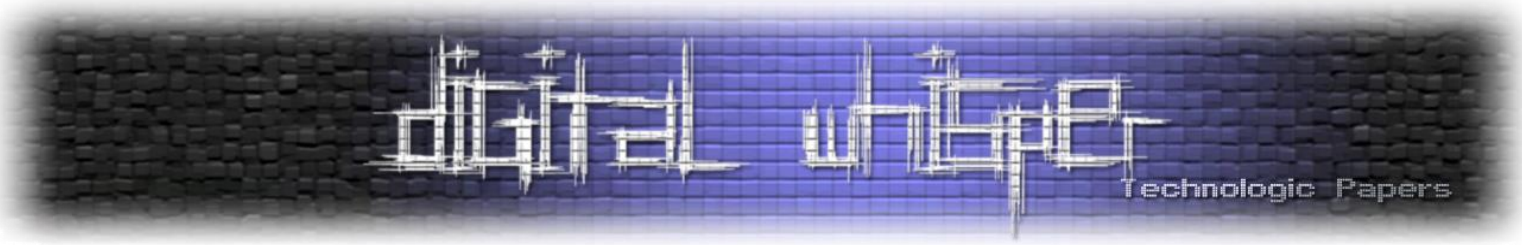


Digital Whisper

גליון 55, נובמבר 2014

מערכת המגזין:

מייסדים:	אפיק קסטיאל, ניר אדר
מוביל הפרויקט:	אפיק קסטיאל
עורכים:	שילה ספרה מלר, ניר אדר, אפיק קסטיאל
כתבים:	d4d, CISA Dragon ושחר קורוט (Hutch)

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

דבר העורכים

ברוכים הבאים לגיליון ה-55! הגיליון של חודש נובמבר!

החודש (ובכלל הזמן האחרון), היו לא מעט אירועים מעניינים בעולם אבטחת המידע, אם מדובר ב-CVE 2011-4862 (החולשה ב-Ironport WSA של סיסקו), אם זה R7-2014-17 (הממצאים של המחקר אודות החולשות ב-NAT-PMP כחלק מה-Project Sonar של Rapid7), אם זה חשיפת ה-MITM של הסינים עבור ה-iCloud של חברת Apple, אם זאת CVE-2014-4114 (חולשת ה-"SandWorm" שהתגלתה לאחרונה ב-OLE של מיקרוסופט), אם זאת CVE-2014-3566 (החולשה "POODLE" שהתגלתה במימושים השונים של SSL), אם אלו מתקפות ה-DDoS השונות שהתגלו דרך SSDP Reflection שהגיעו אף אל מעל ל- 120 GBps!), אם זה אירוע ריקון הכספומטים באירופה בעזרת התולעת Tyupkin שחברת Kaspersky גילתה, ואם זה עוד לא מעט אירועי אבטחת מידע שיצא לנו לשמוע עליהם החודש.

תמיד כשנראה שאי-אפשר להפתיע אותנו וש"כבר שמענו על הכל" - צצה לה איזו ידיעה שמצליחה להפיל אותנו מהרגליים, אם זאת ה-HeartBleed שהפתיעה אותנו בחודש אפריל, ואז לאחריה חולשת ה-ShellShock שהפתיעה אותנו בספטמבר, ועכשיו - POODLE, שאחרי המתקפות CRIME, BEAST, BREACH, ו-LUCKY13 חשבנו שלא נשמע על חולשות ב-SSL בזמן הקרוב...

אך עם זאת, נראה כי עדיין ישנם ארגונים שלא לומדים לקח, ולמרות שאין שום סיבה לחשוב שעולם אבטחת המידע "יעמוד במקומו", נראה שזאת עדיין נקודת ההנחה שלהם, ארגונים כאלה נפרצים על בסיס יומי, ומידע פרטי ומסווג מתפרסם באינטרנט ופוגע בהם, בלקוחותיהם ולפעמים אף מעבר.

כיצד ניתן למנוע זאת? אני לא בטוח עד כמה זה אפשרי. אם הארגון שלך הוצב כמטרה - כנראה שהוא יפרץ, הכל תלוי בכמות המשאבים שהתוקפים מעוניינים להשקיע. וכאן בדיוק אפשר לפעול: אם נצליח להגיע למצב בו לתוקפים לא יהיה שווה לתקוף אותנו, כי העלות של התקיפה תגדל מעל התועלת שהם יקבלו - כנראה שניצחנו, והם יעברו לארגון שפשוט יותר לתקוף.

איך עושים את זה? זאת כבר שאלה אחרת, זאת כבר שאלה שקצרה היריעה מלהכיל את תשובתה, אבל אם תשאלו אותי, ההתחלה של התשובה שלה מתחילה במילה **מודעות**.

וכמובן, לפני שנגש לעיקר הדברים, נרצה להגיד תודה רבה ל-**d4d**, תודה רבה ל-**CISA Dragon**, תודה רבה ל**שחר קורוט (Hutch)** ותודה רבה ל**שילה ספרה מלר** שבזכותם אתם קוראים שורות אלו.

קריאה מהנה!

ניר אדר ואפיק קסטיאל.

תוכן עניינים

Hacking Games For Fun And (mostly) Profit - חלק ב'

מאת d4d

הקדמה

בחלק א' של המאמר הסברנו מה המטרה שאנו רוצים להשיג במחקר והקמנו סביבת עבודה על מנת שנוכל לבצע את הניתוח. הראינו ריצה ראשונית של סביבת העבודה ועצרנו בדרך שבה משתמשים מתחברים לשרת IRC של המשחק WWP.

זהו חלק ב' במאמר, חלק זה מדבר על הנושאים הבאים:

- סוג ההצפנה שבה נעשה השימוש בפרוטוקול
- ניתוח של מנגנון האימות המכניס משתמשים לשרת IRC
- ניתוח ההצפנה של רשימת המשחקים בשרת

החלק הבא במאמר ידבר על הנושאים הבאים:

- פקודות נוספות שיש בפרוטוקול של WWP
- יצירת אמולטור ל-WormNET2 כך שיוכלו לשחק דרכו.

ניתוח ההצפנה של הפרוטוקול

יש כמה פעולות אותן ניתן לבצע על מנת להבין באיזו הצפנה השתמשו במשחק אני אמנה את כולם מהפשוט ביותר למורכב ביותר:

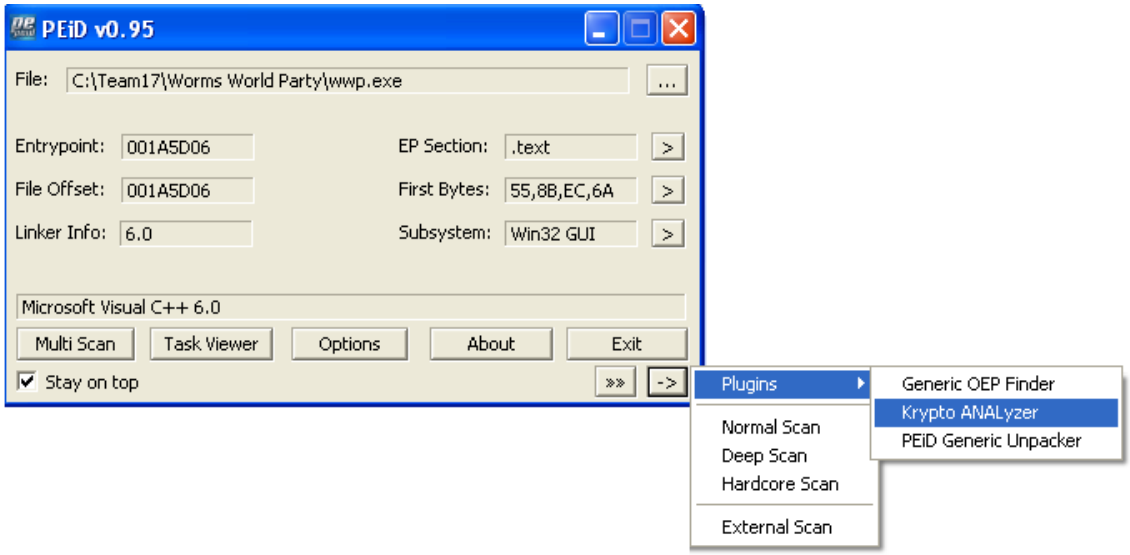
1. שימוש בסקריפטים/פלאגינים ל-IDA Pro/PEiD/Ollydbg אשר יודעים לחפש קבועים בזיכרון.
2. שימוש ב-FLIRT (פיצ'ר הקיים ב-IDA Pro) הנועד לשימוש בעת הצורך בזיהוי פונקציות ספרייה.
3. לקמפל לבד ספריות הצפנה נפוצות עם אותו קומפיילר שהיה בשימוש בעת הקימפול של המשחק המקורי ולהשתמש ב-BinDiff.
4. לעשות Reverse Engineering לפונקציות ההצפנה במשחק המקורי ומתוך כך להבין את האלגוריתם.

אסביר בפירוט את כל השלבים המוזכרים למעלה, ולאחר מכן את תהליך הניתוח שאנו ביצענו, חלק מהבחירות לא היו טובות ואציג גם את הטעויות בכדי שתוכלו ללמדו גם מהן.

שימוש בסקריפטים / פלאגינים

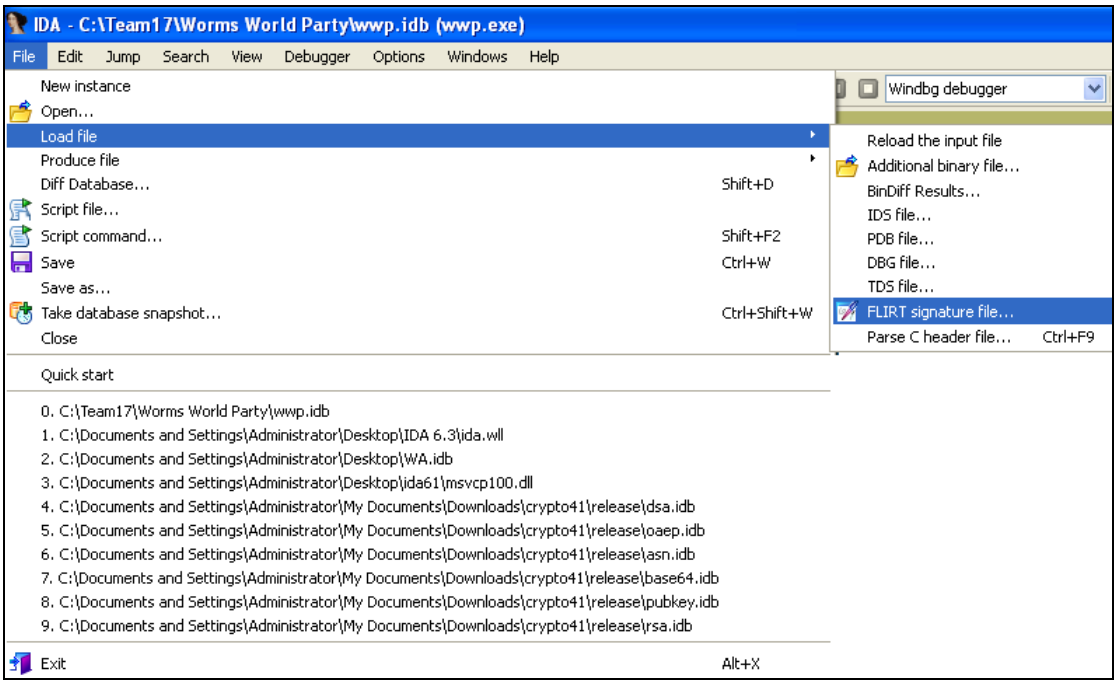
ישנו פלאגין שנכתב ל-IDA Pro בשם FindCrypt אשר יודע לזהות קבועים של צפנים ידועים ולסמן מה שם ההצפנה שהשתמשו בה ואיפה בקוד נמצאים אותם הקבועים.

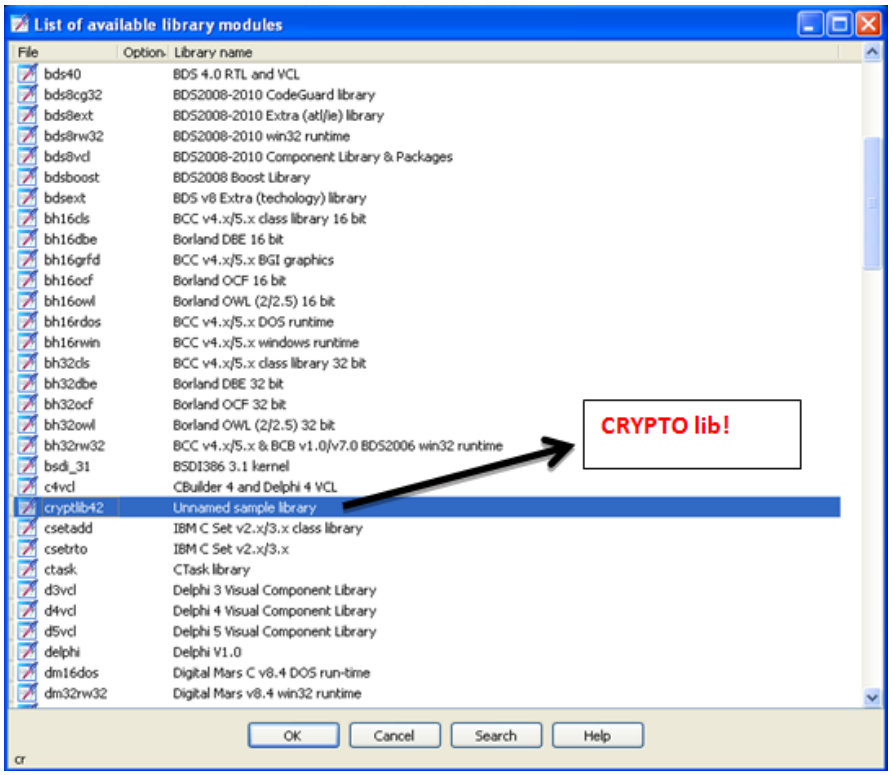
ניתן להשתמש בפלאגין שנכתב ל-PEiD שיכול גם לזהות חתימות של קבצים, למטה ניתן לראות תמונות מתוך PEiD על מנת לזהות הצפנות:



שימוש ב-FLIRT

FLIRT הינו כלי רב עוצמה שיש ב-IDA Pro בעזרתו ניתן לזהות פונקציות של ספריות סטנדרטיות ככה שהוא יביא את השמות וימנע ביצוע Reverse Engineering מיותר. כך ניתן להעלות חתימות ל-FLIFT:





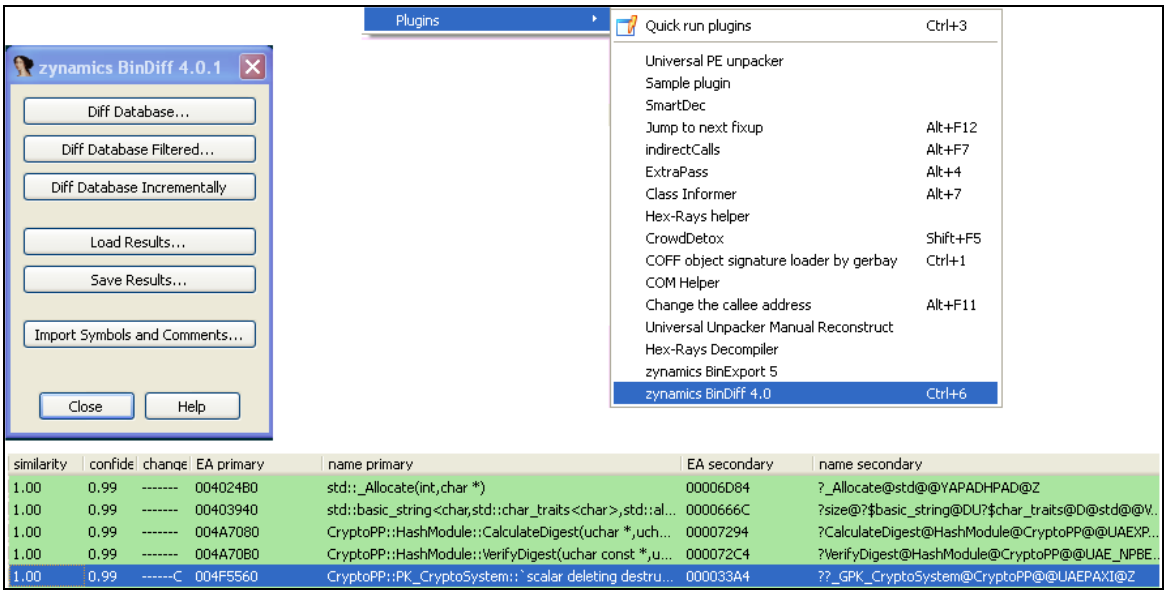
קיימת אפשרות נוספת והיא להעלות חתימות נוספות ל-FLIRT רק צריך לשים אותם בתיקיה sig.

ובנוסף, ניתן גם ליצור חתימות חדשות משלנו, אך לשם כך צריך ספריה סטטית (.lib) של הספריות הצפנה אותם אנו צריכים ואת הכלים של flair 6.x.

שימוש ב-BinDiff

ניתן להשתמש ב-BinDiff על מנת לזהות אם יש דמיון בין פונקציות מסוימות. לשם הדוגמא, נניח כי קימפלנו את כל הספריה CRYPTO, נוכל לפתוח את המשחק ב-IDA עם הספרייה הסטטית שאנו מנסים לבדוק, במקום שניצור לה חתימה.

כך ניתן לבצע זאת בעזרת BinDiff:



שימוש ב-Reverse Engineering

כאמור, ניתן לבצע Reverse Engineering לפונקציות הצפנה על ידי שימוש בדיבאגר ולראות איך הפונקציה מבצעת את הפענוח ולשלוף משם את המפתחות אשר בשימוש.

אפשר כמובן לבצע Reversing לכל ההצפנה ולבצע משהו שיבצע סימולציה לכל המנגנון הצפנה עם המפתחות.

