digital whisper
Technologic Papers

על תכנות low-level, או - כמה נמוך אפשר לרדת?

מאת איתי רויז

הקדמה

אם יש לכם חברים מתכנתים, סביר שלקחתם חלק בשיחה על שפות תכנות מועדפות. יש אנשים שיעדיפו שפות גבוהות (כמו פייתון או ג'אווה) בשביל פשטות התכנון בהן, ויש שיעדיפו שפות נמוכות (סי או אסמבלי) בשביל השליטה הישירה שהן נותנות בזיכרון. כמובן שיש יתרונות וחסרונות נוספים לכל בחירה, אבל באופן כללי ככל שמשתמשים בשפות נמוכות יותר מקבלים שליטה יותר ישירה בזיכרון, במחיר של סיבוך התכנון.

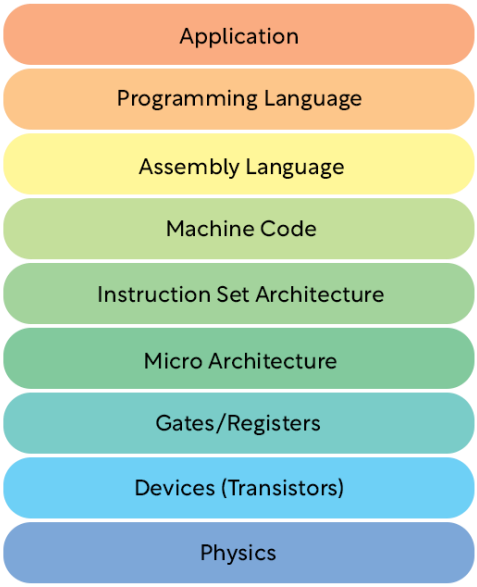
אני זוכר שיחה שהייתה לי עם חברים לפני כמה שנים, שהלכה בערך ככה:

- **חבר:** "תגיד איתי, באיזו שפה יוצא לך לעבוד הכי הרבה?"
- **אני:** "נראה לי שאני מתכנת הכי הרבה בפייתון"
- **חבר:** "פייתון? אז אתה בכלל לא מבין במחשבים... מתכנתים אמיתיים עובדים בסי!"
- **חבר שני נכנס לשיחה:** "סי? מה אתה ילד? ברור לכולם שמתכנתים אמיתיים עובדים באסמבלי"
- **חבר שלישי נכנס לשיחה:** "אסמבלי? תתכנתו בסקראץ' וזהו, מתכנתים אמיתיים כותבים אפסים ואחדות!"
- **חבר רביעי נכנס לשיחה:** "חובבנים... מתכנתים אמיתיים מחזיקים כבל ושולטים על המתח במעבד בעצמם..."

ובזמן שבאותו רגע כולנו צחקנו, נדלקה לי נורה קטנה בראש ורעיון התחיל להתבשל...

רעיון

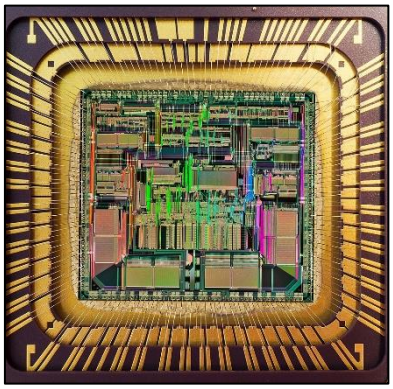
הבדיחה הזו הצחיקה אותנו באותו זמן בגלל דבר שנקרא רמות אבסטרקטציה. כדי לעשות אפילו את הפעולה הבסיסית ביותר במחשב, יש צורך להסתמך על המון רמות אבסטרקטציה שעובדות ומספקות ממשק תקין לרמה שמעל: כדי להדפיס "hello world" למסך שלכם, צריך לכתוב תכנית בשפת תכנות לבחירתכם, שבתמציתה מבקשת ממערכת ההפעלה להציג את המחרוזת שלכם על המסך. למערכת ההפעלה יש ממשק עם החומרה, ורק החומרה יכולה לפנות למסך שלכם עם ערכי הצבעים לפיקסלים הנכונים כך שיוצג על המסך צמד המילים "hello world".



אפילו בהצגה הרב-שכבתית הזו הסתרתי אין-ספור רמות אבסטרטקציה בתוך שפת התכנות האהובה עליכם, בתוך מערכת ההפעלה ובתוך החומרה...

