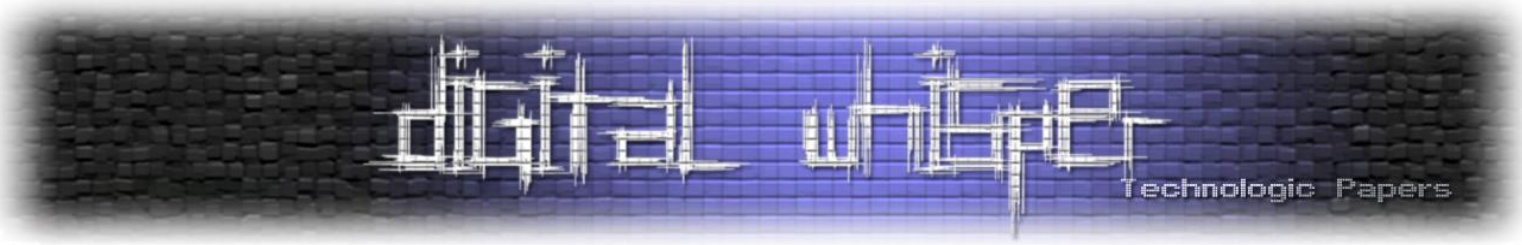


Digital Whisper

גליון 44, אוגוסט 2013

מערכת המגזין:

מייסדים:	אפיק קסטיאל, ניר אדר
מוביל הפרוייקט:	אפיק קסטיאל
עורכים:	שילה ספרה מלר, ניר אדר, אפיק קסטיאל
כתבים:	ישראל חורז'בסקי (Sro), רן לוי, אריק יונאי, אמיתי דן (popshark), יצחק דניאל (iTK98) ואפיק קסטיאל (cp77fk4r).

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

דבר העורכים

ברוכים הבאים לדברי הפתיחה של הגיליון ה-44!

חודש אוגוסט כאן, ואיתו, מלבד החום המשוגע שמסתובב בחוץ, באים הרבה דברים טובים, מהכנס אוגוסט פינגווין בחיפה ועד הכנסים Decfon ו-Blackhat בלאס-וגאס. בדרך כלל, בתקופה הזו, עולם אבטחת המידע רועש וגועש (קצת פחות בגלל אוגוסט פינגווין וקצת יותר בגלל Defcon ו-Blackhat) מפני שבכנסים אלו האקרים חוקרי אבטחת-מידע מכל העולם מפרסמים כל מני מתקפות וקונספטים מעניינים שבלא מעט מקרים משפיעים על תפיסתה של כלל הקהילה בנוגע לנקודות שונות.

שווה מאוד לעקוב אחר הפרסומים בנושא (שלא כמו בשנים שעברו, הפעם לא נפרסם עדכונים בנוגע לכנס, אין לנו את הזמן הפנוי הזה של לנהל את הבלוג בצורה שוטפת), אבל בלא מעט אתרי חדשות ובלוגים בנושא מתפרסמים עדכונים ב-live אודות המתרחש בכנס - שווה להשאר מעודכנים.

ונחזור לארצנו, ולמגזין. החודש מתפרסם ספר בשם "**קרב מוחות (ההיסטוריה הזדונית של וירוסי המחשב)**", של הסופר, הבלוגר והפודקסטר ("עושים היסטוריה") **רן לוי**. הספקתי לקרוא את רב הספר, ולדעתי (ואני אובייקטיבי לחלוטין) הספר כתוב בצורה מעולה, מביא סיפורים מעניינים במיוחד (גם לקהל הלא-טכנולוגי), ובהחלט מספק את הסחורה. דעתי האישית היא שחסרים בספר מספר נושאים מרכזיים כגון הוירוס Conficker, הקבוצה 29a והקהילה VX Heaven (שלדעתי, גם 29a וגם VX Heaven עיצבו רבות את עולם הוירוסים) ועוד, אך עם זאת, אני ממליץ בחום על הספר, במגזין הנוכחי תוכלו להנות מהפרק הראשון של הספר. בנוסף, **הוצאת כתר** סידרה לקוראי המגזין הנחה (פרטים בעמוד 38).

לפני שניגש לסיבה שלשמה התכנסנו כאן היום, היינו מעוניינים להגיד תודה רבה לכל מי שמבזכותו הגיליון התפרסם החודש. תודה רבה ל**ישראל חורז'בסקי (Sro)**, תודה רבה ל**רן לוי**, תודה רבה ל**אריק יונאי**, תודה רבה ל**יצחק דניאל (iTK98)**, תודה רבה ל**אמיתי דן (popshark)**, תודה רבה על הזמן הפנוי שלהם שתרמו לנו, ושהמגזין הינו תוצר שלו וכמובן, לעורכת שלנו - **שילה ספרה מילר**, על עריכת המאמרים. תודה רבה!

קריאה מהנה!

ניר אדר ואפיק קסטיאל.

תוכן עניינים

חדשות

מאת יצחק דניאל (iTK98) ואפיק קסטיאל (cp77fk4r)

KeyJacking

בסוף חודש שעבר, חוקר אבטחה איטלקי בשם רוסאריו ואלוטה הציג מתקפה חדשה ממשפחת המתקפותUI Redressing בשם "KeyJacking", מימוש מוצלח שלה מאפשר לתוקף להריץ קוד על מחשבו של הנתקף. המתקפה מנצלת את אופצית שימוש בקיצורי המקלדת בתוכנת הדפדפן של הגולש וגורמת לו להוריד קובץ כלשהו (לדוגמא - exe) ובעזרת קיצורי המקלדת - להורות לדפדפן להריץ אותו.

ה-PoC שואלוטה הציג הולך כך: משתמש ניגש לקישור על מנת להרשם לפורום או כל אתר אחר, במהלך הקישור עליו למלא טופס המכיל CAPTCHA (בדמו עצמו, ואלוטה מימש אך ורק את ה-CAPTCHA ולא את כלל הטופס, אך זה לא קריטי על מנת הציג את הרעיון), הטופס נראה כך:

במידה והמשתמש אכן ממלא את ה-CAPTCHA - ה-PoC מוריד קובץ exe ומריץ אותו על מחשבו של הקורבן ללא ידיעתו.

איך זה עובד? פשוט מאוד - כאשר נכנסים לעמוד, מורץ הקוד הבא:

```
<iframe id="f1" width="100" height="100"></iframe>
<script type="text/javascript">
       document.getElementById("f1").src="CosmicBreak_BR_setup.exe";
</script>
```

שמורה לדפדפן להוריד את הקובץ CosmicBreak_BR_setup.exe, כל זה אחלה - במקרה של אתר רגיל הדפדפן אמור להציג הודעה שבה על המשתמש לבחור מה לעשות עם ההורדה (האם לשמור את הקובץ, להריץ אותו, או להריץ ולשמור). מה מיוחד ב-PoC? שום דבר, גם כאן נפתח החלון הנ"ל, רק שכאן החלון הנ"ל מופיע מתחת לחלון של ה-CAPTCHA, ואז כאשר המשתמש ממלא את ה-CAPTCHA, הוא לוחץ על האות "R" (כחלק מה-CAPTCHA עצמה) וברגע שהמשתמש לוחץ על האות הנ"ל, הדפדפן מתרגם זאת כאילו המשתמש ביקש להריץ את הקובץ שירד ועשה זאת דרך קיצור המקלדת ("R" היא קיצור של "Run").

לדעתי מדובר במתקפה מדהימה בפשטותה, ושלא כמו ב-ClickJacking, כאן ניתן ממש להריץ קוד על המחשב של הקורבן.

נכון לכתיבת שורות אלה ה-PoC עובד רק על הדפדפן IE, אבל אני מאמין שלאט לאט הקונספט יתפוס וימצאו דרכים נוספות לשלב אותו גם בשאר הדפדפנים.

אתר המפתחים של אפל, נפרץ?

ביום חמישי, ה-18 ביולי 2013, אתר המפתחים של אפל ירד, בעמוד הכניסה לאתר הופיעה הודעה לקונית האומרת כי האתר נתון תחת תחזוקה והמשתמשים מתבקשים להיות סבלניים עד שהוא יחזור. אחרי יומיים בהם האתר עדיין למטה, החלו להתעורר חששות כי אתר המפתחים של אפל נפרץ. תגובה של אפל הגיעה רק ביום ראשון (21 ביולי), בה היא חשפה כי אכן חדירה התבצעה אל מרכז הפיתוח שלה ומידע על המפתחים דלף.

איברהים בליק, מפתח טורקי בן 25, ראה בפעולות האחרונות שלו סיבה לכך שאתר המפתחים של אפל נסגר לתחזוקה. בליק טוען כי לא היו לו כל כוונת זדון, והוא רק רצה לבחון כמה עמוק הוא יכול לחדור דרך הפרצות שהוא מצא. למעשה בליק דיווח על הפרצה לאפל באותו היום שבו האתר ירד מהאוויר לראשונה, וכן הוא דיווח על 13 פרצות שונות בסך הכל החל מה-16 ביולי שאיתן הוא היה מסוגל לדלות מידע על משתמשים אחרים.

אך בליק לא עצר בדיווח, והפעולות הבאות שלו מעלות תמיהה בדבר הכוונות שלו. הוא העלה ליוטיוב קטע וידיאו בו הוא מראה את מידע שהוא הצליח לדלות, בעוד שהקטע המקורי נחסם, ישנו העתק שבו אפשר לראות כי בליק הצליח לדלות מידע על 43 אלף מפתחים (דקה 1:50), לאיזו מטרה, אפשר רק לנחש. הפעולה נעשתה ב-22 ליולי, אך התאריך של הקבצים שאנחנו רואים בשרת ה-FTP שלו מתוארכים ליום לפני כן, לכן לא ברור מתי הפעם הראשונה שבליק החל לדלות מידע ובאיזה היקף.

במקביל לדיווח של בליק החלו מפתחי אפל לקבל דואר המבקש מהם לאפס את הסיסמה, כנראה שלא בליק בעצמו אחראי על זה, אלא משתמשים אחרים שהיו בעלי גישה למערכת דיווח באגים והם בנו סקריפטים בכדי לדלות מידע בעצמם, שכן המידע הזה יכול לשמש לדיוג ממוקד. פעולות שכאלה (דיוג) נעשות כל הזמן, ולא ניתן לקשור את הדיווחים השונים על נסיונות דיוג ישירות אל הפרצה שבליק מצא.

אז מה יש לנו כאן, אתר אחד למטה, חוקר אבטחת מידע אחד שטוען שלא היו לו כוונות זדון, ואלפי אימיילים שנשלחו למפתחי אפל בנסיון להשיג את הסיסמה שלהם. בליק למעשה טעה בשלוש מקומות:

1. בדיווח הבאג הוא נתן דוגמה, שלמעשה היוותה PoC, משם זה הפך במהרה 0Day ואילץ את אפל לסגור את אתר המפתחים.

2. הוא משך מידע על עשרות אלפי מפתחים, כאשר הוא יכל להסתפק בדוגמית הרבה יותר קטנה, מה שלא היה מעלה חשד בכוונותיו.

3. לא למד מטעויות של אחרים. "הפורץ" ל-AT&T, שמצא פרצה דומה במהותה (משיכה של פרטי אישיים מאתר חברת AT&T), קיבל מאסר של 3 שנים.

עם סיום הכתבה נותרנו עם שאלה פתוחה, האם בליק הוא סך חוקר אבטחת מידע ללא כוונת זדון או פורץ לא כל-כך חכם (או גאוני ופשוט אינני מסוגל לראות את התוכנית שלו במלואה). רק לאפל התשובה לכך, והיא גלומה במענה שהיא תבחר לתת בעקבות "הפריצה" לאתר המפתחים שלה.

אנדרואיד, התואם הרע

Bluebox, חברת הזנק שעוסקת בתחום אבטחת מידע של טלפונים ניידים, מצאה כי ישנה חולשה במנגנון של מנהל ההתקנות של אנדרואיד. החולשה מאפשרת לתוקף לקחת קובץ התקנה לגיטימי עם חתימה מסויימת, ולשנותו, כאשר מנהל ההתקנות של המערכת יבחן את הקובץ, הוא לא ישים לב כי נערך שינוי וימצא כי החותמת תקינה גם לקובץ הערוך. חשוב לציין שהחוקר ב-Bluebox מסר אך ורק מידע כללי ורמיזות בדבר הפריצה, הוא העדיף שלא לספק מידע טכני או PoC. החוקר, ג'ף פוריסטל, מבטיח כי ימסור עוד מידע כולל הדגמות בכנס BlackHat2013.

הקהילה לא נותרה אדישה והחלה לחקור, מהו בדיוק אותו באג 8219321 שמוזכר בבלוג. הבאג מדבר על הופעות של קבצים שונים בעלי אותו שם בתוך הקובץ התקנה ודרך ההתייחסות של מנהל החבילות להופעה זו. קבצי ההתקנה (חבילות) למערכת ההפעלה אנדרואיד מגיעים כקבצי APK, הקובץ הזה הוא למעשה קובץ ZIP עם מבנה מסויים. כאשר מופיעים קבצים בעלי אותו שם מנהל ההתקנות למעשה בודק את הקובץ השני, אך מתקין את הקובץ הראשון, וזהו למעשה ווקטור התקיפה, שמאפשר לתוקפים לקחת חבילות לגיטימיות ולהזריק אליהן רוגלות ומזיקים אחרים.

הבאג דווח לגוגל בפברואר 2013 וגוגל שחררה תיקון עבור הבאג, אך בפועל שום יצרנית לא הטמיעה אותו, אף לא גוגל עצמה בנקסוס, מה שמותיר את רוב בעלי מכשירי האנדרואיד חשופים למתקפה זו. חשוב לציין שגוגל הטמיעה בחנות שלה, Google Play, בדיקת לאפליקציות שמעלים, וכאלו שיכילו שמות קבצים זהים יחסמו מהחנות. נכון לעכשיו בטוח להוריד מהחנות אפלקציות של גוגל, אך יש להיזהר ממקורות אחרים מהם אתם משיגים את האפליקציות שלכם לאנדרואיד.

החוקרים ב-BlueBox שבוע לאחר החשיפה שיחררו אפליקציה שיכולה לעזור למשתמשי אנדרואיד לזהות אם המכשיר שלהם מוגן בפני המתקפה הזאת, והאם קיימות אפליקציות מותקנות אשר ניצלו את הווקטור הזה בכדי לתקוף את הטלפון הנייד שלכם. בנוסף, חשוב לציין כי קהילת המפתחים של Cyanogen היא הראשונה שסתמה את הפרצה בגרסת האנדרואיד שלה. כמו כן, אחד המפתחים באותה קהילה שיחרר PoC המראה את הקלות שבה ניתן לנצל את הפרצה הזאת.

לכל המשתמשים שאינם משתמשים בגירסה האחרונה של Cyanogen מומלץ להימנע מלהתקין אפליקציות שמצאו מחוץ לחנות Google Play. כל האפליקציות בחנות נבדקות אם הן מנצלות את ווקטור התקיפה הזה, אך עדיין היו מקרים בהם העלו רוגלות לחנות של גוגל, לכן מומלץ להתקין אפלקציות רק ממפתחים מוכרים. בנוסף, ניתן להתקין את Bluebox Scanner בכדי לבדוק אם הגרסת אנדרואיד שלכם עודכנה כנגד המתקפה, ואם יש אפליקציות אשר מנצלות את אותה מתקפה.

מקורות:

- בלוג BlueBox
- Google Operating System – Unofficial news and tips
- PCWorld
- ComputerWorld
- PoC ששוחרר על-ידי פאו אוליביה פרו

אוגוסט פינגווין 2013

כנס אוגוסט פינגווין הוא הכנס השנתי של קהילת התוכנה החופשית בישראל המאורגן על ידי עמותת המקור. הכנס מהווה הזדמנות למפגש חברתי במקביל להרצאות טכניות, הצגה של יוזמות ופרויקטים חדשים ועדכונים על פעילות בנושאי קוד פתוח ותכנה חופשית בישראל במהלך השנה האחרונה.

השנה, מארגני הכנס החליטו להגדיל את מגוון הנושאים כך שיתאימו גם לאנשים חדשים שלא מכירים תוכנה חופשית.

הכנס יתקיים ביום שישי ה-2 באוגוסט 2013, במרכז הקונגרסים הבינלאומי שחיפה. השנה, הכנס כולל ארבעה מסלולים ובינהם ניתן למצוא מסלול המוקדש כולו לאבטחת מידע והאקינג, המסלול כולל את ההרצאות הבאות:

- Attack by custom malware - ע"י שייע פידמן.
- Man In The Middle - Hands-on - ע"י גיא אדרי.
- Hacking the brain by botnet network and crowd research - דן אמיתי.
- האקתון התקפת מידע - ELFים. לא אלה עם האוזניים המחודדות - יובל נתיב.

ניתן למצוא הרצאה נוספת הקשורה לאבטחת מידע הנמצאת מחוץ למסלול:

- CryptoParty - פרטיות ואנונימיות לכולם - יובל אדם.

על מנת להרשם לכנס ולקבל פרטים נוספים יש להכנס לעמוד הכנס.

ניתן ליצור קשר עם ועד עמותת המקור ומארגני הכנס בכתובת הדואר האלקטרוני הבאה:
board@hamakor.org.il

הסכנה שבקבצי HTML

מאת ישראל חורז'בסקי (Sro)

הקדמה

יש קבצים שכולנו מסכימים שהרצה שלהם היא דבר מסוכן. כאלה הם קבצי exe, reg, bat, cmd או בקיצור - קבצי הרצה. בהתאם, כאשר אתרי אינטרנט ינסו להוריד לנו קבצים אלה נקבל התראה שאלה קבצים מסוכנים ממקור לא ידוע. דוגמא נוספת - שליחת קבצים במייל. קבצים עם סיומת מסוכנת לא ניתן לשלוח במייל, לעיתים אף לא מכווץ בזיפ.

כאן במאמר נחקור קבצי HTML, נראה איך שכולם מתייחסים אליהם כקבצים לא מסוכנים, ננסה לנתח למה, ואז נראה גם כמה שהם מסוכנים, עד כדי קריאת כל הקבצים מהמחשב של ה-Victim. במשפט אחרון של ההקדמה אוסיף שהבדיקות בוצעו על דפדפני Firefox[22] , Chrome[28], IE[10] עדכניים לזמן כתיבת המאמר.