תהליך הניתוח של מנגנון האימות

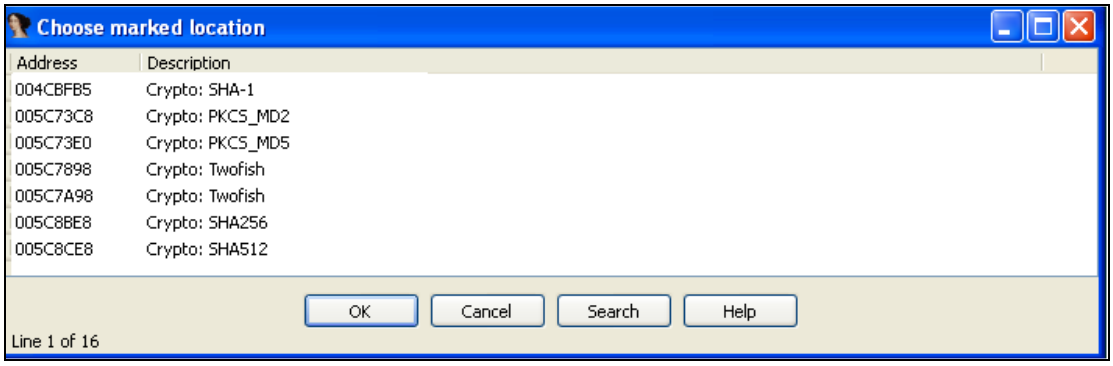
בחלק זה אפרט את תהליך הניתוח שבוצע בפועל על מנת להבין את שיטת ההצפנה שהייתה בשימוש לאימות המשתמש לשרת.

אנו השתמשנו ב-FindCrypt וזיהינו שמדובר בהצפנה בשם Twofish, הצפנה הנ"ל סיימה כאחת מחמשת ההצפנות המועמדות ל-AES אך בסופו של דבר, עקב היותה איטית יותר עבור מקרים מסוימים, היא נפסלה. הצפנה זו הייתה בשימוש ב-WWP אך בוצעו בה כמה שינויים קלים ולכן לא היה ידוע אילו שינויים בוצעו. בעקבות אותם השינויים שבוצעו, בחרנו לבצע Reverse Engineering לקוד ולבדוק אילו שינויים עשו ועד כמה היא שונתה מאחותה המקורית.

בתמונה למטה ניתן לראות את אותו קטע הקוד שהוצג בחלק א', זה היה קצה החוט שלנו בשביל לדעת מה אנו רוצים לפענח:

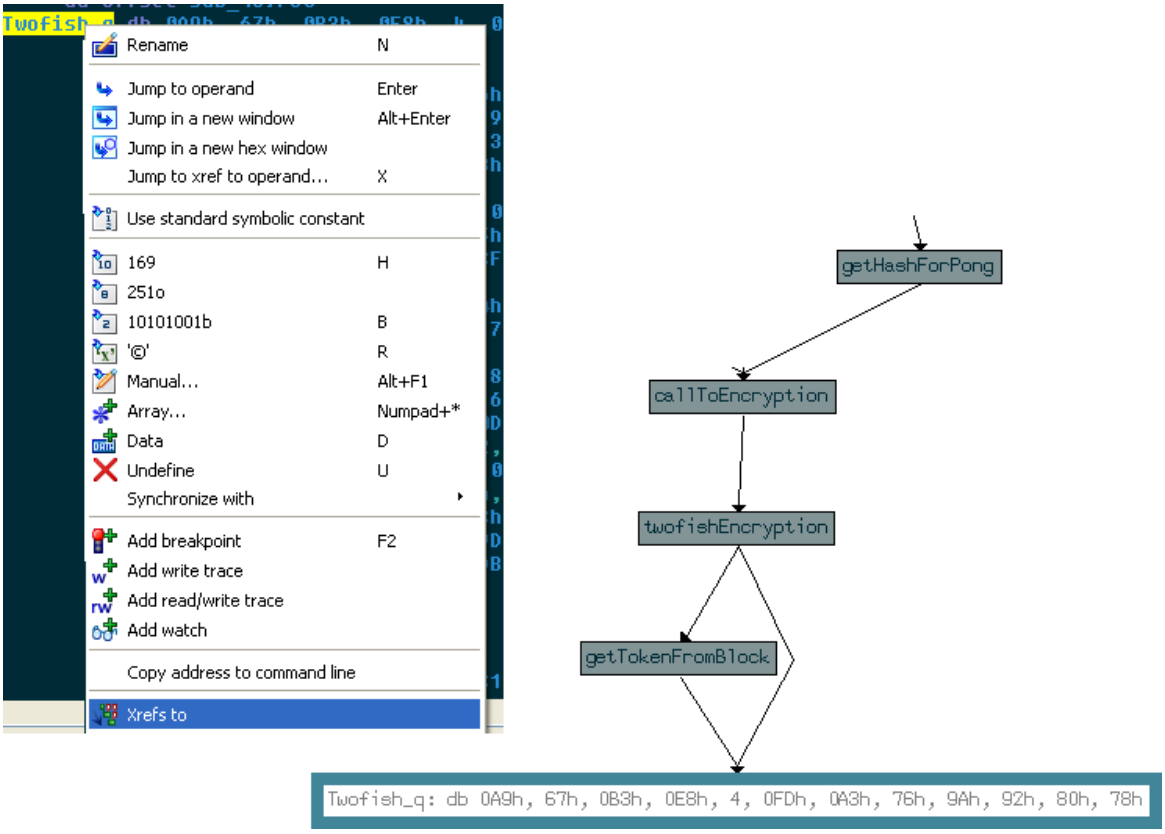
```
.text:0043D55F                          loc_43D55F:
.text:0043D55F 8B 4D 08                 mov     ecx, [ebp+arg_0]
.text:0043D562 89 4D F0                 mov     [ebp+var_10], ecx
.text:0043D565 83 7D F0 00              cmp     [ebp+var_10], 0
.text:0043D569 74 57                    jz      short loc_43D5C2
.text:0043D56B 8D 4D EC                 lea     ecx, [ebp+var_14]
.text:0043D56E E8 41 7F 16 00           call    CString::CString(void)
.text:0043D573 C7 45 FC 00 00 00+       mov     [ebp+var_4], 0
.text:0043D57A
.text:0043D57A
.text:0043D57A 8B 55 0C                 mov     edx, [ebp+buffer]
.text:0043D57D 52                       push    edx
.text:0043D57E B9 EC 83 79 00           mov     ecx, offset keysOffsets
.text:0043D583 E8 84 4F 06 00           call    getHashForPong
.text:0043D588
```

לאחר שימוש ב-FindCrypt קיבלנו References לכתובות בהם יש שימוש ב-Twofish (ctrl+m ב-IDA).
בתמונה הבאה ניתן לראות את מספר הצפנים שנמצאו בשימוש בקוד:



אחת מההצפנות הובילה לפונקציה בשם "getHashForPong" (שזו הפונקציה שנדבר עליה בהמשך - לה אנו זקוקים על מנת לאמת האם משתמש הוא אכן משתמש של המשחק או סתם קליינט IRC).

התמונה הבאה ממחישה זאת לא רע:



לאחר מכן התחלנו להסתכל על הפונקציה getHashForPong ולראות איך היא עובדת. בהתחלה חשבנו להשתמש ב-FLIRT על ספריית CryptoPP, אך בגלל שלא היו לנו את כל הכלים הדרושים על מנת ליצור חתימות - ביצענו ניתוח מלא של הקוד, מה שהתברר כשגיאה שהיה עדיף להימנע ממנה.

חשוב לציין שעל מנת לבצע זאת, כתבנו קובץ DLL שיביא אותנו ישירות לפונקציה getHashForPong בלי להריץ את כל המשחק כפי שהוזכר בחלק א'. הסתכלנו שוב על קטע הקוד שהצגנו לפני:

```
.text:0043D57A 8B 55 0C                 mov     edx, [ebp+buffer]
.text:0043D57D 52                       push    edx
.text:0043D57E B9 EC 83 79 00           mov     ecx, offset keysOffsets
.text:0043D583 E8 84 4F 06 00           call    getHashForPong
.text:0043D588
```

ומההסתכלות, ניתן לראות כי הפרמטר הראשון הוא ה-buffer שמכיל את הקלט הבא:

```
Fjs9GP7Fl1l3+uuu6wcUbesb54ed1JO2JlSuaxJfjqgb9bt7
```

הפרמטר משתנה כל פעם שמתחברים לשרת, אז היה לנו קל יותר להבין מה המפתח להצפנה במקרה הזה. נכנסנו לפונקציה getHashForPong וניסינו להבין מה עושים ל-buffer של הפרמטר הראשון, לאחר חיפושים הגענו למסקנה שמדובר בסוג של base64, אך חשבנו שאולי מדובר במעין שפה שהמפתחים של WWP פיתחו בגלל שהמימוש היה קצת לא שגרתי. הנה קטע מימוש של base64 מתוך ספרייה של CryptoPP:

```
int Base64Decoder::ConvToNumber(byte inByte)
{
    if (inByte >= 'A' && inByte <= 'Z')
        return (inByte - 'A');

    if (inByte >= 'a' && inByte <= 'z')
        return (inByte - 'a' + 26);

    if (inByte >= '0' && inByte <= '9')
        return (inByte - '0' + 52);

    if (inByte == '+')
        return (62);

    if (inByte == '/')
        return (63);

    return (-1);
}
```

[מי שמעוניין לראות את כל המימוש של base64 ב-CryptoPP יכול להסתכל על base64.cpp ו-base64.h בתוך הפרויקט שלהם]

המפתחות של ההצפנה במשחק WWP שמורים בתוך הקובץ בינארי ומקודדים ב-base64 והמשתנה הגלובאלי keysOffsets מכיל את ה-offset של תחילת המפתח, גודל המפתחות הינו 32 בתים (256 ביטים).

בתמונה הבאה ניתן לראות קטע מהבלוק שמכיל את המפתחות עם קידוד של base64:

```
encryptionKeys db 't6bZ54NrRGFsfDjwEgwCIbzr6k2VVIpTyvEjEEK0yM+h60ExS9y3qDliHd1ALvOZz'
     db 'qmedZ6MYuKcUvBnPNwkv0VfrZ3PQ0UWLmcsGmR2yBq6DxNDxEdacE/ZFRF35Frrjj'
     db '3WGbMgnNXgqu4STetbiiE7mPM3A36D8aOBb6YEpsOU9J6c7V4zwEJpyRBXfYH0CI2'
     db 'zYvvfPrCUCYIHRY2hLmRk0PBt4efYJUwEjb+vTYWJ3A/sZ632Ia12csrp2H1dMVdm'
     db 'ZPNXiORo+xS8Y0JOZfYcyV6uanmgwBDyLcOQeejl9dugS3OIbd5wwK3uUhIoIB7d8'
     db 'sN2jWOgkUCTiK3TD+0UfH47q/skfWd4uq+Sr1Le3cSi4gZtEYWCuzl8vChKpjoWb='
     db '=3QGqiy0KS9V+VyqluzdSqLa+LXVcdByAGY1syYLGQru4hsZo6FiAOTRGE0IQty13'
     db 'B2FFF9JJkXShkVrBv+42XSYBBrHfux00fiBeb71CCdcs/cKcRPs9ODeGWG/26+Z9h'
     db 'Ggcmz9F845gSO+z3+4ijxKbQkLYhYYyhzmaFBDaCbxgoWAcl9mwG0H8GDo7Rck0sn'
     db 'cwkvwwk/S1IG3IBJxemeCbshXkGkRSOZ1s2l5KQU/+zrJ+Fbc3YW7jp9voHBiN9S6'
     db 'CNqZHXJ+joCCJw9/cqcOgYW7Vuyt03iRGWLVRSa4maajjVFYMXWmAv4PTW8Sn6xrO'
     db '4U1DatLrabBC6PLEnAEmKM5Ir1zxEwd75COTOWoh+vyRcc0C4cgRKgKse35ga6bCt'
     db 'QL2ha2R5t6uC5Zx/pKKhEH94CRrDDVIk1yejkj2906RFL/dcdKSLZKJ9yo65cES5R'
     db 'cknEXdzKqI86C6X7Iqg7hmqOJLZix9YsdefnWXaEzOHwHC6slgIj3t87aMFcm7M7N'
     db 'dSHLYyFLwYXONQ4toQifWHrRZpVfmfl6yfJ9NHx9+K6cRTdi9UqHBq5tIF15k0u2Q'
     db 'qH/jVmfWsd1nhhX1c/3q6Dil+FuGrvWnUO/OrpaBgR0ztmo0zk37CwGN2Cg4nGNl+'
     db 'x/O97cux+ChvQV6Vfki92ndzDP5Cx1m5PYfJRxksVSbxYGTwUaHIELoLutskRRgtK'
     db '32XBlJWgByiCv3=FmDcPxuJBk3cnXA7YmY2xq6R3+hYaW6sJZzautMWYa5sJmwdJi'
     db 'EaFbVq5Q7myuw+z6L4ryNTsGuIOMKVL2SaX3+MRfDI7jXhBWxca8RJJW/9WWOzUyq'
     db '9F3MxoiOkMItnG+J9UK5FylGUqOUcFwVq4sb1P7KCKbz3Zz1seq7knDaOvU6xae6q'
     db 'kDIkO5Ctscq9FYpoyw/fFEOKqE80wBwLmISN8hgwkP8XBvnemNjO9SWCnahdMRKMM'
     db 'E7I81LjJF7FxveSnmA2z/r4ybqiBSniON79iJqRfwsOCR9WCEHQ3nxHA+h9znsNq5'
     db 'FT0gFRnF+736t7DI6F3FI3WFXZEaR2MoqFmN0TIFFPOTgMCzRO0yURyxV5f6K9C+O'
     db 'bsw0TFN3AbnMyCZUD7MGAqiaF0iHIJRn03pjL8iYuCrdIS4k4BOXyHIYHh1YXkeDR'
     db 'OZcmcr0NPEcaUUiKRgvNLFF/pvRb3otYSizS8FdUVesQJkh1hAANGdsed3K1PcoTN'
     db 'g+mDafagwr+Au176aJCsf9Xcvo29AVyIOnCwNZkTeJQ4C+H9SOk29Ot3ALGjT3Yz1'
     db 'D0H0TMHSmLMYQ6C+tO6YZu5uvdCFRJ9oidkIwiwF7NerBYPPd2RuJ4lauRjVesHur'
     db 'YHK6GKRRj7JKxo8Pg4uJcHIDA+E9=AjGv5+it4uEgzAlxs26VwGwemW6R83JZFS7u'
     db 'mv15HDBbCCRGXCDuNZ0WHCcLDY8oYct6WwofJGUrrm8TvqFpaV5KVqpRJzRtvaEdE'
     db 'fJ7YLW/OJU2LB18Lc5MfBE1M1Ax+1LVORVUbXcGuF/n1lohr8AT2cUFGmd3g5kSc'
     db 'qTaS5b1RnaAhFTn+m6LAlNv/b6dOav71WT7L04TOX4Vedd0dGOW4HjmSrRCrMUFZ8'
     db 'E8cNjpMI6n9Ecxo+nDVhtCRTBRjuUBWFsNzioRugtFXdZx3gX71bCUhsQKDXy7xRB'
     db 'uRgaRC292mTJxK6ZrWalBlzvap3yrjuJDdA+8rSeGu2pDuRw56vyuXoqMAKvS9q7U'
     db 'mgzc9G/f5s/bLB4xT3pbL4AFzvvwd/t4bA4kXZ2W1Bvd+CdGIicFJpLavZTScxyxt'
     db 'AbG9+4XVB8ZOFvbHakOhmCixPirpNCNSuYsuxb0fN3F6IisNuCnTualm+ADH7zFHw'
     db 'JAUcvkYjjsxVtluODqymtYTEb997TEH8+oz8+Jb2XdtPFkOeNXRzQ+YBsqXUtjPTO'
     db 'Z32DE6BG774N39uFo+vYirMGNSHIv1F6a8dCpXXMA+P9IvEylEkpEHRU/xyjCA+26'
```

קטע הקוד לחישוב ה-offset הנכון נמצא במקום הבא:

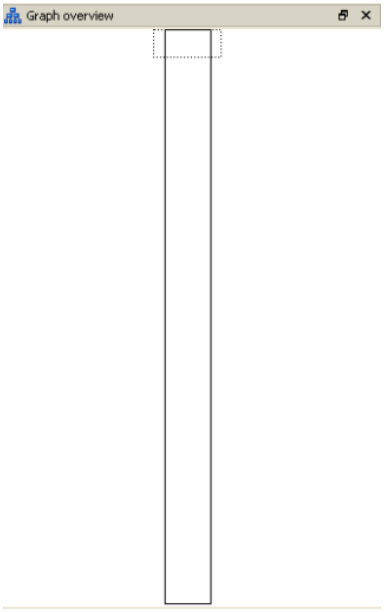
```
.text:004A25C1                  loc_4A25C1:
.text:004A25C1 6A 01                push    1
.text:004A25C3 8B 8D 04 FE FF FF    mov     ecx, [ebp+var_1FC]
.text:004A25C9 51                   push    ecx
.text:004A25CA 6A 01                push    1
.text:004A25CC BA 71 6A 5F 00       mov     edx, offset aLdisable_bmp
.text:004A25D1 8B 85 0C FE FF FF    mov     eax, [ebp+unkClass4]
.text:004A25D7 2B 50 04             sub     edx, [eax+4]
.text:004A25DA 52                   push    edx
.text:004A25DB 8D 4D C0             lea     ecx, [ebp+var_40]
.text:004A25DE E8 6D 7D 02 00       call    calculateStaticKey
```

בכדי למצוא את המפתח, היה צריך לעקוב אחרי כל הפונקציות. בגלל שמדובר ב-C++ חלק מהדברים לא ברורים וצריך לעקוב אחרי כל המשתנים שמעבירים את המפתח עם דיבאגר בשביל להבין את זה ביתר קלות.

