Technologic Papers

Territorial Dispute - כשסוכנויות ביון רבות על חנייה

מאת יהב ארמרמן

מבוא

ב-13 לאוגוסט 2016, משתמש בטוויטר בשם "@shadowbrokerss" מפרסם עמוד Pastebin, ובו קישור לפרוייקט גיטהאב ביחד עם המסר הבא:

"***!!! Attention government sponsors of cyber warfare and those who profit from it !!***

How much you pay for enemies cyber weapons? Not malware you find in networks. Both sides, RAT + LP, full state sponsor tool set? We find cyber weapons made by creators of stuxnet, duqu, flame. Kaspersky calls Equation Group. ... We give you some Equation Group files free, you see. This is good proof no? You enjoy!!! You break many things. You find many intrusions. You write many words. But not all, we are auction the best files."

באנגלית קלוקלת, אותה קבוצה, המכנה את עצמה: "The Shadow Brokers" מציעה למכור כלי תקיפה שנוצרו על ידי היוצרים של Stuxnet, Duqu ו-Flame - שלושה סוגי Malware, תוכנות או כלי ריגול המותקנים על מחשבים, שהיו מהפכניים מבחינת המורכבות והשימוש המבצעי בהם, כאשר הראשון מביניהם - Stuxnet, מוכר בישראל בהקשר אירועי תקיפת הסייבר כנגד הכור הגרעיני בנתנז, איראן ב-2010, אירוע שבעיניי מסמל 'יריית פתיחה' רשמית במלחמת הסייבר שמקיימת ישראל כנגד אויבותייה.

"The Shadow Brokers" משתמשת בשם שחברת Kaspersky שייכה לקבוצה הנגדית-"Equation Group", ומציעה את כליהם לכל המרבה במחיר, וכדי להוכיח שאינם בודים מדמיונם תקיפה וזליגת כלים, הם מפיצים קובץ מסוג tar ובו נמצאים 7 קבצים, כאשר 5 מהם מכילים תמונות המתארות תיקיות עם שמות של חולשות לכאורה, ושני קבצי-tar מוצפנים בסיסמה, המכילים את החולשות שפורסמו. חוקרים הצליחו לשבור את הצפנת הקבצים ואכן למצוא חולשות וכלים שונים וביניהם כלי שו"ב (שליטה ובקרה), אנטי-פורנזיקה ועוד...

Lost in Translation

נדלג כמעט שנה קדימה, ל-14 באפריל 2017. "The Shadow Brokers" מפרסמת שוב הודעה, הפעם כועסת על כך שלא משלמים להם והיא מפרסמת עוד קובץ tar, שהפעם מכיל מספר רב מאוד של חולשות, כלי שו"ב, כלי תקיפה בשם-FuzzBench, הדומה ל-Metasploit. הפרסום מכיל חולשה הנקראת-"EternalBlue",

התוקפת את מימוש פרוטוקול SMB בWindows, אשר תוקנה על ידי מיקרוסופט כחודש לפני הפרסום. החולשה הנ"ל נוצלה בתקיפה Wannacry, אשר הייתה מתקיפות ה-Ransomware החמורות ביותר שנצפו, בעיקר כי חלק גדול מהמטרות היו מערכות מחשוב של ארגונים כמו בתי חולים, שבהם עקב הצורך לפעילות תמידית לרוב אינם מעודכנים ובכך לא מגיעים אליהם עדכוני אבטחה.

"Equation Group", לפי הדעה הרווחת, היא חלק מארגון ה-NSA, סוכנות אמריקאית האחראית על תחום הסיגינט בארה"ב. הסוכנות אחראית על מעקב, איסוף וחקר מודיעין האותות, והגנה על מערכות המחשוב בארה"ב. במונחים היותר מוכרים לקהל הרחב, הסוכנות מקבילה בתחום עיסוקה ליחידה 8200.