האם גוגל ומייקרוסופט חושבים שקבצי HTML מסוכנים

שיטות ההפצה הקלות ביותר של קבצים הינם הורדה מאתר אינטרנט או שליחה במייל. בדוגמא הבאה שמתי את קובץ ה-PoC (final.htm) יחד עם קובץ downloader.php בתיקיה, והקוד של downloader.php הוא:

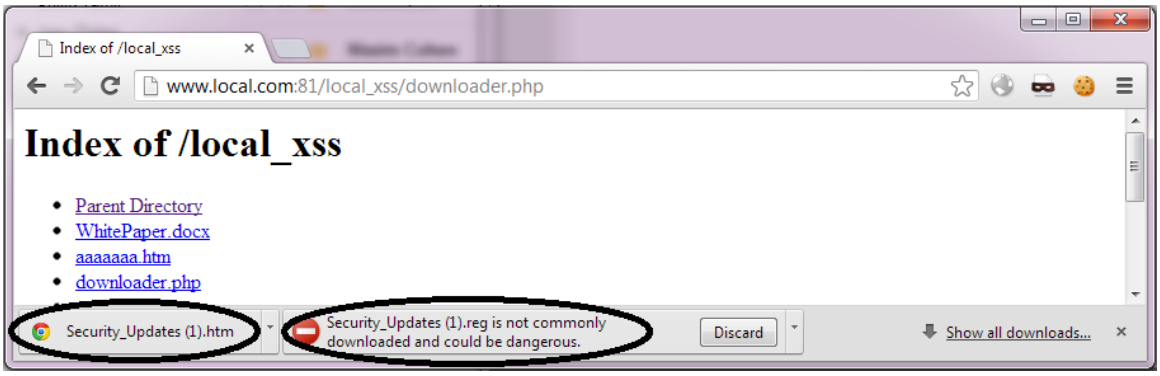
```
<?php
 header('Content-Type: text/html');
 header('Content-Disposition: attachment; filename=Security_Updates.htm');
 echo file_get_contents('final.htm');
?>
```

הקוד שולח לדפדפן 2 כותרים (Headers) על כך שזה קובץ להורדה, שם הקובץ נקבע ל-Seucirty_Updates.htm כדי לבצע Social engineering למשתמש, שיריץ אותו. זה הכל, ב-3 שורות גרמנו לדפדפן להוריד קובץ HTML, בעת ריצה שלו שום דפדפן לא מתריע על כך שזה קובץ מסוכן ממקור לא ידוע.

גולש שגולש לדף downloader.php יקבל את התוצאות הבאות:

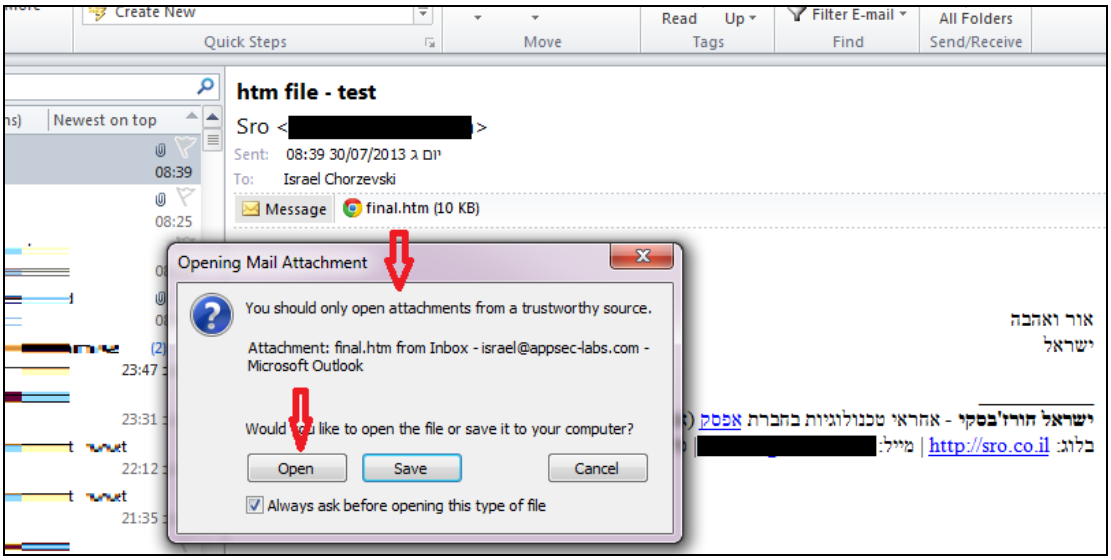
דפדפן	תוצאה
פיירפוקס	שואל אם להוריד או להריץ
כרום	מוריד מיידית
אינטרנט אקספלורר	שואל אם להוריד או להריץ

השוואה גרפית ניתן לראות בתמונה הבאה:

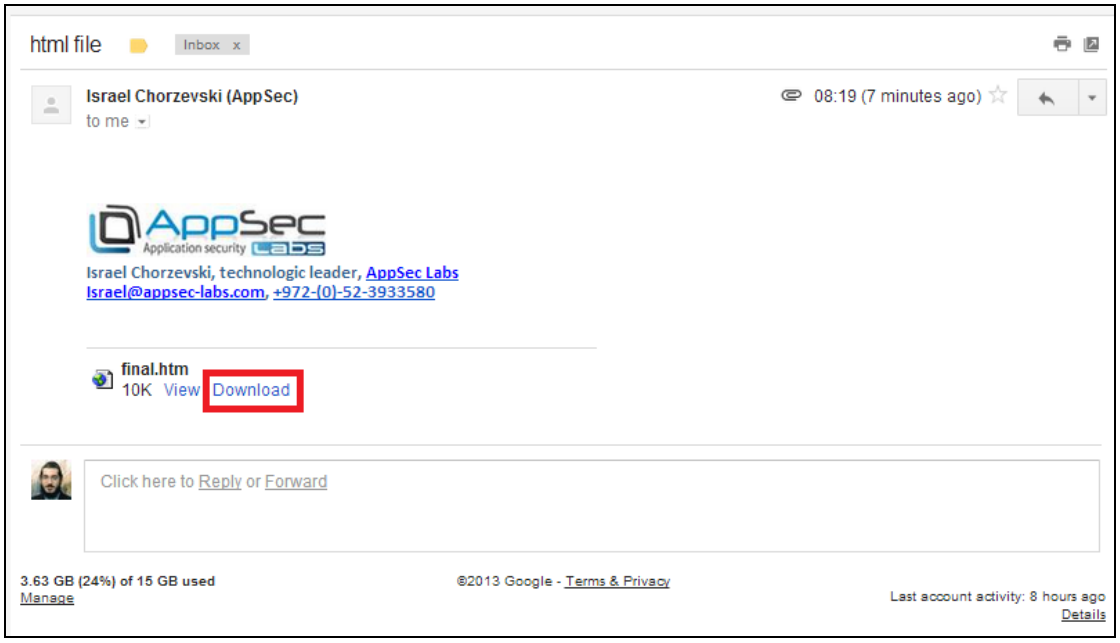


ניתן לראות איך קובץ reg מציג אזהרה בדפדפן כרום, לעומת קובץ htm שיורד מיידית.

שיטת הפצה נוספת, כאמור, היא באמצעות אימייל. כידוע, מערכות אימייל חוסמות אימיילים שמכילים קבצים בעלי סיומות מסוכנות. בתמונות הבאות ניתן לראות איך גם gmail וגם outlook לא חסמו הורדה והרצה של קובץ htm שנשלח במייל. אאוטלוק:



:Gmail



מדוע קבצי HTML לא נחסמים? ככל הנראה החשבון הוא כזה. אם יש לי (כתוקף) 0-day (בעיית אבטחה לא ידועה / לא מתוקנת) על אחד הדפדפנים שאותו אני רוצה לנצל עם java script, אוכל להפעיל אותו על המשתמש באמצעות שליחת לינק במייל, או כשהוא גולש לאתר שלי - במקום לגרום להורדת קובץ. אז אם כן, מה מפריע לנו שהוא יפעיל את הקובץ מהמחשב המקומי שלו? בהמשך נראה שהדפדפנים מנסים להתייחס לקובץ שרץ לוקאלי, באופן זהה לקובץ שרץ מרחוק, אולם נגלה שכל שיטת חישוב ה-origin שלהם שאמור לבצע את הפרדת ההרשאות לוקה בחסר.

לקרוא את נתיב ההורדה (FPD - Full Path Disclosure)

זה ה-"Exploit" הכי בסיסי לקובץ HTML שרץ לוקאלית. רק לקרוא את ה-document.location שמציג ה-URL. בקובץ לוקאלי ה-URL הוא הנתיב המלא של הקובץ. בעיה מסוג זה מכונה Full Path Disclosure, זה רק חשיפת מידע, אבל מידע שמאפשר לנו לבצע מתקפות שונות של קריאת קבצים (מתקפות שכאלה מנצלות חולשות בדפדפנים, פלאגינים והרחבות של דפדפנים שכתובים בצורה לא מאובטחת).

מידע נחמד נוסף שבדר"כ נחשף, זה שם המשתמש של הגולש, כי דיפולט ההורדה יהיה פעמים רבות התיקיה Downloads שב-My Documents שבנתיב המלא זה משהו כמו:

```
C:\users\VICTIM\Downloads
```

עם שם המשתמש אפשר לבצע Brute Force ל-RDP ועוד. אם עדיין לא השתכנעתם למה זה שימושי, חכו לסעיף הבא, רק נסכם קודם את הדפדפנים הפגיעים: כולם.

דפדפן	תוצאה
פיירפוקס	מציג נתיב לוקאלי מלא
כרום	מציג נתיב לוקאלי מלא
אינטרנט אקספלורר	מציג נתיב לוקאלי מלא

לקרוא קבצים מהכונן הקשיח (LFD - Local File Disclosure)

אני חושב שזה הממצא הכי חמור בסיפור הזה, אנחנו מדברים על קריאת קבצים מהארד-דיסק. כאן מן הראוי להרחיב מעט על מודל ה-SOP (Same Origin Policy שמונע גישה לקובץ של מערכת/דומיין אחר) בקבצים לוקאלים. יש לנו שלוש גישות:

- כל קובץ הוא "ממלכה" משלו, אין לאפשר זליגת מידע בין קבצים, אפילו באותה תיקיה.
- כל תיקיה היא אפליקציה, "סביר" שקובץ בתיקיה ירצה לגשת לקבצים שנמצאים לידו / בתיקיות משנה.
- אין דומיינים כי זה קובץ לוקאלי, לקובץ יש גישה לכל יתר הקבצים במערכת הקבצים, אפילו בכונן אחר.

לגבי קריאת קבצים (AJAX, או אם תרצו - CORS), כרום ואינטרנט אקספלורר הולכים בגישה של "כל קובץ לחוד" ואף קובץ לא יכול לקרוא קובץ אחר. פיירפורקס לעומתם תומך בגישה ליברלית והוא מאפשר לכל קובץ לגשת לקבצים אחרים בתיקיה / בתיקיות משנה.

כך שכשאני מוריד קובץ HTML לשולחן העבודה ופותח אותו, הוא יכול לקרוא את כל הקבצים שיש לי על שולחן העבודה, ואת הקבצים שבתיקיות שבשולחן העבודה. הוא רק צריך לדעת (או לנחש) את שמותיהם. לקרוא את הקבצים ולשלוח אותם לשרת של התוקף.

כעת נותר לי להודות על כך שאני לא מוריד קבצים ישירות על כונן C עצמו. זה אומר קריאה של כל הקבצים בכונן C., קבצים בעלי כל סיומת שהיא exe, txt, הכל.

כעת אפשר להבין למה וכמה לדעת את נתיב ההורדה עוזר לנו, כי כדי לקרוא קובץ אנחנו צריכים לדעת את שמו ומיקומו.

דפדפן	תוצאה
פיירפוקס	מאפשר לקרוא כל קובץ שהוא מהתיקיה הנוכחית ומתיקיות המשנה שלה
כרום	חוסם קריאה של קבצים אחרים
אינטרנט אקספלורר	חוסם קריאה של קבצים אחרים

סריקת כוננים קשיחים

יכול להיות שימושי בכל מיני מקרים (אפי' עבור הדבר הפשוט - זיהוי המשתמש. למרבית המשתמשים אין כונן Q, אז אם פעמיים נתקלנו במשתמש עם כונן Q, כנראה שזה אותו אחד). בכל אופן, אנחנו יכולים לבצע סריקה ולדעת איזה כוננים זמינים במחשב.

בגדול הטכניקה היא פשוטה, כדי לבדוק אם כונן Q קיים, ניצור Iframe, נגדיר לו ארוע onload שיקרא לפונקציה שתבצע רישום שה-Iframe נטען. בנוסף, נגדיר ל-Iframe את ה-src ל:

```
file:///Q:/
```

כעת זה מתחלק בין הדפדפנים. כרום מגיב זהה בין מקרה שהכונן קיים למקרה שהוא לא קיים. פיירפוקס מריץ את onload רק אם הכונן קיים, IE מריץ את onload רק אם הכונן **לא** קיים.

זוג פונקציות Java Script שכתבתי (אחת ל-FF, אחת ל-IE) מבצעות סריקה על כל הכוננים ומחזירות רשימה בתוך פחות משניה. פונקציות אלה ובאופן כללי PoC לכל המוזכר במאמר ניתן להוריד בלינק שיופיע בסוף.

באמצעות אותה שיטה ניתן ב-Firefox וב-Internet Explorer לבדוק אם קובץ כלשהו קיים. במקום להגדיר את ה-src ל-file:///Q:/ נגדיר אותו ל-file:///c:/windows/win.ini ונבדוק האם ארוע onload רץ על ה-Iframe.

סריקת פורטים (או: הוכחה שכרום טוב פי 4 מ-FF)

אציין שאפשר לסרוק גם עם JS שרץ מאתר בעת גלישה רגילה לאתרי אינטרנט, ולא רק מדף לוקאלי. אבל מהבדיקות שביצעתי, מתברר שסריקה מדף לוקאלי עובדת הרבה יותר טוב ובאופן יותר מהיר.

איך מבצעים סריקת פורטים? הצורה המהירה היא באמצעות ניסיון קריאת דף עם AJAX (ליתר דיוק, כאן זה כבר CORS, אנחנו מנסים לקרוא דף מדומיין אחר). אפשר לנצל את זה בשביל סריקת פורטים לוקאלית, ואפשר להשתמש בזה בשביל לסרוק פורטים של מכונות אחרות ברשת/בעולם.

מנסים לקרוא את הדף http://127.0.0.1:8080/. בכל מקרה לא נוכל לקרוא אותו (אלא אם כן הוא מוגדר ומציג כותרים שאומרים לדפדפן שמותר לקרוא אותו גם מדומיין אחר, שלא סביר שזה יקרה), אבל מה שכן נוכל למדוד זה את הזמן שלוקח עד שהדפדפן מסיים לטפל בבקשה.

כעת יש שלוש אפשרויות:

1. בפורט 8080 קיים שרת HTTP, במקרה כזה הדפדפן מסיים את הבקשה (=קורא את התשובה. למרות שהוא לא מציג לנו אותה) בתוך פחות מחצי שניה.
2. בפורט 8080 קיים סרוויס שהוא לא HTTP, במקרה כזה הדפדפן מצפה ל-Response, אבל השרת לא יחזיר לו כזה (בטח לא אחד תקין), מה שאומר שהדפדפן יחזיק קונקשן עד ל-Timeout.
3. אין שום סרוויס/שרת שמאזין בפורט 8080, אורך הטיפול בבקשה של הדפדפן הוא בסביבות שניה בודדת.

אז כעת ננסה "לקרוא" את כל הפורטים, נגדיר טיימאאוט של 3 שניות ל-AJAX ונמדוד את אורך ה-Response. אם האורך הוא פחות מחצי שניה, או 3 שניות (טיימאאוט שהגדרנו), סימן שיש איזשהו סרוויס שמאזין על הפורט הזה. אחרת כנראה שאין מישהו שמאזין על הפורט.

סריקת הפורטים הזו היא לא ממש מדוייקת, אבל כמה שהיא חושפת זה עדיין הרבה ! נזכור שאם המשתמש שלנו נמצא מאחורי NAT (או אפי' כל FW אפליקטיבי, הרבה תוכנות מאזינות רק על ה-loopback), אין לנו אפשרות לסרוק אותו רגיל מבחוץ עם Nmap.

ואיפה מה שהבטחנו לגבי ביצועי דפדפנים?

בהתחלה כתבתי קוד שמבצע סריקה, וכל פעם סורק פורט בודד. התוצאה היא - כדי לסרוק 10 פורטים, צריך 10 שניות.

מי שחושב על workers, אז לא. לא ניתן להוציא בקשות AJAX מ-worker. לכן החלטתי לנצל את מנגנון התורים שקיים בדפדפנים מודרנים. ה"טיפול" של הדפדפן בבקשת AJAX מתבצע א-סינכרונית, כך שניתן להריץ במקביל ניסיון קריאה של הרבה פורטים, והדפדפן יטפל בכולם במקביל.

לדפדפנים ול-JS יש נטייה לא לתת תמיכה מלאה של ביצועים, והם לא מבצעים את מה שהם מבטיחים (נתקלתי בזה גם ב-setInterval שאם הגדרתם לו שייגש ל-DOM, לא תוכלו באמת להריץ אותו אפי' לא כל עשירית שניה, הוא ירוץ מתי שבא לו). מה שקורה זה שנפתחים X ת'רדים, וכל היתר מחכים. מבחינתנו זה שהם מחכים יוצר בעיה - כי פתאום בקשה לפורט סגור שאמורה לקחת שניה, תארך 4 שניות רק כי היא "חכתה". ואנחנו נתקלים ב-False positive (זיהוי שגוי, חושבים שמצאנו משהו). לכן גם העליתי את הטיימאאוט משניה וחצי לשלוש שניות, למקרה ומשהו יעכב את הדפדפן לרגע.

אבל בכל זאת, עדיין הדפדפנים יכולים להריץ כמה וכמה בקשות במקביל. זה מאוד משתנה, כי זה תלוי בעומסים שעל הדפדפן, כנראה שזה גם תלוי בכח המחשוב של המחשב שמריץ אותו, וזה גם תלוי בפורטים הפתוחים במחשב. אם יש פורט ומאזין מאחוריו שרת HTTP, הדפדפן יסיים לטפל בפורט הזה במהירות והוא יתפנה לבא בתור, אם יש פורט שמאחוריו יש סתם מישהו שמאזין, הדפדפן ייתקע איתו שלוש שניות שלמות.

בממוצע גיליתי שכרום מאפשר סריקה של 200 פורטים באופן יחסית יציב, פיירפוקס 50 פורטים, ואינטרנט אקספלורר 5 פורטים. אני מדבר על סריקה מקבילית כמובן. ניתן תמיד להריץ אותם אחד אחרי השני.

החלק האופטימי בסיפור, זה שאם הדפדפן התעכב מכל סיבה שהיא, סך הכל זה אומר שנקבל את הפורט כ-False positive, אז מה שאנחנו נבצע זה סריקה נוספת על כל הפורטים שמצאנו ונאמת שאכן מישהו מאזין אליהם. לא ביג דיל. הרבה יותר טוב ממצב הפוך שבו מתפספסים פורטים. לשמחתנו, אנחנו לא שם.

דפדפנים פגיעים? כהרגלנו בקודש - כולם.

דפדפן	תוצאה
פיירפוקס	מאפשר לסרוק - מהירות בינונית
כרום	מאפשר לסרוק - מהירות גבוהה
אינטרנט אקספלורר	מאפשר לסרוק - מהירות נמוכה

יצויין שהדפדפנים חוסמים פורטים מסויימים ולכן נמצא אותם שהם כאילו תמיד "סגורים" (זה משתנה מעט בין הדפדפנים).