לאחר שימוש בדיבאגר שלפנו את המפתח הנכון שצריך לשלב אימות המשתמש, המפתח שקיבלנו היה:

```
\xD5\x2D\x31\x08\xFB\x0F\x54\x9E\x6D\x7A\x0F\xFD\xEE\xDC\x21\x9A\xD4\xA6\x84\x24\x6D\x61\xFA\x8A\xAE\x98\x96\xA1\xF1\x63\x1A\x8D
```

המרנו את כל קוד האסמבלי לקוד C (ידנית), המרנו בשלבים חלקים של ההצפנה (לפי הסדר שמופיע בפונקציה getHashForPong) ואחת הפונקציות שנאלצנו לנתח זה הפונקציה המוצגת בגרף:



הפונקציה המוזכרת למעלה היא פונקציה שמבצעת הרבה חישובים מתמטיים שקשורים להצפנה של Twofish הנקראים פייסטל, זו אחת הפונקציות הכי חשובות להצפנה, והיה הכי קשה לבצע לו סימולטור שיבצע את אותה פעולה בדיוק.

ב-Twofish קיימים 16 מבני פייסטל מה שיצר קוד מאוד ארוך, אנחנו לא מתמטיקאים וקריפטוגרפים שיכולים להבין למה החישובים האלה מצליחים לייצר הצפנה מספיק טובה, אז נאלצנו לשחזר את הכל בצורה ידנית. לאחר מכן מצאנו ספרייה סטנדרטית של twofish והצלחנו להחליף את הקוד המגעיל שלנו בקוד שלה.

שמנו לב שהפונקציות CryptoPP לא השתנו אך נוספה עוד פונקציה שהמפתחים של WWP הוסיפו. המפתחים של WWP הוסיפו את הפונקציה הבאה:

```
.text:004A2745 C6 45 FC 05          mov     byte ptr [ebp+var_4], 5
.text:004A2749 8B 95 30 FE FF FF    mov     edx, [ebp+var_1D0]
.text:004A274F 83 EA 10             sub     edx, 10h                ; 0x14
.text:004A2752 52                   push    edx                     ; length
.text:004A2753 8D 85 D0 FE FF FF    lea     eax, [ebp+halfPart]
.text:004A2759 50                   push    eax
.text:004A275A 8D 8D C0 FE FF FF    lea     ecx, [ebp+partOfSecondBlock]
.text:004A2760 51                   push    ecx
.text:004A2761 8D 8D 9C FE FF FF    lea     ecx, [ebp+var_164]
.text:004A2767 E8 64 6B 02 00       call    sub_4C92D0              ; combination of 2 buffers output the hash for pong
```

מהסתכלות על הפונקציה, ניתן לראות כי זו מורידה 16 מהגודל של הבאפר. למה? מסתבר שהם מורידים את ה-16 בתים ראשונים של כל באפר שהם מפענחים, זה אומר שה-16 בתים הראשונים בכל ההצפנה שמגיעה בפרוטוקול לא מעניינים אותנו. הפענוח בפונקציה מתחיל רק ממיקום 16 בבאפר.

בתמונה הבאה ניתן לראות את קטע הקוד שמראה את מה שניסיתי להסביר:

```
.text:004C92E5
.text:004C92E5                  ; while(i > 0) {
.text:004C92E5
.text:004C92E5 8D 68 01               lea     ebp, [eax+1]
.text:004C92E8
.text:004C92E8                  loc_4C92E8:                             ; CODE XREF: sub_4C92D0+51↓j
.text:004C92E8 8B 44 24 18            mov     eax, [esp+10h+buf2]       ; buf2[0x10]
.text:004C92EC 8B 56 20               mov     edx, [esi+20h]
.text:004C92EF 8A 18                  mov     bl, [eax]                 ; buf2[0x10 + i]
.text:004C92F1 40                     inc     eax
.text:004C92F2 89 44 24 18            mov     [esp+10h+buf2], eax       ; buf2++
.text:004C92F6 8B 46 1C               mov     eax, [esi+1Ch]
.text:004C92F9 3B D0                  cmp     edx, eax
.text:004C92FB 75 08                  jnz     short loc_4C9305
.text:004C92FD
.text:004C92FD
.text:004C92FD 8D 4E 04               lea     ecx, [esi+4]
.text:004C9300 E8 5B FF FF FF         call    encryptHalfPart
.text:004C9305
.text:004C9305                  loc_4C9305:                             ; CODE XREF: sub_4C92D0+2B↑j
.text:004C9305 8B 46 20               mov     eax, [esi+20h]            ; pos
.text:004C9308 8B 4E 18               mov     ecx, [esi+18h]            ; partOfAuthPong
.text:004C930B 03 C8                  add     ecx, eax                  ; partOfAuthPong[pos]
.text:004C930D 8A C3                  mov     al, bl
.text:004C930F 8A 11                  mov     dl, [ecx]
.text:004C9311 88 19                  mov     [ecx], bl
.text:004C9313 32 C2                  xor     al, dl                    ; buf2[i] ^ buf[i]
.text:004C9315 8B 56 20               mov     edx, [esi+20h]
.text:004C9318 42                     inc     edx                       ; pos++
.text:004C9319 47                     inc     edi                       ; i++
.text:004C931A 89 56 20               mov     [esi+20h], edx
.text:004C931D 88 47 FF               mov     [edi-1], al               ; authpong hash
.text:004C9320 4D                     dec     ebp
.text:004C9321 75 C5                  jnz     short loc_4C92E8
```

ולאחר מכן, הגענו לתוצאה הבאה:

יצרנו אמולטור ב-C++ שיודע לדמות את אותה פעולה בדיוק, (מימשנו את ההצפנה בדיוק כמו שעשו חברת Team17). חשוב לציין שלקח לנו מספר ימים שלמים על מנת כתוב את האמולטור. זו הייתה אחת הטעויות שלנו: במקום לנסות לקמפל מחדש את הקוד באותו קומפיילר כמו שקומפל במשחק.

בעת השימוש ב-Twofish יש מספר דרכים לביצוע תהליך ההצפנה, לא ממש ידענו לומר באיזה מוד היה שימוש בשיטה שלנו, בהצפנה של בלוקים קיימים המודים הבאים: OFB, CFB, CBC ו-ECB. הסבר נוסף ניתן למצוא כאן.

הדרך הנכונה הייתה לקחת תוכנה כגון כל דפדפן שהוא, שיכול לבצע פענוח/הצפנה של התוכן ולנסות על ידי ניסוי וטעייה לקבל את אותו אפקט. יותר מאוחר מצאנו את האתר: online-domain-tools.com. אשר דרכו ניתן לפענח ולהצפין חזרה את צופן Twofish. ביצענו זאת, וזו התוצאה שקיבלנו:

```
ASCII "1A3FD9008D18355D76D18A033B4A57528A47E4C0"

00000000  90 ca 37 7f 53 a6 a7 31  d3 fd a6 e7 c5 c1 23 42
00000010  1a 3f d9 00 8d 18 35 5d  76 d1 8a 03 3b 4a 57 52
00000020  8a 47 e4 c0
```

- למעלה, בצבע **שחור** זה מה שקורה במשחק בפועל בקלט מסוים.
- למטה, כפי שאנו יכולים לראות רק במיקום 0x10 יש לנו תיאום מלא של אותו באפר.

באתר עצמו יש אפשרות לבחור באילו מודים נרצה להצפין ולפענח, וכך גם גילינו שהשתמשו במוד בשם: CFB. לאחר מכן כתבנו קוד ב-PHP שאפשר לשלוח לו את מה שהשרת מצפה לקבל. (קישור לקוד מופיע בסוף המאמר).

כעת, אני מעוניין להזכיר כי בחלק א' דיברנו על כך שחברת Team17 שינו את שרת ה-IRC שלהם והוסיפו לו פקודות נוספות, בהן היו AUTHPING ו-AUTHPONG. פקודות אלו נוספו ככה שבנוסף להגנה על השרת עם סיסמא, יהיה בעצם תהליך Challenge-Response. ראשית, בעת החיבור, נשלח AUTHPING:

```
AUTHPING wormnet.team17.com$1 18B1443B13E28B5E64A79E43643E1C1F0951EFAF
```

ולאחר מכן הקליינט של המשחק היה שולח AUTHPONG באופן הבא:

```
GET http://wormnet2.team17.com:80/wwpweb/RequestAuth.php?Secret=wormnet.team17.com
$1&Challenge=18B1443B13E28B5E64A79E43643E1C1F0951EFAF
```

תפקיד הבקשה הזו הוא ליצור קלט שאותו רק השרת והקליינט של המשחק יודעים.

מפתח הקליינט קרא לפקודה זו: "<ANSWER>", דוגמא לקלט שכזה:

```
<ANSWER Gk1nVu8VE4bVinJTXTLz4v+ph5OUnkmj5+Qs4NJpcMkhUW10>
```

המשחק מפרסר את הפקודה הזו, ובעת מציאתה הוא מפעיל את הפונקציה getHashForPong. וכמובן בכל פעם שאנחנו מנסים לגשת לדף הנ"ל, אנו מקבלים תשובה אחרת.

חשוב לא לטעות כאן, זו אותה תשובה, אך בגלל ש-16 הבתים הראשונים לא מעניינים אותנו הם יוצרים אקראיות לקלט, ככה שיהיה קשה יותר לעשות ברוטפורס לכל האפשרויות ולמנוע Reply Attacks. כעת ניתן להתחבר לשרת ה-IRC ישירות מה-mIRC/xchat. אך יש עוד מספר שלבים שעלינו לבצע בשביל להצליח להיכנס לערוצים ניתן להסתכל על מה שכתב StepS:

All steps are required, including the first three. How to connect to the WWP server externally (the fastest way for now):

1. Load wormnet2.team17.com/wwpweb/welcomeloginform.php
2. Load LoginForm.php?ServerId=1
3. Load Login.php?UserName=*IRC_nickname* (it must only contain letters and numbers, the ` char isnt allowed).

If you are **NOT** redirected to welcomeloginform.php, then everything works and ready for IRC connection.

4. Open the IRC client and connect to wormnet2.team17.com:6677 using the *IRC_nickname* that was provided for Login.php, **case sensitive**.
5. Receive the AUTHPING request with the *Secret* (such as *wormnet.team17.com$1* or *wormnet.team17.com$2*) and with the *Challenge* parameter (example: 4CB909A23B22276FF2594E916130793AC45664D6). **You have only 2 minutes to go now.**
6. Fetch RequestAuth.php?Secret=*Secret*&Challenge=*Challenge*
7. Receive the 48-characters answer in the <ANSWER> tag.
8. Into the authping.ini file (in WWP's folder), input *Secret* and the *Answer* into the respective strings.
9. Start WWP.exe: you will get your encrypted answer immediately. It's also automatically copied to your clipboard, so you only have to paste it into the IRC. DONE.

Note 1: steps 1-3 can be performed *after* starting the IRC connection, but they still must be performed, because they authorize you on the server, and you get a correct <ANSWER> when requesting auth.
Note 2: the User-Agent doesn't matter at all.
Note 3: words in italics are variables.
Note 4: if Login.php fails, then make sure cookies are enabled. It might play some role here.
Note 5: it doesn't matter how many times you reload RequestAuth.php. Although the answer is different, the encrypted result is exactly the same!

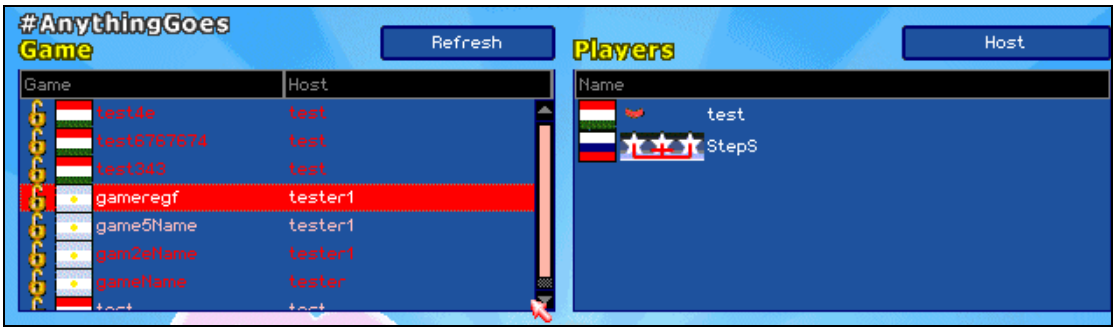
סעיף 8 לא רלוונטי לנו בגלל שיש לנו אמולטור שעובד ב-PHP וגם ב-C++ ☺.

ישנן עוד פקודות מיוחדות שהמשחק מפרסר אבל הם מעבר לחלק ב' של המאמר ויוסברו בחלק ג'.

ניתוח רשימת המשחקים ב-WWP

בחלק זה אפרט את תהליך הניתוח שבוצע על מנת להבין איזה נתונים נשמרים על כל משחק ואיך מתבצעת ההצפנה של רשימת המשחקים. כאשר משתמש מארח משחק, מתווסף לרשימת המשחקים משחק שכל המשתמשים אשר נמצאים מחוץ לערוץ רואים אותו ויכולים לבחור אם להצטרף אליו, לכל ערוץ יש רשימת משחקים נפרדת.

בתמונה למטה ניתן לראות את רשימת המשחקים לדוגמא:



משחקים שצבועים באדום אלו משחקים עליהם אי אפשר ללחוץ ונועדו לציין שיש בעיה במשחק ספציפי.
כל דקה מתבצע רענון אוטומטי של הרשימה על ידי שליחת בקשת HTTP הבאה:

```
GET http://wormnet2.team17.com:80/wwpweb/GameList.php?Channel=AnythingGoes
```

לאחר בקשה זו אנו מקבלים את רשימת כל המשחקים בצורה מוצפנת כמו שנראה פה:

```
HTTP/1.1 200 OK
Date: Mon, 20 Oct 2014 18:47:18 GMT
Server: Apache/2.2.16 (Debian)
X-Powered-By: PHP/5.3.3-7+squeeze14
Vary: Accept-Encoding
Content-Length: 382
Connection: close
Content-Type: text/html

<GAMELISTSTART>
<GAME WaCcerxJWRfFj+7+M6FnArRxaPMXGNUTrEn6+LCh4kZGyaXkA/7T/Bmb1MnvlBpZwCI/
iR5qYr3bStc85prZpfXpB72TuHnH3dMhwaEuwPd+hWug+s+LNPiumo7W2qHPHm97x5JGd/
H7xqjkPx7wIJSKMrlsmQxddtisYbCfb6i6BNLIci5+aHmZXWWvv/
tI5Mh4xeUMu5VORk8nLfmgGHzKDjPfOczvF2lgZxeqgZjCmLqlFllz04YOjFznMTOjSYgDGiLvWnRzl5l5wk1IwTR
lMANK1jonObK5S3BgEz6ZWiyxAjZ365sR7RP0ifWqh+VcT8yj5g6OcFjj4kry7A==>
<GAMELISTEND>
```

הפקודות **<GAMELISTSTART>** ו-**<GAMELISTEND>** אלה פקודות מיוחדות של המשחק המציינות התחלה וסוף של רשימת המשחקים. הפקודה **< ... GAME>** זו פקודה שמציגה את הנתונים של כל משחק אותן נפרט בקרוב.

כאשר הבנו שרשימת המשחקים מוצפנת, שמנו לב שמדובר באותה פונקציה שמפענחת את המידע שצריך בשביל לפתור את ה-Challenge-Response, מה שעשינו על מנת לאמת זאת, היה לשים Breakpoint על אותה פונקציה שמבצעת את הפענוח של ההצפנה וניסינו על ידי כך לראות את הפרטים

שנשמרים עבור כל משחק. התמונה הבאה מציגה את הפונקציה עליה שמנו breakpoint ב-Ollydbg, זו אותה פונקציה שראינו קודם לכן ב-IDA Pro:

```
004C92D0 r$  8B4424 0C     MOV EAX,DWORD PTR SS:[ARG.3]
004C92D4  .  56            PUSH ESI
004C92D5  .  8BF1          MOV ESI,ECX
004C92D7  .  8BC8          MOV ECX,EAX
004C92D9  .  48            DEC EAX
004C92DA  .  85C9          TEST ECX,ECX
004C92DC  .v 74 48         JZ SHORT 004C9326
004C92DE  .  53            PUSH EBX
004C92DF  .  55            PUSH EBP
004C92E0  .  57            PUSH EDI
004C92E1  .  8B7C24 14     MOV EDI,DWORD PTR SS:[ARG.1]
004C92E5  .  8D68 01       LEA EBP,[EAX+1]
004C92E8  >  8B4424 18     MOV EAX,DWORD PTR SS:[ARG.2]
004C92EC  .  8B56 20       MOV EDX,DWORD PTR DS:[ESI+20]
004C92EF  .  8A18          MOV BL,BYTE PTR DS:[EAX]
004C92F1  .  40            INC EAX
004C92F2  .  894424 18     MOV DWORD PTR SS:[ARG.2],EAX
004C92F6  .  8B46 1C       MOV EAX,DWORD PTR DS:[ESI+1C]
004C92F9  .  3BD0          CMP EDX,EAX
004C92FB  .v 75 08         JNE SHORT 004C9305
004C92FD  .  8D4E 04       LEA ECX,[ESI+4]
004C9300  .  E8 5BFFFFFF   CALL 004C9260
004C9305  >  8B46 20       MOV EAX,DWORD PTR DS:[ESI+20]
004C9308  .  8B4E 18       MOV ECX,DWORD PTR DS:[ESI+18]
004C930B  .  03C8          ADD ECX,EAX
004C930D  .  8AC3          MOV AL,BL
004C930F  .  8A11          MOV DL,BYTE PTR DS:[ECX]
004C9311  .  8819          MOV BYTE PTR DS:[ECX],BL
004C9313  .  32C2          XOR AL,DL
004C9315  .  8B56 20       MOV EDX,DWORD PTR DS:[ESI+20]
004C9318  .  42            INC EDX
004C9319  .  47            INC EDI
004C931A  .  8956 20       MOV DWORD PTR DS:[ESI+20],EDX
004C931D  .  8847 FF       MOV BYTE PTR DS:[EDI-1],AL
004C9320  .  4D            DEC EBP
004C9321  .^ 75 C5         JNZ SHORT 004C92E8
```

ידענו איזה פרטים נשמרים על כל משחק ב-WA בגלל ששם לא הייתה הצפנה, אז היה יותר קל לבצע השוואה באופן הזה. כך נראית רשימת המשחקים ב-WA:

```
<GAMELISTSTART>
<GAME normal1v1PROxONLY NNNxKilobyte 2E8B706E.catv.pool.telekom.hu 18 1 0 5914622 1686590792><BR>
<GAME Longplay VolvoDriver 109.90.196.30 15 1 0 5914630 1686586692><BR>
<GAMELISTEND>
```

שמנו לב שב-WWP הוסיפו שדה שמחשב את ה-Checksum של כל הנתונים כדי לבדוק אם לא שונה המידע באמצע "בדרך מסתורית". במידה וה-Checksum לא נכון, המידע לא יוצג ברשימת משחקים.