אז מה הבעיה? הבעיה היא שבעוד רמות אבסטרקטציה מפשטות לנו את התכנון ככה שנוכל לתכנן את הרמה הבאה (אף אחד לא יכול לתכנן שרת http ע"י קביעת רמות המתח בכל חלק של המחשב בכל רגע נתון) הן גם לוקחות לנו מידה משמעותית של חופש, ומגבילות את השימוש שלנו בחומרה שברשותנו אך ורק למה שהיא תוכננה לעשות...

בתור מהנדס סקרן, תמיד עניין אותי מה אני יכול לגרום למחשב לעשות, ויותר ספציפית מה אני יכול לגרום למחשב לעשות שהוא לא אמור לעשות. בסוף כל מחשב הוא רק ערמת מתכת וסיליקון (וחומרים אחרים) שאנחנו מזרימים בה חשמל בדרכים שונות ובתמורה היא פולטת אור, קול וגלים אחרים לפי חוקים מוכתבים מראש (בין אם אלה חוקי הפיזיקה או חוקים שהכתיב מתכנן החומרה או התוכנה). אם אנחנו יודעים את החוקים האלה, מה מונע מאיתנו להזרים את החשמל איך שנבחר ובכך לגרום למכונה לעשות כל מה שנרצה?



מוטורולה 68040: בסה"כ ערימת מתכת וסיליקון. כ-1.2 מיליון טרנזיסטורים, הושק ב-1990

ערימת מתכת וסיליקון, מסודרת קצת אחרת... נסו להריץ על זה DOOM

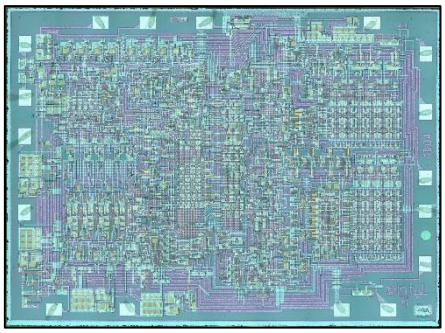
חזרה למציאות

אני שומע אתכם אומרים לי דרך המסך: "אבל איתי, אתה צוחק איתי? המעבד ורוב החומרה שמקיפה אותו סגורים בתוך אריזה סגורה וחתומה, אין שום סיכוי שנצליח להגיע לחלקים הרלוונטיים במעבד ולהעביר את המתח שאנחנו רוצים. חוץ מזה, מבנה המעבד הוא מהמסובכים שבהמצאות האנושות והתכנון שלו לא תמיד זמין לציבור, איך נדע איפה ואיך צריך לשנות את המתח כדי לגרום לאפקט שאנחנו רוצים? חוץ מזה, ה"חוטים" שבמעבד הם דקים יותר מפי אלף משערה אנושית, זה לא קנה מידה שאפילו מכונה מדויקת יכולה לעבוד בו, בקיצור אין לנו סיכוי"

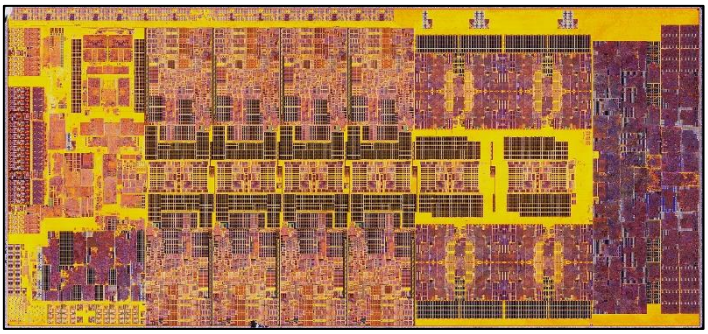
ואתם כמובן צודקים (חוץ מהחלק שאין לנו סיכוי).

המשימה של לשלוט בחישוב המתבצע על גבי המעבד ברמה החשמלית היא בלתי אפשרית (לפחות במחשבה ראשונה). גם אם היא אפשרית היא תכלול מחקר ארוך ומדויק על מעבד ספציפי, תכנון מדוקדק שטעות הכי קטנה בו יכולה להביא לקריסת המעבד וכשלון הניסוי (במקרה הטוב) ולשריפת המעגלים והפיכת החומרה לפסולת אלקטרונית (במקרה הרע).

אבל בתור מהנדסים (מתכנתים, האקרים, כל אחד איך שהוא מחשיב את עצמו) המטרה שלנו היא לקחת את הבלתי אפשרי ולהפוך אותו לאפשרי. יותר במדויק, לקחת את הבלתי אפשרי ולהפוך אותו לתיאורטית אפשרי, את התיאורטית אפשרי להפוך לקשה מאוד, את הקשה מאוד לפרק לבעיות קטנות ולפתור אותן אחת-אחת, עד שבסוף אנחנו נשארים עם משהו אפשרי.