בין הקבצים שפורסמו קיימת תיקייה בשם-TeDi, המכילה 4 סקריפטים בפייתון, שבעקבותם נכתב מאמר זה, מהסתכלות על הקובץ-utils.py , אחד הקבצים שהודלפו, אנו מוצאים את השורה:

```
tedilog = getLogger('TERRITORIALDISPUTE')
```

ומכאן אנו מגלים את שמו המלא של המודול, "Territorial Dispute" או בעברית: סכסוך טריטוריאלי.

אסקור כאן תוקפים שונים, אשר יכונו גורמי תקיפה, או בשפה המקצועית נקראים APT, עבור Advanced Persistent Threat. גורמי תקיפה אלו, שונים מהתקפות הסייבר ה"רגילות" המוכרות לנו, בכך שהן ברובן אופרטוניסטיות, והמטרה שלהם היא כסף, קידום אינטרס פוליטי או התרברבות. ה-APT מטרתם היא עמוקה יותר, ותהיה מכוונת כלפי מטרה או סקטור מסוימים, למשל: משרדי ממשלה, או תעשיות ביטחוניות, בין אם לצורכי ריגול, איסוף מודיעין, הרס ופגיעה כלכלית, ומבצעי תודעה. ה-APT יהיו קבוצות בעלות מימון גבוה, לרוב על ידי ממשלות, אשר יכונו- Nation State Actor, ולעיתים על ידי ארגוני פשע גדולים. מצורף קישור למאמר רחב של חברת Imperva על מהו APT, מחזור החיים שלו, ודרכים להתמודדות איתם.

במאמר אחקור את המודול, אערוך הסבר טכני על הקוד ודרכי הפעולה שלו, אדבר על המניע של ארגון בגודלו של ארגון ה-NSA להחזקת מודול זה ואיך הוא יכול לשמש ככלי משמעותי לשמירת חשאיות וכן אסקור חלק מחתימות ה-Malware אותן המודול מחפש.

ניתוח טכני

כדי להבין את פעולת המודול, נתחיל בניתוח התיקייה. התיקייה שהודלפה מכילה 4 קבצים:

1. datastore.py
2. ifthen.py
3. sigs.py
4. utils.py

נתחיל ב-datastore.py, המכיל משתנים המאותחלים ל-None. קובץ זה משמש ליצירה ואחסון של משתנים גלובלים בהם יהיה שימוש בקבצים האחרים במודול.

הקובץ השני -ifthen.py, אחראי על איסוף המידע למשנים של ה-Datastore, והשמתם.

נמשיך לקובץ השלישי- utils.py. לפי השם אפשר להסיק שהקובץ מייצא פונקציות עבור שימוש המודול, ואכן אפשר לראות זאת מהסתכלות על ה-References לפעולות בתוך התיקייה. נתמקד בשלוש הפונקציות שלהן יש Reference מחוץ לקובץ.

בשלוש הפונקציות עליהן נסתכל יש שימוש במתודות מספרייה שנקראת-dsz. הספרייה מהווה מימוש לפעולות של DanderSpritz - פריימוורק לכלי Post Exploitation. כלומר, כלים שנפעיל לאיסוף מידע או מעבר בין מטרות אחרי החדירה הראשונית למטרת התקיפה.

לשמחתנו, הקוד כתוב כ-"Framework Code". כלומר, קוד עם דוקומנטציה פנימית ברמה גבוהה, שמות פונקציות ומשתנים בעלי משמעות ובאופן כללי קוד שכתוב ברמה מאוד גבוהה:

```
def file_exists(path, name):
    path = path_normalize(path)
    cmd = (u'fileattributes -file "%s\\%s"' % (path, name))
    (cmdStatus, cmdId) = dsz.cmd.RunEx(cmd.encode('utf8'),
dsz.RUN_FLAG_RECORD)
    if cmdStatus:
        [attrib_value] = dsz.cmd.data.Get('file::attributes::value',
dsz.TYPE_INT, cmdId)
        if (attrib_value > 0):
            return True
        else:
            return False
    else:
        return False
```