דוגמא לרשימה שמצאתי ברשת על כרום:

1, 7, 9, 11, 13, 15, 17, 19, 20, 21, 22, 23, 25, 37, 42, 43, 53, 70, 77, 79, 80, 87, 95, 101, 102, 103, 104, 109, 110, 111, 113, 115, 117, 119, 123, 135, 139, 143, 179, 194, 210, 389, 443, 465, 512, 513, 514, 515, 526, 530, 531, 532, 540, 556, 563, 587, 601, 636, 993, 995, 2049, 4045, 6000, 6667.

הפרדה בין אתרים לוקאלית במחשב (local storage, web SQL)

כעת אנחנו מגיעים לישורת האחרונה. בדרך כלל אנחנו גולשים לאתרי אינטרנט בשרתים אחרים, וגם אם לוקאלית זה כשאנחנו מריצים שרת HTTP על המחשב שלנו. אולם, כיוון ש-Java script ו-HTML5 קיבלו יכולות כל כך חזקות, אין זה מן הנמנע שפשוט נלחץ "דאבל קליק" על קובץ HTML ראשי והוא יהווה אפליקציה עצמאית (מזכיר לכם Windows 8?).

זה לא נפוץ כ"כ היום, אבל קיים. נתקלתי בזה באפליקציה שפותחה באפסק (iNalyzer - מערכת לשליטה באפליקציות אייפון, ניתן איתה לראות את המבנה הלוגי של המסכים ולהפעיל אותם ישירות תוך כדי עקיפת לוגיקה שממומשת ב-GUI, ועוד ועוד). נתבקשתי לכתוב שם קטע קצר בקובץ HTML שמשתמש ב-Local storage, וגיליתי שקובץ אחד יכול לגשת למידע ששמר קובץ אחר.

אז בואו נראה מה יש לנו - אפליקציה (= קובץ HTML) שמיועדת לרוץ לוקאלית, שומרת מידע ב-Storages של HTML5 (כמו Local storage, Web SQL). כל קובץ HTML אחר שרץ (וכבר ראינו בהתחלה שלגרום למשתמש להוריד ולהריץ קובץ זה סיפור די קל) יכול לגשת למידע שמאוחסן שם. גם אם אין מידע רגיש עדיין לפי שמות המשתנים / טבלאות ניתן ללמוד שהיוזר משתמש באפליקציה פלונית, מידע שיכול לשמש אותנו בשביל מתקפות שונות (או בשביל לא לבצע מתקפות - כי נתקלנו בחוקר שאנחנו לא רוצים שיגלה אותנו).

דפדפן	תוצאה
פיירפוקס	Local storage - ניתן לקרוא את המידע שנשמר ע"י קבצים מאותה תיקייה Web SQL - המנגנון כלל לא נתמך בפיירפוקס
כרום	Local storage & Web SQL - ניתן לקרוא את המידע שנשמר ע"י קבצים מכל ההארדיסקים, אפי' מכוננים אחרים
אינטרנט אקספלורר	לא פגיע (רק הקובץ עצמו יכול לגשת למידע)

מעקב פיזי - GEO location

זה פשוט באג מעניין. לפחות מייקרוסופט לא יוכלו להגיד שזה פיצ'ר.
כשאתר מנסה לקבל מהדפדפן נתוני מיקום (GEO location), הדפדפן מבקש הרשאה מהמשתמש. למשתמש יש שתי אפשרויות בדרך כלל (תלוי בדפדפן ובגרסה), או אישור חד פעמי, או אישור רב פעמי. אישור חד פעמי זה אומר לדף עצמו עד שהוא נסגר, כך שהדף יכול לבקש אותו שוב ושוב. בכל מקרה, האישור אמור להיות פר אתר (Origin) ולא לזלוג.
מתברר ש-IE שומר את ההרשאה של בקשת נתוני GEO Location לקבצים לוקאלים (קרי: ה-Origin שלהם), ככל הנראה, לפי שם הקובץ ! לא לפי כונן / תיקיה / נתיב מלא.

התוצאה? אם המשתמש גלש לקובץ:

```
C:\big-deal-application\index.html
```

ואישר לדף בדפדפן לקבל נתוני GEO Location. ואני מדגיש - אישור חד פעמי (Allow once). לאחר מכן אם המשתמש גולש באותו טאב (= כרטיסיה) לדף:

```
D:\my_downloads\index.html
```

הדף יוכל לקבל את מיקום ה-GEO Location ללא בקשת אישור מהמשתמש. כי מבחינת הדפדפן זה אותו דף שמנסה לקבל שוב את הפרטים.

דפדפן	תוצאה
פיירפוקס	כל קובץ מבקש אישור לחוד
כרום	קובץ לוקאלי לא יכול לקבל נתוני GEO Location
אינטרנט אקספלורר	באותו טאב, כל קובץ בעל אותו השם יוכל לקבל את נתוני ה-GEO location ללא בקשת אישור נוספת, גם אם הקובץ נמצא בכלל בתיקיה/כונן אחרים

לסיכום

מה שראינו כאן, זה:

1. מערכות מתייחסות לקבצי HTML כאל קבצים לא מסוכנים, בעוד הצלחנו בפועל לקרוא קבצים אחרים מה-Hard disk.
2. דפדפנים לא מבצעים ניהול נכון של Origin לקבצים לוקאלים, וכך קובץ אחד יכול לגשת ל-Storages ששמרו קבצים אחרים.

מה שראינו כאן זה ממצאים שמצאתי בזמן כתיבת המאמר, בטוחני שיש פונקציות נוספות שהדפדפנים לא מטפלים בהם כראוי / לא מטפלים בהם באופן זהה ויש כאן כר למחקר נוסף.

כמשתמש אחשוב פעמיים לפני שאריץ קובץ HTML ולא אסמוך על כך ש"אם האתר היה רוצה, הוא כבר היה מריץ את הסקריפטים / 0-days שלו מ-Remote.

קישור לקובץ PoC למתקפות שהוזכרו במאמר:

http://www.digitalwhisper.co.il/files/Zines/0x2C/the_dangers_of_html_file.zip

על המחבר

ישראל חורז'בסקי, אחראי טכנולוגיות (Tech leader) בחברת AppSec Labs. בנוסף, יועץ אבטחה, פנטסטר, מרצה לבדיקות אבטחה ו-Secure coding ועוד. אשמח לקבל פידבקים / שאלות / הערות / הצעות במייל:

israel@appsec-labs.com

פרק מתוך הספר "קרב מוחות" (ההיסטוריה הזדונית של וירוסי המחשב)

מאת רן לוי

היום שבו מת האינטרנט

"לרוב אנחנו מדמיינים שמחשבים יגברו על האדם על ידי כך שיהיו הרבה יותר חכמים ממנו. יכול להיות שאנחנו נפסיד למשהו שהוא ממש ממש מטומטם."

מארק לודוויג, "הספר השחור הענק של וירוסים למחשב", 1995

כל מי שצפה בכמה סרטים הוליוודיים בימי חייו, מכיר את הקלישאה. אם בסרט כלשהו מופיע וירוס מחשבים, מי שכתב אותו נופל כמעט מיד לתוך אחד מתוך כמה טיפוסי "האקרים" קבועים: למשל, נער מתבגר ומרדן שניחן באינטליגנציה גבוהה, או מדען אקסצנטרי מבריק אך בעל מוסר מפוקפק. תהא הדמות אשר תהא, דבר אחד אינו משתנה: כותב הווירוס הוא גאון, אדם מוכשר משכמו ומעלה.

הקלישאה ההוליוודית הזו נשענת על הנחה סמויה נפוצה מאוד בקרב הציבור הרחב, שלפיה וירוס הוא דבר מורכב ומתוחכם ביותר, ולכן מי שמסוגל לכתוב וירוס למחשב הוא בהכרח אדם יוצא דופן. אחרי ככלות הכול, לא מדובר בתוכנה ככל התוכנות: וירוס צריך לדעת לפעול באופן עצמאי בתוך המחשב, והוא מסוגל גם ליצור לעצמו העתקים חדשים. צאצאים, אם תרצו.

נכון, יש וירוסים בעלי רמת תחכום יוצאת דופן - ואת חלקם אף נפגוש בספר זה. נכון גם שכדי לכתוב וירוסים למחשבים נדרשת השכלה מסוימת בתחום המחשב: ידע בסיסי בתכנות, לכל הפחות. אך גאונות אינה חלק מהעניין.

בפרק זה נספר את סיפורם של כמה מהווירוסים המוקדמים ביותר, ושל האנשים שכתבו אותם. חלקם היו אנשים חכמים ומוכשרים ללא ספק, אבל איש מהם לא היה "מיוחד במינו". כפי שנגלה מיד, הווירוסים לא באו לעולם כתגלית מרעישה של מדען מחשבים מבריק, אלא הומצאו שוב ושוב לאורך עשרות שנים, בידי אנשים שהגיעו מרקע מקצועי, חברתי והשכלתי מגוון ביותר.

משחק של הישרדות

שלושה מדענים צעירים ישבו במשרדו של אחד מהם ושוחחו. השנה הייתה 1961, והמקום - מעבדות בל, בקומפלקס גבעת מוריי שבניו ג'רזי.

מעבדות בל ריכזו אליהן, עוד מאז שנות השלושים של המאה העשרים, את מיטב המוחות המבריקים של המדע וההנדסה בארצות הברית. החידושים וההמצאות שזרמו משערי המעבדות שנה אחרי שנה - הטרנזיסטור והלייזר הם רק דוגמאות בודדות - קבעו את הכיוון שאליו יתקדם עולם הטכנולוגיה כולו לאורך עשורים רבים. בתחילת שנות השישים היה המחשוב תחום המחקר הלוהט ביותר - וזה גם היה עיסוקם של השלושה.

באותו היום גלגלו השלושה שיחה סתמית על הא ועל דא. כפי שקורה לעתים קרובות אצל אנשי מחשבים נלהבים, השיחה פנתה עד מהרה לנושא לא כל כך רציני אבל מושך מאוד: משחקי מחשב. למרות שגודלו של מחשב ממוצע תפס אז חדר שלם, עלה מיליוני דולרים והיה פשוט להפעלה רק אם אתה דוקטור - היו קיימים כבר כמה משחקי מחשב. איקס-עיגול ושחמט היו הנפוצים ביותר, ומשחקי יריות בסיסיים החלו להופיע בערך באותה התקופה. עם זאת, בכל המשחקים המחשב היה משתתף פסיבי בלבד: לא יותר מאשר דף נייר מתוחכם מאוד, אם תרצו.

אחד מהשלושה, ויקטור ויסוצקי (Vyssotsky), העלה רעיון שונה בתכלית: מה אם, תהה ויסוצקי בקול, נוכל לגרום למחשב לשחק את המשחק בעצמו? להיות באמת ובתמים חלק מהמשחק?

מחשבותיו גירו את סקרנותם של שני עמיתיו, דאגלס מקלרוי (McIlroy) ורוברט מוריס (Morris), והם האזינו בקשב רב לתוכנית ששטח בפניהם.

זירת ההתמודדות תהיה זיכרון המחשב, החלק שבו מאוחסן המידע. ה"גלדיאטורים" יהיו תוכנות שיילחמו ביניהן על השליטה בשטח אחסון זה: הן ינסו להשמיד זו את זו - או במילים אחרות, למחוק זו את זו מהזיכרון. הקרבות יתנהלו ללא מגע יד אדם: מרגע שסיים המתכנן האנושי לכתוב את התוכנה ולטעון אותה לתוך זיכרון המחשב, עליה להתמודד עם גורלה לבדה. אם התוכנה כתובה היטב, וליתר דיוק, כתובה טוב יותר מהתוכנה המתחרה, היא תשרוד ותזכה להילחם בסיבוב הבא מול מתמודד נוסף. ואם לא? אז לא.

כשסיים ויסוצקי לשטוח את רעיונותיו, כבר היו כל השלושה נלהבים וקצרי רוח לנסות את כוחם במשחק. ריח עז של תחרותיות עמד באוויר.

שלושתם היו אשפי תוכנה, ובשנים שלאחר מכן ישפיעו רבות על עולם הטכנולוגיה, כל אחד בדרכו: ויקטור ויסוצקי, הוגה המשחק, יהפוך לראש חטיבת מחקר במעבדות בל. דאגלס מקלרוי יהיה אחד מהחוקרים החשובים בתחום מערכות ההפעלה ויזכה לעיטורים ולפרסי הצטיינות רבים, ורוברט מוריס עתיד להיות

מדען בכיר בסוכנות הביטחון האמריקנית הסודית, ה-NSA. המשחק היה אמנם ניסוי מדעי שביקש לבחון רעיון חדשני בעולם המחשב, אך בעבורם הוא היה גם הזדמנות לראות מי מהם הוא המתמטיקאי והמתכנת הטוב ביותר - ומי יוכל לבנות את התוכנה האולטימטיבית שתמחץ את מתחריה.

בבוקר שלמחרת הגישו השלושה למנהליהם בקשה רשמית לקבל הקצבת זמן מחשב, שכן שעות מחשב היו באותם הימים עניין יקר. סביר להניח שהשם שנבחר לפרויקט גרם לכמה מנהלים לגרד בראשם בתימהון:

"דארווין: משחק של הישרדות ו(בשאיפה) אבולוציה."

הבקשה אושרה: המנהלים במעבדות בל הכירו את המדענים שלהם, וידעו שגם לרעיונות המשונים והבלתי שגרתיים ביותר יכולות להיות השלכות בלתי צפויות. ואכן, "דארווין" עמד להיות ציון דרך משמעותי בתולדות המחשב: הפעם הראשונה שבה תוכנה שכפלה את עצמה באופן עצמאי.

החוקים שקבעו השלושה למשחק היו בסיסיים למדי, והזכירו מאוד את המשחק המוכר "צוללות". נניח, לצורך ההסבר, שזיכרון המחשב בנוי מרצף של תאים מחוברים, כמו דף חשבון משובץ. כל תוכנה תופסת כמה וכמה תאים: תוכנה קטנה ופשוטה תתפוס כמה תאים בודדים, ותוכנה מתוחכמת יותר תשתרע על מרחב גדול יותר של משבצות.

המשחק מתנהל בתורות, ובכל תור יכולה תוכנה לנסות ולתקוף את יריבתה. "פעולה התקפית", בהקשר הזה, שקולה לשיגור טיל על משבצת כלשהי, ופגיעת הטיל מוחקת את כל מה שנמצא במשבצת באותו הרגע. אף אחת מהתוכנות היריבות אינה יודעת היכן נמצאת יריבתה, ולכן הירי נעשה "על עיוור" - ניחוש מושכל ותו לא. כמו ב"צוללות", לכל תוכנה מותר שיהיו כמה וכמה עותקים פעילים בו זמנית, מעין צי של צוללות זהות.

אולם להבדיל מ"צוללות", התוכנות נהנות מ"מטריית הגנה" בגודל של עשרים משבצות. המשמעות היא שאם תוכנה תופסת שלושים משבצות, למשל, עשרים מתוכן יהיו חסינות מפני פגיעה, והעשר הנותרות - אלה שמבצבצות מחוץ למטרייה - חשופות בצריח.

הבדל נוסף בין "דארווין" ל"צוללות" הוא שהתוכנות מסוגלות לנוע על לוח המשחק. התזוזה נעשית על ידי העתקה של המידע ממקום למקום, כמו מרבה רגליים שמרים ומוריד כל רגל בתורה.

ויקטור ויסוצקי פתח את המשחק ברגל ימין. התוכנה שכתב הייתה קטנה יותר מאלה של מקלרוי ומוריס, ולכן רק חלק קטן ממנה נותר חשוף ופגיע מחוץ למטריית ההגנה, והיא ניצחה את התוכנות האחרות בסבבים הראשונים.

אך לא לעולם חוסן, ודאג מקלרוי תפס את ההובלה עד מהרה. בשנים מאוחרות יותר, הסטודנטים של מקלרוי סיפרו עליו בהערצה שהוא "חולם בבינארי", וכישוריו המעולים כמתכנת באו לידי ביטוי גם ב"דארווין". הוא כתב תוכנה שהייתה מסוגלת לתקוף ולזוז, ועם זאת, תפסה רק 15 תאי זיכרון. מטריית ההגנה, נזכור, נפרשת על פני עשרים תאים, כך שהתוכנה של מקלרוי הייתה בלתי ניתנת להשמדה.

כדי להמשיך במשחק החליטו השלושה להקטין את מטריית ההגנה כך שהתוכנה הזעירה של מקלרוי תהיה בכל זאת חשופה חלקית. במשך כמה וכמה סיבובי משחק נותר הגלדיאטור הקטן בלתי מנוצח, עד שהצליחו מוריס וויסוצקי, בקושי רב, להקנות לתוכנות שלהם את התחכום הנדרש כדי להתמודד עמו.

ייתכן שבשלב זה נדמה היה שלמשחק לא יהיה סוף נראה לעין. על פניו, כל אחד מהמדענים הצעירים היה מוכשר דיו כדי לשכלל ולפתח את התוכנות שלו כדי שיתמודדו בהצלחה עם כל טריק שהעמיד מולם מישהו מהמתחרים האחרים. אך אז הגה רוברט מוריס רעיון מפתיע.

האתגר הגדול שניצב בפני כל תוכנה הוא לנסות ולנחש היכן נמצאים התאים החשופים והלא מוגנים של התוכנה היריבה. מכיוון שבכל סיבוב ניצבה התוכנה כנגד מתמודדת חדשה ובלתי מוכרת, כל שיכלה לעשות היה לנסות ולנחש את מיקום התאים החשופים ולהתפלל שהניחוש יהיה נכון.

אך רוברט מוריס הבין שההנחה שלפיה אין דרך לדעת מראש את זהותו של היריב הבא אינה נכונה תמיד. אוהדי כדורגל כבר יודעים ש"קבוצה מנצחת לא מחליפים": התוכנה שניצחה בסיבוב לא תשתנה, מכיוון שהיא כבר מוצלחת מספיק. התוכנות המפסידות, עם זאת, חייבות להשתפר כדי להתאים את עצמן ולשרוד בסיבוב הבא. זהו גם אחד מעקרונות היסוד של האבולוציה הביולוגית: הברירה הטבעית מכתיבה שרק יצורים שמסוגלים להתאים את עצמם לסביבתם ישרדו וישגשגו. כמו באבולוציה הביולוגית, המפתח להתאמה טובה יותר לסביבה הוא תורשה.

מוריס יצר תוכנה סתגלנית, אדפטיבית בלעז, תוכנה המסוגלת ללמוד את יריבתה תוך כדי הקרב ולהשתנות כדי לנצל את נקודת התורפה שלה.

בפעם הראשונה שהתמודדה מול תוכנה בלתי מוכרת, היא תקפה משבצות באופן אקראי, אך הקפידה לזכור מה הייתה תוצאת התקיפה. אם הבחירה הייתה שגויה והמשבצת הייתה ריקה, בסיבוב הבא ידעה לבחור משבצת אחרת. אם ההתקפה הייתה מוצלחת והתוכנה היריבה "הושמדה" (נמחקה מהזיכרון), התוכנה של מוריס הייתה יוצרת עותקים משוכפלים חדשים של עצמה, שהיו מצוידים בידע שאותו ירשו מאמם, דהיינו - איזו משבצת צריך לתקוף כדי להשיג ניצחון מהיר.