אינטל 4004: כ-2300 טרנזיסטורים. המעבד הראשון שהודפס כולו על מעגל משולב יחיד, הושק ב-1971

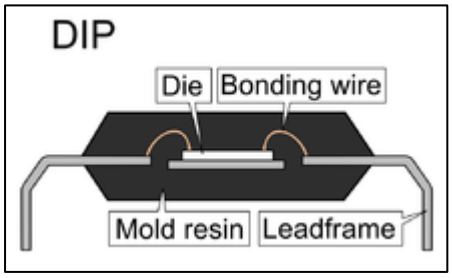


אינטל Raptor Lake (i9-13900K): דור 13, כ-12 **מיליארד** טרנזיסטורים, הושק באוקטובר 2022. עובי הטרנזיסטורים ברפטור-לייק הוא 5nm, בניגוד ל-10um באינטל

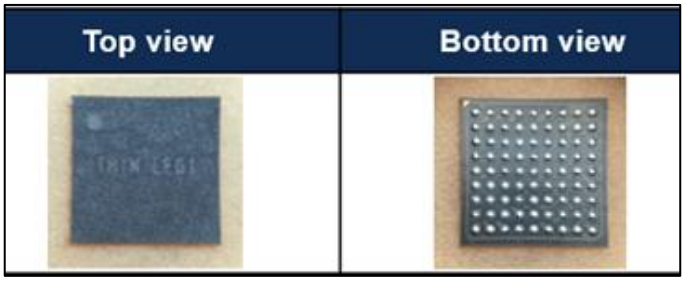
אז איך מתחילים?

בעיה ראשונה: אריזה

נתחיל בבעיה הראשונה: המעבד (כלומר השבב עצמו) סגור בתוך אריזה בלתי פתיחה (ראו ערך אריזת מעגלים משולבים). האריזה של רוב השבבים בימינו (בעיקר אלה שמצויים במכשירים אלקטרוניים לשימוש פרטי) עשויה מסוג כלשהו של פלסטיק, ופלסטיק זה חומר שאפשר לשייף/לשבור (אם כי צריך לעשות את זה בזהירות). לא איכנס לכל השיטות להסיר אריזה של שבב, אבל בדרך כלל נעשה שימוש בשילוב של חומרים שממיסים את הפלסטיק ושיוף פיזי עדין.



אריזת DIP: בד"כ לצ'יפים פשוטים/זולים יותר, כאלה בעלי שטח פנים קטן יחסית. בבחירת האריזה יש הרבה שיקולים, החל מהעלות עד קלות הבדיקה וגם שיקולים פיזיקליים-הנדסיים.



אריזת BGA (יותר במדויק FCBGA): ראשי תיבות של flip-chip ball grid array, באריזה הזו חיבורי המתכת נמצאים מתחת למעגל המשולב ושבבות הטרנזיסטורים נמצאות למעלה (מתחת למצע הסיליקון). כיום רוב המעבדים המודרניים ארוזים בשיטה הזו, מה שעוזר מאוד לתקיפה שלנו- בזכותה אנחנו יכולים לגשת ישירות לטרנזיסטורים בלי לעבור את כל שכבות המתכת שמחברות ביניהם.

בעיה שנייה: מבנה המעבד

בנוגע לבעיה השנייה, זה אכן נדבך מרכזי במחקר. גם אם אנחנו מקלים על עצמנו בלתקוף מעבד שתכנונו ידוע לציבור, עדיין יש צורך בהבנה עמוקה של מבנהו הפיזי של המעגל המשולב, דהיינו איזה "חוט" מחובר לאיפה. זוהי עבודת נמלים אינסופית (אך אפשרית), ולצרכים פרקטיים מתמקדים באזורים ספציפיים עם מבנה מחזורי וקל לזיהוי שאנחנו יודעים שיהיו מעניינים עבורנו (בדרך כלל זכרונות[1]). מצד אחד זה נכון שכשאנחנו מתעסקים במעבדים מודרניים אנחנו מגבילים את עצמנו מלהתעסק בתהליך החישוב עצמו[2], מצד שני צריך לזכור שבמעבדים לשימוש כללי (בניגוד לחומרה ייעודית, ראו את ההבדל כאן) גם המידע וגם ההוראות לעיבוד המידע (כלומר הקוד) שמורים בזכרון כזה או אחר (ראו ערך ארכיטקטורת פון-נוימן). לכן שליטה חשמלית גם אם היא רק בזכרונות עדיין תתן לנו יכולת ביצוע קוד גנרית על ידי המעבד.