נסתכל על הפונקציה file_exists, ראשית נשים לב לפרמטרים (path,name), מבלי להסתכל על מימוש הקוד אפשר להסיק שהפעולה מקבלת שם של קובץ, ומיקום במערכת הקבצים, ובודקת אם הוא קיים. ממעבר על מימוש הפעולה, אפשר לראות שהפונקציה מרכיבה פקודה ומכניסה אותה לתוך משתנה-cmd, ולאחר מכן מריצה אותה בשורת הפקודות של-DanderSpritz. לבסוף, ומקבלת חזרה Tuple ופורקת אותו לשני משתנים: cmdStatus, ו-cmdId. אם-CmdStatus חזר כ-True, והקובץ אכן קיים, הפונקציה מחזירה True, אחרת תחזיר False.

הפונקציה reg_exists דומה מאוד לקודמת בפעולתה, רק שבמקום לחפש קיום של קובץ, היא מחפשת מפתח ב-Registry, מעין מסד נתונים ענק שמכיל נתונים עבור תוכניות Windows השונות.

האחרונה, limitedget, אחראית על להביא את הקובץ המבוקש בפונקציה זו פחות אתעמק במאמר זה.

הקובץ sigs.py:

sigs.py הוא הקובץ המשמעותי ביותר עבור מאמר זה ונבין זאת דרך ניתוחו.

מהסתכלות ראשונית על הקובץ, ניתן לשים לב לכך שמוגדרות בו הרבה פונקציות, כ-54 במספרן, ששמותיהן דומים מאוד, ומכילים שני Prefixes, הראשון: find, והשני: get, שאחריהם מגיעים מספרים סידוריים. אפשר לראות גם שבסוף הקובץ יש שני משתנים המשמשים לייצוא, ומכילים מערכים המכילים את הפונקציות הנ"ל.

בנוסף לכך, בקובץ מוגדרות פונקציות, לדוגמת get_03-, שגופן מכיל רק את המילה pass, מילה שמורה בפייתון המשומשת בפונקציות שעדיין לא מומשו, אבל כדי למנוע אי הרצה של הקוד עקב פונקציה ריקה הנמצאת בתוכו (הקוד מצפה לאינדנט שלא קיים), משתמשים במילה pass, שגורמת לחזרה של הפונקצייה עם הערך None.

אחרי שהבנו את המבנה הבסיסי של הקובץ, נתחיל לעבור על הפעולות ולהבין מה מטרתן:

```
def find_08():
    if ('s7otbxsx.dll' in datastore.SYSTEMROOT_FILE_SET):
        return True
    if ('mrxcls' in datastore.SERVICE_NAME_SET):
        return True
    if utils.file_exists(('%s\\inf' % datastore.SYSPATH_STR),
'mdmcpq3.pnf'):
        return True
    return False
```

נסתכל על הקוד של find_08, בפונקציה קיימים שלושה תנאים:

1. האם קיים הקובץ 's7otbxsx.dll' ב-System Root, מאגר קבצי הליבה של Windows?
2. האם קיים Service בשם 'mrxcls'?
3. האם קיים הקובץ mdmcpq3.pnf ב-SystemPath? (הבדיקה נעשית באמצעות file_exists, מ-utils.py).

ובכן, הבנו שהפונקציה מחפשת כל מיני פרמטרים במערכת ההפעלה, ומחזירה True אם היא מוצאת אחד מהם.

כדי לנסות להבין מהם אותם פרמטרים, ערכתי חיפוש באינטרנט:

1. התוצאה הראשונה בחיפוש שם הקובץ מראה לנו הקובץ הוא ספרייה לרכיבים של חברת Siemens.
2. התוצאה השנייה בחיפוש אחר שירות מערכת מראה לנו שהשירות משמש לטעינת Malware.
3. הקובץ השלישי כבר חוסך לנו להסתכל על תוצאות שהן לא ראשונות בגוגל, וצוענ לנו שהקובץ נקרא גם WORM_STUXNET.

