הפקולטה למדעי המחשב
ע"ש הנרי ומרילין טאוב

הטכניון

SHIELDING AGAINST EVIL MAID ATTACKS

Protect Your Unattended Device from Unwanted Physical Attacks - Windows 10 Editions

מגישים:

אלעד כהן & קשת מאיר

מנחה:

יובל רון

תוכן עניינים

רשימת גרפים ואיורים

מטרת הפרויקט

פרויקט זה נועד לתת פתרון בזמן אמת לאיום מסוג "Evil Maid", אשר בו התוקף מנצל גישה פיזית למכשיר נעול, ללא נוכחות הבעלים, על מנת לקבל גישה לא מורשית למידע השמור על המכשיר או לבצע פעולות עליו. ההתקפה יכולה להתבצע תוך ניצול של חולשת אבטחה במנגנון הנעילה של מערכת ההפעלה, המאפשרת גישה למכשיר גם ללא פרטי הזיהוי. במסגרת הפרויקט היה עלינו לחקור את סוגי ההתקפות השונות מהן נדרש להגן, לאפיין את פרופילי התקיפה האפשריים במסגרת האיום ולהציע פתרונות מתאימים להתמודדות. המיקוד בפרויקט הינו במחקר ניטור אירועים חשודים בזמן שהמחשב במצב מסך נעול, אשר עלולים להעיד על התרחשות תקיפת Evil Maid. התוצר הרצוי שהוגדר עבור הפרויקט הינו אפיון מערכת זיהוי והגנה מפני האיום עבור מערכת ההפעלה Windows 10, והרחבה אפשרית הינה הוספת התרעה למשתמש בדרכים שונות.

רקע

Evil Maid הינה התקפה מבוססת ניצול של גישה פיזית למכשיר נעול, ללא נוכחות המשתמש, על מנת לפרוץ אליו ולגשת למידע אישי רגיש[1]. שם המתקפה נובע מתרחיש אפשרי בו המשתמש משאיר ללא השגחה את המחשב הנייד שלו במצב נעול בחדר המלון. בזמן שהמשתמש שוהה מחוץ לחדר, התוקף, שהינו עובד ניקיון "תמים" של המלון (Evil" Maid"), מנצל את הגישה שיש לו מתוקף תפקידו אל החדר ופורץ למחשב הנעול כדי לבצע פעולות זדוניות. כמובן שהתקפות מסוג זה אינן מוגבלות לתרחיש המתואר ועלולות להתרחש בכל זמן ומקום שבו המכשיר האישי נמצא ללא השגחה – בספריות, בתי קפה, שדות תעופה, ועוד. ההתקפות יכולות להתבצע על סוגי מכשירים שונים (מחשבים, טלפונים ניידים וכו') ועל מערכות הפעלה שונות (Windows, Android וכו').

הגדרת האיום

התקפות Evil Maid לרוב פוגעות באחד או יותר מעקרונות אבטחת המידע הבאים: סודיות המידע, שלמות המידע ואמינות המידע. במהלך ההתקפה נחשף מידע של המשתמש שלא היה אמור להיות חשוף במצב נעול, שונה או נמחק המידע שהיה שמור במכשיר או שנפגעה אמינות המידע שעתיד להיווצר. בתרחיש הנפוץ התוקף מבצע את המתקפה תוך ניצול של חולשת אבטחה לא ידועה במנגנון הנעילה של המחשב וללא שום צורך בידע קודם על הסיסמה של המשתמש. רבות מהחולשות מאפשרות לתוקף לבצע פעולות דוגמת הרצת קוד, גניבה, מחיקה או קריאה של קבצים במכשיר ואף ביצוע פעולות בשמו של המשתמש, כגון העברת כספים ופרסום ברשתות חברתיות.

סקירת ספרות

בסקירת הספרות התמקדנו בשני תחומים - חולשות שנמצאו שיכולות לשמש להתקפת Evil Maid ופתרונות קיימים להתקפות אלו. בעוד שהתקפות מסוגים אחרים מקבלות מענה מגוון מגופי אבטחת המידע בעולם, התקפות Evil Maid

[1] https://en.wikipedia.org/wiki/Evil_maid_attack

אינן זוכות למענה מקיף ונדמה שתחום זה נדחק לשוליים, וזאת על אף שקיימות חולשות רבות בתחום, ומתפרסמות בו חולשות חדשות באופן תדיר.

ממחקרים קודמים בנושא עולה כי מערכות הפעלה המאפשרות פתיחת אפליקציות על גבי מסך הנעילה נוטות להיות יותר חשופות לחולשות אלו. אפליקציות לדוגמה הינן עוזר/ת קולית[2], אפשור התחברות לרשתות אלחוטיות חדשות[3], שירותי נגישות דוגמת Narrator, ועוד.

חולשות קודמות בתחום

בסקירת הספרות מצאנו פרסומים רבים על חולשות המאפשרות התקפות Evil Maid, רובן התגלו במערכות הפעלה או במוצרים נלווים למערכת ההפעלה כמו עוזרות קוליות, על ידי חוקרים עצמאיים וחוקרים מהאקדמיה. אלו פרסמו את החולשה לאחר ביצוע תהליך הסגרה מול החברה שמפתחת את מערכת ההפעלה בה נמצאה החולשה. אנו התמקדנו בחולשות אשר התגלו במערכת ההפעלה Windows 10, ואלו יפורטו להלן, אך במהלך המחקר נתקלנו בחולשות שהתגלו במוצרים נוספים דוגמת העוזרת הקולית Siri.

במחקר "On the Security of Voice Assistants on Lock Screens"[4] שפורסם ע"י יובל רון במאי 2020, נדונה לעומק הסוגייה של פגיעות מחשב להתקפות בעת המצאות במסך נעילה, בדגש על חולשות הנובעות משימוש בעוזרת הקולית (VA – Voice Assistant). כחלק מהמחקר הוצגו מעל עשר תקיפות המנצלות חולשות הנוגעות למערכת הפעלה Windows. בכל אחת מהתקיפות שהוצגו ניתן לראות מאפיינים דומים: פנייה לעוזרת הקולית כדי לנצל שירות הממומש באופן לא מאובטח (לדוגמה: Photo reminder - שימוש בתזכורות עם תמונה מצורפת, או חיפוש ב-Bing באמצעות פקודות קוליות) וניצול חולשה בממשק המשתמש לביצוע פעולה זדונית (לדוגמא: לחיצה על קישור המאפשר פתיחת File Explorer, וצפייה בקבצים של המשתמש).

לדוגמא, החולשה Open Sesame[5] נובעת מתכנון לקוי של מנגנון העוזרת הקולית. כאשר משתמש פותח את חלון חיפוש הקבצים ב-Windows, מופעלת ברקע העוזרת הקולית על מנת לסייע לו. באופן לא מכוון כחלק מהמימוש של Cortana, קיים גם הקשר ההפוך – הפעלה של Cortana מאפשרת גישה למנגנון חיפוש הקבצים. משמעות הדבר הינה שבמערכות בהן מאופשרת הפעלה של העוזרת הקולית ממסך הנעילה, תוקף יוכל להשיג גישה למערכת הקבצים ללא סיסמה, להריץ פקודות, להעביר קבצים כרצונו מהתקן חיצוני כולל קבצי הרצה זדוניים ועוד.

קיימות עוד חולשות רבות שפורסמו ומאפשרות התקפות Evil Maid על מערכות Windows. לא נסקור כאן את כולן.

---

[2] חולשה במנגנון Siri במערכת MacOS - https://www.youtube.com/watch?v=iVcX0kRR6Og&ab_channel=YuvalRonSec

[3] https://www.mathyvanhoef.com/2017/02/windows-10-lock-screen-abusing-network.html

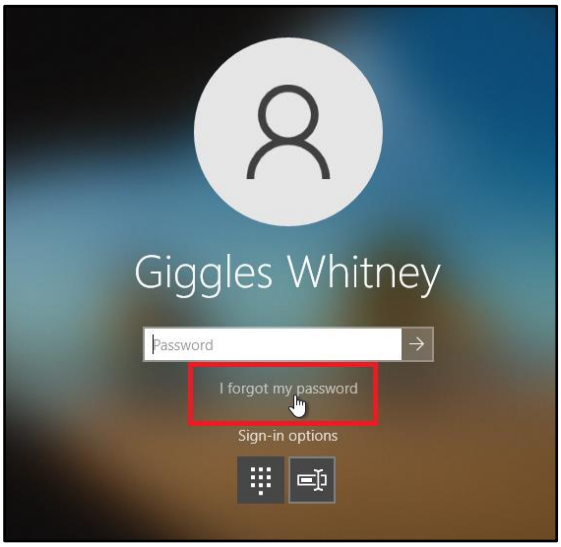
[4] https://www.cs.technion.ac.il/users/wwwb/cgi-bin/tr-get.cgi/2020/MSC/MSC-2020-29.pdf

[5] CVE-2018-8140 - https://www.youtube.com/watch?v=cYeDYV4T3Vo&ab_channel=YuvalRonSec

דוגמה להעמקה – Lock Screen Bypass

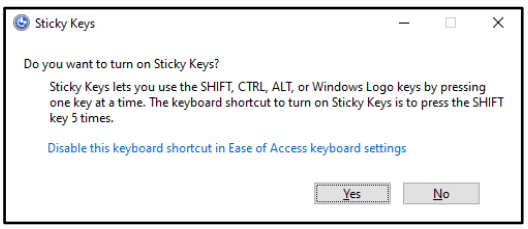
על מנת למקד את הקורא במהלך תיאור המחקר, בחרנו להתמקד בהתקפה עדכנית יותר, שפורסמה בינואר 2021, לפרט לעומק על מימוש ההתקפה, ומאוחר יותר להדגים באמצעותה כיצד המערכת שלנו מגנה מפני סכנות של התקפות מוכרות באופן מלא.

Lock Screen Bypass הינה חולשה ב-Windows 10[6] שהתפרסמה בינואר 2021, העוקפת את מנגנון האימות של Windows ומאפשרת הרצה של קבצים מהתקן אחסון נייד של התוקף. החולשה מתמקדת במנגנון Ease of Access של Windows. התוקף משתמש באפשרות "I forgot my password/PIN" במסך הנעילה, כפי שניתן לראות באיור 1, כדי לבצע יצירה של Session חדש ובו משתמש אוטומטי default-account אשר נוצר או נמחק בהתאם לפעולה.



איור 1 - תמונת מסך נעול הכוללת לינק לשחזור סיסמה

לאחר מכן התוקף לוחץ 5 פעמים על כפתור ה-Shift על מנת להפעיל את מנגנון Sticky Keys. לחיצה על הלינק אשר הופיע בחלון לפני סגירת החולשה (איור 2), תפתח חלון של תפריט ההגדרות ברקע.

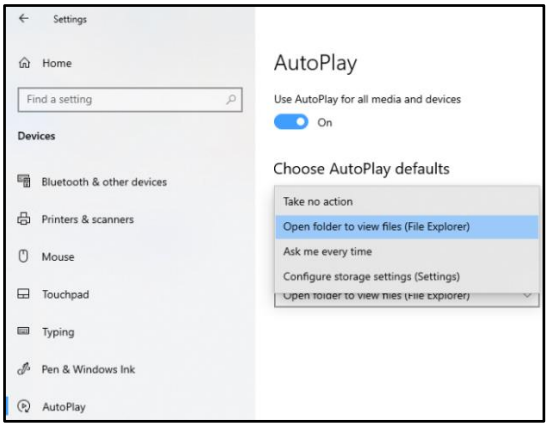


איור 2 - חלון Sticky Keys

כעת, כדי להעביר את הפוקוס אל חלון ההגדרות שנפתח שברקע, התוקף לוחץ מספר פעמים במקום שבו נפתח החלון עד אשר ה-Narrator מדווח על מעבר הפוקוס לחלון ה-Settings. לאחר מכן התוקף משתמש במקשי המקלדת וב-Narrator על מנת להצליח לנווט בתפריטים שמוצגים מעבר למסך הנעילה גם מבלי לראותם.