התוכנה של מוריס, אם כן, ידעה להתאים את עצמה ליריבותיה באופן דינמי, תוך כדי משחק, ללא התערבות יד אדם. מקלרוי וויסוצקי ניסו נואשות למצוא את הדרך להתמודד עם היצור החכם, אך ללא הועיל: לא משנה מה ומי השליכו השניים לזירת ההתמודדות, בתוך זמן קצר מצאו את עצמם מול צי של

לוחמים זריזים ויעילים, שידעו בדיוק היכן צריך לפגוע כדי לנצח. "דארווין" הגיע לכדי סיום, והמנצח לא היה יכול להיות ברור יותר.

עשרים שנה חלפו, וויסוצקי, מקלרוי ומוריס המשיכו כל אחד לדרכו. ספק אם הקדישו מחשבה מרובה למשחק הבלתי שגרתי שהגו.

אך "דארווין" לא נשכח לגמרי. משהו ברעיון שלפיו ניתן לכתוב תוכנות אשר יילחמו זו בזו בתוך זיכרון המחשב וייצרו לעצמן צאצאים חדשים, לכד את דמיונם של חובבי המחשב המעטים ששמעו עליו - רעיונות מרתקים ומוזרים כמו "דארווין" אינם מתים בקלות. הסיפור על אודות המשחק המרתק והמשונה שהמציאו שלושת המדענים במעבדות בל עבר מפה לאוזן, ובחלוף השנים הפך למעין "אגדה אורבנית", מיתולוגיה מודרנית שמקורותיה נעלמו בערפילי הזמן.

אגדה נוספת נולדה ב-1971.
לאורך תקופה ארוכה באמצע המאה העשרים, היה הצבא האמריקני גורם דומיננטי וחשוב מאין כמוהו בהתפתחותם של המחשבים. האמריקנים ניחשו, וצדקו בכך, כמובן, שלמחשב תהיה חשיבות עליונה בשדה הקרב העתידי, ועל כן השקיעו סכומי עתק במחקרים אקדמיים ובטכנולוגיות חדשניות לצורכי פיתוח מערכות נשק. תוצאות המחקרים והפיתוחים זלגו במקרים רבים גם אל השוק האזרחי, ואחד מאותם פרויקטים היה "ארפה-נט" (ARPANET) - טכנולוגיה שפותחה בשנות השישים ואפשרה לחבר מספר רב של מחשבים לרשת תקשורת גדולה. "ארפה-נט" הייתה מאוחר יותר לבסיס רשת האינטרנט המודרנית.

בוב תומאס (Thomas) היה מהנדס מחשבים בחברה אזרחית שסייעה לצבא האמריקני בפיתוח ה"ארפה-נט". הוא וחבריו לקבוצה עמלו על כתיבת תוכנת הדמיה של בקרת תעופה: תוכנית שתדמה את המסכים שרואים מולם פקחי טיסה במגדל הפיקוח.

מערכת ההדמיה הייתה מורכבת מכמה מחשבים אשר חלקו ביניהם את השליטה על המרחב האווירי: מחשב אחד, למשל, עקב אחרי מטוסים שטסו מעל וושינגטון הבירה, ומחשב אחר פיקח על השמים מעל העיר ניו יורק. כל עוד המטוסים בוושינטון הסכימו להישאר בסביבה של הבית הלבן, הכול היה בסדר ולא היו שום בעיות, אבל מדי פעם העלו הנוסעים במטוס את הדרישה "הבלתי סבירה" לטוס דווקא אל מחוץ לעיר, לניו יורק למשל. האתגר של תומאס היה למצוא דרך להעביר את כל הנתונים על המטוס (מהירותו, גובהו וכדומה) ממחשב אחד למחשב השני באופן חלק וללא תקלות, כדי שהמערכת תוכל להמשיך ולעקוב אחר מסלול הטיסה ללא הפסקה.

הפתרון שהגה תומאס היה לכתוב תוכנה קטנה ופשוטה יחסית, אשר תהיה מסוגלת לדלג באופן עצמאי ממחשב למחשב ברשת המחשבים של מערכת בקרת התעופה. הדילוג בין המחשבים היה, למעשה, יצירת עותק חדש וזהה של התוכנה במחשב היעד ומחיקה של העותק במחשב המקור. בכך הדגים

תומאס את היכולת העקרונית להעביר מידע בין מחשבים, כמו נתוני הטיסה של מטוס העובר מהמרחב האווירי של וושינגטון לשמי ניו יורק.

התוכנה של תומאס הייתה אב-טיפוס ראשוני בלבד, ולכן הוא הרשה לעצמו להשתעשע קצת על חשבונם של חבריו לעבודה. בכל מחשב שאליו דילגה התוכנה, הופיעה על המסך ההודעה הבאה:

"אני הקריפר, תפוס אותי אם תוכל!"

"קריפר"(Creeper) פירושו "מתגנב" בתרגום חופשי. ובאמת, לא היה פשוט לתפוס את הקריפר. לפני שהספיק המשתמש המופתע אפילו למצמץ מול ההודעה שעל המסך, כבר היה הקריפר אורז את המזוודות, אפשר לומר, ומעביר את עצמו למחשב הבא ברשת.

אחד מעמיתיו של תומאס היה מתכנת בשם ריי תומלינסון (Tomlinson), והוא החליט להיענות לאתגר. היו אלה חדשות רעות מאוד בעבור הקריפר. ריי תומלינסון היה מתכנת מעולה. למעשה, הוא האדם שהמציא את הדואר האלקטרוני. אם מישהו היה יכול לתפוס את הקריפר, ריי היה האיש.

תומלינסון כתב תוכנה אחרת שהייתה מסוגלת גם היא לעבור ממחשב למחשב ברשת וקרא לה "ריפר" (Reaper, "מלאך המוות"). כמו תום וג'רי בסרטים המצוירים, ה"קריפר" וה"ריפר" דילגו ממחשב למחשב עד שנפגשו, ואז ה"ריפר" היה מוחק את ה"קריפר". מלאך המוות, כמו תמיד, אמר את המילה האחרונה.

לבוב תומאס לא היו כל ספקות לגבי סיכויי ההצלחה של ה"קריפר" שלו כנגד ה"ריפר" של תומלינסון:
"אף פעם לא הקדשתי זמן לשנות את ה'קריפר' כדי שיגן על עצמו, או לאפשר לו להתחמק. גם אם הייתי עושה כן, אני לא בטוח שהוא היה מסוגל לברוח מה'ריפר'. ריי היה [מתכנת] מוכשר מאוד, כנראה המתכנת הטוב ביותר שאי פעם הכרתי."
ריאיון עם המחבר

זוחל, אוכל מילים ומתפשט

ה"קריפר", מסתבר, לא מת לחינם. כמו "דארווין", היו בו כל האלמנטים הדרושים כדי להסעיר את דמיונם של מתכנתים רבים: שכפול עצמי, יכולת עצמאית "לרדוף" אחר מטרה כלשהי וכו'. הסיפור על ה"ריפר" וה"קריפר" עבר מפה לאוזן והיה נדון אולי להישכח ולהיעלם ברבות הימים, אלמלא הגיע לאוזניו של המתמטיקאי הקנדי אלכסנדר דודני (Dewdney).

דודני, בנוסף לעיסוקיו המתמטיים, היה גם חובב מחשבים וסופר. בשנות השמונים והתשעים כתב טור קבוע ב"סיינטיפיק אמריקן", אחד המגזינים המדעיים החשובים והנפוצים ביותר לקהל הרחב. הטור עסק בחידות ובאתגרי מחשבה בתחומי המתמטיקה והמחשב.

כששמע דודני על ה"ריפר" וה"קריפר", התעוררה סקרנותו. הוא ניסה לברר פרטים נוספים על הסיפור, אך העלה חרס בידיו, אולי מכיוון שחלק גדול מהמידע על פרויקט "ארפה-נט" היה עדיין חסוי באותה התקופה. דודני הגיע למסקנה (השגויה) שהסיפור הוא מיתוס שאין לו אחיזה במציאות, אך גישושיו הביאו אותו לגלות את הסיפור על "דארווין" של ויסוצקי, מקלרוי ומוריס. הוא קרא מכתב ששלח אלמוני למגזין מחשבים כלשהו בשנת 1972, ובו תוארו "דארווין" וחוקיו בפירוט רב, וגם הוא, כמו רבים לפניו, נשבה בקסמיו של רעיון השכפול העצמי ו"קרבות התוכנה".

ב-1984 כתב דודני טור ב"סיינטיפיק אמריקן", שבו הציע להחזיר לחיים את "דארווין", בתוספת כמה שינויים בחוקי המשחק אשר יעשו אותו פשוט יותר. באופן זה, קיווה דודני, יוכלו גם חובבי מחשב רגילים, ולא רק מדעני מחשב מנוסים, להשתתף בו וליהנות מהאתגר. הוא כינה את הגרסה שלו ל"דארווין" בשם "מלחמת הליבה" (Core War).

"סיינטפיק אמריקן", כאמור, זוכה לתפוצה אדירה, והמאמר של דודני נגע בנקודה רגישה אצל קוראים רבים.

המושג "וירוס מחשבים" במובן העכשווי שלו של תוכנה זדונית המתפשטת ממחשב למחשב וגורמת לנזק, הופיע לראשונה רק בתחילת שנות השמונים. בפרק הבא נדון בנושא זה באופן מעמיק יותר, אבל חשוב להבין שבאותם ימים ראשונים, רק משתמשי מחשב בודדים נתקלו בווירוס או נדבקו בו בעצמם. "וירוס" היה יצור מיתולוגי-למחצה, כמו המפלצת מלוך-נס: כולם ידעו שהוא מפחיד, אבל אף אחד לא ראה אותו במו עיניו.

אך כל מי שקרא את המאמר של דודני לא יכול היה שלא לראות את הקשר הברור שבין התוכנות הלוחמות של "דארווין" וה"ריפר" וה"קריפר" - ובין הווירוסים שהחלו אז להטיל את חיתתם על משתמשי המחשב. הראשונים היו אבות-הטיפוס של האחרונים, וניחנו באותה תכונה בסיסית שכל וירוס מחשבים ניחן בה: היכולת לשכפול עצמי.

דודני קיבל מכתבים מקוראים רבים שסיפרו לו על ניסיונותיהם הפרטיים ליצור תוכנות המסוגלות לשכפול עצמי. אחד מהם, פרדריק סטאל (Stahl), תיאר כיצד יצר עולם וירטואלי בזיכרון המחשב שבו חיו והתרבו יצורים, ממש כמו חיידקים בצלחת פטרי:

"היצור תוכנת לזחול ב[מרחבי] היקום שלו, כשהוא אוכל מזון (מילים) ויוצר העתקים חדשים של עצמו כשצבר מספיק מזון. [...] יצרתי תוכנה שעקבה אחרי היצורים ובדקה מי מהם עדיין חי... כיניתי אותה 'ידו השמאלית של אלוהים'."

סטאל אפילו ניסה לחקות את מנגנון האבולוציה הטבעית: הוא גרם לכך שבכל שכפול עצמי של יצור, ייפלו שגיאות העתקה זעירות, מוטציות, כך שהיצור החדש יהיה שונה במעט מאבותיו. לעתים יצרו המוטציות יצורים שהיו טובים במעט מקודמיהם, והם התחרו בהם בהצלחה על מזון - המילים שביקום הזיכרון.

זוג קוראים אחר, איטלקים, סיפרו לדודני שמצאו דרך לכתוב וירוס שהיה מסוגל להתפשט ממחשב למחשב, ואפילו להשמיד בקלות את כל המידע השמור על המחשב הנגוע. הם היו משוכנעים שגילו דרך איומה ונוראה להזיק למחשבים שאיש לא גילה לפניהם, נטשו את תוכניתם בחלחלה, ונשבעו שלא לדבר על כך עם איש לעולם, שמא יגלה מישהו קצת פחות מוסרי מהם את השיטה. רק כשקראו את המאמר של דודני, הבינו שהם לא היחידים שהעלו במוחם את הרעיון.

דודני, שהיה מופתע מעוצמת התגובה של קוראיו, כתב את הדברים הבאים במרס 1985:
"כשכתבתי את הטור אודות 'מלחמת הליבה' במאי האחרון, לא היה לי מושג עד כמה רציני הנושא שהעליתי. התיאורים שהבאתי על תוכנות הנעות במרחבי הזיכרון ומנסות להשמיד זו את זו, השפיעו עמוקות [על הקוראים]. לדבריהם של קוראים רבים, שאת סיפוריהם אביא כאן, יש אינספור דוגמאות לתולעים, וירוסים ועוד יצורי תוכנה אחרים, החיים בכל סביבת מחשוב שניתן לדמיין. כמה מן האפשרויות הן כה מחרידות, עד שאני מהסס אם להעלותן על הכתב בכלל."

דוגמה נוספת לווירוס מוקדם, שככל הנראה לא הייתה מוכרת לדודני בזמן שכתב את מאמריו, היא של וירוס שכונה "חיה" ("Animal").

בתחילת שנת 1974, עבד מתכנת בשם ג'ון ווקר (Walker) בחברת הנדסה גדולה בארצות הברית. החברה רכשה, שנים ספורות לפני כן, כמה מחשבים חזקים ומהירים מסדרת UNIVAC. במהלך עבודתו נתקל ווקר בכמה גרסאות ממוחשבות של משחק הילדים הנפוץ "עשרים שאלות". במשחק זה מטרתו של המחשב הייתה לנחש, באמצעות לא יותר מעשרים שאלות, על איזו חיה חושב המשתמש. המחשב היה שואל שאלות בסגנון "האם לחיה יש ארבע רגליים?" והמשתמש היה מאשר או מכחיש, עד שהמחשב היה מצליח (בשאיפה) לנחש את החיה המתאימה.

ברוב המקרים הצלחתו של המחשב הייתה מוגבלת למדי מכיוון שהיו המון חיות, והמחשב גם לא היה חכם במיוחד. היום מנסים לפתור את הבעיה באמצעות הכחדה המונית של זני בעלי חיים, אבל בתקופה ההיא ניסה ווקר פתרונות סופניים פחות. הוא החליט לכתוב מחדש את המשחק, כך שהפעם המחשב יוכל ללמוד משגיאותיו ממשחק למשחק. אם, בתום עשרים השאלות, לא הצליח המחשב לנחש נכונה את החיה, ביקש מהמשתמש לגלות לו את התשובה ולספק לו פיסת מידע אחת הנכונה לגבי החיה הנבחרת, אבל לא לגבי חיה אחרת. למשל, רק סוס יכול להתחרות במירוץ סוסים.

ווקר כינה את המשחק שלו "חיה", וזה הפך למיני-להיט בקרב עובדי ההייטק המשועממים של אמצע שנות השבעים. כה מוצלחת הייתה ה"חיה", עד שג'ון ווקר קיבל בקשות רבות מאנשים בחברות אחרות לשלוח גם להם עותק. בהיעדר דואר אלקטרוני, משלוח של תוכנה היה מסורבל מאוד: ווקר היה צריך להעתיק את התוכנה על גלגל של סרט מגנטי, לארוז אותו בזהירות ואז לשלוח בדואר רגיל. כמו שווקר הגדיר את זה בעצמו - "באסה".

כדי להקל על עצמו, החליט ווקר לפנות לשיטה שניתן לכנותה, בהיעדר מינוח מתאים יותר, "שיווק ויראלי". במקום שהוא עצמו ישלח את המשחק אל המשתמשים, ה"חיה" תמצא את דרכה אליהם בעצמה. הוא הכניס שינוי שגרם למשחק ליצור עותקים של עצמו ברחבי זיכרון המחשב - אבל לא באקראי, כי אם במקומות מסוימים מאוד.

מידע במחשב בכלל, וגם במחשבי היוניבק של ווקר בפרט, מאורגן לרוב במבנה היררכי של תיקיות. אפשר להבין את מבנה התיקיות אם נדמה אותו לעץ: הקבצים מסודרים בתוך תיקייה כמו עלים על ענף בודד, והענפים מסתעפים במבנה היררכי עד הגזע, או תיקיית העל. במקרה של היוניבק, לכל משתמש הייתה תיקייה, או מספר תיקיות מוגבל, ואליה הוא היה רשאי לגשת. שאר התיקיות, על פי שיקול דעתו של האחראי על המחשב, היו חסומות בפניו.

בזמן שהמשתמש היה משחק "עשרים שאלות" מול המחשב, ה"חיה" הייתה פועלת בחשאי ברקע ובודקת לאילו תיקיות יש לו הרשאות גישה. בכל אחת מתיקיות אלו הייתה ה"חיה" שותלת עותק חדש של עצמה. לעתים קרובות הייתה תיקייה אחת משותפת לשני משתמשים ויותר. המשתמש החדש היה בודק את התיקיות שלו ומגלה קובץ חדש ולא מוכר בשם "חיה". הסקרנות האנושית היא דבר נפלא ואמין - כמעט כולם מיהרו לבדוק מה פשר הקובץ המסתורי, וכשגילו שמדובר במשחק, גם שיחקו כמה סיבובים מול המחשב, מתפעלים מחוכמתו ומיכולתו ללמוד משגיאות העבר.

ושוב, בזמן המשחק, הייתה ה"חיה" בודקת את ההרשאות של המשתמש החדש, ומעתיקה את עצמה לתיקיות חדשות, וכן הלאה. באופן זה הייתה ה"חיה" עוברת מענף לענף, עד שלבסוף הגיעה אל הגזע - אל תיקיית העל. תיקייה זו הייתה שייכת לאחראי על המחשב ("האדמיניסטרטור"), ולו הייתה גישה לכל המחשב כולו, כלומר עכשיו יכולה הייתה ה"חיה" להעתיק את עצמה לכל תיקייה ותיקייה בזיכרון המחשב. הכיבוש הושלם בהצלחה.

בתוך שבוע קיבל ווקר דיווח על עותקים של הווירוס שצצו בצדה השני של היבשת הצפון-אמריקנית: ה"חיה" "תפסה טרמפ" על קלטות מגנטיות שהעובדים העבירו ביניהם בדואר. קשה לומר בוודאות מוחלטת, אבל קיים סיכוי סביר שבתוך חודש כל מחשבי היוניבק בארצות הברית (ואולי בעולם בכלל) הכילו עותקים של ה"חיה". זהו פרק זמן קצר עד להדהים, בהתחשב בעובדה שרובה המוחלט של התקשורת בין המחשבים התנהל באמצעות דואר רגיל.

תכונה מעניינת של מחשבי היוניבק באותם הימים הייתה זו: אם משתמש ניגש לתיקיית מידע, המחשב היה מריץ כברירת מחדל את התוכנה האחרונה שנוספה לאותה התיקייה (דבר זה נועד, בוודאי, כדי לחסוך בזמן). דמיינו לעצמכם את המצב הבא: מהנדס מתיישב אל מול המחשב, אולי כדי להפעיל תוכנת בדיקה או לפתור נוסחה מסובכת. הוא ניגש לתיקייה שלו. לפתע מופיעה על המסך השורה הבאה:

"תחשוב על חיה."