אם נקרא על-Stuxnet, נבין שחלק עיקרי בפעולת הוירוס היה לשנות פעולות של בקרים שנקראים: PLC (קיצור של Programmable Logic Controller) אותם רכיבים משמשים בדרך כלל כרכיבי תקשורת בין מכונות תעשייתיות גדולות, במקרה הזה צנטריפוגות איראניות, לבין מחשבים.

אז הבנו שהפונקציה מחפשת סממנים לסיכון (תרגום חופשי מהמונח המקצועי: Indicators of Compromise IoC), של Stuxnet, אותם IoC הם קבצים ושירותי מערכת אשר Malware, במקרה שלנו, Stuxnet, משאיר מאחוריו, ובאמצעותם אפשר לזהות אותו על המחשב בו אנחנו חושדים לתקיפה.

אוסיף שההגדרה הרשמית ל-IoC היא "פיסת מידע פורנזי, לדוגמת דאטה, לוגים וקבצים, המשמשים לזיהוי פעילות זדונית ברשת או מערכת" (ויקיפדיה). לרוב אינדיקטורים אלה יהיו פעילות רשת, קבצים, שירותים ומפתחות רגיסטרי, אך הם יכולים להיות גם משתמשים ש-Malware ייצר לעצמו, עלייה משמעותית בקריאת קבצים, ובמקרה של Ransomware, קבצים מוצפנים בעלי סיומת מיוחדת ו-Ransom Notes.

כדי לוודא שחיפוש הפרמטרים הוא רוחבי לאורך כל פעולות ה-find, נעבור על קוד של פונקציה נוספת:

```
def find_16():
    if ('indsvc32.ocx' in datastore.SYSTEMROOT_FILE_SET):
        return True
    if utils.file_exists(('%s\\temp' % datastore.SYSPATH_STR),
'indsvc32.ocx'):
        return True
    return False
```

הפעם נסתכל על find_16, הפעם נמצא 2 תנאים:

1. האם הקובץ 'indsvc32.ocx' האם קיים ב-SystemRoot?
2. האם אותו קובץ קיים ב-'SYSPATH/temp'?

נחפש את שם הקובץ ונמצא שמדובר ב-Flame, קמפיין Malware ששימש לריגול ברחבי המזרח התיכון.

בקובץ מוגדרות גם פונקציות get, האחראיות על השגת אותם אינדיקטורים, והכנה לשליחתם למפעיל הקוד.

ממצאים נוספים

אז אחרי שהבנו את עיקרון הפעולה העיקרי, נוכל להתחיל לנבור לעומקי המודול ונוכל למצוא בו כמה ממצאים פורנזיים שיוכלו לעזור לנו.

Other_peeps.txt:

אם נסתכל בקובץ ifthen.py, נוכל לראות שהסקריפט פותח קובץ בשם 'other_peeps.txt'. אם נחפש אותו בהדלפה, לא נמצא את השם המדוייק הזה, אבל כן אפשר למצוא קובץ בשם '*gwdef_other_peeps.txt*'. הקובץ מורכב מפונקציות המחפשות 4 קטגוריות: תהליכים, קבצים, שירותי מערכת, ומידע גולמי, המורכב ממפתחות רגיסטרי וסקריפטי חיפוש, אשר יתוארו בהמשך. קובץ זה מתאר את אותם אינדיקטורים שמצאנו ב-sigs.py אך בפורמט אחר, ויותר נוח לקריאה.