מבנה של כל משחק מוצג בשפת C בקטע הבא:

```
struct WWPhost
{
   BYTE checksum[20];
   BYTE hostname[20];
   BYTE username[20];
   DWORD ipInDecimal;
   DWORD gameId;
   DWORD countryFlag;
   BOOL serverUsage;
   BOOL isPasswordGame;
   timer_t timestamp[2];
}
```

ופירוט:

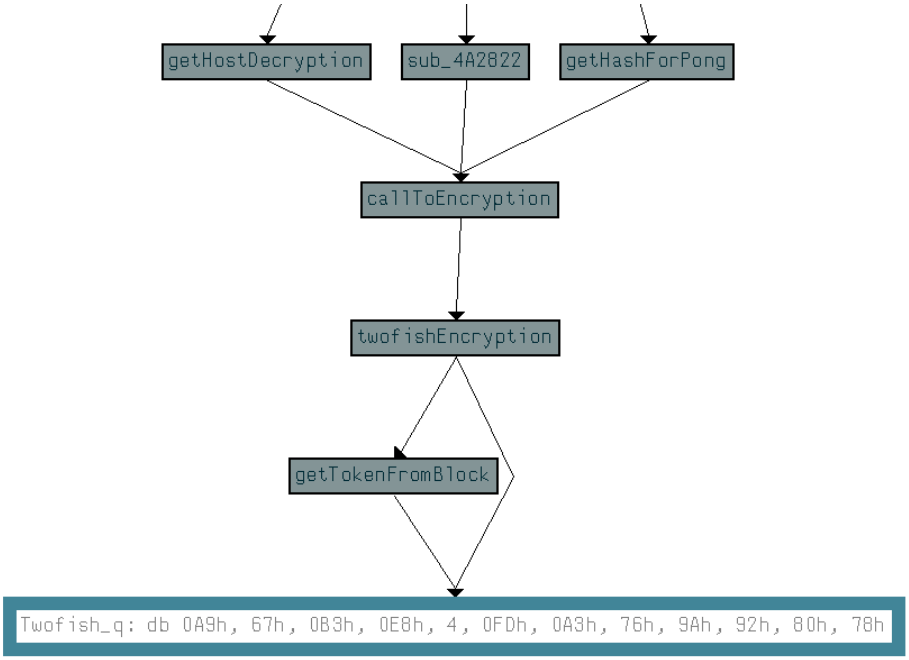
שם שדה	גודל שדה בבתים	תיאור
checksum	20	שדה זה מכיל checksum של כל השדות האחרים על ידי חישוב האש מסוג RIPEMD160
hostname	20	שדה זה מכיל את שם המשחק
username	20	שדה זה מכיל מחרוזת של השם משתמש
ipInDecimal	4	שדה זה מכיל את כתובת ה-IP כמספר רגיל
gameId	4	שדה זה מכיל את ה-ID של המשחק
countryFlag	4	שדה זה מכיל את מספר הדגל, ערכים בין 0 ל-52
serverUsage	4	שדה זה שווה ל-0 אם המשחק תקין, אם המשחק לא תקין הוא שווה ל-1 ויוצג ברשימת משחקים בצבע אדום
isPasswordGame	4	שדה זה שווה ל-1 אם המשחק עם סיסמא, אם לא אז 0
timestamp	8	שדה זה מכיל את מספר השניות שעברו באינדקס 0 שדה זה מכיל את מספר השניות שעברו + 3600 שניות(שעתיים) באינדקס 1

מי ששם לב יוכל לראות שיש די הרבה שינויים בין הפרטים שנשמרים לכל משחק ב-WA ל-WWP. ב-WWP נשמרים הרבה יותר פרטים על מנת לשפר את האבטחה של השרת. השגנו את כל המידע הזה עוד לפני שהתחלנו לבצע Reverse Engineering להצפנה של רשימת המשחקים.

אחרי שביצענו Reversing להצפנה של המשחק קיבלנו תמונה ברורה יותר על מה כל שדה במבנה מייצג, תהליך הניתוח של הצפנת המשחקים יוסבר בקטע הבא.

ניתוח ההצפנה של רשימת המשחקים ב-WWP

בחלק זה נביא את תהליך הניתוח שבוצע על מנת להבין את ההצפנה של רשימת המשחקים. ברשימת משחקים יש גם שימוש ב-twofish הסתכלנו שוב על כמות ה-references שיש. קודם צנזרנו את הקריאות לפונקציות אחרות על מנת למנוע בלבול. התמונה הלא מצונזרת מופיעה למטה:



כעת אנחנו מסתכלים על הפונקציה getHostDecryption שתפקידה הוא לפענח את פרטי המשחק המוצפנים עם twofish. הסתכלנו על הפונקציה getHostDecryption ושמנו לב שיש פעמיים קריאה לפונקציה שדרכה נבחר המפתח הנכון. מכך הסקנו שיש חלק נוסף לרשימת משחקים שנועד כנראה "לסבך את העניינים". בתמונה הבאה ניתן לראות את הפונקציה getHostDecryption בהם יש פעמיים קריאה לבחירת מפתחות:

```
.text:004A2C5A                  loc_4A2C5A:                             ; CODE XREF: getHostDecryption+97↑j
.text:004A2C5A 6A 01                push    1
.text:004A2C5C 8B 8D 2C FE FF FF    mov     ecx, [ebp+var_1D4]
.text:004A2C62 51                   push    ecx
.text:004A2C63 6A 01                push    1
.text:004A2C65 BA C4 63 5F 00       mov     edx, (offset encryptionKeys+1A1Ch)
.text:004A2C6A 8B 85 34 FE FF FF    mov     eax, [ebp+keysOffset]
.text:004A2C70 2B 10                sub     edx, [eax]
.text:004A2C72 52                   push    edx
.text:004A2C73 8D 8D 64 FF FF FF    lea     ecx, [ebp+var_9C]
.text:004A2C79 E8 D2 76 02 00       call    calculateStaticKey

.text:004A2D10                  loc_4A2D10:
.text:004A2D10 6A 01                push    1
.text:004A2D12 8B 95 24 FE FF FF    mov     edx, [ebp+var_1DC]
.text:004A2D18 52                   push    edx
.text:004A2D19 6A 01                push    1
.text:004A2D1B B8 41 78 5F 00       mov     eax, offset aROncontrolerTe
.text:004A2D20 8B 8D 34 FE FF FF    mov     ecx, [ebp+keysOffset]
.text:004A2D26 2B 41 04             sub     eax, [ecx+4]
.text:004A2D29 50                   push    eax
.text:004A2D2A 8D 4D 98             lea     ecx, [ebp+var_68]
.text:004A2D2D E8 1E 76 02 00       call    calculateStaticKey
```

בתחילה ניסינו לעשות Reversing על הפונקציה החדשה, היא הייתה ארוכה מאוד וגם רקורסיבית, מה שהופך את המשימה לקשה מאוד. התחלנו לדמות חלק קטן יחסית מהפונקציות עד שהגענו למספרים של 64 ביט שזה גובל בבלתי נסבל בעת ההמרה ל-C.

בתמונה הבאה ניתן לראות חישובים על מספרים של 64 ביט והרבה:

```
.text:004D0764 8B 0E                    mov     ecx, [esi]
.text:004D0766 33 C0                    xor     eax, eax
.text:004D0768 2B CA                    sub     ecx, edx
.text:004D076A 8B D0                    mov     edx, eax
.text:004D076C 1B C2                    sbb     eax, edx
.text:004D076E
.text:004D076E
.text:004D076E 89 0E                    mov     [esi], ecx
.text:004D0770 F7 D8                    neg     eax
.text:004D0772 8B D0                    mov     edx, eax
.text:004D0774 8B 46 04                 mov     eax, [esi+4]
.text:004D0777 33 ED                    xor     ebp, ebp
.text:004D0779 33 C9                    xor     ecx, ecx
.text:004D077B 2B C2                    sub     eax, edx
.text:004D077D 8B D5                    mov     edx, ebp
.text:004D077F 1B CD                    sbb     ecx, ebp
.text:004D0781 8B 6E 08                 mov     ebp, [esi+8]
.text:004D0784 2B C7                    sub     eax, edi
.text:004D0786 1B CA                    sbb     ecx, edx
.text:004D0788 8B 54 24 20              mov     edx, [esp+18h+token1]
.text:004D078C 03 E9                    add     ebp, ecx
.text:004D078E 89 46 04                 mov     [esi+4], eax
.text:004D0791 89 6E 08                 mov     [esi+8], ebp
```

[הפקודה sbb זו דוגמא לפקודה שמשתמשים בה בד"כ לחישוב מספרים גדולים]

בסוף כנראה שהיינו מצליחים לבנות אמולטור לפונקציות הנ"ל אך התהליך היה לוקח עוד כמה חודשים טובים (במיוחד שהמחקר בוצע רק בסופי שבוע...), ולכן ניסינו גישה אחרת: ניסינו לחרוש את כל האינטרנט ולחפש חתימות לספרייה CryptoPP. מצאנו ב-tuts4you.com מספר גרסאות.

הכנסנו את כל החתימות האפשריות שהיו על CryptoPP והגענו לתוצאה הבאה:

```
mov     edi, [esp+34h+partOfHost]
push    ebx                     ; partOfStaticKey
push    edi                     ; partOfHost
push    ebp                     ; list2
push    esi                     ; list1
call    CryptoPP::PositiveDivide(CryptoPP::Integer &,CryptoPP::Integer &,CryptoPP::Integer const &,CryptoPP::Integer const &)
mov     eax, [edi+8]
add     esp, 10h
```

גילינו שהפונקציה שניסינו לדמות היא פנימית יותר לפונקציה PositiveDivide. הורדנו את CryptoPP וחיפשנו בתוך הפרויקט שימוש בפונקציה זו ואחרות והגענו לתוצאה הבאה:

```
// Add() and Subtract() are coded in Pentium assembly for a speed increase
// of about 10-20 percent for a RSA signature
```

אלה הערות שהיו בקוד, והפונקציות היו כתובות עם inline asm.

בתמונה זו רואים את הדמיון בין הפונקציה שב-IDA לפונקציה שב-VC++:

```
loopstart:
        mov     esi,[edx]
        mov     ebp,[edx+4]

        mov     edi,[ebx+8*eax]
        lea     edx,[edx+8]

        sbb     esi,edi
        mov     edi,[ebx+8*eax+4]

        sbb     ebp,edi
        inc     eax

        mov     [edx+ecx-8],esi
        mov     [edx+ecx-4],ebp

        jnz     loopstart

loopend:
        adc eax, 0
```

```
$loopstart$26150:
        mov     esi, [edx]
        mov     ebp, [edx+4]
        mov     edi, [ebx+eax*8]
        lea     edx, [edx+8]
        sbb     esi, edi
        mov     edi, [ebx+eax*8+4]
        sbb     ebp, edi
        inc     eax
        mov     [edx+ecx-8], esi
        mov     [edx+ecx-4], ebp
        jnz     short $loopstart$26150

$loopend$26149:
        adc     eax, 0
```

הגענו למסקנה שמשתמשים ב-RSA. לאחר מכן החלטנו שאנחנו לא הולכים לעשות Reverse Engineering ל-internals של ה-RSA כפי שמומשו ב-CryptoPP והתחלנו לבצע שלבים אחרים שיעזרו לנו על מנת שנוכל להשלים את העבודה.

החלטנו לקמפל בעצמנו את CryptoPP על מנת ליצור את החתימות בעצמנו. בגלל ש-WWP הינו משחק ישן יחסית היינו צריכים להשיג קומפיילר של VC++ 6. לאחר הקימפול, לא קיבלנו יותר מדי התאמות ואז הגענו למסקנה שבגלל שיש אפשרות לקמפל את הפרויקט לגרסת Debug או לגרסת Release קיימים הבדלים.

בדרך כלל כשמוציאים משחק לשוק משתמשים בגרסת Release שאמורה להיות פחות כבדה עם פחות בדיקות בקוד. אז יצרנו חתימות לגרסת Release של CryptoPP קימפלנו את הספריות כ-lib (יש אפשרות לקמפל גם כ-DLL אך ב-WWP קימפלו אותם בצורה סטטית).

יצירת חתימות ל-FLIRT

על מנת ליצור חתימה ל-IDA Pro אנו זקוקים לכלים של flare 6.x ולקובץ lib.

- בהתחלה אנו יוצרים קובץ pat על ידי שימוש בפקודה הבאה:

```
C:\Documents and Settings\Administrator\Desktop\ida61\flair>pcf cryptlib.lib cryptlib.pat
cryptlib.lib: skipped 2, total 10103
```

- לאחר מכן אנחנו יוצרים קובץ sig על ידי השימוש בפקודה הבאה:

```
C:\Documents and Settings\Administrator\Desktop\ida61\flair>sigmake.exe cryptlib.pat cryptlib.sig
cryptlib.pat (1271): bad pattern
```

- במידה ומקבלים את השגיאה "Bad Pattern" כפי שמוצג, ישנו הצורך להיכנס לקובץ pat ולמחוק את השורות המופיעות במספר. ברגע שסיימנו למחוק את כל השורות שמביאות שגיאה היווצר קובץ sig.
- במידה ויש כמה פונקציות עם התנגשות לאותו שם היווצר לנו קובץ מסוג ext. שבו נצטרך להגיד לאיזה פונקציה להתייחס, ניתן להתעלם מכל הפונקציות.
- לאחר מכן נריץ שוב את הפקודה על sigmake והיווצר קובץ מסוג sig.

את הקובץ שנוצר נכניס לתיקיה sig שיש ב-IDA Pro וכפי שהצגנו בתחילת המאמר נעלה את החתימה ל-IDA.

ניסינו כמה גרסאות של CryptoPP. ניסינו לחשוב מתי יצא WWP ואילו גרסאות יתאימו לנו. הסתכלנו מגרסה 3.1 עד 4.2 שהיו מועמדות לתקופת התאריך בהם קומפל המשחק. מצאנו את התאריכים של ספריות ה-CryptoPP וניתן לראות בתמונה הבאה את התאריכים:

> Crypto++ 1.0 - 23 Jun 1995
> Crypto++ 1.1 - 27 Oct 1995

> Crypto++ 2.0 - 19 Feb 1996
> Crypto++ 2.1 - 10 May 1996
> Crypto++ 2.2 - 02 May 1997

> Crypto++ 2.3 - 17 Jan 1998
> Crypto++ 3.0 - 01 Jan 1999
> Crypto++ 3.1 - 27 Apr 1999
> Crypto++ 3.2 - 20 Mar 2000
> Crypto++ 4.0 - 02 Nov 2000
> Crypto++ 4.1 - 13 Jan 2001

> Crypto++ 4.2 - 05 Nov 2001
> Crypto++ 5.0 - 11 Sep 2002

המשחק עצמו יצא במאי 2001, אז הגענו למסקנה שכנראה WWP השתמשו בגרסה 4.0 של CryptoPP. FLIRT לא עזר לנו למצוא את כל הפונקציות CryptoPP אך הוא עזר למצוא את חלקן, כנראה קימפלו את המשחק עם כל מיני דגלים שיש לקומפיילר - מה שעלול לשנות את קוד האסמבלי הסופי.

לאחר מכן השתמשנו ב-BinDiff על גרסת Release של CryptoPP והצלחנו למצוא עוד כמה פונקציות כמו מוד ה-CFB שהיה בו שימוש ל-Twofish ופונקציות שקשורות ל-RSA. כשהבנו שאכן מדובר בהצפנת RSA חיפשנו חומר באינטרנט על איך עובד RSA ב-CryptoPP וניסינו להבין מה המפתחות אשר משתמשים בהם לרשימת משחקים.

מציאת מפתחות ההצפנה לרשימת משחקים

תהליך מציאת המפתחות מאוד דומה למה שהיה קודם לגבי המפתחות לאימות משתמש. תמונה זו מציגה את הפונקציה בה אנו חושפים את המפתח ל-RSA.

```
.text:004A2C5A                  loc_4A2C5A:                             ; CODE XREF: getHostDecryption+97↑j
.text:004A2C5A 6A 01                push    1
.text:004A2C5C 8B 8D 2C FE FF FF    mov     ecx, [ebp+var_1D4]
.text:004A2C62 51                   push    ecx
.text:004A2C63 6A 01                push    1
.text:004A2C65 BA C4 63 5F 00       mov     edx, (offset encryptionKeys+1A1Ch)
.text:004A2C6A 8B 85 34 FE FF FF    mov     eax, [ebp+keysOffset]
.text:004A2C70 2B 10                sub     edx, [eax]
.text:004A2C72 52                   push    edx
.text:004A2C73 8D 8D 64 FF FF FF    lea     ecx, [ebp+var_9C]
.text:004A2C79 E8 D2 76 02 00       call    calculateStaticKey
```

בתחילה לא ידענו מה גודל המפתח, לא ידענו איך נראה הפורמט של מפתח RSA. על מנת לגלות איך עובד RSA בדקנו מה קורה אחרי שאנו מקבלים את המפתח אחרי שמוציאים את הקידוד של ה-base64. חשוב לציין שכאשר אנו מקבלים offset של המפתח, אנו למעשה מקבלים רצף של בינארי שיותר גדול מהמפתח ולאחר מכן מופעלת עליו פונקציה אחרת שדרכה נבחר את גודל המפתח שצריך לקחת.