[6] CVE-2020-1398 - https://secret.club/2021/01/15/bitlocker-bypass.html

התוקף מאפשר את ההגדרה של AutoPlay להתקני אחסון ניידים, כפי שניתן לראות באיור 3, מה שיוביל להרצה מיידית של קובץ הרצה אשר נמצא בהתקן אחסון נייד בעת החיבור למחשב.

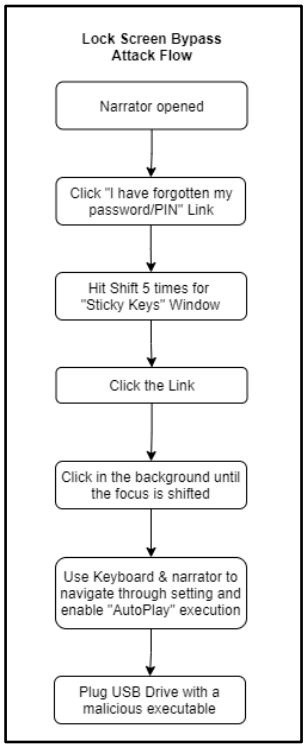


איור 3 – חלון הפעלת AutoPlay בהגדרות המחשב

כעת כל שנותר לתוקף הוא לחבר התקן נייד המכיל קובץ הרצה כרצונו.

הרצת הקוד מאפשרת לתוקף חופש פעולה נרחב, כולל גניבה של קבצים להתקן הנייד, ניצול חולשה נוספת להעלאת הרשאות, פגיעה במכשיר, ביצוע פעולות בשם המשתמש ועוד.

כאשר מסתכלים על תרשים התקיפה, המופיע באיור 4, מנקודת מבט הגנתית, יש לשים לב שסדר הפעולות שהציג מגלה החולשה אינו בהכרח הסדר היחיד המאפשר את ההתקפה. לפירוט נוסף ביצד התמודדנו עם אתגר שינוי הסדר, ראו פרק "המחקר התאורטי – תכנון המערכת" בהמשך.



איור 4 - תרשים זרימה לביצוע מתקפת "Lock Screen Bypass"

חולשה זו היוותה השראה לחולשה נוספת[7] שפורסמה באפריל 2021 ומבוססת על עקרונות דומים. החולשה החדשה הובאה לידיעתנו כאשר המחקר שביצענו היה בשלבים מתקדמים, והיוותה עבורנו כלי נהדר לבדיקת הלוגיקה שבנינו עבור המערכת כנגד תקיפות Evil Maid עדכניות. פירוט נוסף בפרק "הצלחה בזיהוי תקיפה חדשה".

<u>ניסיונות קודמים להתמודדות עם האיום</u>

אחד מהפתרונות שהיו זמינים בעבר לאיום ה-Evil Maid היא האפליקציה Do Not Disturb (DND) ל-MacOS ול-iOS[8]. את האפליקציה היה המשתמש מתקין על ה-Mac הנייד שלו ועל מכשיר ה-iOS שלו, ומסנכרן בניהם. משהאפליקציה הייתה מותקנת, היה המשתמש מקבל התרעה ויזואלית על המסך בכל פעם שמסך ה-Mac היה מורם כי האפליקציה קיימת על המכשיר ומנטרת הרמות מסך, מה שיכל להרתיע תוקף. התרעה הייתה נשלחת באופן מוצפן לכל מכשיר iOS שסונכרן עם האפליקציה שב-Mac. בקבלת ההתרעה למכשיר ה-iOS יכול היה המשתמש לשלוח באמצעותו פקודה למחשב לכבות עצמו או לצלם תמונה של התוקף באמצעות ה-Webcam, שהייתה נשלחת באופן מוצפן חזרה למכשיר ה-iOS. בנוסף, הייתה למשתמש אפשרות להגדיר סקריפט או קובץ הרצה שיופעלו בכל פעם שהמסך מורם, ואף להגדיר לאפליקציה לשמור Log של אירועים חשודים שבוצעו לאחר הרמת מסך (חיבור USB, יצירת תהליכים חדשים וכו').

חשוב להדגיש כי האפליקציה לא ניסתה להבחין בין הרמת מסך חוקית לזדונית, ונשענה כמעט באופן בלעדי על Event של "Open Lid" כבסיס ליצירת התרעה. ההישענות על הרמת המסך כאירוע הבלעדי שעל בסיסו מיוצרת התרעה גרם לשני חסרונות משמעותיים של האפליקציה, ובסופו של דבר הופסקה התמיכה וההפצה שלה. החיסרון הראשון הוא התראות השווא הרבות - מפתחי האפליקציה אמנם זיהו באופן נכון כי התקפות Evil Maid על פי רוב דורשות שמסך המחשב הנייד יהיה מורם, אך חוסר ההבחנה בין הרמת מסך חוקית לזדונית משמעותו התראות על בסיס יום-יומי מהמערכת. נציין כי המפתחים מימשו אפשרויות לכיבוי חלק מההתראות, הנוחה שבהן היא אי-שליחת התרעה, אם ב-5 השניות שלאחר הרמת המסך בוצע אימות TouchID מוצלח. החסירון השני הוא חוסר היכולת להתריע בפני תוקף שניגש על המחשב כשהמסך שלו כבר מורם. כלומר, המשתמש חייב לזכור לסגור את מסך המחשב בכל פעם שהוא מתרחק ממנו על מנת להיות מוגן. התייחסות למאפיינים נוספים של התקפת Evil Maid, כפי שיפורטו בהמשך המסמך, הייתה יכולה לסנן חלק מהתראות השווא ולתת מענה להתקפות שבהן התוקף ניגש למחשב שהמסך בו כבר מורם.

מלבד האפליקציה המתוארת לעיל, אשר כאמור אינה נתמכת כעת, לא מצאנו תיעוד לפתרונות פעילים הנותנים הגנה מפני איום הגישה הפיזית.

[7] ZDI-21-1053 - https://halove23.blogspot.com/2021/09/zdi-21-1053-bypassing-windows-lock.html?m=1

[8] https://digitasecurity.com/donotdisturb/userguide

המחקר

ניתן לחלק את המחקר שביצענו במסגרת הפרויקט לשני סוגים עיקריים של מחקר:

1. מחקר תיאורטי - מטרתו הייתה תכנון של כלי מקיף להתמודדות עם התקפות Evil Maid, ובכלל זה זיהוי האיום והתמודדות אתו. במסגרת חלק זה, למדנו מה החולשות הקיימות וכיצד ניתן לזהות אותן, אך גם מה הם המאפיינים המשותפים להתקפות אלו וכיצד ניתן לזהות התקפות דומות בעתיד.
2. מחקר פרקטי, שמטרתו הייתה לחקור אילו כלים מאפשרים לממש את הכלי שאופיין בחלק התיאורטי.

המחקר התיאורטי - תכנון המערכת

במסגרת המחקר התיאורטי חקרנו את ההתקפות המוכרות בתחום. תחילה אפיינו את האירועים הניתנים לניטור המתרחשים במהלך ההתקפות ואת תרשימי הזרימה שלהם לאורך ההתקפות. לאחר מכן, מצאנו נקודות דמיון בין האירועים המתרחשים בהתקפות השונות, ואפיינו אירועים שיכולים להעיד על כך שמתרחשת התקפת Evil Maid שאינה בהכרח מוכרת למערכת. לבסוף, חשבנו על דרכים להתמודדות עם אירועים חשודים אלו ועם ההתקפות המוכרות.

נציין כי במהלך תכנון המערכת הוקדשה מחשבה גם לאיזון שבין הרצון לזהות פעולות חשודות במכשיר לבין האפשרות שחלק מהפעולות נעשות על ידי המשתמש באופן תמים, ולכן עלינו לאזן בין שליחת התרעות ונקיטת צעדי הגנה לבין הימנעות מהתרעות שווא רבות (בעיה שהייתה קיימת במערכת Do Not Disturb שתוארה לעיל). בנוסף, שלב זה במחקר אמנם תיאורטי, אך הוא מחובר לפרקטיקה של מימוש המערכת (הפרק הבא), ולכן התמקד בהתקפות על מערכת ההפעלה Windows 10, ולא בהתקפות Evil Maid כלליות על כל מערכת הפעלה אפשרית.

מבנה המערכת

המערכת מבצעת שני תפקידים עיקריים - ניטור על אירועים בזמן מצב מסך נעול והתמודדות עם איומים והתקפות על המכשיר. למערכת שלושה מצבים, בהתאם לרמת האיום, והמעבר ביניהם נעשה בעקבות זיהוי של אירועים חשודים בתהליך הניטור. לכל אחד משלושת המצבים קיימות גם תגובות שונות של המערכת כדי להתמודד עם האיום והם אף קוראים למנגנוני ההגנה השונים של המערכת. שלושת המצבים הם:

1. **מצב 0: מצב השגרה** - המצב בו המערכת מתחילה את פעולתה ברגע נעילת המסך. במצב זה אין איום על המכשיר והמערכת רק מנטרת אחר האירועים השונים המתרחשים בו.
2. **מצב 1: מצב איום אפשרי** - המצב בו המערכת מזהה איום בגלל אירוע חשוד שקרה במכשיר, אך עדיין לא מסיקה כי מדובר בהתקפה, ויתכן כי מדובר בפעולה תמימה של המשתמש. במצב זה המערכת תשלח התרעה מינורית למכשיר הנייד של המשתמש (לדוגמה אימייל) כי יתכן שהמחשב שלו נמצא תחת איום. בנוסף, המערכת תנקוט צעדים להתמודדות עם האיום כמו הפעלת מנגנון הגנה מתאים לאירוע החשוד (הרחבה על מנגנוני ההגנה בהמשך).
3. **מצב 2: מצב התקפה אפשרית** - מצב זה מתחלק לשלושה תתי-מצבים:

א. **מצב 2A: מצב בו זוהתה תבנית התקפה מוכרת** - במצב זה המערכת מתריעה כי המכשיר נמצא תחת התקפה מוכרת וודאית. במצב זה תופעל אזעקה במכשיר, תשלח התרעה משמעותית למכשיר הנייד של המשתמש (לדוגמה שיחת טלפון או אזעקה), יופעלו כל מנגנוני ההגנה של המערכת ותצולם תמונה של התוקף.

ב. **מצב 2B: מצב בו זוהתה התקפה אפשרית ולא מוכרת** - במצב זה המערכת מתריעה כי הצטברו אירועים חשודים המעידים על התקפה אפשרית על המערכת. גם במצב זה, כמו במצב 2A, תישלח התרעה משמעותית למכשיר הנייד של המשתמש ויופעלו כל מנגנוני ההגנה הלא-אגרסיביים של המערכת, אך לתוקף הפוטנציאלי תינתן האפשרות להזין PIN (שהמשתמש הגדיר מראש), על מנת להזדהות כמשתמש ובכך להחזיר את המערכת למצב שגרה (מצב 0). אם המשתמש הזין PIN נכון, הרי שמדובר במשתמש לגיטימי (אנו מניחים כי התוקף אינו מכיר את ה-PIN כשם שאינו מכיר את סיסמת ההתחברות של המשתמש), ולכן נאפשר למשתמש להוסיף את הרצף שגרם להתרעה ל-White List, כך שלא נתריע עליו בהמשך.

ג. **מצב 2C** - זהו מצב המשך של מצב 2B, במסגרתו התוקף לא הצליח להזין PIN נכון. במצב זה רמת הודאות להתקפה גבוהה, ולכן נבצע את הפעולות שקיימות במצב 2A שלא ביצענו ב-2B, כמו הפעלת אזעקה.