תיקיית malfind:

תיקייה נוספת שאנחנו יכולים למצוא נקראת malfind. כאשר mal הוא קיצור ל-Malicious, והיא מכילה סקריפטים ב-Batch. הסקריפטים משמשים לזיהוי בדומה לשאר ולחלקם יש חלק נוסף אשר עליו נסתכל:

נסתכל על getsig8.eps ונראה שהמנגנון פועל באופן זהה למנגנון ב-sigs.py, אך במקום להחזיר True או False, הסקריפט מדפיס: *"SIG8 was detected. Do you want to grab the files?"*

כלומר, הסקריפט משיג את הקובץ ואת הגודל שלו, ושואל האם המפעיל רוצה להשיג את הקובץ. אם כן, מועברת קריאה עם מספר החתימה Sig08, ומיקום הקובץ, לפעולה MoveLocalFileToDropbox. נשים לב שבתחילת הקובץ יש include לספרייה בשם DropboxApi, נחפש אותה באמצעות כלי החיפוש של גיטהאב ונוכל למצוא את המימוש של הפעולה. הפעולה אינה מורכבת; היא יוצרת Dropbox, אם אינו קיים כבר, ומעבירה אליו את הקובץ. כל זאת במטרה שיהיה ניתן לשלוח את הקובץ לניתוח חזרה ב'מפקדה'.

DriverList:

הקובץ DriverList, קיים כמה פעמים בהדלפה, פעם אחת כקובץ דאטאבייס של SQLITE, ועוד פעמיים בפורמט טקסט. קיים הבדל מינורי בין הדאטאבייס, לבין קבצי הטקסט, והוא קבצים של המודול Killsuit, המשמש ל"שמירת אחיזה" על המחשבים שהותקפו גם אחרי כיבוי Persistence.

לקבצים הוספה הכותרת "REMOVE ME" וזו כנראה הסיבה שאינם מופיעים בדאטאבייס. להרחבה על Killsuit.

הקובץ נורא פשוט ונסתכל עליו, למען נוחות, בצורת CSV. ה-Database מכיל כ-6 עמודות: שם, תגובה (Comment), סוג, Hash, גודל ועדכון. למשל:

```
name, comment, type, hash, size, updated
crypt32.dll, Crypto API32, ISO_HASH, bd792b7c413bd1b36e7be998f7d323c2c076dc9f,
554496, 20130424
```

השורה הזאת, מייצגת את הספרייה האחראית על כל תחום הקריפטו ב-Windows.

אז הקובץ הזה גדול למדי, ומכיל כמעט 23,000 שורות, ומכיל בעיקר המון דברים שלא מעניינים אותנו כמו קבצי ליבה של Windows, דרייברים לאמצעי זיכרון, כרטיסי מסך, מצלמות ודומיהם, אך הוא מכיל כמה דוגמאות מעניינות:

1. הקובץ מכיל שורות שמכונות PSP - ראשי תיבות עבור-Personal Protection Product, כלומר אנטי-וירוס או מוצר הגנה, אפשר לראות שסוג הקובץ הוא SECURITY_PRODUCT:

```
amon.sys,!!! PSP: ESET Nod32 Anti-Virus !!!,SECURITY_PRODUCT,,,
```

2. כלים, אפשר למצוא בקובץ כלים של ה-NSA, הנמצאים בתוך ההדלפה, ניתן לראות שבהערה רשום שם הכלי, וסוג הכלי הוא TOOL:

```
appinit.sys,*** PEDDLECHEAP ***,TOOL,,,
```

3. Malware - כאן ניתן למצוא 3 קטגוריות שונות- Known Malware, Dangerous Malware, Possible Malware. החלק המעניין פה, הוא אופן הפעולה הנמצא בתגובות, אפשר לראות שהחתימות מסוג-"Known" ו-"Possible" מלוות בהוראה: "*FOLLOW GUIDANCE*", כלומר קיים איזשהו מסמך, שאינו בהדלפה, המסביר מהו אופן הפעולה המומלץ במקרה שמזוהה Malware מוכר.