---

[1] אפשר לראות את אזורי הזכרונות בתמונה של המוטורולה 68040 בעמוד השני - אלו האזורים הנראים כמו מלבנים חלקים, ובפועל מורכבים מאלפי טרנזיסטורים מסודרים במבנה מחזורי.

[2] חשוב לציין שההגבלה הזאת מגיעה רק בגלל הרצון שלנו למנוע סקירה מקיפה (וארוכה מאוד) של כל החומרה, ושאם אנחנו מזהים מקום מסוים בחישוב שמעניין אותנו במיוחד לחלוטין אפשר לתקוף גם אותו. יותר מזה, בחומרות ייעודיות שמבנהן יותר פשוט אפשרי ואפילו רצוי לסקור באופן מקיף את החומרה, בעיקר מכיוון שלעתים ניתן לתקוף נקודה אחת באמצע החישוב במקום נקודות רבות בזיכרון.

בעיה שלישית: קנה מידה

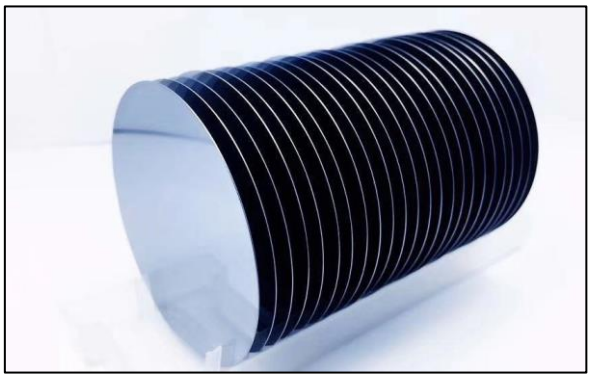
הבעיה השלישית והמרכזית ביותר היא הבעיה של קנה מידה. אילו היינו מגדילים מעבד מודרני לגודל של בניין ממוצע (קנה מידה של עשרות מטרים) גודל המבנים שהיינו רוצים להשפיע עליהם (הטרנזיסטורים) היו בערך בעובי של קור עכביש[3], זה אומר שלכל צורך פרקטי אין לנו שום סיכוי להיכנס לתוך המעבד ולשנות את המתחים כרצוננו. אין שום מכונה כיום או בעתיד הנראה לעין שיודעת לעבוד ברזולוציה כזאת.

אז נשאלת השאלה: אם אין מכונה שאפילו מתקרבת לעבודה בקנה מידה שאנחנו רוצים לעבוד בו, איך מייצרים את המעבדים האלו מלכתחילה? ופה אנחנו נכנסים לעולם המרתק של הנדסת חשמל. אם יש לכם ידע בתהליכי ייצור של מעגלים משולבים - תרגישו חופשי לדלג על הקטע הזה.

איך מייצרים מעגלים משולבים?

כדי להבין איך האנושות הצליחה לייצר מבנים כאלה מורכבים בקנה מידה שבקושי ניתן לתפוס, צריך לדבר קודם על מוליכים למחצה. מוליכים למחצה אלו חומרים שמשנים את תכונות ההולכה החשמלית שלהם כתלות בתנאים שבהם הם שרויים (בין היתר, כתגובה לחשיפה לקרינה). הסיבות ללמה הם מתנהגים ככה נעוצות בחוקים הפיזיקליים שמכתיבים את התנהגותם של חומרים ברמה החלקיקית, ולא ניכנס אליהם במסגרת מאמר זה (קורס נהדר של הטכניון למי שרוצה ללמוד את הנושא לעומק).

המוליך למחצה המרכזי שמשתמשים בו כיום בתעשייה הוא סיליקון, אם כי לצרכים מיוחדים ניתן להשתמש בחומרים אחרים.



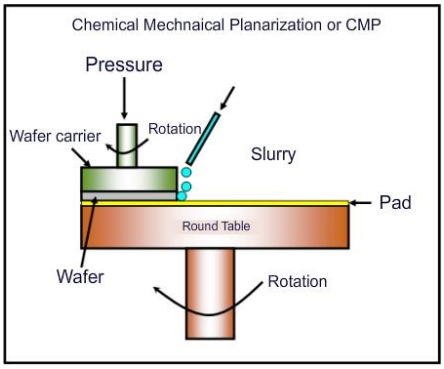
מימין Silicon ingots ו**משמאל** silicon wafers (לא בטוח בנוגע לתרגום לעברית). ה-ingots צריכים להיות 99.999999999% סיליקון טהור (11 תשיעיות). אחר כך חותכים אותם בעזרת מסור מדויק ומשייפים אותם בתהליך שדואג שהפרש הגבהים בין הנקודה הכי נמוכה להכי גבוהה על ה-wafer יהיה פחות מננומטר. הדיוק של התהליך הזה קריטי לכמות המעגלים המשולבים שיודפסו בהצלחה על פרוסת הסיליקון ולכן מושקעים בו