State	Description	Actions
0	No Threat Detected	Continue monitoring
1	Possible Major Threat	Update system log
		Alert the user's external device (minor notification)
		Activate a suitable defense mechanism
2	Known Attack Pattern Detected (A)	Sound an alarm
		Activate all defense mechanisms
		Alert the user's external device (major notification)
		Record video from webcam and show video on screen
		Update system log
		Send forensics to an external source
	Possible Attack Detected (B)	Activate all non-aggressive defense mechanisms
		Alert the user's external device (major notification)
		Validate user by PIN code (with time limit)
		Record video from webcam and show video on screen
		Update system log
		Send forensics to an external source
	Possible Attack Detected and Validation Failed (C)	Sound an alarm
		Activate aggressive defense mechanisms

<u>מנגנוני הגנה</u>

כאמור, במערכת קיימים גם מנגנוני הגנה, שהמערכת מפעילה בהתאם לאירועים החשודים שהיא מזהה. במצבים 2A ו-2B המערכת מפעילה את כל מנגנוני ההגנה, ללא קשר לאירועים שזוהו. אלו הם מנגנוני ההגנה הלא-אגרסיביים:

1. חסימת USB - במסגרת מנגנון זה המערכת מנתקת את כל ה-Driver-ים של התקני האחסון שמחוברים למערכת ב-USB וחוסמת חיבורי USB חדשים למכשיר.

2. חסימת רשת - מנגנון זה יופעל בדרך כלל בעקבות תהליך חשוד של דפדפן אינטרנט. המנגנון יהרוג את התהליך של דפדפן האינטרנט החשוד וינתק את המכשיר מרשתות אליהן התחבר לאחר הנעילה.
3. הריגת תהליכים חשודים - המנגנון יהרוג תהליכים חשודים שיכולים לשמש תוקף, אשר החלו לרוץ לאחר הנעילה של המכשיר. ההגדרה של תהליכים חשודים תובא בהמשך.
4. חסימת Bluetooth - המנגנון מכבה את ה-Bluetooth במכשיר.

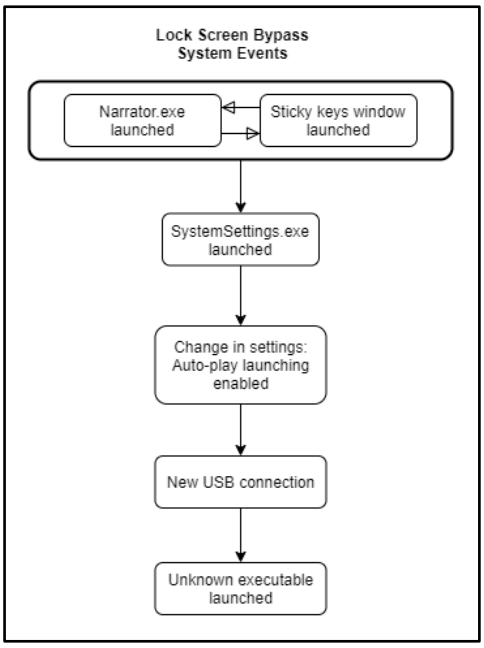
מנגנונים אגרסיביים:

5. כיבוי מחשב - המנגנון מכבה את המחשב. זהו צעד אגרסיבי שכן הוא יפגע בשימוש הרציף של המשתמש, ויתכן שחלק מהעבודה, שלא שמר, ימחק. עם זאת, הוא מאט את התוקף, ואף מונע ממנו פעולות התקפיות שדורשות משתמש Signed-in, שכן כיבוי המחשב גורר Sign-out של המשתמש.

התמודדות עם התקפות מוכרות

המערכת יודעת לזהות ולהגן הן מפני התקפות מוכרות והן מפני התקפות לא מוכרות. בחלק זה נפרט על ההתמודדות עם התקפות מוכרות. במהלך המחקר בחנו התקפות Evil Maid שונות שפורסמו, ניתחנו את האירועים המתרחשים במהלך כל אחת מההתקפות ובנינו תרשימי זרימה המתארים את ההתקפות, המכילים אירועים הניתנים לניטור. במהלך שלב זה בחנו גם האם שינוי סדר בפעולות ההתקפה מאפשרת לתוקף לעקוף את ניטור המערכת, מתוך מחשבה כי התוקף עלול להכיר את אופן פעולתה. לאחר מכן, בחרנו נקודות בהן על המערכת לנקוט צעדים אקטיביים על מנת לחסום את ההתקפה לפני שהתוקף מגיע להישג משמעותי.

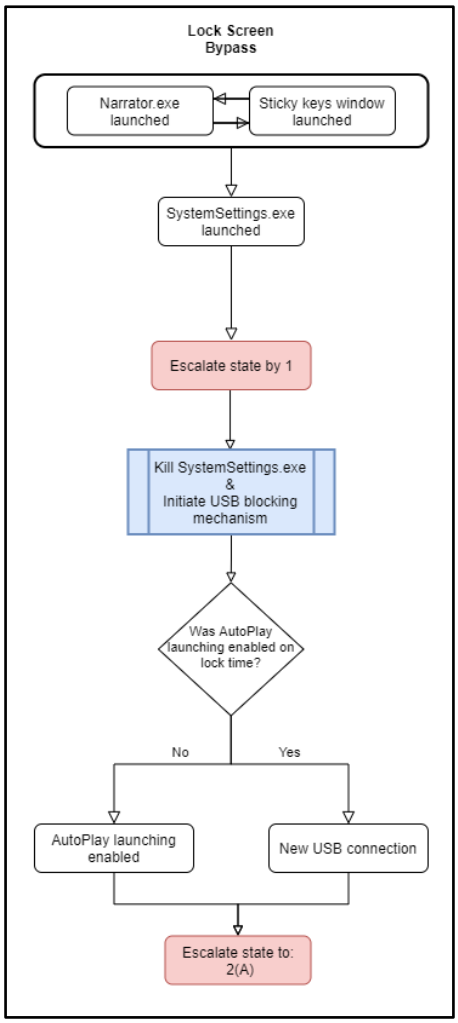
בסקירת הספרות הוצגה התקפת Lock Screen Bypass כדוגמה להעמקה, וכעת נרחיב כיצד המערכת מזהה את המערכת ומתמודדת איתה, כפי שניתן לראות באיור 5). בתחילת ההתקפה מתרחשים שני אירועים, שהסדר ביניהם אינו חשוב לצורך מימוש ההתקפה. אלו הם פתיחת ה-Narrator, שמתבטאת בפתיחת התהליך Narrator.exe, ופתיחת חלון ה-Sticky Keys, שמתבטאת בפתיחת התהליך sethc.exe, אם כי פתיחת תהליך זה היא לא אינדיקציה ודאית לפתיחת החלון (תנאי הכרחי אך לא מספיק). לאחר מכן, יפתח חלון ההגדרות ברקע, מה שילווה בפתיחת התהליך SystemSettings.exe. אירוע זה חייב להגיע אחרי האירועים הקודמים שכן אלו מאפשרים לפתוח את ההגדרות ולהעביר אליהם את ה-Focus. לאחר מכן תשתנה הגדרת ה-AutoPlay, יופיע חיבור USB חדש של התקן אחסון ויתכן וירוץ Executable חדש וזדוני. גם כאן, סדר האירועים חייב להישמר, כי כמובן שחלון ההגדרות חייב להיפתח לפני שינוי ההגדרות, ושינוי ההגדרות חייב להתרחש לפני חיבור ה-USB, כי שינוי הגדרתה-AutoPlay לא תקף ל-USB שכבר מחובר.



איור 5 - תרשים זרימה לפי אירועים להתקפת Lock Screen Bypass

מתוך הבנת האירועים המתרחשים בעת התקיפה בנינו תרשים זרימה לזיהוי והגנה מפני ההתקפה על ידי המערכת, כפי שניתן לראות באיור 6. בבואנו לקבוע כיצד תזהה המערכת את ההתקפה וכיצד תמנע אותה, ניתן לשים לב ששתי הפעולות הראשונות הן יחסית לגיטימיות. משתמש יכול להחליט להשתמש ב-Narrator ויכול ללחוץ על shift מספר פעמים כך שיפתח חלון ה-Stick Keys. לכן, בשלב זה לא נרצה להתריע על איום. אולם, הרצת תהליך ההגדרות SystemSettings.exe היא אינה פעולה לגיטימית ולא אמורה להיות גישה אליהן ממסך נעול. יותר מכך, זוהי פעולה עם פוטנציאל נזק משמעותי, כפי שניתן לראות בהתקפה זו ממש. לכן, בשלב זה כבר נרצה להתריע על איום באמצעות דרדור מצב המערכת ב-1 ולבצע פעולות למניעת המשך ההתקפה - הריגת SystemSettings.exe וחסימת חיבורי USB למכשיר. הדרדור נעשה רק לרמה 1, שכן אלו יכולות להיות פעולות שמשתמש לגיטימי ביצע בטעות, ואינן מעידות בהכרח על התקפה.

בשלב זה ההתקפה נבלמה ולא נגרם נזק ממשי למכשיר הנתקף. עם זאת, נמשיך לתאר את מנגנון ההגנה תחת ההנחה שהתוקף הצליח להתגבר על מנגנון ההגנה הראשון. כעת, המערכת בודקת האם AutoPlay היה מאופשר ברגע הנעילה (מידע שהיא שומרת בזמן נעילה). אם לא, הרי שההתקפה אמורה להמשיך בשינוי הגדרת ה-AutoPlay ושינוי של ההגדרה הזו הוא כבר זיהוי ודאי של ההתקפה ומעבר למצב 2A. אם ה-AutoPlay היה מאופשר כבר קודם, הרי שהתוקף יגלה זאת ולא יצטרך לשנות את ההגדרות, אלא מיד יוכל לחבר את ה-USB שלו, ואז המערכת תזהה זאת כהתקפה מוכרת ותעבור למצב 2A.



איור 6 - תרשים זרימה כולל תגובות המערכת להתקפת Lock Screen Bypass

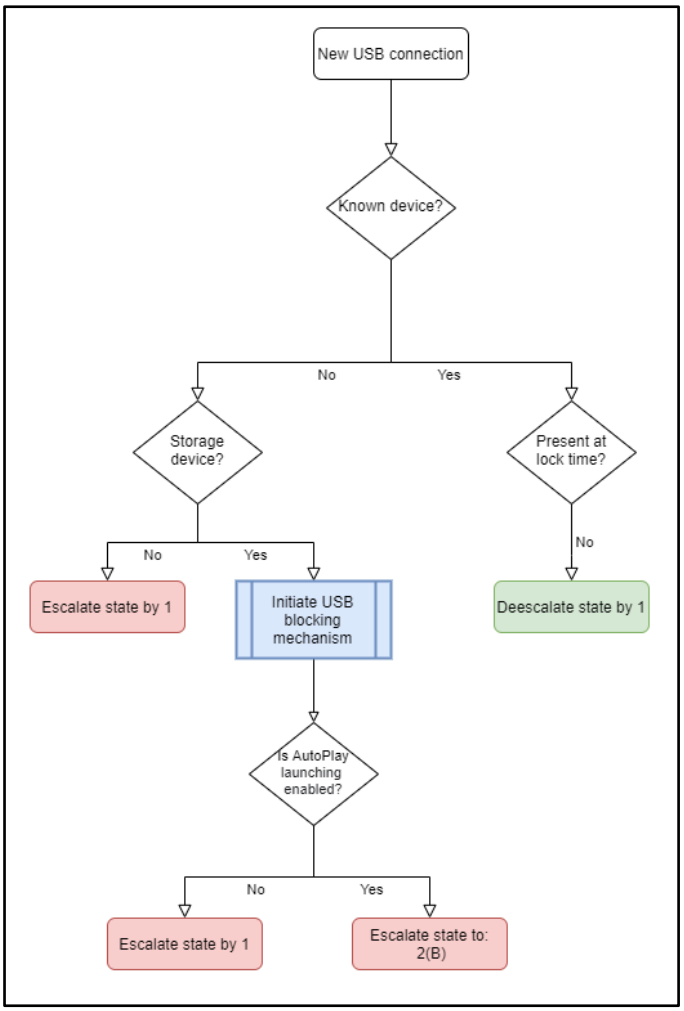
נציין כי מצב שבו התוקף מדלג על כל החלק הראשון של ההתקפה ומיד מנסה את מזלו ומחבר USB מבלי לנסות לפתוח ולשנות את ההגדרות הוא בהחלט אפשרי, אך החלטנו כי אז זו אינה התקפת Lock Screen Bypass המוכרת, אלא התקפה כללית לא מוכרת, והתמודדות עמה תתואר בחלק הבא.

