%A0AAAAuthentication%20AA%20This%20deals%20with%20identifying%20each%20user%20in%20the%20system%20and%20making%20sure%20they%20are%20who they%20claim%20to%20be.%20The%20operating%20system%20makes%20sure%20that%20all%20the%20users%20are%20authenticated%20before allowing%20them%20to%20access%20the%20system.%20The%20different%20ways%20to%20make%20sure%20that%20the%20users%20are%20authentic%20are%3A%A0AA%20Username%2F%20Password%20AAEach%20user%20has%20a%20distinct%20username%20and%20password%20combination%20and%20they%20need%20to%20enter%20it%20correctly%20before%20they%20can%20access%20the%20system.%A0AAUser%20Key%2F%20User%20Card%20AAThese%20users%20need%20to%20push%20a%20card%20into%20the%20card%20slot%20for%20use%20they%20get%20an individual%20key%20on%20a%20keypad%20to%20access%20the%20system.%A0AAUser%20Card%20Attribute%20Identification%20AADifferent%20user%20attributes%20identifications%20that%20a%20 user%20used%20are%20fingerprints%2C%20eye%20retina%20etc.%20These%20are%20unique%20for%20each%20user%20and%20are%20compared%20with%20the%20existing%20samples%20in%20the%20database.%20The%20user%20can%20only%20access%20the%20system%20if%20there%20is%20a%20match.%A0AAOne%20Time%20Password%20AAThese%20passwords%20provide%20a%20one%20time%20security%20for%20authentication%20 purposes.%A0AA%20One%20time%20password%20can%20be%20generated%20exclusively%20for%20a%20login%20every%20time%20a%20user%20wants %20to%20enter%20the%20system.%20It%20cannot%20be%20used%20more%20than%20once.%20The%20various%20ways%20a%20one%20time%20 password%20can%20be%20implemented%20are%20E%28B892%A0AARandom%20Numbers%20AAThe%20system%20can%20ask%20for%20a number%20that%20corresponds%20to%20alphabets%20that%20are%20pre%20arranged.%20This%20combination%20can%20be%20changed%20each%20time%20a%20login%20is%20required.%A0AA%20Secret%20Key%20AA%20hardware%20device%20can%20create%20a%20secret%20key%20related%20to%20 the%20user%20id%20for%20a login.%20This%20high%20level%20chance%20each%20time.%A0AAArja%20AKrist%20CAstro%20AA%20arop=translate

https://www.hon.co.il/%D7%9E%D7%94-%D7%96%D7%94-%D7%90%D7%99%D7%9E%D7%95%D7%AA-%D7%93%D7%95-%D7%A9%D7%9C%D7%91%D7%99-%D7%9B%D7%99%D7%A6%D7%93-%D7%94%D7%95%D7%90-%D7%A2%D7%95%D7%91%D7%93-%D7%95%D7%9C%D7%9E%D7%94-%D7%96

https://upload.wikimedia.org/wikipedia/commons/thumb/2/2f/Priv_rings.svg/1200px-Priv_rings.svg.png

http://www.stolerman.net/studies/os/operating_systems_summary.pdf

https://en.wikipedia.org/wiki/Protection_ring

[illegible]

[https://he.wikipedia.org/wiki/%D7%9E%D7%A0%D7%A2%D7%95%D7%9C_\(%D7%AA%D7%95%D7%9B%D7%A0%D7%94\)](https://he.wikipedia.org/wiki/%D7%9E%D7%A0%D7%A2%D7%95%D7%9C_(%D7%AA%D7%95%D7%9B%D7%A0%D7%94))
<https://www.geeksforgeeks.org/mutex-vs-semaphore/>
<https://docs.microsoft.com/en-us/sysinternals/downloads/autoruns>
<https://www.varonis.com/blog/how-to-use-autoruns/>
<https://techcommunity.microsoft.com/t5/ask-the-performance-team/introduction-to-the-new-sysinternals-tool-rammap/ba-p/374717>
<https://docs.microsoft.com/en-us/sysinternals/downloads/rammap>
<https://docs.microsoft.com/en-us/sysinternals/downloads/vmmmap>
<https://docs.microsoft.com/en-us/sysinternals/downloads/cacheset>
<https://searchwindowsserver.techtarget.com/tip/Tracking-system-resources-with-free-WinObj-utility-from-Sysinternals>
<https://docs.microsoft.com/en-us/sysinternals/downloads/tcpview>