מעניין היה לראות את תגובתו באותו הרגע.

המשתמשים במחשבי היוניבק לא ראו ב"חיה" איום כלשהו אלא שעשוע בלתי מזיק בלבד. ג'ון ווקר, עם זאת, הבין את המשמעות האמיתית של ליקוי האבטחה הזה, כשנוכח במו עיניו עד כמה מהירה הייתה התפשטותה של ה"חיה": לו הייתה לווקר כוונת זדון, שינוי פעוט יחסית בקוד ההפעלה של התוכנית היה מאפשר לו למחוק את כל המידע על המחשב באבחה אחת.

בשלב כלשהו החל ווקר לחשוב על רעיונות שיאפשרו לו למחוק את ה"חיה" ולעצור את התפשטותה. אחד מהפתרונות האפשריים היה ליצור וירוס נגדי, כמו ה"ריפר" של ריי תומלינסון, אבל למזלו של ווקר לא היה בכך צורך. כשנה לאחר תחילת הסיפור הופיע עדכון למחשב, שבאופן מקרי לחלוטין מנע מה"חיה" לקבל את המידע על הרשאות הגישה. המשחק עצמו התנהל כרגיל, אבל כעת כבר לא יכלה החיה להתרבות ולשכפל את עצמה מתיקייה לתיקייה. עדכון זה סימן את סופה של ה"חיה", והיא עברה להיות חלק מדפי ההיסטוריה.

נראה, אם כן, שיצירת "חיים מלאכותיים", לפחות ברמה הפשוטה ביותר, אינה כה מסובכת כפי שניתן היה לחשוב בתחילה. למעשה, היא כל כך פשוטה, עד שאפילו נער מתבגר היה יכול להמציא אותה בעצמו.

היא תיכנס, היא תסתנן, היא תידבק

המחצית השנייה של שנות השבעים וראשית שנות השמונים היו תחילתו של עידן זהב עבור עולם המחשבים. ממוסך קטן בחוף המערבי של ארצות הברית יצאה הבשורה: סטיב ג'ובס וסטיב ווזניאק, שני נערי פלא, יצרו מחשב קטן, ביתי ואישי. לא עוד מחשבים גדולים כמו ארון ויקרים כמו בית - ה"אפל" (Apple) היה "מיקרו מחשב", פשוט, נוח להפעלה ובעיקר זול.

הגרסה השנייה של מחשב האפל, אפל-2, הייתה הצלחה מעוררת קנאה גם בקנה מידה עכשווי. מיליוני יחידות נחטפו מהמדפים בארצות הברית ובעולם כולו. עסקים קטנים רצו מחשבי הנהלת חשבונות, וילדים קטנים רצו לשחק במשחקי חלליות מול המסך. האפל-2 התאים לכולם.

ב-1982, ריץ' סקרנטה (Skrenta) היה נער מתבגר בן 15 בפיטסבורג, ארה"ב. כמו רבים מחבריו, הוא אהב להשתעשע עם מחשב האפל-2 שלו, ובעיקר לשחק במשחקים הרבים שנכתבו עבור המחשב.
ריץ' הסקרן חקר את נבכי המחשב לעומקם, ונעזר בעובדה שהאפל-2 היה נגיש ונוח לתפעול בצורה בלתי רגילה ביחס למחשבים אחרים באותם הימים. בעוד שיצרני המחשבים המסורתיים נטו לשמור בסוד את כל הפרטים הטכניים על המחשבים שלהם, ווזניאק וג'ובס עשו בדיוק את ההפך: יחד עם המחשב קיבלו המשתמשים גם חוברות הסבר מפורטות על כל חלקיק ותא זיכרון במחשב. מי שרצה והיה מוכן להשקיע זמן בלימוד החוברות, יכול היה לכתוב תוכנות ומשחקים לאפל-2 בקלות רבה.

ריץ' השקיע את המאמץ הנחוץ, ולמד בכוחות עצמו לתכנת את האפל-2. בתור בדיחה, הוא כתב תוכנה קטנה שגרמה לכיתוב הבא להופיע על המסך:

"אלק קלונר ['הדישון המשכפל', Elk Cloner]: התוכנה עם האישיות

היא תיכנס לך לדיסקים
היא תסתנן לך לתוך הצ'יפים
כן, זהו הקלונר!

היא תידבק אליך כמו דבק
היא תשנה לך את הזיכרון
שלחו את הקלונר!"

הקלונר היה, כמובן, מתיחה נחמדה ולא מזיקה. הבעיה של ריץ' הייתה לגרום לחבריו להחדיר את התוכנה אל מחשבי האפל-2 שלהם: כולם מסביבו ידעו שריץ' אוהב מתיחות ולא מיהרו לתת לו גישה חופשית למחשבים שלהם. על כן הוא ישב וחשב על דרך להחדיר את הקלונר למחשבים אחרים מבלי לעורר חשד. ברגע של הארה הבין לפתע איזה שינוי יצטרך לעשות בתוכנה, כדי לתת לה את יכולת השכפול העצמי הנדרשת להתפשטות ויראלית:

> "הרגע של ה'אה הה!' היה כשהבנתי שאני יכול בעיקרון לגרום לתוכנה שלי לנוע מעצמה. אני יכול להעניק לה כוח מניע, על ידי כך שאגרום לה להתחבא... וואו! זה יהיה קול!"

העברת מידע בין מחשבי אפל-2 נעשתה באמצעות דיסקטים: ריבועי פלסטיק קשיחים ודקים, שהכילו בתוכם משטח מגנטי עדין שעליו נשמרו הקבצים. הקלונר ידע להסוות את עצמו כך שלא הופיע ברשימת הקבצים, ולהעתיק את עצמו מהדיסקט אל המחשב ללא התערבות המשתמש. מכאן ואילך, בכל פעם שהוכנס דיסקט חדש לכונן המחשב, קלונר היה יוצר עליו עותק של עצמו. רק אחרי חמישים הדלקות של

המחשב הופיעה ההודעה על המסך, והמשתמש ידע שמשהו לא בסדר במחשב, אבל בזמן שחלף סביר להניח שהקלונר כבר הדביק כמות נכבדה למדי של דיסקטים חדשים.

הקלונר עשה עבודה מצוינת, הרבה "בזכות" ההרגלים הרעים של חבריו של ריץ', שנהגו להחליף ביניהם דיסקטים ועליהם עותקים פיראטיים של משחקים. בזמן שיא הסתנן הקלונר למחשביהם של כמעט כל מי שהכיר ריץ' (כולל, באופן מפתיע, מחשבו של המורה שלו למתמטיקה!), ומשם למספר בלתי ידוע, אך ללא ספק גדול מאוד, של מחשבים בעולם כולו. הקלונר נחשב לווירוס המחשבים הראשון שהצליח לצאת מגבולות המעבדה או חדר העבודה שבו נוצר, ולהדביק כמות גדולה באמת של מחשבים.

כשהתבגר, המשיך ריץ' לעשות חיל בעולם המחשבים. הוא עסק בפיתוח משחקי מחשב, כתב את הגרסה הראשונה של תוכנה שהפכה (לאחר כמה גלגולים) לאתר האנציקלופדיה הפופולרי "ויקיפדיה", ואף ייסד חברה משלו. אבל לגורל יש רצון משלו: דווקא התוכנה הקטנה והפשוטה ביותר, הווירוס הראשון אי פעם עבור המחשב האישי, היא זו שתיזכר לו לתמיד. כפי שכותב ריץ' עצמו:

> "כתבתי הרבה דברים עבור האפל-2... [אבל] הקוד הכי מטופש שאי פעם יצרתי הוא זה שעורר הכי הרבה עניין, וממשיך לעשות כן עד עצם היום הזה."

מתקפת מוריס

כפי שנוכחנו לדעת, כתיבת וירוסים אינה מורכבת ומסובכת כפי שהיא נדמית ממבט ראשון, והרעיון הבסיסי של תוכנה בעלת היכולת לשכפול עצמי והתפשטות ממחשב למחשב הומצא שוב ושוב פעמים רבות.

אך מה בדבר הנזק שגורם הווירוס? אחרי ככלות הכול, מחשבים מתוכננים ונבנים על ידי אנשי מקצוע מומחים בתחומם. סביר להניח שהגנה על המידע השמור במחשב היא בראש סדר העדיפויות שלהם, שכן מהו מחשב אם לא מכונה לעיבוד מידע? גם אם כל אחד (כמעט) יכול לכתוב וירוס, האם לא נדרשת רמה גבוהה יותר של מיומנות וכישרון כדי ליצור וירוס המסוגל לגרום לנזק אמיתי למערכות מחשב?

הסיפור הבא ימחיש את התשובה לשאלה זו. בנוסף, הוא גם יוכיח לנו שלעתים, לגורל יש חוש הומור ציני ואירוני במיוחד.

רוברט טאפאן מוריס (Morris) היה סקרן.
השנה הייתה 1988, ומוריס בן ה-23 היה סטודנט לתואר שני במדעי המחשב באוניברסיטת קורנל שבארצות הברית. כמו רבים מבני דורו, הוא הוקסם מהפוטנציאל הטמון באינטרנט וביקש לחקור ולגלות את העולם החדש הזה. הוא היה מרותק בעיקר לשאלה עד כמה הוא חסין בפני השפעתם של וירוסים.

מוריס בילה ימים ולילות בחיפוש אחר חולשות שונות של מערכות המחשב, והצליח לאתר שגיאה, "באג", בתוכנה שהייתה נפוצה מאוד והותקנה על מחשבים רבים. הוא פנה לאחראי המחשבים באוניברסיטה שלו ודיווח לו שהבאג שגילה עשוי לאפשר לתוקף חיצוני להשתלט על המחשב דרך הרשת ולעשות בו ככל העולה על רוחו. אחראי המחשבים לא לקח את הדיווח ברצינות, אך מוריס לא ויתר. כדי לבחון את ההשערה שלו לגבי חומרתו של הבאג, הוא כתב תוכנה שתכליתה הייתה להתפשט ולהשתלט על המספר הרב ביותר של מחשבים ברשת, מהר ככל האפשר: זהו סוג של וירוס שמוכר לנו כיום בשם "תולעת מחשבים". תולעת משתמשת בקווי התקשורת שבין המחשבים כדי להתפשט, בדומה לנגיפים ביולוגיים שנעזרים בכלי הדם של הגוף כדי להתפשט בתוכו.

מוריס תכנן את התולעת שלו כך שתנצל את הבאג המדובר - ועוד שלוש חולשות אבטחה נוספות שגילה - כדי לחדור למחשב, ליצור עותקים חדשים של עצמה ולהפיץ אותם לכל המחשבים שהיו מחוברים אליו. כדי לוודא שכל מחשב יידבק בעותק אחד בלבד של התולעת, הוא שילב בתוכה מנגנון בדיקה פשוט: לפני כל ניסיון הדבקה התולעת הייתה בודקת אם המחשב הקורבן כבר מכיל בתוכו עותק קודם. אם התשובה הייתה חיובית, תהליך ההדבקה הופסק והתולעת הייתה פונה לחפש קורבן אחר.

לא הייתה כל כוונת זדון בתוכניתו של מוריס; הוא לא ביקש להזיק למחשבים, אלא רק לבחון את פוטנציאל ההתפשטות של התולעת. אף על פי כן, הוא שילב בתוכה מנגנון פשוט להגנה עצמית. הוא ידע שאם התולעת תתגלה לפני שתספיק להתפשט בכל רחבי הרשת, הדרך הפשוטה ביותר להתגבר עליה תהיה לעצור את תהליך ההדבקה בשלב שבו התולעת בודקת אם המחשב כבר מכיל עותק קודם שלה. אם המחשב הקורבן ישיב ב"כן" כל הזמן, גם אם הוא נקי מתולעים, הוא יהיה חסין מפני השתלטות.
כדי למנוע מצב כזה, תכנן מוריס את התולעת כך שבפעם השביעית שתקבל תשובת "כן", תהליך ההדבקה ימשיך בכל זאת. בסופו של דבר, אמר מוריס לעצמו, רשת האינטרנט כל כך גדולה, מה הסיכוי שאותו המחשב ייתקל ביותר מתולעת אחת או שתיים?

לרוע מזלו, מוריס שגה כאן שגיאה חמורה: הנחת היסוד שלו לגבי הסיכוי הקלוש שאותו המחשב ייתקל בתולעת כמה פעמים, הייתה שגויה לחלוטין. האינטרנט אמנם רחב ידיים, אבל הקשרים בין המחשבים הם ענפים ומרובים יותר מששיער, וכל מחשב הותקף עשרות ומאות פעמים. מכיוון שכל ניסיון הדבקה שביעי היה בהכרח מוצלח, הצטברו בתוך זמן קצר עשרות תולעים פעילות בתוך כל מחשב נגוע, והתוצאות היו הרסניות.

מוריס ידע היטב שמה שהוא מתכוון לעשות כנראה אינו חוקי. כדי להסתיר את עובדת היותו סטודנט של קורנל, מצא דרך להתחבר לאינטרנט דרך אוניברסיטת MIT, ששימשה כמעין תחנת ממסר. בשניים בנובמבר, יום רביעי, בשעה שש בערב, שחרר מוריס את התולעת לרשת דרך החיבור ל-MIT. מוריס קם מכיסאו, מרוצה שהצליח להוציא את הפרויקט שלו לפועל, והלך לאכול כשהוא שמח וטוב לב.

לו ידע איזו דרמה התחוללה בחוץ באותו הזמן, הוא כנראה לא היה מסוגל ליהנות מארוחת הערב שלו. בשעה תשע ועשרים בערב, קצת יותר משלוש שעות לאחר שהפיץ מוריס את התולעת שלו, קיבל האחראי על המחשבים בפקולטה למדעי המחשב של אוניברסיטת יוטה תלונות מכמה אנשים שדיווחו כי המחשב המרכזי של הפקולטה עובד לאט מהרגיל. האחראי (אדמיניסטרטור, כפי שהוא מכונה בעגה המקצועית) ניגש לבדוק את פשר העניין. בדיקה קצרה גילתה לו מיד את הסיבה לאיטיות.

כמו מחשבים רבים באותם הימים, גם המחשב המרכזי של הפקולטה למדעי המחשב שירת כמה וכמה משתמשים בו זמנית, אשר כל אחד מהם ישב מול מסוף מחשב משלו. כמו מלצר העובר משולחן לשולחן במסעדה, המחשב המרכזי הקדיש תשומת לב לכל משתמש בתורו, ומהירות העיבוד הגבוהה שלו יצרה את האשליה כאילו כולם עובדים במקביל זה לזה. אך כפי שגילה האדמיניסטרטור, כמות חריגה של תוכנות בלתי מוכרות פעלה כרגע על המחשב. משמעות ריבוי התוכנות היא שעבר זמן ארוך מדי עד שה"מלצר הווירטואלי" השלים סבב בין השולחנות וחזר אל הלקוח הראשון, ולכן האשליה התפוגגה והמשתמשים החלו לחוש באיטיות התגובה של המחשב.

התוכנות הבלתי מוכרות היו, כמובן, עותקים זהים של התולעת של מוריס, שהדביקו את אותו המחשב כמה וכמה פעמים.

כל עותק של התולעת היה תוכנה עצמאית אשר דרשה תשומת לב מהמעבד.

העומס המוטל על המחשב, כך הבחין האדמיניסטרטור של אוניברסיטת יוטה, היה גבוה פי-שלושה מהרגיל. בזמן שלקח לו לבדוק את העניין, גבר העומס פי-שמונה: נדמה היה שכמות התוכנות הבלתי מוכרות עולה מדקה לדקה. האדמיניסטרטור לא נתקל בתופעה שכזו מימיו. בצעד של ייאוש הוא כיבה את המחשב והדליק אותו מחדש. עשרים דקות מרגע ההדלקה כבר הייתה רמת העומס גבוהה פי-עשרה, והאיטיות הפכה את המחשב לבלתי שמיש כמעט.

שוב כיבוי, ושוב הדלקה. הפעם רמת העומס כבר זינקה לפי שנים-עשר.

אוניברסיטת יוטה לא הייתה היחידה שחשה את נחת זרועה של התולעת.
בסביבות חצות הופיעה ההודעה הבאה באחת מקבוצות הדיון באינטרנט. קבוצות הדיון היו דומות למדי לפורומים המוכרים לנו היום, מעין לוחות מודעות וירטואליים שכל אחד יכול לכתוב בהם ולקרוא את תוכנם, והיו מקום המפגש המקובל של מרבית קהילת מהנדסי המחשבים. ההודעה הראשונה בלוח המודעות שהצביעה על כך שמשהו אינו כשורה הייתה המסר הבא שנשלח על ידי פיטר יי (Peter Yee), עובד סוכנות החלל האמריקנית, נאס"א. ההודעה הופיעה על לוח המודעות בסביבות חצות, כשש שעות מרגע התפרצות המגיפה.

"אנחנו כרגע תחת התקפה של **וירוס אינטרנט** [ההדגשה במקור]. הוא פגע באוני' ברקלי, אוני' סן דייגו, מכון מחקר לורנס ליברמור, בסטנפורד ובנאס"א-איימס..."

פיטר הגדיר את ההתרחשות כ"התקפה" מכיוון שהבין שרק משהו חריג באמת יכול להפיל בבת אחת את המחשבים של מוסדות מחקר גדולים וחזקים כמו ברקלי, סטנפורד ונאס"א. בהמשך ההודעה נתן פיטר כמה רעיונות ועצות איך לנסות ולהתמודד עם האיום הפתאומי הזה, אבל הזמן הקצר שחלף מאז שנתגלתה ההתקפה לא אפשר לו לנתח את התולעת ברצינות, והעצות שנתן לא מנעו את התפשטות התולעת (אם כי, למען ההגינות, הן בהחלט היו בכיוון הנכון).

הסצנה שהתרחשה באוניברסיטת יוטה חזרה על עצמה בכל רחבי הרשת. מהנדסים ואדמיניסטרטורים באלפי אוניברסיטאות, מכוני מחקר, בסיסים צבאיים ומתקנים ממשלתיים נאבקו במחשבים שקרסו תחת גל אחרי גל של תולעים. טלפונים בהולים צלצלו באישון לילה בבתים רבים, מזעיקים מומחים ממיטותיהם כדי לנסות להתמודד עם האויב הבלתי נראה.

בדיקה ראשונית העלתה שאחת מצורות החדירה העיקריות של התולעת הייתה דרך מנגנון הדואר האלקטרוני. על כן, התגובה האינסטינקטיבית של רבים מהאדמיניסטרטורים בארגונים השונים הייתה פשוט לכבות את הדואר האלקטרוני לגמרי. בדיעבד, מסתבר, פעולה זו הייתה, בעוכריהם: מוריס, כזכור, תכנן את התולעת כך שתנצל ארבע חולשות במערכות המחשבים, והחולשה בדואר האלקטרוני הייתה רק אחת מהן. ניתוק הדואר לא מנע מהתולעת להתפשט, אבל הוא מנע זרימה יעילה של מידע בין המומחים, ולמעשה הקשה על ההתמודדות מול התולעת.