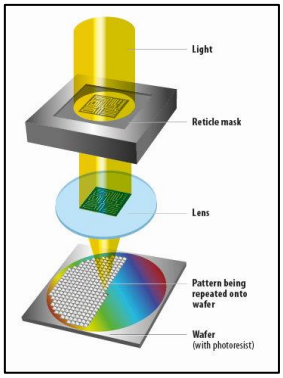
[3] יחס קנה מידה של בערך 1 ל-10 מיליון. כבר היום, גודל השכבה הדקה ביותר שאנחנו מתעסקים איתה בייצור טרנזיסטורים היא בקנה מידה ננומטרי- כלומר כמה אטומים בודדים.

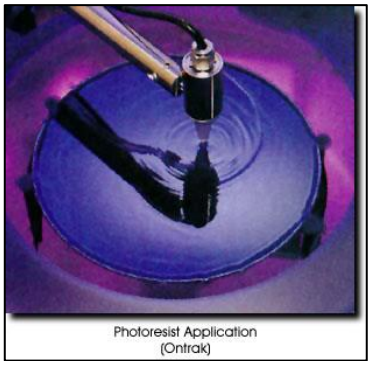
ידידנו הכימאים גם גילו לנו שקיימים חומרים שבחשיפה לקרינה משנים את תכונותיהם (למשל, הופכים מנוזל למוצק). בהקשר של ייצור מעגלים משולבים, החומרים האלו נקראים פוטורזיסט. כשמשלבים את שתי קבוצות החומרים האלו אנחנו מגלים שיש לנו את היכולת להשפיע על מרקם והולכה חשמלית של חומרים בקנה מידה שאנחנו רוצים - נשמע כמו התחלה טובה...

אז איך נראה תהליך של ייצור מעגל משולב? מעגל משולב מיוצר בשכבות, ככה שבגדול אנחנו יכולים לפשט את התהליך לתהליך של יצירת דפוסים בחומר על גבי המצע שאנחנו בוחרים. התהליך נקרא פוטוליתוגרפיה, ובפישוט רב, הוא נראה בערך ככה:

- **דפוזיציה**: יוצרים שכבה של החומר שממנו עשוי הדפוס שנרצה להשאיר על גבי המצע.
- **השמת פוטורזיסט**: מורחים שכבה של החומר הרגיש לקרינה על גבי החומר שנרצה להשאיר.
- **חשיפה לקרינה**: דרך מסיכה[4] (חומר חסין לקרינה שחתוך בצורת הדפוס שנרצה להשאיר) חושפים את הפוטורזיסט לקרינה ואפקטיבית מקשיחים את הפוטורזיסט לפי הדפוס שתכננו.
- **איכול**: מנקים את הפוטורזיסט שלא התקשה, ומשתמשים בחומר שלא ממס את הפוטורזיסט אבל כן את החומר שמתחתיו כדי ליצור את הדפוס הרצוי. אחר כך ניתן להסיר את הפוטורזיסט הקשיח שנותר (בדרך כלל בעזרת חומר שממס את הפוטורזיסט אבל לא את החומר שמתחתיו) ונותרנו עם הדפוס שרצינו.







תמונה של השמת פוטורזיסט, אילוסטרציה של חשיפה לקרינה* ואיכול כימי-מכני. קשה למצוא תמונה מובנת של תהליך הדפוזיציה כי הוא נעשה בתא סגור ובד"כ עם חלקיקים של החומר שאותו נרצה להשאיר מפוזרים בגז- תהליך שנקרא physical vapor deposition (PVD).

*החשיפה לקרינה לא נעשית באופן ישיר כמו בציור, אלא דרך הרבה מראות במכונה גדולה. אתר של ASML, החברה שמייצרת את המכונות האלו

חוזרים על התהליך בסביבות ה-40 פעמים עבור השכבות השונות שבמעגל המשולב, עד שיצרנו את כל המבנים שרצינו ליצור. ניתן לחשוב על התהליך כסוג מסובך של הדפסה בתלת מימד, שבה אנחנו מדפיסים שכבות דו-מימדיות בצורות שונות אחת על גבי השנייה וככה יוצרים מבנה תלת-מימדי מסובך.

[4] את המסיכה מייצרים בגודל פרקטי לייצור ע"י מכונות- גדול משמעותית מרזולוציית הדפוסים שברצוננו להשאיר. אחר כך, באמצעות סדרה של מראות ועדשות, ממקדים את ה"אור" (בפועל קרינה באורכי גל קצרים יותר) שנותר אחרי המעבר במסיכה עד שהוא מגיע לרזולוצייה של הדפוסים שנרצה להשאיר.