לפירוט נוסף ראו נספח א' - תרשימי ניטור והגנה מפני התקפות מוכרות נוספות.

התמודדות עם התקפות לא מוכרות

כאמור, לאחר הבנת ההתקפות המוכרות והאירועים המתרחשים בהן, פנינו להגדרת היוריסטיות כלליות לזיהוי של אירועים חשודים, אשר יכולים להעיד על איום או התקפת Evil Maid, ובחלק זה נתאר אותן. חלק מהאירועים המתוארים בהגדרות ההיוריסטיות מדרדרים את מצב המערכת ישירות ל-2B, וחלק מדרדרים את מצב המערכת ב-1, כלומר אם המערכת הייתה במצב 0 היא תעבור למצב 1 ואם הייתה במצב 1 אז היא תעבור למצב 2B.

1. חיבור USB חדש (ראו איור 7) - התקני אחסון עם חיבור USB מאפשרים לתוקף הן להוציא מידע רב מהמחשב והן להריץ קבצי הרצה זדוניים שהכין מבעוד מועד, ולכן חיבור של התקני USB הוא אירוע שעל המערכת לנטר ואולי אף למנוע במצבים מסומים.

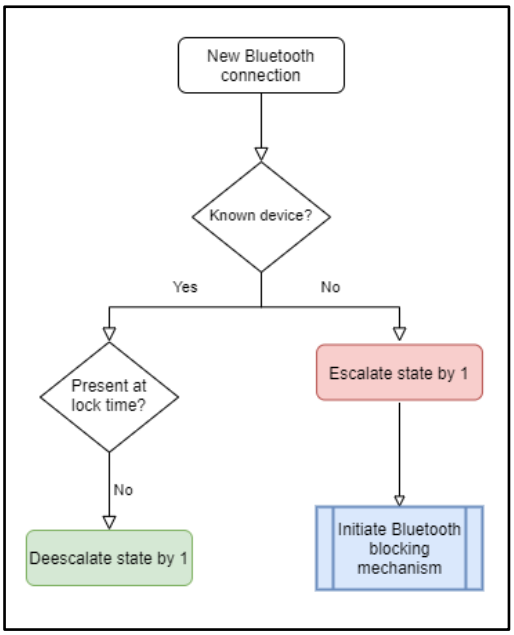


איור 7 - תרשים זרימה הכולל את תגובת המערכת לאירוע מסוג New USB Connection

ברגע חיבור התקן USB על המערכת לבדוק האם הוא התקן שחובר אל המכשיר בעבר, ולכן אינו מהווה התקן זדוני, או שהוא התקן שלא חובר למכשיר בעבר. אם הוא לא חובר בעבר, אם לא מדובר בהתקן אחסון, האיום הוא פחות משמעותי, ולכן המערכת רק תדרדר את מצב המערכת ב-1. אם מדובר בהתקן אחסון, המערכת תפעיל את מנגנון חסימת ה-USB ותדרדר את מצב המערכת ב-1 או ישירות ל-2B, אם AutoPlay מאופשר, כי אז ברור כי התוקף יכול להיות בעיצומה של התקפה.

אם התקן ה-USB חובר בעבר למכשיר, ברור כי מדובר בהתקן לא זדוני, ואף נרצה להקטין את מצב המערכת ב-1 אם הוא נמצא במצב גבוה יותר, כיוון שחיבור של USB מוכר הוא פעולה שמאפיינת משתמש לגיטימי. תוקף שמכיר את מנגנוני המערכת עלול לנצל מצב שבו למכשיר הנתקף כבר היה מחובר USB של המשתמש לפני התקיפה, ולהערים על המערכת בכך שינתק ויחבר אותו מחדש בעת ההתקפה כדי להקטין את מצב המערכת. לכן, הקטנת מצב המערכת ב-1 תעשה רק אם ה-USB המוכר שחובר לא היה מחובר ברגע הנעילה.

2. חיבור Bluetooth חדש (ראו איור 8) - תוקף יכול לעשות שימוש במכשיר Bluetooth לצורך התקיפה. בדומה לחיבור USB חדש, גם כאן נרצה לדעת האם המכשיר שהתחבר הוא מכשיר שמוכר מהעבר, ולכן מאפיין את המשתמש, ואף יגרום להורדה של מצב המערכת ב-1, או שמא מדובר במכשיר שלא חובר מעולם, ואז נרצה להפעיל את מנגנון חסימת Bluetooth. במקרה זה גם נדרדר את מצב המערכת ב-1.

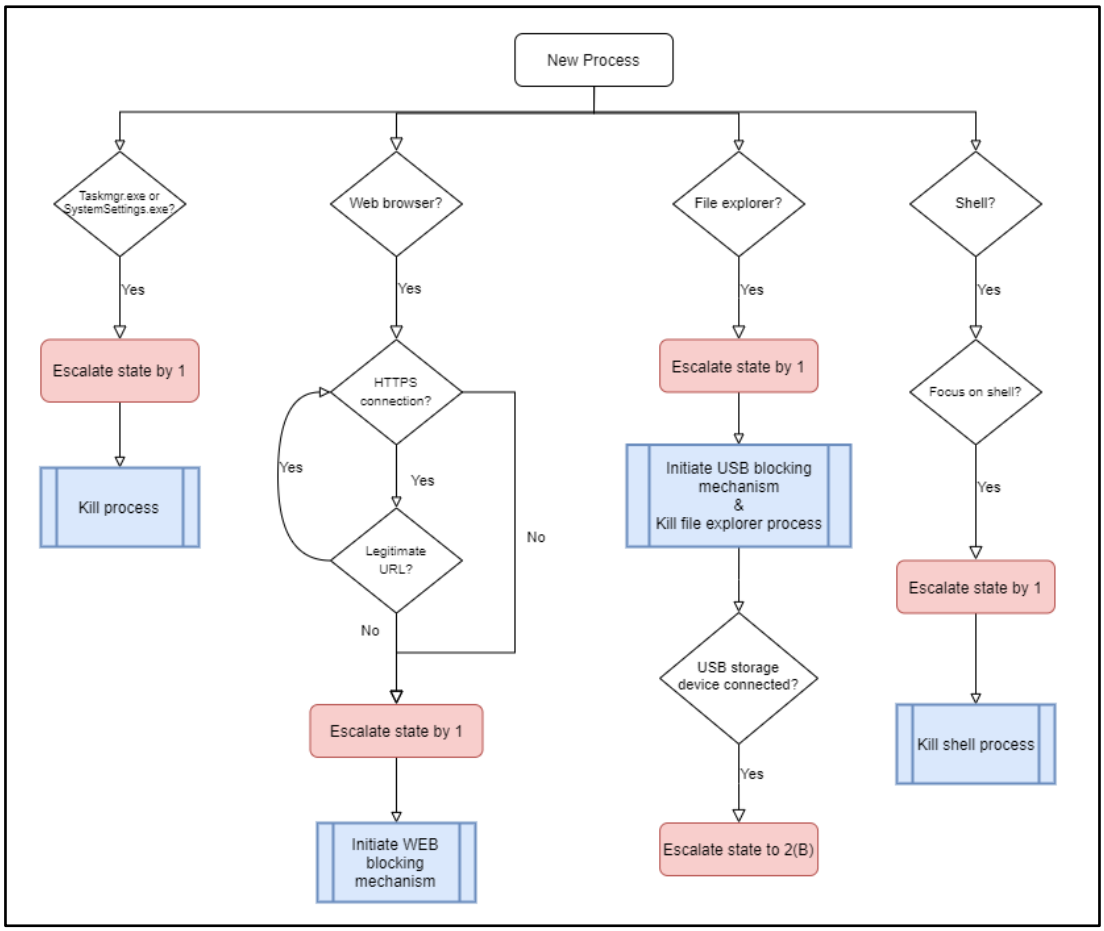


איור 8 - תרשים זרימה הכולל את תגובת המערכת לאירוע מסוג New Bluetooth Connection

3. ריצת תהליך חדש (ראו איור 9) - התחלה של ריצה של תהליכים חדשים במערכת היא אירוע שקורה כל הזמן, גם במצב מסך נעול. לכן, עלינו לנטר רק על שני סוגים של תהליכים:

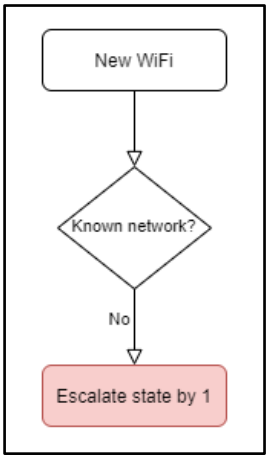
א. תהליכים שלא אמורים להיפתח בזמן מסך נעול ולכן פתיחתם היא איום ועלינו להרוג אותם מיד - taskmgr.exe, SystemSettings.exe, File Explorer ו-File Picker UI.

ב. תהליכים שיכולים להיפתח בזמן מסך נעול מסיבות לגיטימיות, אך יכולים גם לשמש תוקף לפעולות זדוניות, ולכן עלינו לבחון את אופן השימוש בהם, ועל פיו להחליט האם להרוג אותם או לא. דוגמה לתהליכים שכאלה הם דפדפני אינטרנט. כאשר מדובר בדפדפן אינטרנט, שיכול להיפתח על ידי אפשרויות מובנות במסך הנעילה, נרצה לבדוק מה השימוש שנעשה בו - האם ההתחברות נעשית לאתרים לגיטימיים והאם היא נעשית ב-HTTPS, ורק אם היא חורגת מכללים אלו נהרוג את התהליך. גם תהליך Shell, על אף פוטנציאל הנזק הגדול שלו, יכול להיפתח באופן לגיטימי בזמן מסך נעילה, לדוגמה על ידי Scheduled Tasks. אולם, אם ה-Focus עבר אל תהליך ה-Shell החדש, כלומר הקלט מהמשתמש מגיע ל-Shell, מדובר באפשרות לא לגיטימית שלא נרצה לאפשר, ולכן נהרוג את התהליך.



איור 9 - תרשים זרימה הכולל את תגובת המערכת לאירוע מסוג New Process

4. חיבור Wi-Fi חדש (ראו איור 10) - תוקף יכול לנסות לבצע מגוון התקפות על ידי התחברות לרשת Wi-Fi אחרת (אפשרות שקיימת במסך הנעילה של Windows 10). לכן, אם נעשה חיבור לרשת Wi-Fi חדשה בזמן מסך נעול, והיא לא רשת שהמכשיר התחבר אליה בעבר, על המערכת לדרדר את מצב המערכת ב-1.



איור 10- תרשים זרימה הכולל את תגובת המערכת לאירוע מסוג New Wi-Fi

מימוש המערכת

על מנת לממש את המערכת לפי הדרישות הנ"ל היינו זקוקים למידע ממערכת ההפעלה: מתי קורים אירועים שהגדרנו כדורשים ניטור, מהם החיבורים שקרו בעבר וכמו כן רצינו לבצע פעולות אקטיביות כאשר אירועים חשודים מתרחשים. להלן פירוט המחקר שביצענו על מנת להצליח במימוש המערכת.

ניטור אירועים ב-Windows 10

חלק ממערכות ההפעלה מכילות מנגנוני Instrumentation[9] שנועדו להקל על מפתחי אפליקציות לזהות כאשר קורים אירועים מסוימים ולהגיב אליהם. ב-Windows 10 קיימת תשתית נרחבת של דיווח על Event-ים, המאפשרת למפתח לקבל דיווח דרך מגוון ממשקים (API) שונים. היה עלינו לבחור את הממשק המתאים ביותר עבורנו כדי לקבל מידע על אירועים שקורים במערכת ההפעלה. הדרישות עבור הממשק הנבחר היו: פתרון מהיר ובזמן אמת, שיאפשר לנטר על כל ה-event-ים שהגדרנו כנדרשים, וכן פשוט על מנת לאפשר עמידה בלוחות הזמנים של הפרויקט.