בינתיים, בקורנל, התחילו השמועות על ההמולה והבלגן ברשת האינטרנט להגיע גם אל רוברט מוריס. מתיאור הנזקים ודרכי החדירה למחשב הבין מוריס מיד שהתולעת שלו היא שאחראית לכל המתרחש. אפשר רק לדמיין איזו אימה מילאה אותו כשתפס את ההשלכות של מעשיו. הוא, הסטודנט בן ה-23, הפיל במכה אחת רשתות מחשבים של הצבא, הממשלה והאקדמיה בכל רחבי ארצות הברית. הוא ניחש שה-FBI ושאר רשויות החוק האמריקניות כבר פתחו במצוד נרחב אחריו. הוא צדק.

מוריס מיהר להתקשר לחבר טוב מאוניברסיטת הארוורד בשם אנדי סודו (Sudduth). הוא הסביר לו את המתרחש וביקש ממנו לשלוח (בעילום שם) הודעה לאחת מקבוצות הדיון עם הוראות כיצד למנוע את התפשטות התולעת. אנדי, ככל הנראה, לא השתכנע לחלוטין שמוריס לא מנסה למתוח אותו, אבל הסכים בכל זאת לשלוח את ההודעה.

"דיווח על וירוס אפשרי:

ייתכן שווירוס מסתובב חופשי באינטרנט.

הנה תמצית ההודעה שקיבלתי:

אני מצטער.

הנה כמה צעדים כדי למנוע הדבקה עתידית:

[פרטים טכניים]

אני מקווה שזה עוזר, אבל יותר מזה, אני מקווה שזו מתיחה."

אבל איתרע מזלו של מוריס: התולעת שלו השביתה גם את המחשבים של אוניברסיטת הארוורד, וכתוצאה מכך ההודעה של אנדי סודו "נתקעה" בדרך. חלפו יומיים עד שההתנצלות וההסברים הופיעו לבסוף על לוח המודעות של קבוצת הדיון - עיכוב טרגי שהפך אותם ללא רלוונטיים כלל.

המלחמה על שלום כדור הארץ

במקרה לגמרי, באותו השבוע התכנסו כארבעים מומחי מחשבים מכל רחבי ארצות הברית לוועידה מקצועית שגרתית באוניברסיטת ברקלי שבקליפורניה.

הוועידה כבר עמדה להינעל, למעשה, כשהגיעה הבשורה על התקפת התולעת. אט אט התבררו ממדיו האמיתיים של האירוע, ומרבית המשתתפים בוועידה התקשרו כדי לבטל את הטיסות חזרה הביתה. אוניברסיטת ברקלי הפכה, בזכות ריכוז המוחות, הניסיון והידע שהיו שם באותו הרגע, למרכז העצבים של המלחמה בתולעת של מוריס.

האווירה בברקלי, כפי שתיאר אותה אחר כך אחד המשתתפים, הייתה מחשמלת. תכנות הוא מקצוע "אפור" למדי, ומתכנתים רגילים לעבוד לבד או בקבוצות קטנות. שיא ההתרגשות שמור, בדרך כלל, לשליח הפיצה הדופק בדלת. לפתע פתאום הם מצאו את עצמם בתפקיד קו ההגנה היחיד כנגד פלישה של אויב לא מוכר למחשבים ממשלתיים נושאי מידע רגיש וסודי. ככל שידעו באותו הרגע, התולעת עשויה להתחיל ולהשמיד מידע בסיטונות בכל רגע נתון ולפגוע בביטחון הלאומי של ארצות הברית. הם היו הקומנדו, יחידת העילית להגנת האינטרנט. הכול מסביב היה בקריסה, טלפונים בהולים זרמו מכל רחבי העולם, והם היו במרכז העניינים. הם היו על קו העונשין בזריקה המכרעת לאליפות. הם עמדו בשער בפנדל האחרון של הגביע העולמי. זו הייתה חוויה שאיש מהם לא ישכח לעולם.

בתום עשרים וארבע שעות של עבודה מאומצת הצליחו המומחים בברקלי לנתח את התולעת של מוריס במלואה. הם הצליחו לזהות את כל המנגנונים שבהם השתמש מוריס כדי להדביק את המחשבים, תוך שהם עובדים בשיתוף פעולה עם מומחים מאוניברסיטאות נוספות. מרגע שהבינו המומחים את דרך פעולתה של התולעת, הם יצאו בשורת הנחיות לכל מרכזי המחשבים במדינה כיצד לחסל את התולעת ולהפסיק את התפשטותה. שלושים ושש שעות מרגע תחילת האירועים נסתיים למעשה העניין, והחל תהליך שיקום הנזקים והסקת המסקנות.

הידיעה על המתרחש ברשת האינטרנט תפסה את הכותרות הראשיות בכל כלי התקשורת, אבל עד שהבינו שם את מה שהתרחש, כבר היה מאוחר מדי, והעיתונאים נותרו עם הרבה דיווחים "בדיעבד".

למזלם, עדיין היו כמה כותרות עסיסיות למצות מהעניין, במיוחד כשנתבררה זהותו של כותב התולעת וזהותו של אביו.

רוברט מוריס נתפס יומיים לאחר שהתולעת שלו נעצרה סופית.
מוריס, שלא היה בדיוק אב-הטיפוס של הפושע המשופשף, לא ניסה לרדת למחתרת או להתחמק מרשויות החוק. הוא הניח שה-FBI יתפוס אותו במוקדם או במאוחר, והעדיף לטוס בחזרה הביתה, שם שכר עורך דין טוב ולא הוציא את האף מהדלת.

אז גם גילו העיתונאים שרוברט מוריס הוא למעשה בנו של רוברט מוריס האב, מיודענו ממשחק "דארווין" של שנות השישים, ובאותו הזמן אחד מבכירי המומחים לנושא אבטחת מידע ב-NSA, הסוכנות להגנה לאומית. האירוניה שבייחוסו של מוריס הבן לא נעלמה מעיניהם של העיתונאים, והם אזכרו אותו בהנאה שוב ושוב בכל דיווח. מוריס האב, מצדו, שמר על ממלכתיות. הוא הודה שייתכן כי בנו אחראי לאירוע - מוריס הבן עצמו לא הודה רשמית באחריותו, בשלב זה - אך ציין שהמקרה עשוי להניב רווח מסוים לציבור.

> "[האירוע] העלה [את עניין אבטחת המידע] לתודעת הציבור באופן ניכר. סביר להניח שזה יגרום לאנשים להיות זהירים יותר וקשובים יותר לסכנות כאלה בעתיד."

מעניין לציין, בהקשר הזה, את השוני בין גישותיהם של אנשי המקצוע וכלי התקשורת לאירוע. הדיווחים העיתונאיים ציירו את מוריס כסטודנט צעיר ופוחז, ילד מחשבים מבריק אך פזיז שצריך לעשות לו "נו נו נו" מצד אחד, אבל גם אפשר, בזכותו, להתגאות בדור הצעיר והמתוחכם של ארצות הברית. הציטוט הבא לקוח מתוך כתבה עיתונאית שהתפרסמה מיד לאחר שנודעה זהותו של מוריס הבן.

> "...בקורנל, נציג הפקולטה למדעי המחשב של אוניברסיטת קורנל צחקק ואמר, 'אנחנו מנסים לשמור עליהם [על הסטודנטים שלנו] שלא ישתעממו יותר מדי. כנראה לא השתדלנו מספיק...'"

כפי שניווכח לראות בהמשך הספר, יחס אמביוולנטי זה של הצלפה שהיא גם טפיחה על השכם, שכיח למדי כשמדובר בכותבי וירוסים שנתפסים.

אנשי המחשב, לעומתם, "קטלו" את מוריס ללא רחמים, ואפילו הפגינו כלפיו איבה מסוימת. לדידם, הוא חילל את המקדש, הוא פגע באינטרנט "שלהם". הם כעסו עליו. למעשה, נדירות הן פיסות התוכנה שעברו בדיקה, ניתוח ופירוק לגורמים בצורה כל כך אינטנסיבית כפי שעברה התולעת של מוריס. אלמלא הנסיבות שבהן התרחש האירוע, היה הדבר משול להשפלה פומבית אכזרית למדי: כל שגיאה וכל טעות טיפשית שעשה מוריס בקוד התוכנה שלו נחשפו במלואן ונותחו בפרוטרוט, כשכל בודק נהנה לנעוץ בו סכין חדה. כך נכתב, לדוגמה, באחד מאותם ניתוחים טכניים:

"...נראה שהקוד הוא תוצר של מתכנת לא מנוסה, לחוץ בזמן או מרושל. האדם (או האנשים) שבנו את התוכנה הזו הם, כנראה, חסרי כל הבנה בסיסית..."

אך עם מבחן התוצאה לא ניתן היה להתווכח: התולעת של מוריס הייתה אולי "טרנטה", אבל לפניך. עבור קהילת אנשי המחשבים, "התולעת הגדולה" של "יום חמישי השחור" הייתה קריאת אזהרה ברורה מאין כמוה, דלי מים קרים שהעיר אותם באחת לתוך מציאות מצמררת. השגיאות הרבות של מוריס בכתיבת התולעת וחוסר הניסיון הבולט שלו רק הדגישו את העובדה שכל אחד, **כל אחד**, יכול ליצור כלי משחית אימתני שכזה. תם עידן ימי התום, ונעלמו הימים שבהם רק יחידי סגולה היו בעלי גישה אל ארון התקשורת הקדוש שבחדר השרתים באוניברסיטה.

המהירות שבה התפתחה והתפשטה רשת האינטרנט בשנות השמונים הייתה בעוכריה. מאות אוניברסיטאות, בסיסי צבא וחברות מסחריות התחברו לרשת הצעירה בתוך שנים ספורות בלבד, ללא יד מכוונת או פיקוח של רשות מרכזית כלשהי. בהיעדר פיקוח, כל אחד היה חופשי להגדיר את האופן שבו הוא מאבטח את רשת המחשבים שלו. חלק מהרשתות, הצבאיות בעיקר, היו מוגנות היטב וכל התעבורה האלקטרונית הנכנסת והיוצאת מהן עברה סינון קפדני, בעיקר לצורך מניעת ריגול, מן הסתם. רובן המוחלט של הרשתות האזרחיות, לעומת זאת, היו פרוצות למדי. אבטחת המחשבים הייתה, אם כן, בהתאם לשיקול דעתם האישי של מאות ואלפי מהנדסי המחשבים ברחבי הרשת, שרובם המכריע בכלל לא הבין את הסכנות שאורבות מעבר לפינה.

התולעת של מוריס הביאה לכך שבן לילה קמו ועדות רבות שבחנו את נושא אבטחת המחשבים לעומקו, ומדינות רבות בארצות הברית החלו בתהליכי חקיקה מזורזים של חוקים ותקנות שמטרתם לשמור על ביטחונה של רשת האינטרנט.

את ההבנה הפתאומית לגבי החשיבות שבאבטחת המידע, היטיב לנסח יוג'ין ספאפורד, אחד מהמומחים שניתח את התולעת של מוריס:

"הרבה מומחי מחשב בכירים החלו את הקריירה שלהם, לפני שנים רבות, בפריצה למחשבי הפקולטות במכללות שלהם או במקומות העבודה שלהם, כדי להדגים את כישוריהם. אבל הזמנים השתנו... עסקים שלמים תלויים לחלוטין, בחוכמה או שלא בחוכמה, במערכות מחשב. כסף של אנשים, הקריירות שלהם ואולי אפילו חייהם תלויים בהתנהלות תקינה של מחשבים..."

רוברט מוריס הבן הועמד לדין באשמת חדירה וגרימת נזק במזיד למערכות מחשב קריטיות. כבר מתחילת החקירה התברר שקשה מאוד להעריך את הנזק האמיתי שגרמה התולעת שלו. כפי שציינתי קודם, לאף אחד אין שליטה מרכזית על מה שמתרחש ברחבי הרשת, כך שלמעשה כל ההערכות לגבי מידת הנזק היו ניחושים. מומחי המחשב ציינו מספר סביר של כששת אלפים מחשבים (אולי) שנפגעו באירוע, מתוך

כשישים אלף מחשבים שהיו מחוברים לאינטרנט (כנראה). שווי הנזק הכספי שהוצמד לכל מחשב שנפגע בהתקפה נע בין מאתיים דולר לחמישים אלף דולר, תלוי את מי שואלים.

מוריס הורשע. הוטלו עליו שלוש שנות מאסר על תנאי, ארבע מאות שעות עבודה קהילתית וקנס בסך עשרת אלפים דולר. אף על פי כן, גורלו של מוריס שפר עליו: אוניברסיטת קורנל השעתה אותו, אמנם, אך הקפידה שלא להטיל עליו עונש מחמיר שיפגע ממש בקריירה האקדמית שלו. לאחר שריצה את עונשו בעבודות שירות, הוא חזר למסלול הלימודים וסיים אותו בהצלחה מרובה. כיום, רוברט מוריס הוא פרופסור בכיר למדעי המחשב באוניברסיטת MIT, באופן אירוני, אותה האוניברסיטה שבמחשביה עשה שימוש כ"תחנת ממסר" עבור התולעת שלו.

הנזק שחוללה התולעת של מוריס הבן היה לקח חשוב, אבל אולי היו אלה דווקא משובותיו של מוריס האב בשנות השישים, המשחק "דארווין" שהגה יחד עם ויקטור ויסוצקי ודאג מקלרוי, שלימדו אותנו לקח בסיסי ופשוט בהרבה. מוריס האב סיכם אותו כך:

> "אולי התובנה הגדולה ביותר שהרווחנו [מהמשחק] היא ששכפול-עצמי, תכונה מחוכמת ככל שתהיה, היא פשוטה עד מאוד."

כשחוקרים... כתבו אודות וירוסים [למחשב] בשנות השמונים, הם אימצו לעצמם גלימה של 'כהן גדול', וחשבו שהנושא הוא מסתורי ומסוכן מכדי שניתן יהיה לחשוף אותו לעיניהם של פשוטי העם. זה היה שטויות במיץ עגבניות, אבל העיתונות אכלה את זה.

> "...אין שום דבר מסתורי בווירוסים למחשב. ניתן ליצור וירוס פשוט ועובד, בלא יותר מכמה שורות של קוד..."

על הספר

הספר לוקח את הקוראים למסע משעשע, מרתק ומטריד בעקבות ההיסטוריה העולמית של התוכנות הזדוניות. זהו, למעשה, הסיפור הלא יאומן מאחורי וירוסי המחשב. מתברר שההיסטוריה של וירוסי המחשב עתיקה כמעט כמו המחשבים עצמם. בתוך שלושים שנה בלבד הפכו הווירוסים במחשבים משעשוע בלתי-מזיק לתוכנות זדוניות רבות עוצמה, שמטילות איום מוחשי על שלומם של גולשי האינטרנט, ארגונים עסקיים ומדינות שלמות. הכוח העצום והזדוני הזה יכול, למעשה, להרוס את העולם – כפי שאנחנו מכירים אותו כיום. הספר נשען על אינספור ראיונות שקיים המחבר עם גיבורי פרשות הווירוסים המרתקות ביותר בעולם, מספר בין השאר על סטודנט צעיר שהשבית את רשת האינטרנט כולה ליומיים, ועל הרמזים המרתקים המסתתרים בתוך "סטוקסנט", התולעת שתקפה את הכור האיראני.

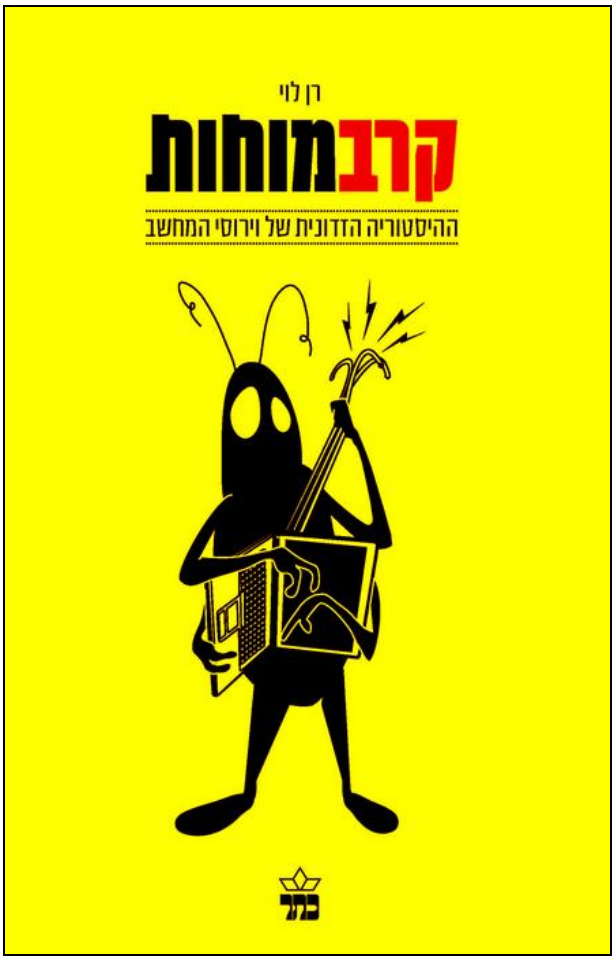
על המחבר

רן לוי, מהנדס מחשבים במקצועו, מפקד דבור בעברו, ומחברם של "פרפטום מובילה" (ספריית מעריב) ו"האוניברסיטה הקטנה של המדעים" (גורדון) הוא בעל התוכנית האינטרנטית "עושים היסטוריה"- שזכתה בלמעלה ממיליון הורדות ובקהל מאזינים נלהב. ספריו הקודמים זכו בביקורות נלהבות.

הנחה בקניית הספר לקוראי המגזין

קוראי המגזין המעוניינים לרכוש את הספר במלואו יכולים לעשות זאת ב-39.90₪ (במקום 98₪ קטלוגי, הנחה רק לקוראי המגזין), לא כולל דמי משלוח. הזמנות דרך החנות של כתר ב: 02-6557837 (בין השעות 10:00 ו-18:00) או במייל:

izik@keter-books.co.il



תכנון והטמעת SIEM בארגון

מאת אריק יונאי

הקדמה

במאמר זה אסקור דרכים ואפשרויות להטמעת מערכת SIEM בארגון. מי שקרא את מאמרי הקודם, יודע שאני משתדל למעט בהגדרות המילוניות היבשות, אך אסקור בקצרה לריענון הזיכרון:

> מערכת SIEM היא מערכת המאפשרת שליטה ובקרה (שו"ב) על אירועי אבטחת מידע בארגון, ע"י איסוף התראות מרכיבים שונים, וע"י חיבור ותעדוף של ההתראות, מייצרת תמונת מצב כוללת של מצב אבטחת המידע בארגון (לרוב מצב אבטחת המידע בארגון, אך בהחלט ניתן להשתמש במערכת SIEM גם בכדי לתת מענה לצוותים אחרים בארגון).

הטרמינולוגיה משתנה מעט ממוצר SIEM אחד למשנהו, אך העיקרון זהה אצל כולם.

מערכת ה-SIEM תאסוף / תקבל לוגים מרכיבים שונים, כגון: Firewall, IPS, VPN, שרתי Windows, שרתי Web, DB, וכו', ותעביר את הלוגים תהליך (Correlation בין השאר) בכדי לקבוע בסופו של דבר מה לעשות עם הלוג, והאם להפוך אותו ל-Event.