CryptoPP מחלקת לבלוקים של 0x100 את הבינארי אחרי שמוציאים את הקידוד של ה-base64, ככה שאם המפתח גדול מ-0x100 אז הבתים שבבלוק הבא ישלימו את המפתח.

כך נראים 0x100 הבתים הראשונים של המפתח:

```
30 82 01 22 30 0D 06 09 2A 86 48 86 F7 0D 01 01
01 05 00 03 82 01 0F 00 30 82 01 0A 02 82 01 01
00 DB 5B B9 77 B1 81 8F 86 4B 76 E1 A4 0A C5 DA
D1 82 AF 38 40 6D 98 49 53 E3 D1 82 2E EE D3 7F
CF 46 E4 51 5E AE BE DD 13 C0 61 92 39 4C E8 15
D5 52 F4 52 49 85 94 70 DE 49 9A 18 1F B3 B3 2B
39 40 84 61 30 B6 C2 A5 08 2B 6F 80 35 FF ED 7F
D4 1C 34 86 D2 84 F7 2E B8 BF 55 E3 51 9F C3 35
2A F3 B2 9A 4E 2D 07 59 95 04 CE 4A CB 24 8B 12
69 72 E3 1B D8 45 77 1B 52 36 DA 6C 4D 6E DF 29
70 D9 3C 8A AC 9B C6 D9 7D 83 16 E5 0A BE 3B 9E
DC 07 CB 7F 30 86 E1 48 A3 64 0D B9 D8 34 57 8F
0D 7E CE B0 76 14 1F 76 15 32 D1 63 66 86 25 F8
C9 37 09 90 31 29 8C 39 B8 E5 98 AC F0 A6 C0 88
9B C7 5F DA B2 49 1C 6F 5D E9 8D 59 B2 4D 7D C8
68 23 03 35 66 88 C8 F7 5F E6 48 01 57 8A 32 3E
```

כאמור, בהתחלה לא ידענו מה גודל המפתח. חיפשנו חומר על RSA וכל מיני פורמטים של מפתחות הנמצאים בשימוש ב-CryptoPP. שמנו לב שיש כמה קבועים שמופיעים במפתחות RSA כדי לציין אם

המפתח הוא public או private. מצאנו שמפתחות RSA מתחילים באמת ב-?? ?? 30 82 המפתח במקרה שלנו השתמש בפורמט PEM. במפתחות RSA נעשה שימוש בקידוד ASN.

הנה תמונה מתוך MSDN.

```
0008:    |  30 82 01 22                        ; SEQUENCE (122 Bytes)
000c:    |  |  30 0d                           ; SEQUENCE (d Bytes)
000e:    |  |  |  06 09                        ; OBJECT_ID (9 Bytes)
0010:    |  |  |  |  2a 86 48 86 f7 0d 01 01  01
         |  |  |  |     ; 1.2.840.113549.1.1.1 RSA (RSA_SIGN)
0019:    |  |  |  05 00                        ; NULL (0 Bytes)
001b:    |  |  03 82 01 0f                     ; BIT_STRING (10f Bytes)
001f:    |  |     00
0020:    |  |     30 82 01 0a                  ; SEQUENCE (10a Bytes)
0024:    |  |        02 82 01 01               ; INTEGER (101 Bytes)
0028:    |  |        |  00
0029:    |  |        |  dc 66 85 5c 25 71 f1 f7  7b 00 0b 78 42 74 5e dd
0039:    |  |        |  20 b6 e9 5b 3f 75 22 84  e9 d7 cf 4a 17 56 ce e6
0049:    |  |        |  8b e9 17 ef 83 e9 45 30  c0 1d 54 bb 3b e9 db 90
0059:    |  |        |  77 04 81 89 a7 84 05 13  9a ba 36 90 c0 1b be f9
0069:    |  |        |  4f f0 c9 dc b5 ab dd 98  35 3f 9f 4f a1 37 34 cb
0079:    |  |        |  0b 33 c2 ce e7 f3 88 2e  ba b9 5a 8f 31 85 8b e9
0089:    |  |        |  f3 df 7e c0 f2 8e 61 0b  58 2b 14 a4 a4 8d 1b 53
0099:    |  |        |  8b 35 d3 be 3a 1f fd cc  8c 04 d1 a3 74 29 87 35
00a9:    |  |        |  89 f5 ad d1 db 61 f8 28  0c fb 2b 03 fb 96 0b 13
00b9:    |  |        |  19 be 6e 68 18 90 cd da  59 38 7a df 61 3d 92 34
00c9:    |  |        |  69 e1 54 38 70 15 ff ff  fc 4a ce 4b fa b3 11 03
00d9:    |  |        |  ab f5 9c bb 8e 1f 29 b7  c0 df 2f 46 fe 44 3e f1
00e9:    |  |        |  83 53 c0 12 0e 6a eb d8  4d f9 ab 92 07 6f 2d 3e
00f9:    |  |        |  af 7b c9 d6 23 b1 4a dd  5f f8 57 8b 00 fc d7 eb
0109:    |  |        |  55 82 bb 67 5c c3 a7 3e  df 0a 41 3f 7e 7b 05 5e
0119:    |  |        |  5c 55 4f 9d 19 4d e3 24  a2 dc 6c 68 c3 c9 8c 89
0129:    |  |        02 03                     ; INTEGER (3 Bytes)
012b:    |  |           01 00 01
```

אם מבצעים השוואה בין התמונה שב-MSDN למפתח שקיבלנו מקבלים התאמה מלאה שמדובר באותו פורמט. לאחר עוד קצת חיפוש, הבנו שמדובר גם במפתח RSA בגודל 2048 bits. המפתח שקיבלנו ב-WWP הוא ה-Public key!

WWP משתמשים בהצפנת RSA כדי לאמת את המקור של המשחק, הסיבה לכך היא כדי שלא יוכלו ליצור משחקים בשרת אחר.

המפתח בפורמט PEM של WWP מוצג בקטע הבא:

```
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA21u5d7GBj4ZLduGkCsXa
0YKvOEBtmE1T49GCLu7Tf89G5FFerr7dE8Bhkj1M6BXVUvRSSYWUcN5Jmhgfs7Mr
OUCEYTC2wqUIK2+ANf/tf9QcNIbShPcuuL9V41GfwzUq87KaTi0HWZUEzkrLJIsS
aXLjG9hFdxtSNtpsTW7fKXDZPIqsm8bZfYMW5Qq+O57cB8t/MIbhSKNkDbnYNFeP
DX7OsHYUH3YVMtFjZoYl+Mk3CZAxKYw5uOWYrPCmwIibx1/askkcb13pjVmyTX3I
aCMDNWaIyPdf5kgBV4oyPrnbwrJIohYJy+rQ2YUZ2mb0Y7ltRl5R3ByGPJ2UL9E9
4wIDAQAB
-----END PUBLIC KEY-----
```

לאחר שהשגנו את המפתח של ה-RSA, שלפנו את המפתח שבשימוש להצפנה של twofish. בקטע קוד הבא ניתן לראות שמתבצע חישוב לאיזה offset מתאים המפתח:

```
text:004A2D10                   loc_4A2D10:
text:004A2D10 6A 01                   push    1
text:004A2D12 8B 95 24 FE FF FF       mov     edx, [ebp+var_1DC]
text:004A2D18 52                      push    edx
text:004A2D19 6A 01                   push    1
text:004A2D1B B8 41 78 5F 00          mov     eax, offset aROncontrolerTe
text:004A2D20 8B 8D 34 FE FF FF       mov     ecx, [ebp+keysOffset]
text:004A2D26 2B 41 04                sub     eax, [ecx+4]
text:004A2D29 50                      push    eax
text:004A2D2A 8D 4D 98                lea     ecx, [ebp+var_68]
text:004A2D2D E8 1E 76 02 00          call    calculateStaticKey
```

קיבלנו את המפתח הבא לאחר שימוש בדיבאגר כפי שעשינו קודם לאימות משתמש ולכן לא נפרט את כל התהליך שוב:

```
\x6C\x91\xA8\xEC\x79\x38\x35\xF4\x7E\x58\x26\x18\x23\x10\x19\x24\x65\x93\x90\x0F\xFE\xCA\x63\xE2\x63\x2E\xB8\x95\xB6\xF3\xAC\x4C
```

כתבנו סקריפט שיודע לפענח את רשימת המשחקים בשפת PHP. וכעת, כאשר אנו שולחים לדף ה-PHP את ה-base64 של המשחק, הוא יודע להחזיר את המידע לאחר ההצפנה. (קישור לקוד מופיע בסוף המאמר).

בקטע הבא ניתן לראות תמונת זיכרון של משחק אחרי הפענוח ב-010 editor:

```
        0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   0123456789ABCDEF
0000h: 46 2D 9B B8 D6 65 30 07 61 FF 9B 75 DC E6 9F 80   F-›¸Öe0.aÿ›uÜæŸ€
0010h: 2E 86 A0 9E 67 61 6D 65 6E 61 6D 65 00 00 00 00   .† žgamename....
0020h: 00 00 00 00 00 00 00 00 68 6F 73 74 6E 69 63 6B   ........hostnick
0030h: 6E 61 6D 65 00 00 00 00 00 00 00 00 0C 0D 0E 0F   name............
0040h: 00 39 36 22 00 00 00 00 00 00 00 01 00 00 00 00   .96"............
0050h: 00 00 00 00 53 47 C3 06 53 47 DF 26               ....SGÃ.SGß&
```

אחרי שהצלחנו להבין איך עובדת ההצפנה ניסינו להבין איך בדיוק מחושב ה-checksum. ה-checksum מחושב בטווח 0x14-0x5B. כתבנו סקריפט ב-PHP שמנסה לעשות hash עם כמה אלגוריתמים שונים המכילים 20 בתים. ניסינו SHA1 ועוד אך הם לא תאמו את התוצאה המתבקשת, לבסוף, האלגוריתם RIPEMD160 היה זה שהתאים. בהתחלה חשבנו ש-SHA1 היה האחד שהתאים בגלל ש-flirt טען שמשתמשים ב-SHA1 אך התברר שזה לא היה נכון.

קיימת פונקציה נוספת שיש בה שימוש ב-twofish עליה לא דיברנו: הפונקציה נמצאת בשימוש לפקודה בשם **<CHECK ...>** זו פקודה שנמצאת בשימוש בשביל לבדוק אם הדיסק של המשחק מקורי או לא על ידי חישוב checksum, לא ממש נכנסנו לזה כי זה לא מעניין וניתן להסתדר בלי זה ליצירת אמולטור לשרת משחק.

סיכום

בחלק זה פירטנו באיזה הצפנה השתמשו חברת Team17 על מנת להגן על הפרוטוקול של המשחק.

- הסברנו איך עובד אימות המשתמשים לשרת IRC.
- הסברנו איך נראה מבנה של כל משחק שיוצרים ב-C.
- הסברנו איך עובדת ההצפנה של רשימת המשחקים.
- הסברנו איך ליצור חתימות עם FLIRT

בחלק הבא אנו נפרט איך יצרנו אמולטור ל-WWP שדרכו ניתן לשחק אחד עם השני ונפרט על עוד פקודות הקיימות בפרוטוקול.

קישור לקטעי הקוד המוסברים במאמר:

http://www.digitalwhisper.co.il/files/Zines/0x37/PHP_Scripts.rar

על מחבר המאמר (d4d)

מחבר המאמר עוסק ב-Reverse Engineering ואוהב לחקור משחקי מחשב והגנות, לכל שאלה שיש או ייעוץ ניתן לפנות אליו בשרת IRC של NIX בערוץ:

#reversing

בכתובת האימייל:

llcashall@gmail.com

או דרך האתר:

http://www.cheats4gamer.com

כלכלה קיברנטית

מאת CISA Dragon

הקדמה

ב-7 באפריל 2013 הותקפו אתרי אינטרנט ישראלים בארץ ובעולם במבצע שנודע כ-OpIsrael. המתקפה התרחשה בהמשך למבצע "עמוד ענן" של צה"ל בסוף 2012 ברצועת עזה וכתגובה למדיניות ישראל בשטחים. למבצע הקיברנטי קדמו פרסומים רבים וכן סרטוני תעמולה[1] של ארגון Anonymous אשר הוביל את ההתקפה. מארגני המבצע איימו לנתק את מדינת ישראל מהאינטרנט וקבעו רף חדש במערכה התודעתית וביחסים בין טרור קיברנטי למדינה. במהלך השבועות שקדמו להתקפה הורגשה בשוק המקומי תכונה רבה לקראת העתיד לבוא. בארגונים רבים, במיוחד באלה הנמנים על המגזרים המועדים להתקפות על רקע לאומני, התקיימו ישיבות הכנה, בוצעו פעולות מנע וטיוב למערכות ההגנה, אורגנו התייעצויות עם גורמים מקצועיים ונאסף מודיעין לקראת ההתקפה. במהלך ההתקפה עצמה, אשר נמשכה כל סוף השבוע שקדם ל-7 באפריל וכן במהלך אותו יום ובימים לאחר מכן, הותקפו אתרים של חברות וארגונים מכל הסוגים, רשויות ציבוריות, משרדי ממשלה, גופים בטחוניים וכל יעד אחר שהמתקיפים שייכו למדינת ישראל או למוסדות המזוהים עימה.

עד כאן העובדות מוכרות ומקובלות. השאר, הופך לפרשנות סובייקטיבית למדי. בסוף מתקפת ה-7 באפריל התפרסם בשם Anonymous ציוץ מעניין ב-Twitter ובו נתוני ביניים המסכמים את ההתקפה:

> #Anonymous partial damage report, 100k+ websites, 40k Facebook pages, 5k twitter & 30k Israeli bank acc got hacked ~ **$3-plus billion damage**[2]

המספרים שסופקו על ידי מארגני ההתקפה הם פנטסטיים בכל קנה מידה. מדהימה במיוחד היא הערכת הנזק למשק הישראלי אשר עומדת, על פי המארגנים, על כ-3 מיליארד דולרים. לשם המחשה: עלות הנזק הישיר למשק בגין אי-אספקת חשמל במשך יממה עומדת על כ-4.2 מיליארד דולר[3]. על פי התבטאות של שר הביטחון, סכום זה דומה לעלות מבצע "צוק איתן" למערכת הביטחון[4]. האומנם?

[1] http://www.youtube.com/watch?v=q760tsz1Z7M
[2] https://twitter.com/Op_Israel
[3] http://www.themarker.com/dynamo/1.1691873 (הנתון בקישור מתייחס להשבתה של שעה. שער הדולר שחושב: 3.6 שקלים)
[4] http://pplus.ynet.co.il/articles/0,7340,L-4566748,00.html

אחרי שמסיימים לגחך, כדאי לחשוב ברצינות: האם רק החברים מ-OpIsrael הגזימו בהערכות הנזק שלהם? מספיק לקרוא כמה דוגמאות ממחקרים שונים המתפרסמים בעולם לגבי היקף הכלכלה הקיברנטית השחורה כדי להבין שידידינו שונאי ישראל אינם היחידים הלוקים בחשוביהם[5]. דוגמא לסוגיה ניתן למצוא במחקר, שפרסמה בשנת 2009 חברת מקאפי, אשר העריך את סך הנזק העולמי כתוצאה מפעולות קיברנטיות בלתי לגיטימיות בסכום הדמיוני של **טריליון דולר בשנה**[6]. מחקר זה נוצל בשעתו בידי חלק מאנשי הממשל כדי לשכנע את הנשיא אובמה לתקצב בנדיבות את המאמצים הקיברנטיים של ארצות הברית. ביולי השנה פורסמה הערכה נוספת של מקאפי בשיתוף המרכז האמריקאי ללימודים אסטרטגיים ובינ"ל (CSIS). באופן מפתיע, השנה המספרים צנחו דרמטית ועמדו על הטווח של 20-140 מיליארד דולרים "בלבד" בשל מה שכונה במחקר "שינוי מתודולוגיית המדידה"[7]. ממצאים אלה גרמו ללא מעט הרמות גבה בקרב מתנגדי התוכנית הקיברנטית האמריקאית והעלו תהיות לגבי אמינות הנתונים אשר מהווים בסיס למדידת נזק במימד הקיברנטי. הנתונים הללו והמחקר מאחוריהם החזירו לקדמת הבמה דיון ותיק בדבר הקושי הרב בכימות ההוצאה על אבטחת מידע וסייבר ובמיוחד על אי הוודאות הרבה האופפת כל ניסיון לבצע חישוב בתחום.

כלכלה קיברנטית

אחד הנתונים המקובלים בשנים האחרונות בתעשייה הוא שהפשיעה הקיברנטית היא אחת משלוש הכלכלות הלא מדווחות[8] הגדולות ביותר (ביחד עם תעשיית הסמים והפורנו). עוד נתון מעניין הוא שעל פי מחקר של הפורום הכלכלי העולמי (WEF), התקפות קיברנטיות הן אחד הסיכונים המרכזיים שמאיימים על עולמנו והסיכון השני בסיכונים הטכנולוגים הקיימים כיום[9]. מכל מקום, קשה למצוא מישהו שאינו רואה בתחום הקיברנטי האפל משהו מטריד "וגדול". אבל עד כמה באמת "גדול"?