לעומת זאת, אופי הפעולה בקטגוריה השלישית, "Dangerous Malware", הוא: "*SEEK HELP ASAP*", כלומר לפנות לעזרה במיידי. אנחנו יכולים להסיק מזה שבפרספקטיבה הרחבה של ה-NSA, קיימת דאגה מגילוי של סוגי Malware הקיימים בשטח, ואינם משוייכים לגורמי תקיפה משמעותיים.

4. Sigs - אותן חתימות שפגשנו מספר פעמים במאמר, ההוראה לחתימות היא "*FOLLOW GUIDANCE*" וגם להם יש אופי פעולה מוגדר.

5. Warning - כאן ניתן לראות כלי בטיחות שלא נכנסו לקטגוריית ה-PSP, כמו Packet Sniffers ואמולטרים, ודרייברים שאינם מזוהים אך כנראה מהווים סיכון שניהם נמצאים תחת הכותרת: "*UNKOWN - PLEASE PULL BACK*", עם הוראה למפעיל לצאת מהמכונה אותה תוקף. ניתן לראות שתי דוגמאות לכלים של גופים ידידותיים, כנראה גופים אחרים בתוך ה-NSA או בסוכנויות הביון האחרות בארצות הברית, וההוראה עבורם היא "*SEEK HELP ASAP*" כמו במקרה של "Dangerous Malware".

חוץ מאלו, קיימת עוד שורה אחת, שלגביה לא הצלחתי למצוא שום מחקר או חומר לקריאה באינטרנט. שם הקובץ הוא fast16.sys, וההוראה שלו היא: "*NOTHING TO SEE HERE - CARRY ON*". למה יש קובץ שמוגדר כ-WARNING והכיתוב שלו הוא שונה מאוד משאר הניסוחים, אני מאמין שזה כבר נושא למחקר אחר.

מניעים

כל המודול שסקרנו, הפעולות והקבצים והשונים, לעניות דעתי, כולם משמשים כדי שה-NSA, מלבד המטרה הבסיסית של מניעת חשיפת כלים לגורמי תקיפה שונים, יוכל להיות מעין "העין שבשמים", שצופה על עולם ה-Malware והריגול הממוחשב, לראות את כל המתרחש, ולהיות מסוגלים לנתח לעומק את תקיפות הסייבר בהן הם צופים.

סיכום

במאמר זה הצגתי אירוע הדלפת כלים המיוחסים ל-NSA, המכונה "Lost in Translation", ביצעתי ניתוח טכני על מודול TeDi, דרכי פעולתו, מנגון זיהוי החתימות העיקרי, ממצאים נוספים הקשורים לזיהוי החתימות וזיהוי IoC הנמצאים בהדלפה ודנתי במניעים לקיום מודול זה.

ברצוני להודות ל**אמיר ר'** על עריכה לשונית וליטוש המאמר, ול**אפיק קסטיאל**, עורך המגזין עבור ליווי במהלך הכתיבה וליטוש והכנת המאמר לפרסום.

על הכותב

יהב ארמרמן, בן 19 מהדרום, בוגר שנת שירות מטעם הצופים ומתגייס לצה"ל במהלך חודש אוקטובר, מתכנת ועוסק באבטחת מידע ורשתות כ-5 שנים, בשנה האחרונה בתחום ה-Malware, Reversing ומחקר APT, לפניות, הערות ושאלות מוזמנים לשלוח הודעה!

מקורות לקריאה נוספת

הפייסטבין שבו פורסמה ההודעה הראשונה, מאורכב:

https://archive.ph/20160815133924/http://pastebin.com/NDTU5kJQ

קישור ל-github בו מאוכסן Lost in translation:

https://github.com/misterch0c/shadowbroker/tree/e3a069bea47a2c1009697941ac214adc6f90aa8d

קובץ העוקב אחרי APT שונים ושיוכם ברחבי העולם:

https://apt.threattracking.com

מחקר של מעבדת CrySys, בו נעזרתי המון במהלך המחקר:

https://www.crysys.hu/publications/files/tedi/ukatemicrysys_territorialdispute.pdf