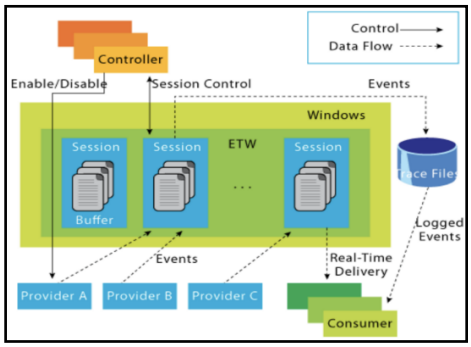
לאחר סקירת האפשרויות, התמקדנו בשני ממשקים מובילים: Event Tracing for Windows (ETW) ו- Windows Event Log. לבסוף החלטנו להמשיך עם Windows Event Log, בעיקר לאור היותו פתרון פשוט משמעותית לעומת ה-ETW, אשר לו הרבה יכולות מקיפות שלא נדרשו עבור פרויקט זה. להלן סקירה של יכולות הממשקים שבחנו:

א. ETW – Event Tracing for Windows

ETW[10] הינה תשתית ניטור האירועים הנרחבת ביותר של מיקרוסופט עבור Windows, ותפקידה לאפשר ניטור בזמן האמת על מגוון אירועים ברמת גרעין מערכת ההפעלה, או אפליקציות שרצות על מערכת ההפעלה ותומכות ב-ETW. את האירועים ניתן לשמור לקובץ log לקריאה אוחרת על ידי אפליקציות אחרות.

ה-API מחולק לשלושה רכיבים עיקריים:

- **Providers** – אפליקציות המספקות דיווח על אירועים (events) בעת התרחשותם.
- **Controllers** – אפליקציות השולטות בהתחלת/הפסקת הניטור, ומאפשרות את ריצת ה-Providers.
- **Consumers** – אפליקציות הצורכות את האירועים בעת התרחשותם.



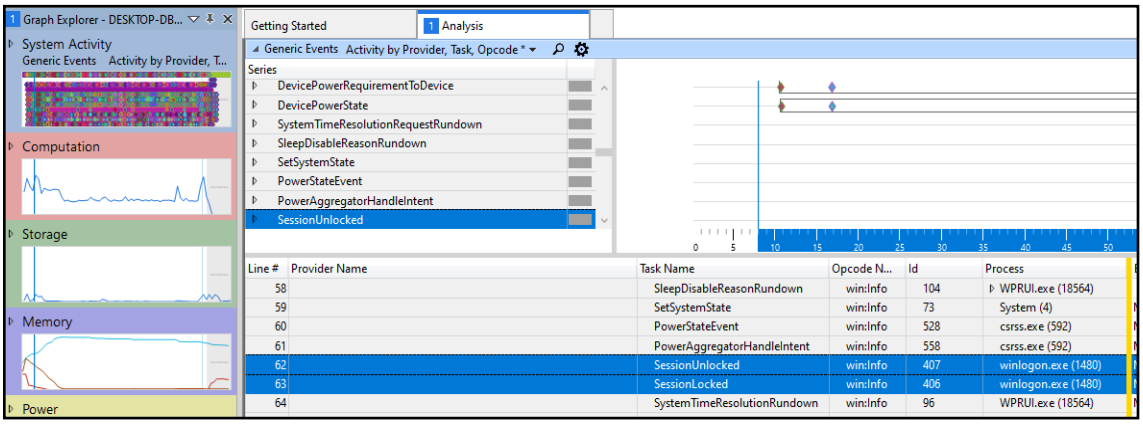
איור 11 - תשתית ה-ETW, מתוך התיעוד של מייקרוסופט[11]

[9] כלי מדידה שונים שנועדו לאסוף מידע על מצב המערכת.

[10] https://docs.microsoft.com/en-us/windows/win32/etw/about-event-tracing

[11] https://web.archive.org/web/20200725154736/https://docs.microsoft.com/en-us/archive/blogs/ntdebugging/part-1-etw-introduction-and-overview

קיימים מספר כלים המאפשרים לעבוד עם ה-API של ETW. כלי לדוגמא הינו Logman, כלי שמגיע עם Windows, ומאפשר לבצע פעולות כ-controller (לדוגמא – לייצר או לעצור event trace session, לאפשר provider) באמצעות חלון ה-Command line. כלים נוספים ושימושיים המאפשרים להתממשק עם אירועים של תשתית ה-ETW הינם כלי ה-Windows Performance Toolkit[12]: Windows Performance Recorder (WPR) ו-Windows Performance Analyzer (WPA), באשר WPR מאפשר הקלטה של אירועים בזמן אמת לפי Provider-ים נבחרים ושמירה של התוצאות לקובץ לוג (.etl), ואילו WPA מאפשר תצוגה של התוצאות באופן ויזואלי המקל על מחקר והסקת מסקנות.



איור 12 – הקלטה שניתחנו באמצעות WPA, המכילה אירוע Lock

ב. Windows Event Log

Windows Event Log[13] הינה התשתית העדכנית לניטור ו-logging שמציעה Microsoft עבור Windows. היא עושה שימוש באירועים המנוטרים גם על ידי תשתית ה-ETW, אך מוכוונת בעיקר לכתיבת Log-ים עבור מערכת ההפעלה ואפליקציות נוספות. היא מאפשרת ניטור על אירועי System וכן אפליקציות נוספות. כאשר אירוע מתרחש, הוא מוכנס ל-log או ה-Channel המתאים עבורו לפי הגדרות המערכת. המושגים של Provider ו-Consumer משמשים באופן דומה להגדרת ספקים מול צרכנים.

כאמור, תשתית זו מאפשרת לאפליקציה המספקת אירועים (Provider) לפרסם אותם באמצעות שני מקורות: הראשון הינו event log channel (ערוץ) והשני הינו event tracing log file (קובץ), וכך ה-Consumer יכול לבחור כיצד לצרוך את האירועים לאחר מכן. קיימים 4 סוגים של Channel-ים: Admin, Operational, Analytic ו-Debug. כל אחד מסוגי הערוצים מכיל אירועים מסוג אחר[14]. מעתה נתייחס לכתיבה ל-log ככתיבה ל-Channel, מאחר והאירועים הנכתבים והשמות משמשים באופן זהה.

ה-Channel-ים/Log-ים מחולקים לשתי קבוצות:

---

[12] https://docs.microsoft.com/en-us/windows-hardware/test/wpt/

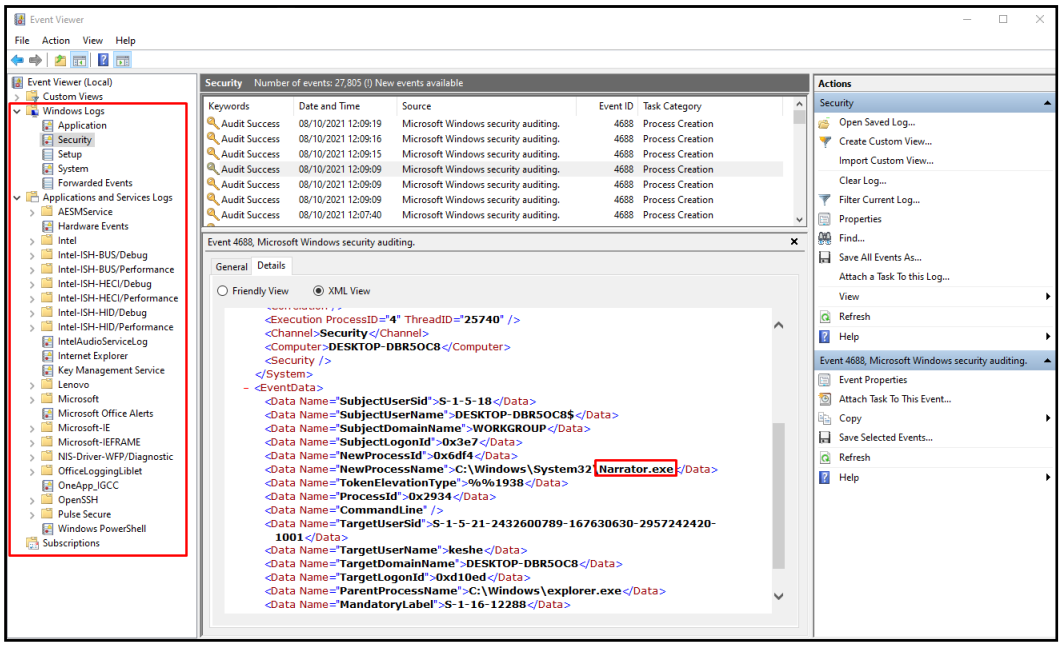
[13] https://docs.microsoft.com/en-us/windows/win32/wes/windows-event-log

[14] https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/cc722404(v=ws.10)

Windows Logs – חמישה לוגים המשמשים בעיקר לאירועי System של מערכת ההפעלה: Security, Setup, System, Forwarded Events Application,.

Application and Services Logs – לוגים עבור אפליקציות או רכיבים המפותחים עבור מערכת ההפעלה.

האירועים נשמרים כיום בפורמט בינארי evtx. אשר ניתן להמרה לקובץ xml באמצעות Windows Event Log API, או לקריאה באמצעות הכלי Windows Event Viewer (eventvwr.msc - MMC Snap-in).



איור 13- חלון Event Viewer הכולל אירוע יצירת Process מסוג "Narrator.exe" בחלק מה-Security Log

ה-Windows Event Log מאפשרת לצרוך אירועים באמצעות ביצוע שאילתה ברגע נתון, או באמצעות "הרשמה" ל-event מסוים. כאשר מבצעים הרשמה ניתן לבחור את סוג האירוע עליו יש לנטר, את המקור (Channel או קובץ) וכן האם נדרש ניטור כולל אירועי עבר, או רק עבור אירועים עתידיים.

קונפיגורציית ההקלטה והדיווח על האירועים

כדי שנוכל לנטר על האירועים שמעניינים אותנו, ראשית יש לאפשר אותם.

התשתית שמציעה Microsoft על מנת לנהל את הניטור על אירועים מתייחסת לשני סוגי מדיניות לניטור: הראשון הינו Basic Audit Policy, והשני הינו Windows Advanced Security Auditing Policy. קיימים ביניהם הבדלים רבים[15], הנוגעים בעיקר לעבודה מול מחשבים ברשת עם Domain מנוהל, אך לא נדון בהבדלים השונים כאן.

בנוסף, קיימות גרסאות שונות (editions) של מערכת ההפעלה Windows 10. אמנם תשתית הדיווח על האירועים דומה בכולן, אך בגרסאות שלא יכולות להצטרף ל-Domain מנוהל, לדוגמא Windows 10 Home edition, לא קיימת גישה לכלים המאפשרים לערוך Group Policy, ובפרט Audit Policy, (לדוגמא secpol.msc המאפשר עריכה של ה-

[15] https://docs.microsoft.com/en-us/windows/security/threat-protection/auditing/advanced-security-auditing-faq

Local Basic Policy, או gpedit.msc). בחנו מספר פתרונות להתמודדות עם הבעיה, ולבסוף בחרנו להשתמש בכלי Auditpol.exe[16], כלי Command line המאפשר לערוך, לצפות לשמור ולהגדיר את ה-Audit Policies במערכת ההפעלה. לדוגמא, השורה הבאה אפשרה לנו להפעיל ניטור על event מסוג Process Creation:

"auditpol /set /subcategory:"Process Creation" /success:enable /failure:enable"

מימוש ההרשמה לאירועים

כאמור, Windows Event Log מאפשרת לתשאל ולהירשם לדיווח בעת התרחשות אירוע מ-channel או מקובץ log. קובץ ה-winevt.h header מכיל מגוון פונקציות ומבני נתונים המאפשרים למפתח אפליקציית Consumer לממש את ההרשמה לאירוע באמצעות קוד.