כעת אנחנו יודעים עובדה קריטית לעתיד המחקר ולתחום: כיוון שכרגע ובעתיד הנראה לעין המבנים שמרכיבים את המעגלים המשולבים מיוצרים בעזרת קרינה מסוגים שונים, תמיד מובטח לנו שתהיה שיטה להשפיע על המבנים הללו בעזרת אותה קרינה (בעצם, כיוון שהקרינה מייצרת את הטרנזיסטורים שלנו היא תמיד חייבת להיות יכולה להשפיע ברזולוציה שלהם).

אז איך משפיעים על מעגלים משולבים בעזרת קרינה?

עכשיו כשאנחנו יודעים שניתן להגיע לרזולוציה שבה אנחנו רוצים לעבוד בעזרת קרינה אלקטרומגנטית, טבעי שנבחר באותה קרינה גם כדי לנסות לשלוט על המתח באותה רזולוציה, האינטואיציה היא שקרינה נושאת אנרגיה, אנרגיה מכל סוג יכולה להפוך לאנרגיה חשמלית, ואנרגיה חשמלית מובילה לשינוי במתח שאנחנו רוצים. לעזרתנו שוב נחלצת הפיזיקה, ומיידעת אותנו על תהליך שנקרא גנרציה אופטית, מבלי להיכנס לפיזיקה יותר מדי (ממליץ בחום על ערך הויקיפדיה של "מוליך למחצה" ועל הקורס שצירפתי למעלה) הארה על מוליך למחצה (אפקטיבית, הכנסת אנרגיה לחומר) יכולה לגרום לפיצול של מטען חשמלי (אטום ניטרלי הופך לגרעין טעון חיובית ואלקטרון טעון שלילית) וליצירת זרם בתוך המוליך למחצה. התהליך שבו אנחנו משתמשים בקרינה אלקטרומגנטית כדי לייצר זרם במעגלים משולבים וכך להשפיע על עבודתם נקרא באנגלית EMFI: Electro-Magnetic Fault Injection.

משמו של התהליך אפשר להבין שהכוונה היא בעיקר לייצר שגיאות, כלומר להקריס את התוכנה או את אלגוריתם שרץ על גבי המעבד. ואכן, מדובר בפרקטיקה מאוד דורשנית, כדי להפוך ביט אחד צריך לייצר זרם במקום ספציפי, והרבה יותר קל לשנות ביט אחד כדי להפוך פקודה חוקית לבלתי חוקית (ברוב הפעמים לא כזה משנה איזה ביט הופכים כדי להקריס את האלגוריתם - פשוט מקרינים הרבה על אזור הזכרון בנסיון להשחית אותו). אבל צריך לזכור: אם יש לנו את היכולת לשלוט בביט אחד, יש לנו את היכולת לשלוט בכולם.

פרקטיקה

אז מה עושים עם זה בפועל? נגיד ויש לנו יכולת לשלוט על תוכנם של ביטים בזכרון, מה השלב הבא? המחשבה הראשונה תהיה לכתוב תכנית זדונית (למשל כזו ששולחת מידע פרטי לשרת בשליטתנו) ולקמפל אותה לקוד בינארי בהתאם לארכיטקטורת המעבד שאותו אנחנו תוקפים, לטעון אותה לזכרון הפקודות שמצאנו בשלב המחקר ע"י השמה של כל הביטים במקום הנכון, וניצחנו...

אבל דבר כזה ייקח המון זמן, אפילו התכנית הזדונית הכי קטנה לאחר קמפול תהיה בגודל מאות רבות של בתים שהן אלפים רבים של ביטים, ובהנחה שאנחנו תוקפים ביט-ביט מתקפה שכזאת תיקח כמות זמן שאנחנו לא רגילים אליה בעולם אבטחת המידע - קנה מידה של שעות.

היינו רוצים למצוא את המקומות הכי טובים לתקיפה מהסוג שפיתחנו, מקומות שבהם תקיפה של מספר קטן של ביטים תביא לנו שליטה מוחלטת במחשב. פה מגיעה לעזרתנו העובדה שמעבדים מודרניים לא תוכננו לעמוד בסוג כזה של תקיפות, לא מבחינת מימוש פיזי בחומרה ולא מבחינת ארכיטקטורת פקודות (ISA). סט היכולות שאנחנו יכולים להשיג בעזרת תקיפה יחסית קצרה מהסוג שפיתחנו הוא רחב מאוד, ונותן לנו אפשרויות שקשה מאוד להשיג עם תקיפה קונבנציונלית.