מליסה הת'אווי (Melissa Hathaway)[10], אשר כיהנה בממשל בוש הבן כמנהלת כוח המשימה הבין-משרדי לנושאי סייבר וכיועצת במועצה לבטחון לאומי בממשל אובמה, הציגה בעת ביקורה בישראל ביוני 2013 תוצאות שני מחקרים אשר נעשו במדינות שונות בעולם על השפעת הפשיעה הקיברנטית על התמ"ג[11]. על פי התוצאות בבריטניה עולה כי השפעה זו נעה סביב 1.8 מהתמ"ג (!). על פי אותו מחקר בריטי, אשר נערך בשנת 2011 על ידי חברת המחקר Detica בשיתוף המשרד לאבטחת סייבר, עולה כי סך ההשפעה

[5] http://www.ponemon.org/local/upload/file/2012_US_Cost_of_Cyber_Crime_Study_FINAL6%20.pdf , http://www.symantec.com/about/news/release/article.jsp?prid=20120905_02 ואחרים

[6] http://news.cnet.com/8301-1009_3-57594989-83/cyberattacks-account-for-up-to-$1-trillion-in-global-losses/ ,

[7] http://www.theverge.com/2013/7/23/4547506/new-study-says-cybercrime-may-cost-140-billion-annually

[8] שאין משלמים מיסים בגינן.

[9] http://www3.weforum.org/docs/WEF_GlobalRisks_Report_2013.pdf

[10] http://www.cigionline.org/publications/2013/5/change-conversation-change-venue-and-change-our-future

[11] תמ"ג: תוצר מקומי גולמי. מדד מרכזי בכלכלה למדידת גודלה הכלכלי של מדינה.

של אירועי פשיעת סייבר (ללא אירועי טרור או לוחמה קיברנטית) על הכלכלה הבריטית הינו כ-27 מיליארד לירות סטרלינג בשנה, כשמתוכם כשליש (9.2 מיליארד) מקורו בגניבת קניין רוחני (IP) מעסקים[12]. אם ננסה להקיש מנתונים אלה על הכלכלה המקומית שלנו, נגלה שסך ההשפעה הצפויה של אירועי סייבר על רקע כלכלי בישראל צריך להסתכם בכ-4.5 מיליארד דולר בשנה[13]. וואו....זה הרבה. לא פלא שאותו מחקר התקבל בשעתו בספקנות רבה על ידי המומחים. מדובר בסכום השווה לכ-66% מסך שוק מערכות המידע בישראל[14]. האם באמת ייתכן שסך אירועי הפשיעה הקיברנטית בישראל שווים שני שליש משוק מערכות המידע המקומי???

אבל מדוע בעצם מעניין אותנו למדוד כמה עולה לנו הסייבר הזה? מכיוון שלכולנו ידוע שמי שאינו מודד לא יכול לנהל ועל כן, ניתן להניח, שאין לרגולטורים הישראלים בתחום הסייבר (המטה הקיברנטי\המפקח על הבנקים\רא"מ\רמו"ט וכל השאר) כלים לשפר את הפיקוח, את הפניית התקציבים ולכוון את הטכנולוגיה לכיווני התפתחות רצויים על מנת להביא למיצוי יכולת ההתגוננות של מדינת ישראל עם איומים קיברנטיים ממשיים ולא דמיוניים.

האתגר המרכזי כאן הוא אפוא מדידה אפקטיבית של השפעת עולם הסייבר על הכלכלה. אבל מה עומד מאחורי הביטוי מדידה אפקטיבית? מחקר שפורסם בשנת 2010 על ידי Fafinski, Dutton, Margetts מאוניברסיטת אוקספורד ניסה לעשות סדר במדידה של פשיעת סייבר. מחקר זה הצביע במפורש על חלק מהקשיים המרכזיים באיסוף מידע אמין לביצוע המדידה[15]:

עודף במקורות מידע - בעולם שבו אנו חיים קיים מגוון עצום של מקורות מידע. גם בתחום מחקר הסייבר קיימים עשרות רבות של גופים בינלאומיים ומקומיים אשר עוסקים בנושא. הקושי העיקרי כאן הוא שאין תקן אחיד לאיכות הנתונים ואין שום הבטחה לגבי נכונותם.

תיעוד חסר של אירועי סייבר - בהמשך לנקודה הראשונה ולמרבה האירוניה, הפחד מפרסום שלילי או ריגול בקרב ארגונים שסבלו מאירוע סייבר, העדר חובת דיווח והעדר תקן לצורת הדיווח מביאים לכך שלכלל תושבי כדור הארץ אין כיום תמונת מצב מהימנה לגבי אירועי סייבר והשפעתם המצטברת על הכלכלה.

סקרי סייבר - כמו שכבר הוזכר למעלה, אין מתודולוגיה מוכרת לביצוע סקרים בנושא הנזק הנגרם כתוצאה מאירועי סייבר. כמו כן, אין כמעט סקרים אשר ביצעו עבודה מקיפה למדידה של הנזק הנ"ל. יש לציין כי סקרים ורטיקלים נחשבים למדוייקים יותר וניתן באמצעותם לקבל, לעתים, תמונת מצב טובה על הנעשה במגזר מסויים. עדיין, תמונה כללית על שוק או מדינה אינם בנמצא.

[12] https://www.gov.uk/government/publications/the-cost-of-cyber-crime-joint-government-and-industry-report

[13] בנתוני תמ"ג שמסתכמים ב-248 מיליארד דולר בשנת 2012. החישוב הוא: 1.8%*248.

[14] http://www.slideshare.net/jimmyschwarzkopf/stki-summit-2012-israeli-it-market

[15] http://www.law.leeds.ac.uk/assets/files/staff/FD18.pdf

ניגוד אינטרסים והטיות - סקרים רבים מבוצעים על ידי גורמים אשר יש להם אינטרס מסויים בשוק של פשיעה קיברנטית. חברות אשר מוכרות מוצרים בתחום עלולות לפרסם נתונים מנופחים על מנת לשדר בהלה ודחיפות ולנסות לשכנע לקוחות לרכוש את מוצריהם.

שפה ורטוריקה - אוכלוסיות שונות משתמשות בטרמינולוגיה שונה ועל ידי כך מקשות על כלל הציבור לקבל תמונה מהימנה של האירועים. דוגמא טובה לכך היא הדרמטיזציה שעיתונאים עושים לתחום הסייבר בכיסוי שלו במדיות השונות. דוגמא נוספת מישראל היא השיח הקולני, לעתים, אשר מובל על ידי נבחרי ציבור שונים שמנסים למשוך את תשומת הלב הציבורית לנושא באמצעות תיאורים אפוקליפטים ודימויים מוגזמים[16].

מחקר נוסף[17] שפורסם ב-2012 ובוצע גם הוא בהזמנת המשרד הבריטי לאבטחת סייבר ניסה לייצר מסגרת ומתודולוגית למדידת עלות פשיעת סייבר, תוך הפקת לקחים מהמחקר של חברת Detica אשר הוזכר לעיל.

על פי מחקר זה, ניתן לחלק את העלויות לשלוש קבוצות, שאם סוכמים אותן מקבלים את סך העלות לחברה כתוצאה מפשיעה קיברנטית:

1. עלות ההגנה. למשל: רכישה של תוכנת אנטיוירוס. (הצד השני של המטבע: הרווח של יצרן האנטיוירוס).
2. עלויות ישירות. למשל: גניבה של כרטיסי אשראי, מאמץ לבודד וירוס וכו'.
3. עלויות לא ישירות כתוצאה מאירוע סייבר. למשל: איבוד האמון בבנקאות באינטרנט, פגיעה מוניטינית (הצד שני: רווח של חברות ייעוץ, חברות תדמית וכו').

בכל אחת מהקבוצות קיים צד של הפשיעה עצמה וצד של תשתית התומכת בפשיעה.

עורכי המחקר מציעים להפריד בין תחומים שלגביהם קיים מידע ונתונים אשר מאפשרים לבצע מדידה כמו הונאות בכרטיסי אשראי, הונאות בבנקאות באינטרנט, זיופים שונים בתחום המוסיקה והוידאו ועוד לבין תחומים שאין לגביהם נתונים כלל. בכל תחום מתחומי ההונאה השונים נבחר מקור המידע האמין ביותר לדעת החוקרים וכך הורכבה התמונה השלמה. החוקרים התבססו על הנחה נוספת, לפיה בריטניה אחראית ל-5% מסך התמ"ג העולמי[18] וזאת כדי לבצע מנפולציות על נתונים מהרמה העולמית לרמה המקומית וההיפך. סך הממצאים מרוכזים בטבלה הבאה[19]:

[16] http://main.knesset.gov.il/Activity/committees/Science/News/pages/pr_980_01011900.aspx
[17] http://weis2012.econinfosec.org/papers/Anderson_WEIS2012.pdf
[18] http://data.worldbank.org/country/united-kingdom
[19] המספרים המודגשים חושבו בהתבסס על הנחות ומחקרים בכל תחום ואילו המספרים לצידם, שאינם מודגשים, חושבו על פי חלקה היחסי של בריטניה מסך התמ"ג העולמי.

Table 1: Judgement on coverage of cost categories by known estimates

Type of cybercrime	UK estimate	Global estimate	Reference period	Criminal revenue	Direct losses	Indirect losses	Defence cost
Cost of genuine cybercrime							
Online banking fraud							
– phishing	$16m	**$320m**	2007	×?	×?		
– malware (consumer)	$4m	**$70m**	2010	×↓	×↓		
– malware (businesses)	$6m	**$300m**		×↓	×↓		
– bank tech. countermeasures	$50m	**$1 000m**	2010				×?
Fake antivirus	$5m	**$97m**	2008–10	×	×		
Copyright-infringing software	**$1m**	**$22m**	2010	×	×		
Copyright-infringing music etc	$7m	**$150m**	2011	×↓			
Patent-infringing pharma	**$14m**	**$288m**	2010	×			
Stranded traveller scam	$1m	**$10m**	2011	×↓			
Fake escrow scam	$10m	**$200m**	2011	×↓			
Advance-fee fraud	**$50m**	$1 000m	2011	×↓			
...							
Cost of transitional cybercrime							
Online payment card fraud	**$210m**	$4 200m	2010		(×)		
Offline payment card fraud							
– domestic	**$106m**	$2 100m	2010		×↓		
– international	**$147m**	$2 940m	2010		×↓		
– bank/merchant defence costs	$120m	**$2 400m**	2010				×↓
Indirect costs of payment fraud							
– loss of confidence (consumers)	**$700m**	$10 000m	2010			×?	
– loss of confidence (merchants)	**$1 600m**	$20 000m	2009			×?	
PABX fraud	**$185m**	**$4 960m**	2011	×	×↓		
...							
Cost of cybercriminal infrastructure							
Expenditure on antivirus	**$170m**	$3 400m	2012				×
Cost to industry of patching	$50m	**$1 000m**	2010				×?
ISP clean-up expenditures	**$2m**	$40m	2010			×?	
Cost to users of clean-up	**$500m**	$10 000m	2012			×?	
Defence costs of firms generally	$500m	**$10 000m**	2010				×?
Expenditure on law enforcement	**$15m**	$400m	2010				×
...							
Cost of traditional crimes becoming 'cyber'							
Welfare fraud	**$1 900m**	$20 000m	2011	×	(×)		
Tax fraud	**$12 000m**	$125 000m	2011	×?	(×)		
Tax filing fraud	-	**$5 200m**	2010	×	(×)		
...							

על פי נתונים אלו, עלות פשיעת הסייבר בבריטניה עומדת על כ-18.5 מיליארד דולר (0.77% מהתמ"ג הבריטי). עם זאת, יש לשים לב לפרמטר מאוד חשוב בטבלה: במחקר נלקחו בחשבון סעיפי עלות להונאות מתחומי המס והרווחה מתוך הבנה כי חלק גדול מהאינטראקציה בין האזרח לרשויות היא אלקטרונית. כך, למשל, הונאות מס נעשות כיום, במקרים רבים, במערכות ממוחשבות ועל פי מחקר זה יש לראות בהן הונאות של תחום אשר הופך לקיברנטי. עם זאת, יתכן שלא כולם יסכימו לפרשנות זו, שכן בדוגמה זו מהות העבירה היא לא קיברנטית, למרות שהאמצעי הטכני לביצועה - כן. שני חלקים אלה תורמים כ-14 מיליארד דולר עלות הכוללת. אם ננקה את שני הסעיפים הללו נראה כי סך העלות כתוצאה מפשיעת סייבר בבריטניה עומדת על כ-4.5 מיליארד (0.19% מהתמ"ג הבריטי). **אם נקיש מנתונים אלה על השוק הישראלי נגלה כי 0.19% מהתמ"ג הישראלי מסתכמים ב-460 מיליון דולר[20]. וזו, על פי מחקר זה, היא עלות הפשיעה הקיברנטית בישראל.**

[20] החישוב: תמ"ג בישראל (248 מיליארד $ בקירוב) כפול 0.19%: 248B*0.19%=460M

מדידת סייבר בישראל

בסעיפים הקודמים הוזכרו מספר דוגמאות לנסיונות חישוב של השפעת הפשיעה הקיברנטית על הכלכלה ברחבי העולם.

במאי 2012 פרסם המשרד לביטחון פנים דו"ח מקיף העוסק בנזק הכלכלי מתופעת הפשיעה במדינת ישראל[21] אשר עומדת על פי הדו"ח (בנתוני 2011) על כ-4 מיליארד דולר. כשמנתחים את הדו"ח ניתן לראות כי עלות הפשיעה הקיברנטית בישראל שחושבה בסעיף הקודם בהתבסס על מחקר בריטי (460 מיליון דולר) **גדולה פי 2.7** בקירוב מעלות הנזק הכלכלי כתוצאה מעבירות מין או **פי 4** מעלות הנזק כתוצאה מעבירות רצח. לכן, הנתון המעניין ביותר בדו"ח זה הוא דווקא הנתון שלא מופיע בו. הדו"ח אינו מספק נתונים כלשהם על פשיעה קיברנטית ואף אינו מזכיר את הנושא כנתון שיש להתחשב בו.

חברות מחקר בתחום מערכות המידע והטכנולוגיה מבצעות בכל שנה הערכה לגבי גודל שוק מערכות המידע בכלל ושוק אבטחת המידע בישראל בפרט. שוק אבטחת המידע הוערך על ידי חברת המחקר הישראלית STKI בשנת 2012 בכ-400 מליון דולר[22]. סכום זה חושב באמצעות שני כלים עיקריים: איסוף נתוני תקציבים מחברות וארגונים מובילים במשק וכן מתשאול של ספקים ויצרני פתרונות אשר פעילים בשוק המקומי. מכיוון שהנתונים אשר נאספו היוו מדגם מייצג בלבד, היה על מנהלי הסקר לבצע מכפלות שונות על מנת לקבל נתון שיתייחס לכלל השוק המקומי.

במהלך 2012 נעשה, כאמור, ניסיון צנוע לכמת את ההשפעה הכלכלית של אירועי אבטחת מידע בשוק המקומי מעבר לתקציב השוטף של אבטחת המידע[23]. לצורך כך התבצע מדגם בקרב כשישים ארגונים, מוסדות וחברות מהגדולים במשק. במדגם נשאלו מנהלי אבטחת המידע ואנשי מקצוע רלוונטים אחרים לגבי מספר האירועים המשמעותיים שחוו במהלך 36 החודשים שקדמו לסקר וכן לגבי סך הנזק אשר נגרם להם כתוצאה מאירועים אלה. "אירוע אבטחה משמעותי" הוא מושג מאוד חשוב בסקר זה. הוא הוגדר בסקר כאירוע שבגינו נגרם הפסד ישיר של ימי עבודה מחוץ לתכנון או הזמנה לא מתוכננת של שעות ייעוץ. טיפול באירועים חריגים, ניתוח לוגים ואפילו ביצוע חקירה כתוצאה מאירוע שהתרחש אינם מוגדרים כאירוע חריג אם התרחשו כחלק שגרת העבודה של צוות האבטחה האורגני. דגימה מקבילה לצורך אימות התבצעה מול תשע מתוך חברות ייעוץ אבטחת המידע הגדולות במשק: כאן התבקשו מנהלי החברות להעריך את מספר האירועים אשר חוו לקוחותיהם במהלך שנת 2011 (ללא ציון פרטים מזהים על הלקוח) ואת עלות הנזק שנגרמה, להערכתם, כתוצאה מאירועים אלה.

---

[21] http://mops.gov.il/Documents/Publications/CrimeDamage/CrimeDamageReports/CrimeDamageReport2011.pdf

[22] http://www.slideshare.net/jimmyschwarzkopf/stki-summit-2012-israeli-it-market

[23] http://www.slideshare.net/shaharmaor/information-security-stki-summit-2012shahar-geiger-maor-12059675

סיכום תוצאות הסקר הצביע על הממצאים הבאים:

1. בארגון ממוצע במשק (לא כולל צבא וקהילת המודיעין) התרחש **אירוע משמעותי כל 18 חודשים**.
2. על פי נתוני ביטוח לאומי ורשם החברות קיימים בישראל כ-236 ארגונים שבהם יותר מ-1000 עובדים נכון לשנת 2012. הנחת עבודה ראשונה היתה שארגונים אלה, הכוללים את המגזר הפיננסי, התעשיות הביטחוניות, חברות השירותים והתקשורת, משרדי הממשלה וגופי תעשיה מובילים, ריכזו את רוב פעילות הסייבר של המשק (למעט הצבא). אם נכפיל את מספר האירועים המשמעותיים בממוצע בשנה לארגון במספר הארגונים הגדולים הללו נקבל 2/3 * 236 = 156 **אירועים משמעותיים ברמת כלל המשק בשנה אחת.**
3. הנחת העבודה הבאה של הסקר היתה שבישראל ישנם לא מעט אירועים לא מדווחים ולכן לא נלקחו בחשבון בסעיף 2 לעיל. על כן הורחב הנתון בדבר מספר האירועים לצורך החישוב ל-**400 אירועי אבטחה משמעותיים בישראל בשנה**.
4. הנסקרים השיבו כי אירוע משמעותי גורם לדעתם **להפסד של כ-50 שעות עבודה בממוצע**. יש לזכור כי הממוצע חושב על כלל המגזרים במשק אשר השיבו לסקר.
5. הערכת עלות הנזק בממוצע לאירוע היתה הערכה סובייקטיבית ומאוד שמרנית ועמדה על **50 אלף דולר בממוצע לאירוע**.
6. הנסקרים התבקשו, כאמור לעיל, להעריך את סך הנזק כתוצאה מאירועי סייבר כאחוז מכלל ההכנסות של הארגון. רוב מנהלי אבטחת המידע העריכו כי מדובר בעלות מאוד שולית (פחות מ-**1% מכלל הכנסות ארגונם בשנה**). רוב היועצים העריכו כי עלות הנזק עומדת על **1%-5% מכלל ההכנסות**. על פי נתוני הבורסה לניירות ערך המתפרסמים מעת לעת, סך הכנסות של כלל החברות הנסחרות במדד ת"א 100 עמד בשנת 2011 על כ-200 מיליארד דולר[24]. כדי לחשב את עלות הנזק כתוצאה מאירוע אבטחת מידע ברמת כלל המשק בוצעה הערכת נזק באופן שמרני: כ-0.1% מכלל הכנסת החברות במדד ת"א 100, כלומר **200 מיליון דולרים נזק שנתי כתוצאה מאירועי סייבר**.
7. הערכת הנזק הסובייקטיבית שפורסמה בסופו של דבר בסקר עמדה על **20 מיליון דולר** בלבד, שכן ההערכה שמובעת בסעיף 6 לעיל שיקפה עלות נזק ממוצע של כ-500 אלף דולר לכל אירוע אבטחה משמעותי. בשיחות אימות עם חלק מהמשתתפים בסקר התקבל נתון זה בספקנות רבה ולכן פורסמה ההערכה החדשה.