כדי לצרוך אירוע בעל מאפיינים ספציפיים (לדוגמא: יצירה של תהליך בעל שם מסוים) יש לייצר XML Query שישמש כפילטר מוגדר מראש. כדי לבצע פעולת הרשמה ל-event, יש לקרוא לפונקציה EvtSubscribe.

```
#define QUERY_LOCKED \
    L"<QueryList>" \
    L"  <Query Path='Security'>" \
    L"    <Select>Event[System[EventID=4800]]</Select>" \
    L"  </Query>" \
    L"</QueryList>"

hLockSubscription = EvtSubscribe(NULL, NULL, NULL, QUERY_LOCKED, NULL, NULL,
(EVT_SUBSCRIBE_CALLBACK)LockCallback, EvtSubscribeToFutureEvents);
```

איור 14 - דוגמת קוד: שאילתת XML המפלטרת לפי Lock event ID, לאחר מכן קריאה לפונקציה EvtSubscribe()

הפונקציה EvtSubscribe() מציעה שני מודלים לביצוע Subscribe:

א. Push Subscription

הרשמה שמאפשרת הרצה של פונקציית Callback ברגע התרחשות האירוע. הפונקציה נשלחת למערכת ההפעלה בעת ההרשמה ל-Event ומורצת באופן אוטומטי, מיד לאחר ההתרחשות.

א. Pull Subscription

הרשמה שלאחריה מורצת לולאה הבודקת באופן אקטיבי האם קיימים event-ים נוספים אשר מתאימים ל-Query, כלומר לפילטר שהוגדר.

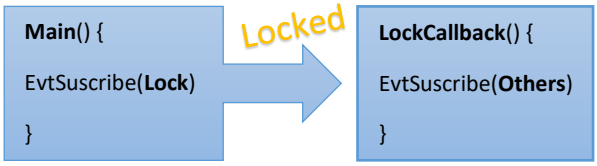
לכל אחד מהמודלים יתרונות וחסרונות. דוגמא ליתרון בולט של ה-Pull Model הינו עניין הסנכרון – כאשר מדובר במערכות גדולות שאוספות אירועים ממקורות רבים, קל יותר לבצע זאת בתצורה מנוהלת אקטיבית. היתרון הבולט של ה-Push Model בא לידי ביטוי בזמן התגובה – פונקציית ה-callback רצה מיידית כאשר מערכת ההפעלה מדווח שהתרחש האירוע. במערכת הנוגעת לאבטחה פיזית של מחשב מפני תקיפה, יש חשיבות קריטית לזמני התגובה

[16] https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/auditpol

שימנעו ביצוע פעולות נוספות בהתקפה, בנוסף קנה המידה של המערכת הינו מצומצם כיום – איסוף אירועים בודדים מוגדרים מראש ממחשב יחיד. לפיכך בחרנו ב-Push Subscription.

מכיוון שמיידיות התגובה הינה בתיעדוף עליון, לאחר שכתבנו את הקוד, וידאנו שהתגובה אכן מתבצעת באופן מידי בעת התרחשות האירוע על ידי הדפסה של הזמנים לקובץ ה-log של המערכת שלנו, המודפס בעת הריצה.

על מנת לחסוך פעולות ותגובות מיותרות ולהגביר את יעילות זמן הריצה של המערכת, בחרנו לבצע הרשמה ל-event-ים רק בעת ביצוע פעולה של Screen Lock, והסרה של ההרשמה בעת ביצוע Screen Unlock תקין.

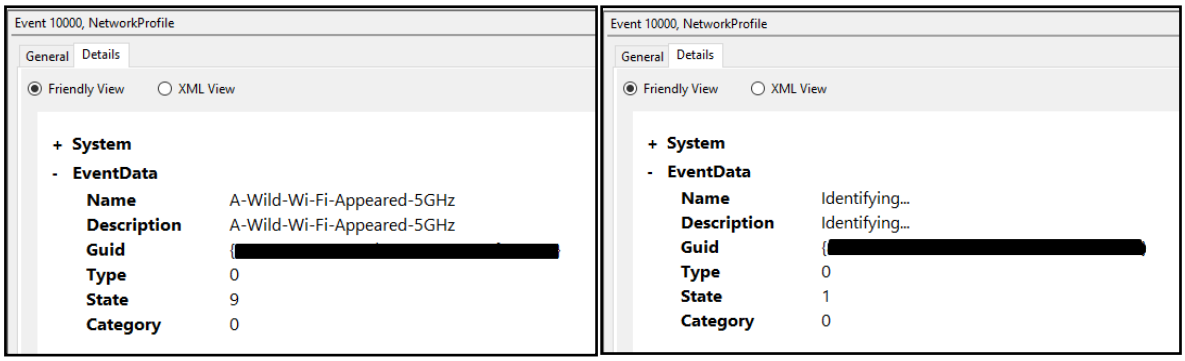


איור 15 - תהליך ההרשמה לאירועים במערכת

ההיגיון בבחירה זו הינו שאם פעולה חשודה מתבצעת על ידי משתמש **בעל יכולת לבצע Unlock באופן חוקי**, זוהי התקפה שאינה מוגדרת כהתקפת Evil Maid, אלא דורשת כלי ניטור מתקדמים ולכן לא מנוטרת על ידי המערכת.

כתיבת שאילתה נכונה לסינון האירועים דרשה מאיתנו הבנה ב-Syntax של Xml Query, וכן הבנה של תוכן כל אחד מה-Event-ים עליהם רצינו לפלטר. לכל אחת מהפעולות שרצינו לנטר היו מספר event-ים שיכלו להסגיר את התרחשותה. לדוגמא, עבור נעילת המסך יכולנו להשתמש ישירות ב-event "The workstation was locked" בעל id 4800, או לנטר את פעילות התהליכים LogonUI.exe/Lockup.exe. לאחר שמצאנו אירוע מועמד המתאים לניטור, היה עלינו לחוק מתי בדיוק הוא מתרחש, ולוודא שהוא מהווה אינדיקציה יחידה רק לפעולה שרצינו לנטר, ולא לפעולות אחרות. לדוגמא, עבור חיבורי USB חדשים רצינו לנטר על האירוע Removeable Storage (id - 4663) שמופיע כחלק מה-Security log, אך לאחר מחקר נוסף גילינו שהוא מתרחש רק בעת הגישה לקבצים השמורים בתקן ולא בעת החיבור עצמו, לפיכך בחרנו לנטר על טעינת ה-Driver-ים שמתרחשת מיד לאחר חיבור.
בנוסף, היה עלינו להבין לאיזה channel עלינו להירשם על מנת לקבל את האירוע, אירועים רבים מדווחים במספר Channel-ים שונים ברמות מידע שונות.
לדוגמא, עבור חיבורי רשתות Wi-Fi, היו מספר ערוצים שהכילו הודעות על חיבורי רשתות אלחוטיות או חוטיות חדשות, מבין האפשרויות בחרנו להירשם לערוץ "Microsoft-Windows-NetworkProfile/Operational"' (חלק מה-Application and Services Logs), אשר הכיל את ה-Event שהתרחש צמוד לניסיון החיבור לרשת החדשה:

```
#define QUERY_CONNECTED_WIFI \
    L"<QueryList>" \
    L"  <Query Path='Microsoft-Windows-NetworkProfile/Operational'>" \
    L"    <Select>Event[System[EventID=10000] and
EventData[Data[@Name='Name']!='Identifying...']]</Select>" \
    L"  </Query>" \
    L"</QueryList>"
```



איור 16 - התאמה של השאילתה Wi-Fi Connection כך שתפלטר החוצה הודעות לא רלוונטיות

כאשר חקרנו את ה-event על ידי סימולציות חיבור לרשת Wi-Fi חדשה, שמנו לב שלא כל ההודעות מסוג 10000 נוגעות לחיבור רשת חדשה, לדוגמא הודעה חוזרת שבמקום שם הרשת מופיע "Identifying...". השתמשנו ב- XML כדי לסנן הודעות כאלו ולא להתייחס אליהן.

בנספח ב' מצורפת טבלה המכילה כל ה-Event-ים עליהם בחרנו לנטר, כולל תיאור מפורט מתי הם מתרחשים.

אחזור מידע השמור במערכת ההפעלה

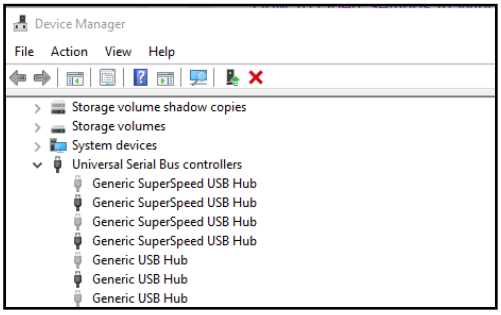
לאחר שההרשמה ל-Event-ים הושלמה, נותר לממש את פונקציות ה-callback בהתאם לכל event.

בחלק מהתגובה לאירוע מסוים, לדוגמא חיבור USB חדש, רצינו לבקש מידע השמור המערכת ההפעלה, לדוגמא לבקש את היסטוריית חיבורי ה-USB, כדי לחפש בה את המזהה של ההתקן שחובר ולבדוק האם מדובר בהתקן מוכר. כך נוכל להעריך את הסיכוי שאנחנו נמצאים בתרחיש של מתקפת אמת.

לצורך כך, היה עלינו לוודא שהקוד שלנו רץ בהרשאות מתאימות. לשם כך:

1. תמיד ביצענו הרצה כ-Administrator.
2. היה עלינו לממש פונקציה אשר בזמן ריצה מאפשרת לבצע פעולות בהרשאות גבוהות[17].

כדי למצוא מקום המכיל את היסטוריית חיבורי התקני אחסון ב-USB חקרנו את אופן החיבור של התקני אחסון למחשב באמצעות DeviceManager (devmgmt.msc) ו-Event Viewer. גילינו שקיימת אפשרות להציג את היסטוריית החיבורים ב-Device Manager על ידי בחירת האפשרות "Show hidden devices" בתפריט view:

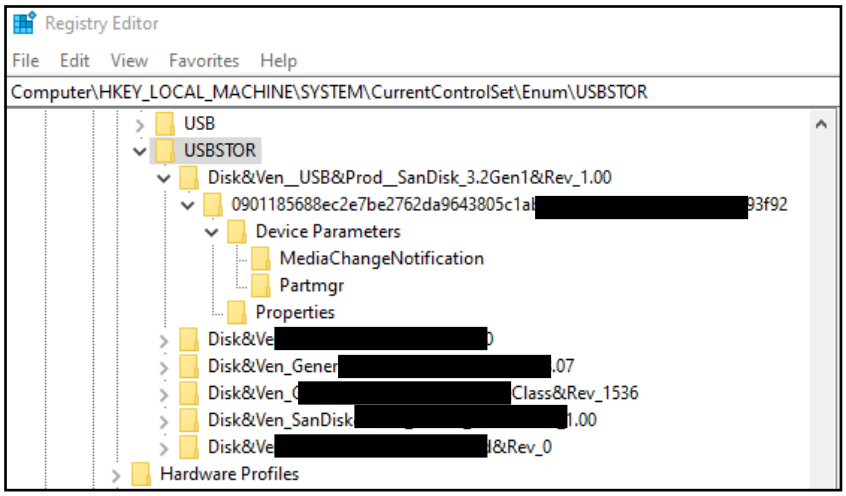


איור 17 - הצגת Hidden Devices תחת USB Controllers, באפור מסומנים התקנים שלא מחוברים כעת

[17] לפירוט נוסף ראו CppPrivilileges.h בקוד הפרויקט.