אז מה אפשר לעשות עם מספר קטן של היפוכי ביטים?

שינוי פקודות

הרבה מעבדים מודרניים משתמשים בארכיטקטורת פקודות מסוג CISC - ראשי תיבות של Complex Instruction Set Computer, וכשמה כן היא, ארכיטקטורת פקודות מסובכת. זה אומר שאורך פקודה במעבדי CISC יכול להשתנות, מה שמסבך את התכנון אבל מאפשר להקטין קבצי הרצה (מאחר והפקודות הקצרות שמורות לפעולות נפוצות יותר, בממוצע הקוד הבינארי קצר יותר). התוצאה הבלתי רצויה של הארכיטקטורה הזאת היא שקידוד הפקודות הנפוצות צפוף יותר, כלומר סביר שבעזרת מספר קטן יחסית של היפוכי ביטים נוכל לעבור מפקודה חוקית אחת לאחרת.

כמובן שעם תקיפה קצת יותר ארוכה יהיה אפשר לכתוב כל פקודה אחרת כרצוננו, אבל זה תהליך יותר ארוך ששמור לפקודות בעלות חשיבות גדולה בקוד. בהרבה מקרים נעדיף רצף ארוך של פקודות נפוצות (שיצרנו ע"י היפוך מספר קטן של ביטים בכמה פקודות נפוצות) על פקודה אחת לא נפוצה (שיצרנו ע"י היפוך מספר גדול של ביטים בפקודה אחת), למשל לצורך פיזור רעש אלקטרומגנטי.

בקרת זרימה

כדי להשיג שליטה במחשב ולבצע קוד כרצוננו יש צורך בתקיפה של הרבה ביטים, לכל הפחות אחד על כל פקודה שנרצה לשנות. לאורך עשרות או מאות פקודות הכמות הזו מצטברת, ויכולה להאריך את התקיפה שלנו משמעותית. בפועל, בהרבה מקרים יספיק לנו לתקוף ביט אחד או שניים כדי לקבל שליטה על ביצוע קוד במעבד.

בקרת זרימה בקוד (למשל באסמבלי) מתבצעת ע"י השוואה בין שני ערכים, קפיצה לקטע קוד אחד אם מתקיים יחס כלשהו ביניהם ולקטע קוד אחר אם לא. תוצאת ההשוואה הזאת (שמתבצעת במקומות שונים כתלות בסוג ההשוואה ובארכיטקטורת המעבד) נשמרת ברגיסטר כלשהו, שלצרכים שלנו נראה כמו תא זכרון רגיל לחלוטין (ישנם סוגים שונים של זכרונות בתוך מעבד, אבל באופן כללי כולם מורכבים ממוליכים למחצה / טרנזיסטורים, כלומר אנחנו יכולים לשלוט על תוכנם באותה שיטה). אם נדע איפה נשמר הערך שנבדק ע"י התכנית כדי להחליט לאיזה קטע קוד לקפוץ, נוכל לתקוף אפילו רק ביט אחד, ולהסיט את ביצוע הקוד לקוד שאנחנו רוצים שיתבצע. בכך נוכל לגרום לביצוע קוד שלא היה אמור להתבצע, ולקבל הרשאות שלא היינו אמורים לקבל.

בנוסף, יש לנו יכולת לדלג קדימה ואחורה בקוד, אם נתקוף את הרגיסטר שאחראי על מיקום הפקודה שעתידה להתבצע (IP/PC/EIP) נוכל לדאוג למשל שפקודת השוואה לא תתבצע, פקודת בקרת הזרימה שאחריה תבדוק את הערך הקודם של הרגיסטר שמכיל את תוצאת ההשוואה (שיכול להיות שונה מהערך התקין), ונוכל לכוון ביצוע קוד בעזרת תקיפה של ביט אחד בלבד!

