**תיעוד קוד**

**קיצור דרך להפעלה של QEMU - ~/start\_vm.sh**

**דרייבר** – תוכנה שנמצאת במע' ההפעלה ותפקידה לקשר בין תוכנה לחומרה.

**Kernel** – חלק שבו פועלת מע' ההפעלה עצמה והיא שולטת על חומרה.

**User mode**  -מרחב שבו רצות תוכנות רגילות.

**SYSTEM call** - System Call הוא מנגנון שמאפשר לתוכנת משתמש (User Mode) לבקש שירותים מהקרנל – כמו קריאה/כתיבה לקובץ, גישה לזיכרון, או שליחת מידע לרשת

. **LKM** – לינוקס קרנל מודול, כל קרנל בנוי ממודולים (קוד) שמבצע תהליכים במע' ההפעלה ובמאחורי הקלעים. כשבונים LKM ניתן להוסיף למע' ההפעלה לינוקס מודולים שאנחנו רוצים בקוד שאנחנו רוצים. וניתן לפרוק ולטעון אותו במידת הצורך/רצון.

**LBR LAST BRANCH RECORD** – טכנולוגיה של אינטל שמאפשר שמירת היסטוריה של הסתעפויות בקוד, למשל שורות כמו If return while וכו'. כל הסתעפות נקראת branch. LBR שומר רשימה של ההסתעפויות האחרונות שקרו.

**MSR MODEL SPECIFIC REGISTERS –** מאפשר גישה לרג'יסטרים מיוחדים שמראים מה קורה בcpu.

**CPUID**- מקבל מספר leaf ו4 מקומות לשמור מידע והוא שם שם את המידע על הcpu.

**EAX** - אחד מארבעת הרגיסטרים שפוקדת CPUID מחזירה אליהם מידע. כשקוראים עם leaf 1, EAX מחזיר מידע על הדגם, ה־family, ה־stepping והגרסה של המעבד.

**EBX** רגיסטר נוסף שפוקדת CPUID מחזירה אליו מידע. כולל למשל אינדקס למותג (Brand Index), גודל cache line, מספר ליבות לוגיות, ו־APIC ID.

**ECX** רגיסטר של תכונות מודרניות של המעבד. כל ביט בו מציין תמיכה באפשרות מסוימת (למשל ביט 27 מציין אם המעבד תומך ב־LBR).

**EDX** רגיסטר של תכונות ותיקות של המעבד. כל ביט מציין תכונה כמו FPU, MMX, SSE, TSC ועוד.

**MSR\_IA32\_DEBUGCTL** רג'יסטר ב־CPU שמכיל ביטים שמפעילים/מכבים תכונות דיבוג, כולל את הפעלת LBR. אם ביט 0 בו שווה ל־1, LBR מופעלת.

**MSR\_LASTBRANCH\_TOS** רג'יסטר שמכיל את האינדקס של ההסתעפות האחרונה שנשמרה ב־LBR (TOS = Top Of Stack). משמש כדי לדעת איפה נמצאת ההסתעפות הכי חדשה בטבלת ההיסטוריה.

**MSR\_LBR\_FROM\_BASE** כתובת הבסיס של רשימת רג'יסטרים ב־CPU שמכילים את כתובת המקור (FROM) של כל הסתעפות שנשמרה ב־LBR. כל רג'יסטר אחריו (680, 681, 682 וכו') מכיל את ה־FROM של הסתעפות אחרת.

**MAX\_LBR\_ENTRIES** – קבוע בתוכנה שמציין כמה רשומות LBR ננסה לקרוא (למשל 32). לא קשור ישירות למעבד אלא משמש אותנו כדי להגביל את הלולאה בקריאת ההיסטוריה.

**Rdmsrl-** קורא את הערך מהרגיסטר המסוים ומכניס אותו למשתנה

**Wrmsrl-** כותב את מה שיש במשתנה לתוך רגיסטר.

* כדי לעשות mount לתיקיה של הפרויקט –

Cd /mnt/host

sudo mount -t 9p -o trans=virtio hostshare /mnt/host

**טיפ לזכור:**

* **CPUID** ⇒ "מה מותר לי?" (האם בכלל יש LBR)
* **DEBUGCTL** ⇒ "האם LBR כרגע פעיל?"
* **LBR\_CTL** ⇒ "אם הוא פעיל – באיזה מצב בדיוק?"

**file\_operations**

**הגדרה:**  
זוהי טבלה שמכילה כתובות של פונקציות שהקרנל משתמש בהן כדי לדעת איך לטפל בקובץ מכשיר (כמו /dev/lbr\_device).

**תיאור פשוט:**  
כאשר תהליך בלינוקס (כמו תוכנה שאתה מריץ) מנסה לבצע פעולה על קובץ – כמו פתיחה (open), קריאה (read) או שליחה של פקודה (ioctl) – הקרנל מחפש את הפונקציה המתאימה בטבלה הזו ומריץ אותה.

**miscdevice**

**הגדרה:**  
מבנה שמייצג התקן פשוט במערכת (/dev/...) עם תמיכה קלה לרישום בדרייבר.

**תיאור פשוט:**  
במקום לכתוב הרבה קוד כדי לרשום "מכשיר" בלינוקס, משתמשים ב־miscdevice שעושה את זה אוטומטית. לדוגמה, הוא יוצר עבורך את הקובץ /dev/lbr\_device, קובע הרשאות, ומקשר אותו ל־file\_operations.

**דברים שקשורים לקוד**

1. כאשר מריצים InsMod אנחנו מפעילים את הlbr. ישר כשמסיימים את הInit הקרנל מבצע context switch ומכבה את הlbr עוד לפני שאנחנו נכנסים לExit ומכבים בעצמנו את הlbr.