לאחר מחקר נוסף הבנו שנוכל למצוא את המקור ממנו האפליקציה Device Manager קוראת את המידע ב-Registry. חיפשנו אילו ערכים ב- Registry משתנים בעת חיבור פיזי של התקן, ומצאנו מספר מקומות מתאימים, לדוגמא HKEY_LOCAL_MACHINESYSTEMCurrentControlSetEnumUSBSTOR, וכן המיקום שנבחר לבסוף SYSTEM\CurrentControlSet\Enum\USBSTOR. לא מצאנו תיעוד רב בנושא ערכי ה-Registry שמצאנו, ומפאת קוצר הזמן לאחר בחינה קצרה בחרנו להתקדם עם האפשרות השנייה המצוינת לעיל. נדגיש כי אמנם נראה שהערך שבחרנו עונה לדרישה בזמן הקצר של ביצוע הפרויקט, אך נדרש מחקר נוסף על מנת לאפיין בדיוק מה גורם למחיקה של התקן מהרשימה, ובאיזו תדירות מתרחשת מחיקה שכזו. ייתכן שמחקר המשך מעמיק יותר על ערכי Registry יניב ערכים מתאימים יותר.

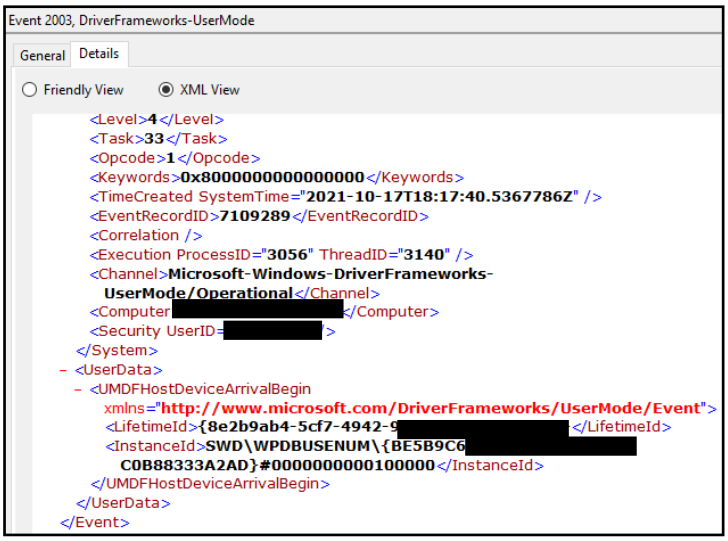
על מנת לקרוא מפתחות וערכים מה-Registry עשינו שימוש בספרייה winreg.h המכילה מגוון פונקציות לקריאה, כתיבה ועריכה של ה-Registry. מהסתכלות ב-Key הרלוונטי ב-Registry, הבנו כי עלינו לבצע חיפוש רקורסיבי בעץ הערכים מכיוון שכל הנתונים אודות ה-device נמצאו תחת מפתח המכיל את שמו:



איור 18 - מבנה הערך המכיל את היסטוריית חיבורי USB ב-Registry

לפיכך בחרנו לממש את החיפוש בהיסטוריה באמצעות הפונקציה RegSaveKey(), אשר שומרת את המפתח הנבחר וכל תתי המפתחות תחתיו בקובץ קבוע מראש, כך שנוכל לחפש את המזהה שנבחר בתוכן הקובץ, וכך לזהות האם המכשיר שחובר הינו מוכר מהעבר או לא.

לאחר שהשווינו בין המידע החוזר מה-Event המודיע על חיבור Device, ולבין המידע השמור בהיסטוריה ב-Registry, בחרנו לבצע את הבדיקה באמצעות מזהה ה-Instance ID, שכן להבנתנו מדובר במזהה ייחודי ל-Device (שמות נקבעים לפי יצרנים ועלולים להיות זהים).



איור 19 - המידע החוזר ב-Event המתריע על Usb Device שחובר למחשב

על מנת להצליח לזהות חיבור של התקן לא מוכר בעת מסך נעול באופן עדכני בכל נעילה, החלטנו לשמור קובץ עדכני של היסטוריית חיבורי ההתקנים הניידים בכל נעילה מחדש, כך נדע בוודאות האם מדובר בהתקן שהיה מחובר גם לפני הנעילה, או שמדובר בהתקן חדש ולא מוכר, אשר עשוי להצביע על מתקפה.

בנוסף, ניתן לבצע בדיקה זהה עבור רשתות Wi-Fi לא מוכרות באמצעות שימוש בערכי Registry מתאימים, לדוגמא HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\NetworkList\Profiles.

תגובה אקטיבית

במהלך המחקר מימשנו גם תגובות אקטיביות להתקפות. כאשר מצב המערכת עולה ל-1, המערכת שולחת התרעה למשתמש. אנחנו בחרנו לממש את ההתרעה הזו באימייל שישלח לחשבון המשתמש. המערכת משתמשת בפונקציה WinExec, המאפשרת הרצת פקודות מתוך קוד C++, כדי לשלוח פקודה באמצעות הכלי curl, המאפשר שליחת מידע על גבי הרשת במגוון פרוטוקולים, בין היתר שליחת הודעות אימייל בפרוטוקול SMTP, כפי שבחרנו לבצע במימוש שלנו. שתי פעולות אקטיביות נוספת שמימשנו הן כיבוי המחשב והפעלת אזעקה עבור מקרים קיצוניים. כיבוי המחשב נעשה על ידי העלאת ההרשאות עבור התהליך של המערכת, כך שיוכל לכבות את המחשב, ואז כיבוי באמצעות ExitWindowsEx(), פונקציית מערכת שמסופקת על ידי Windows. הפעלת האזעקה נעשית באמצעות הפונקציה mciSendStringA() שמאפשרת שליחת פקודות להתקני מולטימדיה.

בנוסף, ניסינו לממש חסימת USB-ים אקטיבית כאמצעי למניעת המשך התקפה. ממחקר שביצענו בנושא גילינו שמרבית הממשקים המוצעים למפתחים/אנשי אבטחה שרוצים לבצע חסימה שכזו מאפשרים חסימה כללית של חיבורי USB עתידיים למכשיר, ולעיתים גם דורשת Restart, ולא קיים פתרון פשוט לחסימה נקודתית לפי White list. לדוגמא, ניתן לחסום חיבורים באמצעות Group Policy בגרסאות Windows 10 מנוהלות[18], אך רצינו למצוא פתרון שיתאים גם לגרסת Windows 10 Home. מצאנו כי ניתן להוסיף ערך Registry מסוים[19] ולהוסיף לו פרמטר

[18] http://woshub.com/how-to-disable-usb-drives-using-group-policy/
[19] Computer\HKEY_CURRENT_USER\SOFTWARE\Policies\Microsoft\Windows\RemovableStorageDevices

Deny_All עם הערך 1. הוספת ערך זה תחסום את הגישה להתקני האחסון שיחוברו למחשב באמצעות USB, והשינוי אף יכנס לתוקף מבלי לאתחל את המחשב בניגוד לערכי Registry רבים אחרים. עם זאת, במהלך ניסיונותינו ראינו כי החסימה לא מתבצעת באופן מידי, ויתכן כי תעבור למעלה מדקה מרגע השינוי עד לרגע החסימה, זמן מספיק להתקפה. פתרון נוסף שעלה כחלק מהמחקר הינו שימוש בספרייה cfgmgr32.h המכילה פונקציות לשליטה ב-Driver-ים שונים, לדוגמא הפונקציה CM_Request_Device_EjectA() המאפשרת ניתוק של Device לפי מזהה ספציפי. אנו מאמינים כי ניתן לבצע לולאה שתבצע Eject לכל התקני האחסון במחשב עד שהשינוי יכנס לתוקף, כאשר נבצע סינון לפי רשימה שמורה מראש של התקנים המותרים לחיבור בכדי לא להעיק על המשתמש עם ניתוקים מיותרים של מכשירים בעלי חיבור USB. אנו משאירים זאת כביוון מחקרי להמשך.

הוכחת יכולת - הצלחה בזיהוי התקפה חדשה

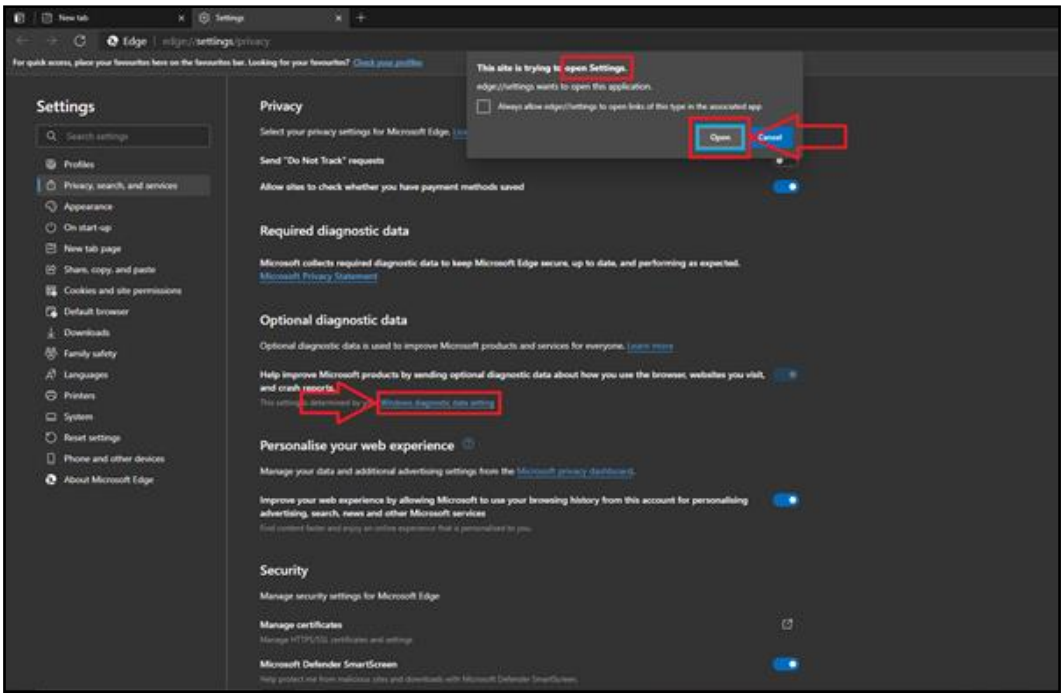
בשלהי המחקר ולאחר שלב המחקר התיאורטי והגדרת המערכת, פורסמה התקפה חדשה[20] בעקבות התקפת Lock Screen Bypass, אשר עליה הרחבנו במחקר זה. הייתה זו הזדמנות עבורנו לבחון את הגדרת המערכת אל מול איום חדש, שלא היה מוכר בזמן תכנונה. נתאר כעת בקצרה את החולשה החדשה ונראה כי המערכת התיאורטית שלנו הייתה עוצרת את ההתקפה לפני מימושה המלא, ולמעשה לפני שנגרם נזק כלשהו.

ההתקפה נמצאה בהשראת התקפת Lock Screen Bypass, וקיימים קווי דמיון רבים בין שתי ההתקפות. בתחילת ההתקפה התוקף מפעיל את ה-Narrator, ומתחיל לעבור בין סדרה ידועה מראש של קישורים בתוך המסך הנעול, עד שהוא מגיע לחלון "Sign in with a security key", שבו יש קישור להסבר על אפשרות זאת.

איור 20- קישור עם חולשה בחלון Sign in המאפשר פתיחת חלון דפדפן ברקע המסך הנעול

התוקף פותח את הקישור באמצעות דפדפן האינטרנט Edge. ברקע נפתח הדפדפן וה-Focus מועבר אליו. התוקף מנווט בהגדרות הדפדפן באמצעות ה-Narrator, עד שמגיע להגדרה בשם "Windows Diagnostic data setting", אשר פתיחה שלה פותחת את ההגדרות של Windows 10.

[20] ZDI-21-1053 - https://halove23.blogspot.com/2021/09/zdi-21-1053-bypassing-windows-lock.html?m=1



איור 21 - לינק המאפשר פתיחת חלון Settings דרך אפליקציית edge

המטרה של התוקף היא להגיע להגדרת ה-AutoPlay ולחבר USB, אך ברגע שנפתח התהליך SystemSettings.exe, המערכת תהרוג אותו ותתריע למשתמש, ובכך תמנע את ההתקפה.