Event יהיה אירוע (או התראה), שייווצר מלוג או ממספר לוגים שונים, ושיש לו חשיבות כלשהי ולרוב גם ידרוש התייחסות או תגובה.

ליוויתי מספר פרוייקטים של הטמעת SIEM בשלבים שונים, ואני חייב לציין שהטמעת מערכת SIEM לרוב היא הטמעה מורכבת וארוכה, הדורשת הרבה מבחינותרבות (כגון כסף, זמן, סבלנות, והמון ניסיון מקצועי). לעומת מוצרים רבים, בכדי להביא מערכת SIEM למצב של פעילות "איכותית" (התראות "אמיתיות" של אירועי אבטחת מידע הדורשים תגובה), יש צורך בהמון זמן הטמעה ותחזוקה, וצורך ביצרן איכותי + אינטגרטור מקצועי עם המון ניסיון במוצר הספציפי.

לאחר ליווי של מספר הטמעות SIEM, אני יכול לומר שרובן נכשלו (בפרספקטיבה של הדרישות המקוריות שהיו מהמוצר), ובסוף נשארה קופסה שחורה בחדר שרתים, בעל ערך מועט ביותר, ממספר סיבות שאפרט בהמשך. אנסה לומר זאת בעדינות, כנראה שאין פתרון SIEM אשר "מחברים והוא עובד", או כפי שאינטגרטורים מסויימים נוהגים לומר: "תן לי יומיים עבודה - תראה אילו התראות אני מוצא לך!".

אז זהו, שכנראה אין חיה כזאת בתחום ה-SIEM.

ועכשיו, נצלול פנימה ☺

סיבות להטמעת SIEM בארגון

אסקור את שתי הסיבות העיקריות להטמעת מערכת SIEM:

1) רגולציה - הסיבה הנפוצה להטמעת SIEM בארגון. אני מעריך שרוב הארגונים מתחילים את ההטמעה של מערכת SIEM מתוך דרישה של רגולציה כזו או אחרת, המחייבת את הארגון לאחסן ולנטר את הלוגים המכילים פעולות הנוגעות למידע רגיש, במקום מרכזי, ושתהיה לארגון אפשרות לקבל תמונת מצב כוללת על מצב אבטחת המידע בארגון, לרוב ע"י שליחת דו"חות מתוזמנים ממערכת ה-SIEM. לרוב, הטמעות מסוג זה הן פשוטות וקצרות יותר מהאפשרות שאציג מיד, ולרוב המטרה היא לסמן "V" על דרישות רגולציה. הערך של פתרון ה-SIEM בארגון בשיטה זו, מנקודת מבט של אבטחת מידע, יהיה מוגבל ברוב המקרים.

2) רצון לשפר את אבטחת המידע בארגון - במצב זה, בניית פתרון ה-SIEM יבוצע בצורה "נכונה" יותר ברוב המקרים, אך ההטמעה תהיה מורכבת יותר. ארגונים המעוניינים לשפר את מערך אבטחת מידע בוחרים פעמים רבות להטמיע מערכת SIEM, כפתרון "שליטה ובקרה" (שו"ב), המאפשר ניהול אירועים מנקודה מרכזית. כמו כן, מערכת SIEM גם מקנה יכולות תחקור (Forensic), כך שבמקרה אירוע ניתן לחפש מידע לאחור.

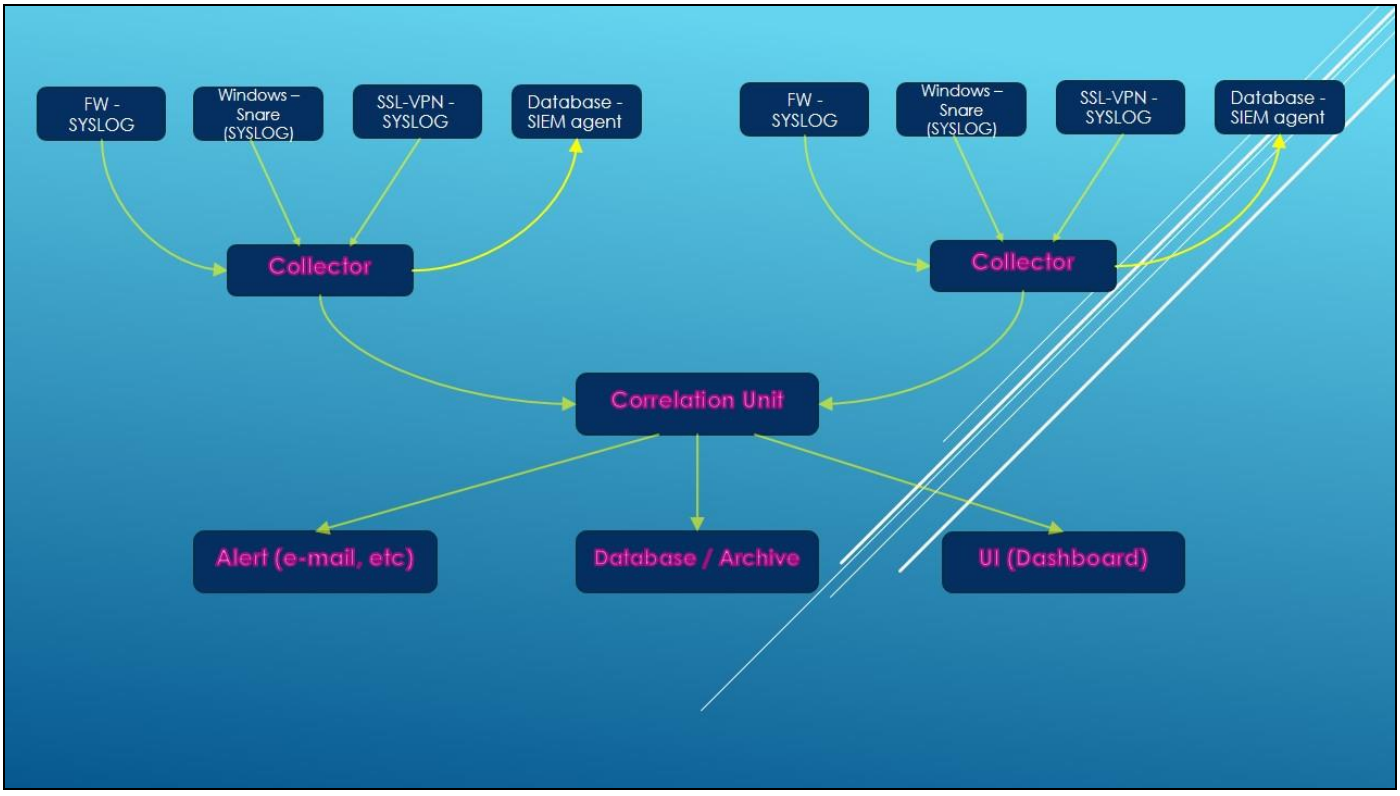
הבעיה והפתרון

כמובן שאין באפשרותנו לעבור על כל הלוגים הנכתבים במערכות הארגון, ובוודאי שלא בו-זמנית מרכיבים שונים, בזמן אמת. מערכת SIEM מאפשרת לקבל במקום מרכזי אחד, עם Dashboard אחד, את הלוגים מרכיבים שונים ברשת, ועליהם יבוצעו פעולות שיאפשרו בסופו של תהליך לקבוע האם הלוג יהפוך לאירוע (Event), והאם יש לטפל בו.

מבנה ורכיבי המערכת

ליצרני SIEM שונים יש שמות שונים לרכיבים שונים, אך אפרט מהם הרכיבים הכלליים המשותפים למערכות SIEM (כאמור השמות יכולים להיות שונים מיצרן ליצרן).

מערכת SIEM היא תוכנה המוכרבת ממספר רכיבים, לעיתים תהיה מותקנת על Appliance ייעודי, ולפעמים על Open server או שרתי VM. חלק מהפתרונות מבוססי Linux, וחלקם מבוססי Windows. אצל רוב היצרנים ניתן יהיה להפריד את הרכיבים השונים לשרתים נפרדים במקומות שונים ברשת.



[מבנה מערך SIEM סטנדרטי]

למערכת ה-SIEM יש כמה רכיבים:

1) **Collector** - רכיב אשר אמון על איסוף / קבלת הלוגים מהרכיבים השונים. רכיב זה לרוב יודע לקבל / לאסוף לוגים במספר פרוטוקולים, כאשר בדר"כ הנפוץ מביניהם יהיה SYSLOG.
בחלק מההתקנים ברשת נגדיר שליחת SYSLOG לרכיב (שרת) ה-Collector (כאשר החיבור יפתח מהתקני הרשת, לכיוון שרת ה-Collector).

בהתקנים אחרים ברשת (שרתי Database לצורך העניין), יתכן ויהיה צורך להגדיר Authentication ברכיב ה-Collector, אשר **יפנה לשרת** ה-Database **וימשוך את המידע הרלוונטי אליו.**

התקנים אחרים ישלחו מידע ע"י Agent אשר יותקן על אותו רכיב (לצורך העניין שרתי Windows), וה-Agent הוא זה שישלח את הלוגים הרלוונטים לרכיב ה-Collector בפרוטוקול כזה או אחר.

2) **Parsing** - רכיב אשר בדר"כ יהיה חלק מרכיב אחר.
תפקידו של רכיב זה הוא לפרק את הלוגים "הגולמיים" (לא מחולקים לשדות, ועל כן חסרי משמעות למערכת ה-SIEM), לשדות מוגדרים אשר יוכלו להיכנס לטבלאות קבועות ב-DB, בכדי שניתן יהיה לתת להם משמעות ולהגדיר על-פיהם תסריטים (יפורט בהמשך).

הלוג הבא (במצב "נקי", ללא Parsing), מדגים כיצד ה-Collector מקבל לוג מ-Firewall ב-SYSLOG:

```
"14:55:20 accept gw.foobar.com >eth1 product VPN-1 & Firewall-1 src
10.5.5.1 s_port 4523 dst 10.10.10.2"
```

ובכן, כנראה שלרובנו יהיה קל יחסית להבין מה כתוב פה פחות או יותר, כי ראינו כאלה ואנחנו מסוגלים להשתמש בהיגיון בריא בכדי לנתח את המידע. ובכן, ל- SIEM, כך מסתבר, לא יהיה קל לנתח את המידע.

מערכת ה-SIEM זקוקה ל-Parsers, מערך של Regular expressions, שיעניק משמעות לחלקי הלוג השונים. לכל פתרון SIEM קיים מערך Parsers, אשר מתרגם את הלוג הגולמי, ללוג בעל משמעות.
בדוגמא זו, הלוג (ללא Parsing):

```
"14:55:20 accept fw-1.test.com >eth1 product VPN-1 & Firewall-1 src
10.5.5.1 s_port 4523 dst 10.10.10.2"
```

לאחר Parsing יראה פחות או יותר כך:

Time of event	Action	Firewall IP	Interface	Source	Source port	Destination
14:55:20	accept	fw-1.test.com	eth1	10.5.5.1	4523	10.10.10.2

אתייחס לבעיות צפויות בתחום ה-Parsing בהמשך המאמר.

3) **Correlation Unit** - הרכיב מבצע ניתוח של הלוגים (לאחר שלב ה-Parsing), ומחליט האם ליצור התראה (Event). ה-Correlation Unit **מורכב מתסריטים (Scenarios)**, שהם לצורך העניין ליבת מערכת ה-SIEM.

תסריט הוא קבוצה של **חוקים**, המגדירים ל-Correlation Unit כיצד להתייחס ללוגים השונים.

לצורך הדוגמא, **אציג תסריט פשוט, המוגדר מחוק אחד בלבד, כאשר מטרתו היא להתריע כאשר נוצר Domain Admin חדש ב-Active Directory.**

החוק אומר כך:

אם:

א. המקור הוא Active Directory (ניתן להגדיר את החוק ע"פ סוג הרכיב, כתובת IP, וכו').

ב. ומגיע לוג המכיל פעולה X (יצירת Domain Admin חדש).

ג. הפעולה התרחשה בין השעות 21:00 ל-7:00 (החלטתי שאלו שעות חשודות שבהן אינני מצפה לפעילות כזו, לצורך הדוגמא).

אז: שלח התראה (דוא"ל לצורך הדוגמא) למנהל הרשת עם פירוט הפעילות (מי יצר, מתי, וכו'), ברמת דחיפות Medium.

דוגמא ל**תסריט נוסף, המורכב משני חוקים (ששניהם חייבים להתקיים יחדיו). המהות של התסריט, היא למנוע מצב שבו אדם יגנוב זהות של עובד ויצליח להתחבר ב-VPN מחוץ למשרד, בעוד שהעובד נמצא בתוך המשרד.**

חוק א':

אם:

א. המקור הוא Active Directory.

ב. מגיע לוג המאשר שמשתמש X ביצע פעילות בדומיין, **מתוך רשת המשרד**.

חוק ב':

אם:

א. תנאי: חוק ב' יתקיים **רק אם חוק א' התקיים בחצי שעה האחרונה** (אם חוק א' התקיים לפני יומיים, חוק ב' יהיה חסר משמעות, ועל כן כל התסריט לא יופעל).

ב. המקור הוא SSL-VPN.

ג. משתמש X הזדהה בהצלחה מול ה-VPN.

אז: אם חוק א' + חוק ב' התקיימו ביחד, אז שלח התראה למנהל אבטחת מידע בארגון + המשתמש שביצע את הפעולה (בכדי לקבל מהמשתמש תגובה לגבי הפעילות: האם הוא זה שביצע את הפעולה, או זהותו נגנבה?), עם פירוט הפעילות, בדרגת דחיפות Critical.

ודוגמא אחרונה לתסריט קצר המורכב מחוק אחד (תלוי במוצר SIEM), שמטרתו היא **להתריע בפני Brute force על SSL-VPN מכתובת מקור אחת**.

החוק:

אם:

א. המקור הוא SSL-VPN.

ב. מגיע לוג עם הודעת "Failed login" מכתובת Public IP כלשהי.

אז: שלח התראה למנהל אבטחת המידע בארגון, **אך התנאי הוא שהחוק יופעל כ-30 פעמים בטווח זמן של חמש דקות** (אחרת החוק והתסריט לא יופעלו).

מערכת SIEM תכיל לרוב **מספר תסריטים שונים**, שכל אחד מהם **יכיל חוק אחד או כמה חוקים.**

ע"י מערך של תסריטים, יכול הארגון לקבל התראות ולהגיב במהירות לתסריטים אותם הוא הגדיר כקריטיים להמשך פעילות תקינה של הארגון.

מחזור חיים סטנדרטי של התראה:

- ☒ הלוגים נשלחים / נאספים ע"י ה-Collectors מהרכיבים השונים הרשת.
- ☒ הלוגים עוברים תהליך של Parsing (ע"פ Regular expressions), ומוכנסים לטבלאות הרלוונטיות (Date, IP addresses, actions, etc.).
- ☒ הלוגים מועברים (לאחר ה-Parsing) ל-Correlation Unit, **ונבדקת התאמה בין הלוגים לתסריטים השונים.**
- ☒ במידה וקיימת התאמה בין לוגים לתסריט (במידה והלוגים עונים על הדרישות של החוקים באחד התסריטים), מתבצעת פעולה, לרוב שליחת התראה למשתמש (קיימים מוצרי SIEM אשר יודעים "להגיב אוטומטית" לאירועים, כגון להריץ סקריפטים, וכו').
- ☒ לאחר מכן הלוגים / ההתראות ישלחו ל-Database (יפורט בהמשך), לצורך אפשרויות תחקור או Archive.

4) מאפיינים נוספים למערכות SIEM:

במערכות SIEM רבות יופיעו מספר רכיבים נוספים, אשר נותנים ערך מוסף חשוב ולפעמים קריטי בבחירת המוצר המתאים:

א. **Compression** - קיימים שני סוגים של דחיסה: הראשון - דחיסה של התעבורה, והשני - דחיסה של המידע המאוחסן (Database, logs, etc). כאשר עוברים לוגים רבים על גבי קווי WAN, דחיסת המידע יכולה להיות קריטית, במקרים מסויימים משהו שלא ניתן להסתדר בלעדיו.
בדקו היטב מה יכולות המוצר בנוגע לדחיסת המידע שעובר על גבי הרשת, במידה וקווי ה-WAN יקרים ללבכם.

הסוג השני - דחיסה של המידע על גבי Storage. מבלי להיכנס עמוק מדי לסוגיה זו, מערכת SIEM גם מקנה יכולות תחקור, מאחר וניתן לחפש לאחור לוגים / אירועים. לדחיסת המידע באחסון (ולא רק על גבי רשת התקשורת), יש משמעות גדולה, כי היא מגדילה את הזמן שניתן לתחקר לאחור אירועים.

ב. **תחקור (Forensic)** - לכל מערכת SIEM יש יכולת חיפוש / תחקור כזו או אחרת, אך איכותה הוא נושא חשוב מאוד. מה ניתן לחפש ואיך, והכי חשוב: **באיזו מהירות יזרמו התוצאות.** בזמן אירוע, נהיה מעוניינים לחפש כתובת IP, שם משתמש, וכו'. אני יכול לומר באחריות שקיימות מערכות SIEM בהן **תוצאות חיפוש של 24 שעות אחורה (על מיליוני שורות לוג), יגיעו רק לאחר שעות ארוכות ואף ימים (!)**, ואילו קיימות מערכות SIEM **אשר יהיה ניתן לחפש חודשים אחורה על כמות אדירה של שורות לוג, ולקבל את התוצאות תוך דקות בודדות (ואף בפחות).**

קחו בחשבון את נושא מהירות החיפוש במערכת השיקולים בבחירת מוצר SIEM (זה נושא יותר משמעותי ממה שהוא נראה על פני השטח, מאחר וחיפוש איטי לאחור יכול לפגוע משמעותית ביכולות התחקור).

ג. **Dynamic C&C database** (או IP Reputation) - רכיב אשר מכיל רשימה של כתובות IP ודומיינים זדוניים, ומתעדכן בתדירות גבוהה.
ניתן להגדיר תסריטים אשר יתריעו בפני תעבורה הנכנסת / יוצאת מהרשת לכתובות זדוניות, ובכך לזהות Malwares שונים (ראו מאמר בנושא "Malwares 2.0, ודרכי התמודדות בארגון": http://www.digitalwhisper.co.il/files/Zines/0x2A/DigitalWhisper42.pdf).
רכיב יעיל מאוד כנגד סוסים טרויאנים ותולעים.

ד. **Compliance Reports** - במידה והארגון נדרש לעמוד ברגולציה (או ידרש לעמוד באחת כזו בעתיד), כדאי לוודא שקיימים דו"חות Out-of-the-box לרגולציות שונות (רוב המוצרים תומכים בדו"חות מסוג זה).

בעיות צפויות ונקודות קריטיות ל-POC

העצה הטובה ביותר שקיבלתי לפני הטמעת SIEM, היה לעשות PoC מעמיק וארוך! אם יש מוצר שהערך של POC בו הוא הגדול ביותר, לדעתי זה מוצר SIEM.