מסקנה: יותר מדי הנחות עבודה ופחות מדי נתונים עובדתיים שיתמכו בסקר. הערכות הנזק הסובייקטיביות מסייעות אומנם לגיבוש כיוון כללי להערכת עלות הנזק, אך הן אינן תחליף להמצאות

[24] מטרת הצגת נתון זה היא לקבל אומדן מקורב של הכנסות כלל המגזר העסקי (שחלק משמעותי ממנו נכלל בסקר). ברור שיש הטייה מובנית, שכן משרדי ממשלה למשל כלל אינם נכללים בחישוב זה.

נתונים אמיתיים "מהשטח". מיעוט האירועים שהתרחשו בפועל בשוק הישראלי בשנים אלה[25] והעדר ניתוח בדיעבד של עלות הנזק בקרב רוב הארגונים הופכים סקרים כדוגמת הסקר לעיל לבעייתיים מאוד.

הקשר בין מדידה לביטוח סייבר

תשלומי פרמיה של ביטוח נקבעים על ידי מספר רב של פרמטרים ומושפעים מאוד מסוג התכולה המבוטחת, התעשייה, הניסיון של המבטח ועוד. ברוב המקרים מתכננים מודלים לקביעת הסיכון הגלום בנושא המבוטח וההסתברות לאירוע ביטוחי באותו תחום, ממש כמו ניהול סיכונים שכולנו מכירים מהעולם הטכנולוגי. בשוק חדרי המחשב לאירוח אתרים בארה"ב (hosting) גילו לפני שנים לא רבות כי קשה לייצר פוליסות ביטוח שיהלמו את הצרכים של הלקוחות מכיוון שחברות הביטוח לא מכירות מספיק את התחום. מצב זה הביא לכך שלא נמצאו מבטחים לחלק מאתרי האירוח הגדולים. כדי להתמודד עם הבעיה דרשו חברות הביטוח שמתקני האירוח יעברו הסמכה של גופים מקצועיים חיצוניים ובלתי תלויים, אשר יגדירו מדדים מוסכמים ומוחשיים להערכת השרידות של המבנים ושל התשתיות. הערכות אלה דורגו במדרגות מיוחדות (tiers) על פי רמת השרידות של המתקן, של מערכות התשתיות, החשמל, המים ועוד. כך נוצר תחום מדידת השרידות בחדרי המחשב ונסללה הדרך להתפתחות של תוכניות ביטוח מותאמות לצרכי אתרים אלה. כיום מדובר בסטנדרט לפיו מתכננים חדרי מחשב על פי רמת השרידות שלהם וכל לקוח יכול לקבל לידיו בצורה שקופה את רמת הדירוג של האתר ולגזור מכך את רמת הביטחון שהמידע שלו ישמר ויאובטח במתקן לאורך זמן.

גם בתחום הסייבר החלה להתפתח מגמה מעניינת מצד חברות הביטוח שהחלו להציע תוכניות לביטוח מפני נזקי סייבר. על פי מחקר של חברת Ponemon ב-31% מסך החברות והארגונים בארה"ב קיימת מדיניות ביטוחית בתחום הסייבר וכ-39% מהארגונים האחרים מתכננים לרכוש תוכנית ביטוחית דומה[26]. בישראל החלה מגמה דומה וחלק מחברות הביטוח כבר החלו בשיווק תוכניות ביטוח מותאמות לכיסוי נזקים קיברנטיים[27].

אבל למעלה כבר צויינה הבעייתיות בהערכת הנזק וגודל שוק נזקי הסייבר ולכן צף ועולה פעם נוספת הספק הבסיסי לגבי יכולתן של חברות הביטוח להעריך בצורה טובה את עלות הנזק כתוצאה מאירועי סייבר ומהם לגזור את גובה הפרמיה ללקוחות.

כאן יש לציין עיקרון חשוב לפיו פועלות חברות הביטוח ואשר מקל מאוד את מתן המענה הביטוחי גם בתחום בעייתי זה: פרמיית ביטוח נקבעת ברוב רובם של המקרים לאחר בניית מודל כלכלי וניהול סיכונים. עם זאת, במקרים בהם אין למבטח מספיק כלים להעריך את הסיכון הוא עשוי לפנות לעקרונות קמאיים יותר של היצע וביקוש ולבחון מהי העלות שהלקוחות יהיו מוכנים לשלם עבור תוכנית הביטוח המוצעת.

[25] הסקר נערך לפני אירועי "ההאקר הסעודי" ולא כלל התייחסות אליו.
[26] http://www.ponemon.org/local/upload/file/Cyber%20Insurance%20white%20paper%20FINAL%207.pdf
[27] http://www.albit.co.il/news/news-7a-7-2013.htm

סביר להניח שמחירי הביטוח בתחום הסייבר ישתנו ככל שחברות הביטוח ילמדו להכיר את הלקוחות, את השוק הזה ולצבור ניסיון. עם זאת, שיווק תוכניות ביטוח המבוססות על ניסיון נרכש ולא על עקרונות של ניהול סיכונים מדגימה לנו דרך עוקפת גם בהתמודדות עם תחומים אחרים שמושפעים מכלכלת הסייבר.

דוגמת הביטוח מדגישה לנו נקודה מאוד חשובה להמשך הדיון: הערכת הסיכון והמדידה שלו היא תנאי להבנה טובה יותר שלו ולחלוקה אופטימלית של משאבים.

מודל למדידת אירוע סייבר

הצורך במודל כלכלי אשר יתאר במושגים מקובלים גם את התופעה שנקראת סייבר הולך וגדל ככל שמתרבים אירועים קיברנטיים אשר משפיעים על הכיס שלנו. מסמך זה אינו מתיימר להציע מודל אמיתי ויציב למדידת הנזק הגלום באירועי סייבר, אולם הוא נועד לגרות את הדיון בדבר חשיבות פיתוח מודל\ים בתחום וקידום העיסוק בנושא.

אם נחזור לפסקת הפתיחה של מסמך זה, ניתן להשתמש ב-OpIsrael כמקרה בוחן לבדיקת הנקודות שיש להתייחס אליהן במודל:

גבולות גזרה - דיון על מדידה של אירועי סייבר חייב להכיל הגדרה מדוייקת לגבולות הגזרה. "מהו סייבר?" היא שאלה פילוסופית כמעט עם יותר מתשובה אחת נכונה. כחלק מכל עבודה או מחקר רציני אשר יעסקו במדידה של אירועי סייבר למיניהם, יהיה על מבצעי המחקר להגדיר במפורש מה הם כוללים או לא כוללים תחת המחקר ואילו "סוגי" סייבר יחושבו במסגרת המדידה. יש לציין כי רוב העבודות והמחקרים הנוגעים בהשפעת הסייבר על הכלכלה והחברה מתייחסים לפשיעה קיברנטית על סוגיה השונים ואינם נוגעים בהיבטים הביטחוניים-לאומיים.

נקודה נוספת למחשבה היא האם יש למדוד את כלל העלות של סייבר על המשק או את עלותם של אירועים חריגים כעלות נפרדת? בדוגמת OpIsrael ניתן למדוד את האירוע פעמיים: כחלק מעלות כוללת של תחום אבטחת המידע (ביחד עם כל תקציב החברה על אבטחת מידע וסייבר) או בפני עצמו ביחד עם אירועים חריגים אחרים (ואז ניתן לחלץ את העלות הנוספת למשק כתוצאה מאירועים "חריגים").

זהות הארגון - מהו אופיו הקיברנטי של הגוף שסובל מנזק קיברנטי (משרד ממשלתי, אתר לקניות ברשת, בנק). עד כמה הנוכחות שלו באינטרנט חשובה לעסקים? על פי נתוני חברת See Security, ב-OpIsrael הותקפו קצת יותר מ-300 אתרים (בניגוד למספרים אשר פורסמו על ידי מארגני ההתקפות). מסקירת האתרים עולה כי רובם אתרים פרטיים קטנים ורק מיעוטם אתרי ממשלה וגופים ציבוריים מוכרים[28].

[28] http://hackingdefined.org/opisrael/rss.xml

אופי ההתקפה - האם הגוף סבל באופן ישיר מההתקפה? אם כן, מהו אופי הנזק (השבתת שירות, השחתת אתר, גניבת מידע)? ההתקפות על האתרים ב-OpIsrael התחלקו על פי הפירוט הבא: 181 אתרים סבלו מהשחתה (Defacement), 128 אתרים סבלו מהתקפות למניעת שירות (DDOS) ו-12 אתרים מזליגת פרטים רגישים[29]. מצד שני, ארגונים רבים אחרים, בעיקר גופים רגישים ובעלי צביון לאומי החזיקו כוח אדם נוסף בעת ההתקפה ואף הפעילו כלים טכנולוגים (קיימים או חדשים) על מנת להטיב את התמודדותם עם ההתקפה. במובן זה יש לחשב אותם כחלק מסך "הנזק" של האירוע.

עיתוי - פרמטר שהתגלה כרלוונטי מאוד ב-OpIsrael הוא עיתוי ההתקפה: חלק מאתרי הממשלה הותקפו בשעות הלילה ולפנות בוקר ה-7 באפריל. ברוב המקרים דווח כי מדובר בהתקפות למניעת שירות. מיותר לציין, כי התקפה למניעת שירות אשר מבוצעת על אתר ממשלתי בשעות הלילה אינה אפקטיבית והנזק הישיר בעטיה קטן יחסית.

נזק למוניטין - חישוב הנזק למוניטין הוא אחת המשימות הקשות ביותר בכל מודל כלכלי מהסוג הזה. מכיוון שאין, ככל הנראה, נוסחה מוסכמת לחישוב גודל זה, יש להעזר ביוריסטיקות שונות. דרך אחת יכולה להיות על ידי ביצוע ניתוח לאחר מעשה להפסדים כספיים, כגון נטישת לקוחות, שניתן לזקוף אותם בדיעבד לאירוע קיברנטי מסויים וחישוב הנזק המוניטיני בהתבסס על הערכה זו. דרך אחרת יכולה להיות שימוש בתקציב השיווק והדוברות של הארגון ככלי עזר לחישוב מכיוון שהם הכלים המרכזיים בשמירה על מוניטין הארגון: בצורה זו ניתן לחלק את התקציבים הללו בחלק היחסי מהשנה שבו התרחש אירוע קיברטי. לדוגמא: אם תקציב השיווק והדוברות עומד על מיליון שקלים בשנה, אזי הנזק התדמיתי באירוע OpIsrael שנמשך (לצורך החישוב) 7 ימים הינו 19,178 = (7*365)/1M ₪. יש לציין כי מעלות זו צריך לקזז את החיסכון הצפוי בעלויות של הגברת מודעות העובדים בנושא אבטחת מידע לאורך השנה.

הערה חשובה אחרת מתייחסת לנקודת הייחוס השונה בין המגן והמתקיף בכל הקשור לנזק מוניטיני. באירוע OpIsrael הדבר בא לידי ביטוי בצורה מצויינת: במהלך ההתקפה הותקפו, כאמור, אתרים רבים באמצעות כלים למניעת שירות. כחלק ממהלך ההגנה, נחסמו על ידי ארגונים ומוסדות רבים פניות ממדינות מסויימות אשר הוגדרו כבעייתיות או פעילות בהתקפה (הפעלה אקטיבית של חוקי geo-location). כתוצאה מהפעלת חוקים אלה נחסמה בפועל תעבורת אינטרנט לגיטימית לאתרים ולצופה מהצד השתקפו שתי צורות שונות של אותה מציאות. מנקודת מבט ישראלית האתרים פעלו כרגיל והנזק המוניטיני נראה מזערי. אולם, מנקודת המבט של המתקיפים ואזרחים רבים אחרים ברחבי העולם, האתרים הושבתו לכל דבר ועניין ועל כן ההתקפה הוכרזה כהצלחה והנזק המוניטיני הועצם.

קיזוז הפסדים - בראיה מקרו כלכלית, יש להוסיף למודל מנגנון לקיזוז הנזקים בעזרת חישוב ההכנסות מההתקפה. עד כמה שזה נשמע מוזר, בהתקפה קיברנטית יש לא מעט גורמים אשר מגדילים משמעותית את ההכנסות שלהם. כאלה הן חברות הייעוץ והשירותים אשר מזרימות כוח אדם ללקוחותיהן ומעבירות

[29] http://hackingdefined.org/opisrael/searching.php

ללא הרף דיווחי מודיעין בזמן אמת על התפתחות ההתקפה. כאלה הן גם יצרניות המוצרים וחברות האינטגרציה אשר פועלות בימים שלפני ההתקפה לעיבוי מערך ההגנה ומכירת כלים חדשים ללקוחותיהן וכן תחזוקה של הכלים הקיימים בעת ההתקפה על פי צורך. על פי עדויות של חלק מחברות הייעוץ לקראת הכנת מסמך זה עולה כי אחדות מהן הגדילו את מחזורי העסקים בעשרות אחוזים בימים שלפני ובמהלך OpIsrael. אחרים דיווחו כי נרשמה עליה בהכנסות, אם כי "לא דרמטית". אחד ממנהלי החברות סיפר כי ההכנות לקראת OpIsrael סייעו לקדם פרוייקטים מתוכננים שהיו תקועים או כאלה שתוכננו להמשך השנה.

תפקיד המדינה במדידה קיברנטית

מהפסקאות האחרונות ניתן ללמוד, כי הנזק אשר נגרם באירועי OpIsrael היה שולי ביותר ברמה מקרו כלכלית ובכל מקרה רחוק מאוד מהמספרים שפורסמו על ידי המתקיפים עצמם. עם זאת, הערכת הנזק המוניטיני של ההתקפה מקשה מאוד על המדידה האמיתית של סך הנזק אשר נגרם בתודעה העולמית לישראל.

בהמשך לכך, מסמך זה מעלה נקודה כאובה מאוד בכל הקשור להתנהלות המיקרו והמקרו כלכלית בתחום הסייבר. העדר מדידה אפקטיבית לאירועי סייבר פוגעת בחלוקת המשאבים ובהכוונת התקציבים למקומות הנכונים. המדינה והרגולטורים הרלוונטים חייבים לבצע הערכת סיכונים מתמשכת וממצה בנושא הסייבר. המדינה (באמצעות המטה הקיברנטי) תתקשה מאוד להתוות מדיניות סייבר אם לא יהיו בידיה הכלים למדוד את השפעתה על הכלכלה ועל החברה. צעד חשוב ראשון בכיוון הוא מכתב[30] שהוציא בדצמבר 2012 המפקח על הבנקאים לכלל התאגידים הבנקאיים בנושא השלכות של סיכוני אבטחת מידע ותקריות קיברנטיות על הדוח לציבור. במכתב זה מנחה המפקח את הגופים להוסיף לדוחות הכספיים שלהם התייחסות לכל אירוע קיברנטי משמעותי, שכן לאירועים אלה השפעה אפשרית על הדוחות ועל כן יש ליידע את בעלי המניות בגינם.

לפי גישות מסויימות יש לפעול כדי לשנות את המציאות הזו במספר מישורים:

- יש לקדם פיתוח מודל ומתודולוגיה להערכת ומדידת איומי סייבר ולהנחילם במגזרים הרלוונטים במשק.
- יש לקדם מנגנונים לחובת דיווח על אירועי סייבר אשר יכללו הערכת נזק ואת האמצעים אשר ננקטו אצל המדווח על מנת למנוע הישנות הנזק והרחבתו.

30 http://www.boi.org.il/he/BankingSupervision/LettersAndCircularsSupervisorOfBanks/LettersOfTheBankingSupervisionDepartment/201223.pdf

- את הדיווח המגזרי יש לרכז בידי גורם אחד ברמת המשק על מנת שתתקבל תמונת מצב רוחבית ואמינה לגבי השפעת תחום הסייבר על הכלכלה ועל החברה.

דיווח עקבי ואמין על אירועי סייבר ועל עלות הנזק שלהם תעלה מאוד את השקיפות של הארגונים במשק, תמנע שמועות וספקולציות אשר מסמאות את הדיון הציבורי בנושא הסייבר ותפנה משאבים להמשך פיתוח התחום בישראל מול האתגרים הבאים.

The POODLE Attack

מאת שחר קורוט (Hutch)

הקדמה

בחודש אוקטובר האחרון, פרסמו שלושה מהנדסי אבטחת המידע של גוגל Bodo Möller, Thai Duong ו-Krzysztof Kotowicz מתקפה חדשה הנקראת POODLE (CVE-2014-3566). המתקפה למעשה מאפשרת לתוקף לגלות סוד הנמצא בבקשת לקוח העוברת בטווח תקשורת המוצפן באמצעות SSL. בכך, למשל, יכול תוקף לגנוב ממשתמש את ה-Cookie שלו גם אם העוגייה הוגדרה עם מאפייני אבטחה כמו Secure ו-HTTP Only. המתקפה תוקפת תקשורת מוצפנת מבוססת SSLv3.0, ומזכירה ברעיון שלה מתקפות מוכרות כמו ה-BEAST וה-CRIME. נזכיר שגירסא 3.0 שוחררה ב-1996 על ידי מהנדסי Netscape שפיתחו במקור את פרוטוקול ה-SSL.