נציין כי כשהתוקף פותח את הדפדפן הוא יכול לחלופין לנסות לגשת לאתר זדוני, שהכין מראש. אולם אם ינסה לעשות כן, המערכת תזהה שהוא ניגש ל-URL לא לגיטימי, תהרוג את דפדפן האינטרנט, שכן הוא נפתח לאחר נעילת המסך וניגש ל-URL בעייתי, ובכך תמנע את ההתקפה ותתריע למשתמש.

אתגרים משמעותיים

במהלך המחקר התמודדנו עם מספר אתגרים. הראשון הוא גישת ההגנה הייחודית שנדרשה מאיתנו. בעוד שאנטי-וירוסים, תוכנות ההגנה הנפוצות ביותר כיום, מנסות לזהות תוכנות ותהליכים זדוניים, על המערכת שלנו לזהות התנהגות אנושית זדונית, ולדעת להבדיל אותה מהתנהגות אנושית שגרתית. אתגר נוסף שהשתלב לתוך אתגר זה, הוא הצורך באיזון בין הגנה על המחשב לבין הרצון לאפשר פעולות לגיטימיות במסך נעול - אחת הטעויות המרכזיות של ניסיון העבר להתמודדות עם הבעיה, Do Not Disturb, הייתה שהוא יצר המון התרעות שווא למשתמש, ולמעשה המשתמש נתקל בה על בסיס קבוע. אנחנו רצינו שהמשתמש לא יצטרך לראות את המערכת שלנו בשימושו היומיומי, אלא רק בעת התקפה.

אתגר נוסף שהתמודדנו אתו היה שחזור החולשות, שכן החולשות שמפורסמות, נחסמו כבר על ידי Microsoft. החולשה הראשונה שניסינו לשחזר הייתה Open Sesame שהוזכרה בפרק הרקע. היה עלינו למצוא Image ישן של Windows 10, שבו החולשה עדיין לא תוקנה, ולדאוג שלא יבוצעו עדכונים למערכת ההפעלה. מספר ימים לאחר שהצלחנו לשחזר את החולשה, הופסקה התמיכה ב-Cortana עבור גרסת ה-Windows 10 שהשתמשנו בה, והמשמעות המעשית היא שהחולשה נחסמה לחלוטין. ניסינו לשחזר גם את Lock Screen Bypass, אך מכיוון

שעבדנו עם Virtual Machine, שזמני התגובה שלה איטיים משל מכונה רגילה, היה קשה לשחזר את החולשה, שדורשת תיאום זמנים מדויק באחד השלבים. עבור שיחזור חולשה זו אפילו יצרנו קשר עם מי שפרסם את החולשה, אך לא עלה בידו לסייע לנו.

כיווני מחקר נוספים

לאורך המחקר עלו כיווני מחקר נוספים שניתן להתמקד בהם כחלק ממחקר המשך:

1. דרכים נוספות למניעה אקטיבית של התקפות Evil Maid – לדוגמא חסימת חיבורי USB, Wi-Fi ו-Bluetooth והפעלת המצלמה במחשב על מנת להקליט את המאורע.
2. זיהוי שינויים בהגדרות בזמן מצב נעול, לדוגמה שינוי הגדרת ה-AutoPlay במכשיר.
3. זיהוי חיפושים חשודים ב-Cortana ובדפדפנים במסך נעול ובכלל זה גלישה שלא ב-HTTPS.
4. זיהוי שינויים ב-Focus של מערכת ההפעלה וזיהוי כש-Focus עובר לתהליך חשוד, לדוגמה תחילת ריצה של SystemSettings.exe, כאשר המסך נעול.
5. שילוב של זיהוי פנים כחלק ממנגנון הזיהוי והניטור של המערכת.

סיכום

מחקר התקפות Evil Maid הוא תחום פעיל שממשיך להתפתח כל הזמן. על אף שחוקרים מוצאים חולשות חדשות ומפתחים התקפות, אין כיום מענה הגנתי הולם להתקפות מסוג זה, במיוחד עבור משתמשים פרטיים.

במהלך המחקר בחנו התקפות מוכרות בתחום ה-Evil Maid, זיהינו את נקודות הדמיון בין ההתקפות השונות והצענו מערכת תיאורטית להתמודדות עם הבעיה, אשר מבצעת שני תפקידים - ניטור והתמודדות עם האיום. מחקר ההתקפות המוכרות סייע לנו לזהות אירועים המעידים על התקפות אלו ולהגדיר את המערכת כך שתדע להתמודד איתן, אך גם לזהות אירועים שיכולים להעיד על התקפות לא מוכרות ולחזק את הגדרת המערכת, כך שתדע להתמודד גם עם איומים לא מוכרים. לקראת סוף המחקר ולאחר שלב הגדרת המערכת התיאורטית, פורסמה התקפת Lock Screen Bypass חדשה, ולשמחתנו כאשר בדקנו את יכולות המערכת שלנו כהגנה אל מול ההתקפה החדשה, גילינו שהצלחנו ליצור מערכת תיאורטית שיודעת לזהות ולעצור את ההתקפה לפני שמתרחש נזק ממשי לנתקף.

במקביל למחקר התאורטי, ביצענו מחקר פרקטי על מנת למצוא כלים שמאפשרים לממש את המערכת התיאורטית שהצענו. מימשנו גרסת PoC ראשונית למערכת, המיישמת תכונות מרכזיות של המערכת התיאורטית. המערכת מאפשרת לנטר על אירועים חשודים כמו חיבור התקן אחסון לא מוכר, תחילת ריצה של תהליכים חשודים וחיבורים לרשתות Wi-Fi. בנוסף, המערכת מדווחת למשתמש על רצף אירועים החשוד כתקיפה ומבצעת פעולות למניעתה.

ביבליוגרפיה

[1] https://www.cs.technion.ac.il/users/wwwb/cgi-bin/tr-get.cgi/2020/MSC/MSC-2020-29.pdf

[2] https://www.mathyvanhoef.com/2017/02/windows-10-lock-screen-abusing-network.html

[3] https://secret.club/2021/01/15/bitlocker-bypass.html

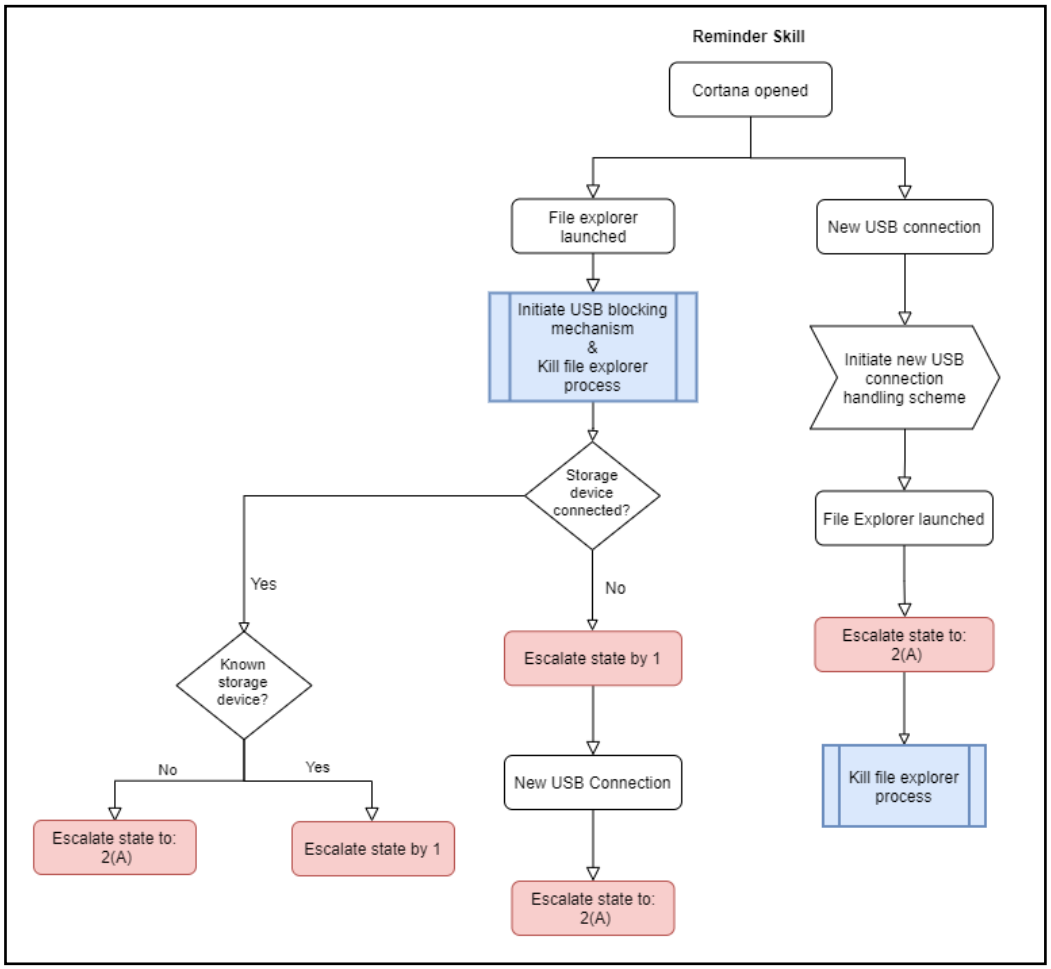
[4] https://docs.microsoft.com/en-us/windows/win32/etw/about-event-tracing

[5] https://docs.microsoft.com/en-us/windows/win32/wes/windows-event-log

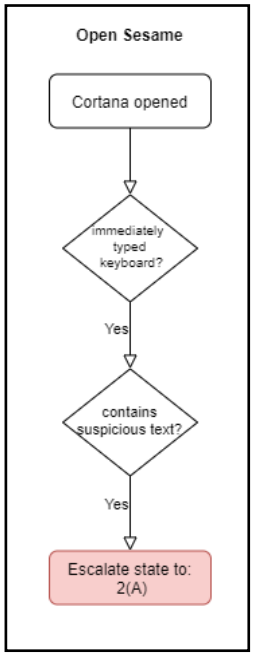
[6] https://www.ultimatewindowssecurity.com/securitylog/encyclopedia

נספח א' - תרשימי ניטור והגנה מפני התקפות מוכרות נוספות

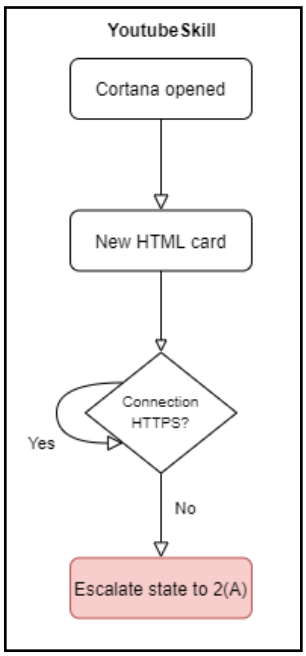
להלן תרשימים נוספים של התקפות מוכרות, אשר מהווים חלק מהגדרת המערכת התיאורטית.



הצגת התקיפה



הצגת התקיפה



הצגת התקיפה

נספח ב' – טבלת אירועים לניטור

טבלה זו מכילה את האירועים ב-Windows 10 עליהם בחרנו לנטר על מנת לזהות פעולות חשודות במערכת ההפעלה. בטבלה ניתן לראות באיזה ערוץ האירועים מפורסמים, מה המזהה שלהם ומתי הם מתרחשים.

Event Name	Channel	Event ID	Occurrence
The workstation was locked	Security	4800	When either a user manually locks his workstation, or the workstation automatically locks its console after a period of inactivity this event is logged.
The workstation was unlocked	Security	4801	When a user unlocks his workstation, you will see this event.
Wireless Local Area Network (WLAN) Extensibility	Microsoft-Windows-NetworkProfile /Operational	10000	Network Connected
Loading drivers to control a newly discovered device	Microsoft-Windows-DriverFrameworks-UserMode/Operational	2003	UMDF Host Process loading drivers for a device
A new process has been created	Security	4688	Event 4688 documents each program that is executed, who the program ran as and the process that started this process.