הגדירו יעדים ומדדים ברורים ומדוייקים להצלחה!

1) אתחיל דווקא מהסוף: אולי אחד הדברים החשובים ב-POC / לאחריו, הוא כתיבת מסמך Incident response.

הגעתם ליעד המבוקש, דאגתם להגדיר את ה-SIEM היטב, תיקנתם, שיפצתם, קניתם, והמוצר עובד פיקס. מה עכשיו? בסופו של יום, יגיעו התראות מה-SIEM, ומישהו יצטרך לטפל בהן. המלצתי החמה היא ליצור מסמך שירכז כמה עשרות אירועי אבטחת מידע (נפוצים וחריגים), ושרשרת הפעולות שיש לבצע לאחר קבלת דיווח על אירוע מה-SIEM.

הקמת צוות SOC (או לפחות אדם שיודע שתפקידו הוא לטפל באירועי אבטחת מידע שיגיעו מה-SIEM), היא פעילות מורכבת ממה שנדמה, ומומלץ מאוד להגדיר פעולות מסודרות לפני שיגיעו

האירועים (מה אני עושה כשמתקבלת התראה על Brute force? מה אני עושה כשמשתמש מתחבר ב-VPN, למרות שהוא במשרד? כיצד מגיבים לאירוע DDoS וכו').

2) בדקו היטב את מודל הרישוי של המוצר, וסכמו לפני הרכישה על אפשרות גדילה. רוב מוצרי ה-SIEM נמדדים ב-EPS (Event per second), שהמשמעות היא כמה לוגים המוצר מסוגל לעבד בשניה. חלק מהמוצרים דווקא לא מגבילים כמות EPS ע"י רישוי (אלא מוגבלים למגבלות חומרה), אך הרישוי מגביל כמות Collectors, או רכיבים אחרים. סכמו מראש על מחירי שדרוג של חומרה / תוכנה / רישוי, במידה ויוצר הצורך בעתיד.

3) נושא שבוודאות ידרוש מכם להשקיע הרבה אנרגיה יהיה תחום ה-Parsers. רוב ה-Parsers, מגיעים מיצרן ה-SIEM. יחד עם זאת, Parsers נכתבים למוצר ספציפי, ולרוב גם לגרסה ספציפית. חשוב מאוד להבין את המשמעויות של העניין.

נניח כי יש לי בארגון Check Point Firewall, ומוצר ה-SIEM תומך ב-Check Point זה נהדר, אבל זה ממש לא מחייב שה-SIEM ידע לבצע Parsing נכון של הלוגים. יכול להיות שה-Parsers במוצר ה-SIEM נכתבו ל-Check Point R70, ואילו בארגון יש לי Check Point R75.40 (לדוגמא), ומבנה הלוגים הוא שונה לחלוטין.

המשמעות במקרה זה (או במקרים של תוכנה שאיננה "תוכנת מדף", אלא תוכנה שפותחה בתוך הארגון), **היא שתצטרכו ליצור את ה-Parsers בעצמכם**. לחלק ממוצרי ה-SIEM יש כלים חצי אוטומטיים לכתיבת ה-Parsers, אך בהחלט זהו נושא שחובה להתנסות בו בעצמכם במהלך ה-POC (מומחה Regular expressions מאוד יכולה לסייע), ולהבין האם כתיבת Parsers חדשים היא משימה לא מורכבת באופן יחסי, או משימה מאוד מאוד קשה.

אני מדגיש את החשיבות של ה-Parsers, מאחר וראיתי במו עיניי מקרים שבהם יצרן X כתב רשימה באורך הגלות של מוצרים נתמכים, ואילו בפועל, רק כ-30% מהמוצרים עברו Parsing כמו שצריך. המשמעות היא שבכדי לתקן את זה, צריך לכתוב עשרות אלפי Regular expressions, ובפועל, ניתן לזרוק את המוצר לפח.

4) במהלך ה-POC למדו כיצד לכתוב תסריטים, ובדקו אותם. לדוגמא, אם כתבתם תסריט שמתריע מפני Brute force על ה-VPN, בדקו בפועל שאכן ה-SIEM ידווח לכם על האירוע.

5) תנו ל-SIEM לאסוף מידע במהלך ה-POC מהרכיבים אותם תרצו לחבר לרשת, במשך כמה שבועות. לאחר מכן, בצעו חיפוש על פרמטר כלשהו (כתובת IP, שם משתמש, וכו'). ראו תוך כמה זמן אתם מקבלים את תוצאות החיפוש.

6) כתבו מסמך תסריטים מפורט ומסודר. במסמך, פרטו מהו התסריט (דיווח על התחברות משתמש דרך VPN, בעוד המשתמש נמצא ברשת המשרד, לדוגמא), אילו רכיבים יש לחבר ל-SIEM לצורך יישום תסריט זה (Active Directory + SSL-VPN), ואת כל שאר הפרמטרים הדרושים (לאחר כמה לוגים ה-SIEM ישלח את ההתראה, באילו שעות, באיזו עדיפות, וכו').

7) כאמור מערכת SIEM היא מערכת שמיועדת לצורכי אבטחת מידע, אך בהחלט ניתן להשתמש בה גם לשימושים נוספים, לצורך העניין לשימושי מחלקת ה-IT: קבלת התרעות על נפילות של שרתים, מעקב אחר פעולות System מסויימות, נפילת Services, ועוד. שימוש מערכת ה-SIEM גם ע"י צוותים נוספים בארגון, יכול להעלות את ערכו וחשיבותו של מוצר ה-SIEM בארגון.

לאמיצים בלבד - Open Source

לגיבורים שבנינו, קיימים פתרונות Open source חינמיים לחלוטין, או מאוד מאוד זולים (יכולים לעלות אחוזים בודדים ממחיר פתרון SIEM של יצרן), אך לרוב הם דורשים ידע מעמיק ביותר ב-Linux (לדעת לכתוב ifconfig ו-reboot לא יספיקו במקרה זה ☺).

בפתרונות אלו, בדר"כ לא יהיו את כל הרכיבים הקיימים בפתרונות בתשלום, אך יש להגדיר היטב את צרכי הארגון ולהשוואתם ליכולות המוצרים, בהחלט יתכן ויהיו פתרונות Open source אשר יתאימו לצרכי הארגון.

לסיכום

כמובן שיש עוד מידע רב על התחום, אך במאמר זה ניסיתי לתת את אבני היסוד להטמעת SIEM, בכדי להימנע מהטעויות הקרדינליות שרבים מאיתנו עשו, בעיקר בבחירת המוצר והאינטגרטור.

אני לא חושב שיש הטמעת SIEM קלה ומהירה (לא רק לחבר את ה-SIEM לחשמל, אלא SIEM שבאמת יעשה עבודה), ודורש תקופה ארוכה מאוד של Fine tuning, אבל בהחלט כדאי לבדוק האם SIEM יכול לסייע לכם.

בתחילת הדרך, כנראה ויגיעו אליכם יותר מדי התראות מהמוצר, ואז לאחר מכן, מדי התראות, ואז עוד פעם יותר מדי, עד שתגיעו למצב שבו אתם מקבלים כמות התראות שניתן להתמודד איתה. מטבעם של דברים, רוב ההתראות שתקבלו יהיו "False-positive", ולא באמת בעלות משמעות - היו חזקים, והמשיכו לבצע Fine tuning על התסריטים שהגדרתם עד שתגידו לתוצאה הרצויה! ואדגיש פעם נוספת את **חשיבות איכותו של האינטגרטור והניסיון המקצועי שלו!**

המצב הגרוע ביותר הוא שתקבלו אלפי התראות ביום, ואף אחד כבר לא יפתח אותן בכדי לבדוק מה יש בפנים.

תודה על תשומת הלב, אריק יונאי.

תקיפת בתי חולים על ידי מטופלים

מאת אמיתי דן (popshark)

הקדמה

לאחר התקיפה על הכור האטומי באיראן ביוני 2010, באמצעות תולעת סטקסנט (Stuxnet), העולם עבר שינוי. ההבנה כי ניתן לבצע פעולות סייבר מתוחכמות הביאה לכך שמדינות החלו לפתח מערכי הגנה ותקיפה קיברנטית, וגם חברות אזרחיות הבינו שיש צורך בהערכות מחודשת לנושא. חשוב להבין איך הצליחו לחדור לתוך המערך האיראני בין היתר מאחר שעלו מספר תיאוריות ביחס לדרך שדרכה הצליחו להחדיר את התולעת. שימוש יעיל במסקנות מאפשר את שיכפול ההצלחה.

בין היתר הועלתה השערה כי באחד מהכנסים שבהם מדעני גרעין אירניים השתתפו, חולקו התקני זיכרון בחינם (Net Stick) וכך, תוך ניצול חולשה אנושית של קבלת מתנות, משתתפי הכנס החדירו את תוכנת התקיפה שהוסוותה בתוך ההתקנים. השערה זו היא הבסיס למחקר שעיסוקו היכן ניתן לתקוף בתי חולים תוך עקיפת מרבית ההגנות הקיימות במעגלי האבטחה החיצוניים.

בתי חולים כמטרת איכות

היכולת לשבש מאגרי מידע של מטופלים לפני ניתוח, לשנות פרטי מטופלים כולל סוגי דם או לפגוע ולשלוט מרחוק במכשירים רפואיים, עלולה להוות מטרת איכות של ארגוני טרור, פשיעה וגופים ואנשים שונים. בעוד שהרווח של ארגוני טרור יהיה פגיעה במורל האוכלוסייה שתפחד לקבל טיפול מבית חולים, ארגוני פשע יכולים לסחוט בצורה זו את בתי החולים כסוג של כופר בתמורה לשחרור מאגר מידע. בו בזמן היכולת לשנות נתונים של פציינט מסוים מאפשרת לגרום לפגיעה בגופו אותו פציינט. לדוגמא, בכיר בממשלה או בחברה עסקית שנסע לקבל טיפול רפואי.

מאחר והיום קיימים רובוטים רפואיים הנשלטים מרחוק ובצורה קולית, חלקם לפרוצדורות רפואיות (דה וינצ'י), ניתן לסמן גם אותם כמטרת איכות. הבעיה שנשארת היא הדרך. בתי חולים רבים מנסים להגן על עצמם ממתקפות קיברנטיות ובניגוד לעבר קיימות הגנות רבות כנגד תוקף חיצוני. אם נחזור לכור האירני, ההשערה בנוגע לשיטת התקיפה הינה מדיה מגנטית לצורך חדירה למערכות הארגון. גם במקרה זה, צריך למצוא את נקודת התורפה הנכונה - שבועת הרופאים.

בניגוד לטיפולים הנערכים בתוך גבולות קופות החולים המנהלת מאגר מידע עצמאי, ישנם טיפולים רפואיים שעל הפציינט להביא אתו את ההיסטוריה הרפואית כדי שיוכל לקבל טיפול הולם. תוצאות של בדיקות רפואיות ניתנות לרוב על גבי מדיה מגנטית. כאשר הפציינט עובר בדיקה בקופת חולים ומגיע לניתוח, הוא מתבקש להביא דיסק עם תוצאות רפואיות רלוונטיות, וזאת נקודת תורפה.

בניגוד לסיפור על טרויה, שבה החדרת הסוס הטרויאני בוצעה בעורמה ובצורת מתנה, כאן בעצם הרופאים פונים בבקשה לכלל המטופלים להביא מדיה מגנטית. זאת הדרך לאפשר טיפול רפואי הולם. מנקודת המבט של התוקף, במקום לתת את ההיסטוריה שלו, הוא תוקף את בית החולים.

בתוך הדיסק שנראה כמו דיסק תוצאות לגיטימי, ניתן להסתיר את כלי התקיפה. וכך, על ידי שימוש במטופלים מתחזים לבצע מתקפות מרובות משתתפים על בתי חולים .תוך זמן קצר ניתן יהיה להחדיר מדיה מגנטית עם כלי תקיפה לכלל בתי החולים במדינה שתותקף.

נקודות תורפה בבתי חולים

לאחר שהבנו את הבסיס והרקע אפשר להתקדם שלב ולהתחיל להתביית על מטרות נוחות להתקפה. מאחר שמערכות רפואיות מנסות לחסוך כסף ומשאבי מערכת עקב זמינות מוגבלת, מטופלים רבים עם שברים או בעיות רפואיות הדורשות צילום פנימי נשלחים לבצע הליך מקדים של בדיקות זולות לפני הבדיקה היקרה יותר.

לרוב המטופל יתחיל בצילום רנטגן רגיל, יעבור ל-CT ורק במידה שאין ברירה, הוא ימתין ויבצע את הבדיקה היקרה בסדרה זו שהנה MRI. לעיתים יש לבצע בדיקה נוספת לאחר תקופה או שבר חוזר. מאחר שבכל שלב התוצאות מתקבלות על ידי מדיה מגנטית, ומכיוון שחובה על הרופאים לראות את ההיסטוריה הרפואית של המטופל ברוב המקרים, הפציינט יכול להביא אתו כל דיסק שיבחר.

אם מחברים את הפאזל לתמונה ברורה, מבינים כי בדיקות רדיולוגיות מהוות דרך יחסית נוחה להכניס מדיה מגנטית תוקפנית לבתי חולים. מאחר שמדובר בעצם בהתקנים לרפואה גרעינית, נסגר כאן מעגל תקיפתי עם מקור השראה לתקיפות עתידיות. תרחיש זה מעלה שאלה נוספת - מהו המכשיר המסוכן ביותר שניתן לפגוע בו?

אחת הדוגמאות היא מטופלים העוברים טיפולים נגד סרטן במכשירי הקרנה. מכשירים אלו הינם בעלי עוצמת הקרנה המיועדת להריגת תאים ביולוגיים. השתלטות על מכשיר מסוג זה לאחר החדרה של מדיה מגנטית, תאפשר את הפיכתו למכשיר שהורג תאים בריאים ובמקרים מסוימים אף ככלי לפציעת אנשים. כלל מכשירי הרדיולוגיה עוברים בדיקות תקופתיות למדידת חריגות קרינה, אך הבדיקות אינן תכופות,

וכפי שכבר למדנו מהמקרה של איראן, ניתן גם לזייף תוצאות של מכשירי ניטור ולמנוע את איתור החריגה.

דוגמה נוספת היא רובוטים רפואיים המבצעים ניתוחים. חדירה למערכת הכריזה של בית חולים בעזרת דיסק מגנטי תאפשר באופן הפשוט ביותר לשלוח הוראות קוליות בזמן ניתוח חודרני לרובוטים המקבלים הנחיה קולית. במקרים מעניינים יותר, ניתן יהיה להשתלט לחלוטין על הרובוט ולפגוע בחיי המטופל.

ברוב בתי החולים בעולם קיימת בקשה מהמטופלים המגיעים למחלקות הרדיולוגיות להביא איתם דיסק עם תוצאות רפואיות. לעיתים גם ניתן להביא תוצאות אלו בהתקן USB. מאחר והשימוש במדיה מגנטית הינו חדש יחסית בעולם הרפואה, ככל שבית החולים מנותק ממחשבים הסיכוי שלו להיות מוגן במקרה זה גדול יותר. לכן, דווקא בתי חולים ומרפאות בעולם המשתמשות בתשלילים מיושנים או נייר להצגת נתונים, מוגנים יותר מפני תקיפות מסוג זה.

לצורך מתקפה מוצלחת תוקף יוכל לבצע הנדסה לאחור של קבצים מסוג DICOM המשמשים את בתי החולים לצורך שמירת נתונים רפואיים, ולדעת היכן להחדיר בהם את קובץ התקיפה תוך חיפוש של Zero Days בקוד התוכנה. מבחינה ויזואלית ופיזית ניתן להעתיק את הקובץ שבדיסק, את הנתונים המודפסים על הדיסק המגנטי ולהטמיע אותם חזרה לתוך הדיסק הנוסף שלתוכו יוחדר אמצעי התקיפה. את הנתונים החזותיים (שם הפציינט, פרטי זהות, ופרטי הבדיקה) התוקף יצרוב בחזית הדיסק השני כך שהדיסק לא יעורר חשד.

איומים נוספים יכולים להגיע מכיוון תיירות רפואית ומסתננים. במקרה של תיירות רפואית, היכולת לשלם למדינה זרה תמורת טיפול מאפשרת לתיירים רפואיים להוות כלי אנושי יעיל במתקפה בווקטור זה, במיוחד מכיוון שלעיתים המטופל מגיע ממדינת אויב. במקביל להם, גם מסתננים הנמצאים במגע עם מבריחי גבול וארגוני טרור עלולים לקבל דיסק תקיפה.

נדרשת היערכות מחדש של בתי החולים

כפי שמאמר זה מציג ניתן כיום להחדיר בקלות אמצעי מדיה מגנטית לצרכי תקיפה בבתי חולים בעולם המערבי. פרצה זו נובעת מהתנהלות אבטחת מידע של בתי חולים בעולם, הכפופה לשבועת הרופאים ומהחובה של הרופאים לרפא, מרכיב אתי הקודם לנהלי מחשוב כאשר ישנה שאלה האם לטפל בחולה או לאבטח מחשב.

בכנס רדיולוגים שבו העברתי הרצאה בנושא נאמר שלא ניתן לצפות מרופא לבצע בדיקה ביטחונית לפציינט. מבחינת פתרונות, צריך לפתח שיטות עבודה חדשות ונדרשת היערכות בכל בתי החולים

ובמערכת הרפואית בכלל בכל הנוגע למטופלים שפוגעים בנהלי בתי החולים ובהתנהלותם. יתכן שאחת המסקנות תהיה שיש לבצע בדיקה ביטחונית לאדם לפני שמאפשרים לו החדרת נתונים רפואיים.

בכל מקרה על בתי החולים והאמונים עליהם להבין כי מתווה האיומים השתנה, ופציינט עוין יכול לפגוע באמצעות מערכות המחשוב בגופם ובשלומם של המאושפזים האחרים.

על המחבר

אמיתי הינו חוקר של סוגיות אבטחת מידע, לוחמת סייבר, פרצות במערכות פיזיות וכשלים בשיטות עבודה. כמו כן, הוא מרצה בנושאי אבטחת מידע בפני פורומים שונים, ונוטל חלק בקבוצת מחקר בנושא סייבר באחת מהאוניברסיטאות בישראל.

דברי סיום

בזאת אנחנו סוגרים את הגליון ה-44 של Digital Whisper. אנו מאוד מקווים כי נהנתם מהגליון והכי חשוב- למדתם ממנו. כמו בגליונות הקודמים, גם הפעם הושקעו הרבה מחשבה, יצירתיות, עבודה קשה ושעות שינה אבודות כדי להביא לכם את הגליון.

אנחנו מחפשים כתבים, מאיירים, עורכים ואנשים המעוניינים לעזור ולתרום לגליונות הבאים. אם אתם רוצים לעזור לנו ולהשתתף במגזין Digital Whisper - צרו קשר! בנוסף, אנחנו עדיין מוסרים חתול מדהים בשם צ'ייסר, מי שמעוניין - שישלח מייל!

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 אוגוסט.

אפיק קסטיאל,

ניר אדר,

31.07.2013