מה החולשה?

השם POODLE הינו ראשי תיבות של **P**adding **O**racle **O**n **D**owngraded **L**egacy **E**ncryption. וכמו שהשם מרמז, המתקפה מבוססת על מניפולציות שהתוקף מבצע על ה-Padding בפרוטוקול. ה-**Padding** ("ריפוד") הינו ערך רנדומלי שנוסף למסר סודי שנשלח על ידי המשתמש כדי להקשות על תוקפים לפענח את התוכן. הריפוד נעשה על ידי אלגוריתם מוסכם לפני שלב ההצפנה, ולרוב אינו תהליך סודי. אפשר לשרשר מספר אקראי כלשהו למסר עצמו לפני ההצפנה כך שהוא מטשטש את המבנה המקורי של המסר. **Oracle** הוא פשוטו כמשמעו - נביא, כלומר גילוי התוכן המוצפן על ידי חולשה במנגנון ה-Padding (למי שרוצה להרחיב בנושא ממליץ בחום לקרוא את הכתבה של An7i, מבוא למתקפת Padding Oracle).

החולשה נמצא למעשה באופן שבו הפרוטוקול SSL 3.0 מבצע את תהליך ה-Padding ואינו חלק מהצפנת ה-MAC (Massage Authentication Code), לפיכך לא מתבצעת וולידציה בדבר אחר ערך ה-Padding והאם הוא זהה לערך שאותו שלח המשתמש.

על MAC מתוך ויקיפדיה:

> "**MAC**, הוא שם כולל לקבוצה של פונקציות עם מפתח סודי המשמשות לאימות (Authentication) והבטחת שלמות מסרים (Message Integrity). פונקציית MAC מקבלת מפתח סודי ומסר באורך שרירותי ומפיקה פיסת מידע קצרה הנקראת תג המשמשת לאימות וזיהוי מסר. אלגוריתם קוד אימות מסרים הינו סימטרי במובן שהשולח והמקבל חייבים לשתף ביניהם מראש מפתח סודי, באמצעותו המקבל יכול לוודא שהמסמך שקיבל אותנטי ושלא נעשה שינוי בתוכנו. ללא ידיעת המפתח לא ניתן לייצר את התג הנכון, ולכן אם נעשה שינוי כלשהו בהודעה, התוקף לא יצליח להתאים את התג בצורה שמתאימה להודעה ששונתה, והמקבל יבחין בשינוי."

משום שה-Padding אינו חלק מה-MAC, ניתן לערוך אותו וע"י כך לגלות מידע סודי הנמצא בבקשת ה-Request כמו למשל Secure & HTTP only Cookie.

מה ההתקפה?

הצפנת ה-SSL(3.0) עובדת כך שהלקוח לוקח את ההודעה הגלויה ומצפין אותה כ-MAC. לאחר מכן מוסיפים את ערך ה-Padding ומשלמים למספר הביטים הנדרש במסגרת ההצפנה (למשל AES128 משתמש במפתחות בגודל 16Byte) כאשר הביט האחרון ב-Padding משמש לספירת גודל ה-Padding. בשלב האחרון מצפין הלקוח את ההודעה באחת משיטות השונות של ההצפנה ב-SSL כך שכל חבילה נראת כך:

[((Message in Plain text)HMAC-SHA-1) + Padding]AES128

נראה איך זה נראה בפועל, נגיד והבקשה שלנו היא:

```
My secret password is So8dYAd14V. I want you to log me in.
```

לפני ההצפנה ההודעה שלנו נראת כך (בהקסדצימלי כאשר כל שורה היא 16 ביטים):

```
4d79 2073 6563 7265 7420 7061 7373 776f  My secret passwo
7264 2069 7320 536f 3864 5941 6431 3456  rd is So8dYAd14V
2e20 4920 7761 6e74 2079 6f75 2074 6f20  . I want you to
6c6f 6720 6d65 2069 6e2e 0abc 3a1f 0cf3  log me in.
02c5 80dd c869 66c0 1b2c 2a53 3c9d 5b00
```

כאשר הקטע המסומן בצהוב מ-BC עד 5b הינו ה-Tag של ה-HMAC-SHA-1 שהוסף אל ההודעה, ו-00 מסמל את גודל ה-Padding.

וכך נראת ההודעה לאחר ההצפנה:

```
33c5 3aa2 08a4 0ecf c408 c6df 0c7d ac47
c9fb a2e4 54c8 a316 78de 1ec2 cc6e 9600
0572 dcfd 25fc c941 e9fd 6ea9 9ca2 9e50
e4c3 5bc9 f4c6 d973 2412 03fb 1615 be93
7faf 2119 c740 becc 9095 c595 034a 6a61
```

במידה ונשנה את הביט האחרון בשורה הראשונה מ-47 ל-46, התוצאה לאחר Decrypt תראה כך:

```
860b 238a 9fc2 b909 6dd6 7642 05a8 85fe
7264 2069 7320 536f 3864 5941 6431 3457  rd is So8dYAd14W
2e20 4920 7761 6e74 2079 6f75 2074 6f20  . I want you to
6c6f 6720 6d65 2069 6e2e 0abc 3a1f 0cf3  log me in.
02c5 80dd c869 66c0 1b2c 2a53 3c9d 5b00
```

השורה הראשונה נהרסה משום שערכנו את המידע, אך אפשרי לראות עכשיו כי האות האחרונה בסיסמא השתנתה מ-V ל- W, הדבר נובע מכך שרוב שיטות ההצפנה של SSL3.0 עובדת בשיטת CBC. CBC עובד כך שלפני שלב ההצפנה בפועל כל בלוק מידע (16 ביטים כמו השורות שאנו עובדים איתם) עושה XOR עם הבלוק לפניו. (שימו לב כי השינוי מ-47 ל-46 הוא שינוי מתמתי כלשהו אלא עריכה שרירותית שלנו)

נחזור לדוגמא, נקח את ההודעה ונוסיף לה בלוק חדש של Padding:

```
4d79 2073 6563 7265 7420 7061 7373 776f  My secret passwo
7264 2069 7320 536f 3864 5941 6431 3456  rd is So8dYAd14V
2e20 4920 7761 6e74 2079 6f75 2074 6f20  . I want you to
6c6f 6720 6d65 2069 6e2e 0abc 3a1f 0cf3  log me in.
02c5 80dd c869 66c0 1b2c 2a53 3c9d 5b00
```

בעקבות כך שב-SSL3.0 ה-Padding הם ערכים רנדומלים ואין עליהם וולידציה, נוכל להשתמש באיזה ערכים שנרצה ל-Padding. נצפין את ההודעה:

```
33c5 3aa2 08a4 0ecf c408 c6df 0c7d ac47
c9fb a2e4 54c8 a316 78de 1ec2 cc6e 9600
3c87 de66 3db9 6961 3cee 12b2 0391 e2ba
e68c 5ff0 800c 72f7 78a6 78be 0866 826e
6889 b648 f1bd cbd7 294a 76b9 a51c 0632
08ab db46 cf99 bc60 c772 e3ce 3d15 c11b
```

ולאחר מכן נקח את המידע המוצפן ונעתיק את השורה בה מופיעה הסיסמא שלנו לסוף ההודעה במקום השורה האחרונה הנוכחית בתור ה-Padding שלנו כך:

```
33c5 3aa2 08a4 0ecf c408 c6df 0c7d ac47
c9fb a2e4 54c8 a316 78de 1ec2 cc6e 9600
0572 dcfd 25fc c941 e9fd 6ea9 9ca2 9e50
e4c3 5bc9 f4c6 d973 2412 03fb 1615 be93
c9fb a2e4 54c8 a316 78de 1ec2 cc6e 9600
```

אם נבצע Decrypt להודעה כעת נקבל את התוצאה הבאה:

```
4d79 2073 6563 7265 7420 7061 7373 776f  My secret passwo
7264 2069 7320 536f 3864 5941 6431 3456  rd is So8dYAd14V
2e20 4920 7761 6e74 2079 6f75 2074 6f20  . I want you to
6c6f 6720 6d65 2069 6e2e 0abc 3a1f 0cf3  log me in.
a562 4102 8f42 84d3 d87e 9c65 7e59 2682
```

מדוע 82?, בתהליך ההצפנה אנו עושים XOR עם הערך 93 שהוא בבלוק שמעל לבלוק הכתום. בהצפנה המקורית ה-XOR נעשה עם הערך 47 (ניתן להסתכל באיור הראשון המציג את המצב לאחר ההצפנה, בערך שבסוף השורה הראשונה). עכשיו באמצעות חישוב מתמטי פשוט נוכל לגלות מה הערך: במידה ו-X הוא האות האחרונה בסיסמא אותה אנחנו רוצים לגלות, נחשב:

$$X \oplus 47 \oplus 93 = 82 \underset{47 \oplus 93 = D4}{\Leftrightarrow} X \oplus D4 = 82 \underset{X \oplus A = Y \Leftrightarrow X = Y \oplus A}{\Leftrightarrow}$$

$$X = D4 \oplus 82 \Leftrightarrow X = 56$$

56 בהקסדצימלי משמעותו V, כפי שאנו יודעים זאת אכן האות האחרונה מהסיסמא המקורית שלנו.

[מקור התמונות: https://www.dfranke.us/posts/2014-10-14-how-poodle-happened.html]

כיצד מתבצעת ההתקפה?

על מנת לבצע מתקפה זו תוקף צריך להיות במצב של MITM כלפי תעבורת הלקוח. לרוב, יידרש התוקף להוריד את פרוטוקול החיבור של המשתמש באמצעות TLS_FALLBACK_SCSV, פאקטה זו מבקשת מהשרת לעשות שימוש בפרוטוקול ה-SSL 3.0, בשלב לאחר מכן התוקף יצטרך להריץ קובץ JavaScript על הלקוח שיגרום לו לשלוח את אותה הודעה עם הסוד באופן קבוע

כפי שאנחנו יודעים בחיים האמיתיים אין לנו את המידע המפוענח שהשתמשנו בו קודם כדי לחשב את ערך הסוד, אז למעשה נצטרך להשתמש בטריק קצת שונה אך עם מתמתיקה דומה.

מכיוון של SSL 3.0 בודק את תקינות ה-Padding רק לפי גודל ה-Padding (Padding byte) נוכל לנחש את הערך שנמצא בבלוק שמעל הסיסמא. כלומר הערך ההקסדצימלי 93 יוחלף בערכים הקסדצימלים (מ-00 עד ff) עד שנקבל את גודל ה-Padding Byte המקורי, או במקרה שלנו 00. למשל, אם נחליף את הערך 93 ב-11, שזה 82 (מהמסר שפענחנו קודם) $\oplus$ 93 כך:

```
e4c3 5bc9 f4c6 d973 2412 03fb 1615 be11
c9fb a2e4 54c8 a316 78de 1ec2 cc6e 9600
```

→ Y

93⊕00⊕Y(00-ff)=Z(82)

93⊕ Y(00-ff)=Z(82)

כאשר Y הוא מי שהחלפנו ב-11, ו-Z הוא ערך שאינו ידוע. אך ברגע ש-Y ישתווה במשוואה ל-Z השרת אמור לאשר את ההודעה, ולא להוציא הודעת שגיאה, משום שה-Padding Byte לאחר Decrypt יהיה 00 כפי שהוא ציפה. ובכך אפשר לחשב את 82 אחורה ולפיכך לחשב את ערך הסיסמא כפי שראינו בדוגמא קודם:

$$X \oplus 47 \oplus 93 = Z \ ,Z=82$$

$$X \oplus 47 \oplus 93 = 82 \underset{47 \oplus 93 = D4}{\Leftrightarrow} X \oplus D4 = 82 \underset{X \oplus A = Y \Leftrightarrow X = Y \oplus A}{\Leftrightarrow}$$

$$X = D4 \oplus 82 \Leftrightarrow X = 56$$

למעשה ההתקפה על ה-Padding עובדת בעצם פעם אחת מתוך 256 פעמים כאשר 255 פעמים השרת יחזיר שגיאה, אך פעם אחת מתוך 256 הפעמים המתקפה תעבוד ונוכל לבצע את החישוב ה-XOR. כאשר שנרצה להתקדם כדי לגלות עוד חלק מהסוד בעזרת ה-Padding Byte, נצטרך לקדם את כל הסוד בביט אחד, נניח והסוד שלנו הוא ה-Cookie אנחנו צריכים להוסיף עוד תו אחד למשל אל ה-URL כדי שנוכל לחשוף את האות הבאה בעוגיה, למשל אם ביקשנו מהמשתמש לבקר בדף הבא למשל:

http://www.digitalwhisper.co.il/0x76/

לאחר שגילינו את האות הראשונה בסוד נשלח את המשתמש אל:

http://www.digitalwhisper.co.il/0x76/?

וכך נוכל להתקדם אל האות הבאה בסוד וכאשר נגלה גם את האות הזו נוסיף עוד ביט:

http://www.digitalwhisper.co.il/0x76/?D

וכן הלאה עד שנחשוף את הסוד כולו (כאשר כל בקשה כזו מוסיפה ביט אחד).

חדי ההבחנה יבחינו ש-SSL 3.0 שוחרר בשנת 1996 ובוודאי הרהרו לעצמם למה שמישהו ישתמש ב-SSL היום ולא ב-TLS גם מבחינת השרת וגם מבחינת הדפדפנים שעדיין תומכים. העיקרון שעומד מאחורי ההחלטה הוא תמיכה לאחור, כלומר אם המחשב שלי רוצה להגיע לאתר אינטרנט אך השרת ישן ואינו יודע לדבר כלל ב-TLS. אנו נדבר איתו ב-SSL בשביל למנוע מניעת שירות לאתר.

חשוב לציין שמשתמשים במערכת Windows ושרתי Windows עד גרסאות של Windows 8 בגרסאות ה-Desktop ו-2012 בגרסאות ה-Server עדיין תומכים ב-SLL 3.0 באופן ברירת מחדל. בנוסף לכך, TLS_FALLBACK, הינה למעשה חבילה בתוך TLS המבקשת לבצע downgrade לגרסא מוקדמת יותר של TLS או ל-SSL וכבר שומשה בעבר כבסיס למתקפת ה-Renegotiation של Marsh Ray.

החבילה למעשה נכתבה במקור כדי שלא ידרשו לכתוב Extensions ל-TLS על מנת לאפשר התקשורת אל מול שרתים ישנים. כאשר לקוח שולח אל שרת בקשת TLS_FALLBACK והוא מסתיר את האפשרות שלו להתחבר בעזרת TLS ומצהיר שהוא תומך רק ב-SSL.

תוכלו להשתמש באתר הבא כדי לראות האם השרת שלכם תומך ב-TLS_FALLBACK, הידוע לשמצה:
https://www.tinfoilsecurity.com/poodle

שימו לב כי יכול להיות שהאתר שלכם תומך ב-TLS_FALLBACK אך אינו תומך ב-SSL 3.0 מה שמבטל את האופציה למתקפה.

כיצד ניתן להתגונן?

כרגע ההנחייה הינה לבטל לגמרי את התמיכה ב-SSL 3.0 משם שהפרוטוקול אינו בר תיקון בתצורתו הנוכחית, ואין הצדקה משאבית לתקן אותו. בפיירפוקס הודיעו שלא יספקו תמיכה ב-SSL 3.0 מגרסא 34 של פיירפוקס. משתמשי Chrome, Safari יוכלו למצוא הסברים כיצד לבטל את SSL3.0 כאן (משתמשי Explorer תמצאו כאן תיקון אוטומטי, יפתור לכם עוד הרבה בעיות ☺)

תודות

תודה **לליאור ברש** שהכניס אותי לעולם אבטחת המידע ועל שנים של הדרכה והכוונה. תודה **לשייע פידמן** ותודה **לכל משפחת באגסק** שמלווה תומכת ומלמדת. תודה אחרונה ומיוחדת **לעידן כהן** שעזר לי לערוך את המאמר.

מקורות מידע וקישורים להמשך קריאה

- http://googleonlinesecurity.blogspot.in/2014/10/this-poodle-bites-exploiting-ssl-30.html
- https://en.wikipedia.org/wiki/Transport_Layer_Security#SSL_3.0
- https://www.openssl.org/~bodo/ssl-poodle.pdf
- http://thehackernews.com/2014/10/poodle-ssl-30-attack-exploits-widely_14.html
- https://www.dfranke.us/posts/2014-10-14-how-poodle-happened.html
- http://en.wikipedia.org/wiki/Padding_(cryptography)
- http://en.wikipedia.org/wiki/Message_authentication_code
- http://he.wikipedia.org/wiki/%D7%A7%D7%95%D7%93_%D7%90%D7%99%D7%9E%D7%95%D7%AA_%D7%9E%D7%A1%D7%A8%D7%99%D7%9D
- http://www.digitalwhisper.co.il/files/Zines/0x10/DW16-3-PaddingOracle.pdf
- http://www.jbisa.nl/download/?id=17683062
- http://www.uniroma2.it/didattica/netsec/deposito/4_tls3.pdf

דברי סיכום

בזאת אנחנו סוגרים את הגליון ה-55 של Digital Whisper, אנו מאוד מקווים כי נהנתם מהגליון והכי חשוב- למדתם ממנו. כמו בגליונות הקודמים, גם הפעם הושקעו הרבה מחשבה, יצירתיות, עבודה קשה ושעות שינה אבודות כדי להביא לכם את הגליון.

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

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

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

www.DigitalWhisper.co.il

"Talkin' bout a revolution sounds like a whisper"

הגליון הבא ייצא ביום האחרון של שנת 2014.

אפיק קסטיאל,

ניר אדר,

31.10.2014