Use-case נפוץ יותר: התרחיש הקריפטוגרפי

בעוד תקיפה של מעבדים מודרניים לשימוש כללי בהחלט אפשרית בשיטות שתיארנו, תרחיש התקיפה פחות סביר. מעטות הפעמים שתהיה לנו גישה פיזית למעבד שאנחנו רוצים לתקוף, ועוד יותר מעטות הפעמים שתהיה לנו גישה פיזית למעבד בתנאי מעבדה. למרות שכולי תקווה שבעתיד פיתוחים בתחום הלייזר יהפכו תקיפה שכזו לקלה יותר לביצוע, כרגע היא דורשת עמדת לייזר ממונעת שהיא גם גדולה מאוד (לא ניתנת לנשיאה) וגם יקרה מאוד. התרחיש היותר מעניין כרגע עם הטכנולוגיה שבידינו, הוא התרחיש של חומרה מאובטחת. יישומים רבים בחומרה (למשל ארנקי קריפטו) מבטיחים הגנה מוחלטת על מידע (גם במקרה של נפילת החומרה לידי גורם זדוני) בעזרת שמירת מפתחות הצפנה קריפטוגרפיים על חומרה שנגישה רק למכשיר עצמו (ראו ערך TPM). מתקפה כמו שלנו תוכל לשנות את מפתחות ההצפנה האלו למפתחות כרצוננו, ובכך בעצם לבטל את ההצפנה המובנית במכשיר, תרחיש מסוכן מאוד כשמדובר על מכשירים שמופקדים לשמור על כמויות גדולות של מטבעות דיגיטליים...

אנקדוטות אחרונות

יש מספר דברים שיכולים להקל את התקיפה שלנו, למשל במקרה של שליטה על מידע חסוי: בניגוד לשינוי של פקודות יש פעמים שנרצה להשאיר בזכרון ערך מסוים ולא נדע מראש מה הערך שקדם לו. במקרה כזה היה עוזר לנו לגלות את הערך הקודם, כדי שנוכל לשנות רק את הביטים ההכרחיים - מה שיכול לקצר לנו את התקיפה משמעותית...

כלומר אנחנו מחפשים מתקפות על החומרה שידליפו לנו מידע, אידאלית לפני שהתחלנו את התקיפה. לעזרתנו מגיע סוג מתקפה נפוץ בעולמות החומרה שנקרא מתקפות ערוץ צד (SCA או Side-Channel Attacks באנגלית). המתקפות האלה מאפשרות לנו, בהינתן דלף מידע בתהליך (בד"כ צריכת חשמל או תזמון שונה לפקודות) לגלות מידע שאמור להיות חסוי בפנינו. זהו נושא שלם כשלעצמו שלא איכנס אליו במאמר זה (אולי נושא למאמר המשך) אבל הרבה פעמים נעשה שימוש בתקיפת ערוץ צד בשילוב עם EMFI (התקיפה שלנו) כדי להשיג יכולת קריאה / כתיבה גנרית בזכרון שמור.

סיכום

כשמחשיבים את העולם הפיזי כממשק תקיפה ליישומים בתוכנה, נגלה בפנינו עולם שלם של יכולות שלא חשבנו עליהם בכלל, הרבה מעבר לתקיפות נקודתיות. בין היתר, ישנם שימושים רבים לקרינה אלקטרו-מגנטית בכל הנוגע לעולמות החומרה, בתחומים כמו אבטחת מידע, אימות חומרה (וידוא שחומרה זרה שקיבלנו לא מכילה חלקים זדוניים, רלוונטי במיוחד בעולם שבו רוב החומרה שלנו מיוצרת בחו"ל), ותהליכי ייצור. במאמר זה סקרתי סוג אחד של מתקפה שמשתמשת בקרינה אלקטרומגנטית, בצורה יחסית בסיסית. אני מעודד את כל מי שהנושא מעניין אותו ללמוד עוד, בין אם מהקישורים שצירפתי או מחומר שקיים באינטרנט. התחום הזה מתפרש על עולמות ידע שונים שנבנים אחד על השני ומתערבבים אחד בשני, יש אינסוף דברים חדשים ללמוד...

וברוח זו, אם פספסתי משהו / טעיתי במשהו / יש משהו שאתם רוצים לתקן - תרגישו חופשי לפנות אלי במייל: itayroiz@gmail.com

תודה ל**ד"ר איתמר לוי** ול(בקרוב ד"ר) **אור נחום** על ההדרכה האקדמית, ו**לגילעד סבוראי** על העזרה בפרויקט. היה הרבה יותר קשה להיכנס לעולם המסובך הזה בלעדיכם!

ותודה ל**אפיק קסטיאל** על המגזין ועל עריכת המאמרים שקשה להפסיק לקרוא.

מקווה שנהניתם!

מקורות, תמונות ותרשימים

רמות אבסטרקציה, אינטל 4004, אינטל Raptor Lake, מוטורולה 68040, פסולת אלקטרונית, אריזת DIP, אריזת BGA, גבישי סיליקון, פרוסות סיליקון, השמת פוטורזיסט, פוטוליתוגרפיה, CMP

על המחבר

בן 19, סטודנט שנה רביעית לתואר ראשון בהנדסת מחשבים. חובב אבטחת מידע, מחשבים (תוכנה וחומרה) ובעצם כל נושא שקשור לטכנולוגיה.