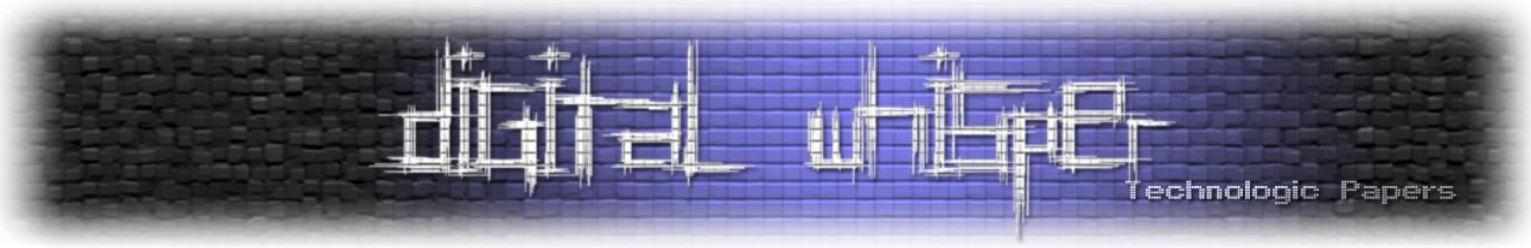


DNS Cache Poisoning

מאת אפיק קסטיאל (cp77fk4r)

הקדמה

זהו המאמר השני בסדרת המאמרים שהחלה מנושא ה-HTTP Attacks. המאמר הראשון בסידרה דיבר על מתקפת Response-Splitting, וכמו שנכתב במאמר הראשון, בסידרה זו נסקור מספר התקפות אשר מנצלות את אופי ואופן פעולותיהן של מספר פרוטוקולים. במאמר הזה נתמקד במתקפה המתבצעת על שירות ה-DNS. למתקפה קוראים "DNS Cache Poisoning", ובאמצעותה מתבצעות רוב מתקפות ה-Pharming למינהן, שאנו שומעים עליהם לאחרונה.

הרעיון ב-Pharming או ב-Phishing (מדובר בשני דרכי פעולה שונים) הוא לגרום למשתמש לחשוב שהוא גולש או מתקשר עם גורם אמין ורלוונטי מסויים, אך במציאות, הגולש נמצא באתר אחר- לרוב גורם זדוני.

כמו שאתם יודעים, האינטרנט מורכב מכתובות IP, כתובת IP יכולה להכיל רק ספרות מ-0 עד ל-255, מופרדות בנקודות (למרות שכיום חלק נכבד מהשירותים תומכים גם ב-IPv6 שמצוין כערכים הקסדצימאלים ויכולים להכיל גם את התווים A-F, אבל הרעיון הוא אותו רעיון). כדי להתחבר לרשת האינטרנט חובה על האובייקט (כל גוף המבקש להתחבר לאינטרנט), לקבל כתובת IP שתוקצה אך ורק לו, ובאמצעותה יהיה ניתן לזהותו (במקרה של נתבים ומתגים הם מקבלים כתובת אינטרנט חיצונית ומאפשרים למספר עמדות קצה להשתמש בה ע"י תיוגן בכתובות IP פנימיות לנתב). למה? בכדי ששאר "משתתפי" האינטרנט יוכלו לדעת לאיפה לשלוח את המידע שאמור להגיע לאותו אובייקט, וכך תוכל להתנהל תקשורת תקינה. לפני שהחל השימוש בשרתי ה-DNS, היתה חובה לדעת את כתובת ה-IP שלהם, ע"מ ליצור איתם קשר. אם היינו רוצים להתחבר לשרת מסויים דרך ה-BBS שלנו, היינו חייבים להכניס את כתובת ה-IP של השרת אליו רצינו להתחבר, זאת אומרת, שהיינו צריכים לזכור או לרשום הרבה מאוד מספרים, שלא אומרים יותר מדי, וזה דבר קשה יחסית ומעצבן. לכן, בשנת 1983, פותח פרוטוקול ה-DNS (קיצור של Domain Name Server) ע"י שני סטודנטים.

הרעיון שלהם היה ליצור מאגר מידע אשר יכיל רשימה ובה שמות מתחמים (באותיות **ולא** במספרים), ולצידם - כתובות ה-IP של המחשבים המארחים. כך, אם היינו רוצים לגשת לשרת שכתובת ה-IP שלו היא

65.33.123.212, לדוגמא, במקום לזכור את כל הספרות האלו, היינו פשוט צריכים לזכור את שמות האתרים (כמו שאנו זוכרים היום כתובות). הרעיון נחל הצלחה ענקית.

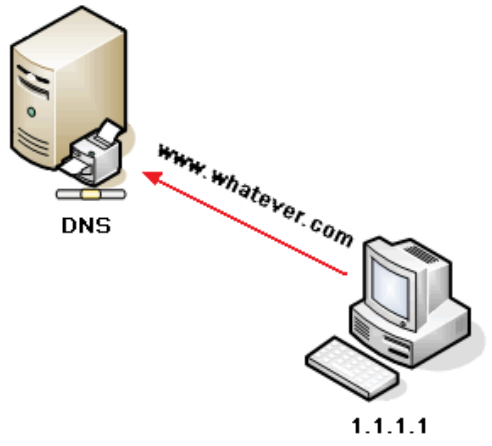
איך זה עובד?

אם אני רוצה להגיע לאתר בשם: www.whatever.com, אז אני כותב בדפדפן שלי, בשורת הכתובת, את כתובת האתר. המחשב לא יודע להתחבר לכתובת כזאת, כי לא מדובר פה בכתובת IP נומרית שהוא מסוגל להבין, אז הוא ניגש לשרת ה-DNS, ומתשאל אותו לגבי הכתובת הזאת. שרת ה-DNS בודק ברשימה שלו לאיזה IP שייכת הכתובת שהוא נשאל לגביה, ומחזיר את כתובת ה-IP למחשב. אז הדפדפן שלי ניגש לכתובת ה-IP ושולח בקשת HTTP בכדי להשיג את המידע.

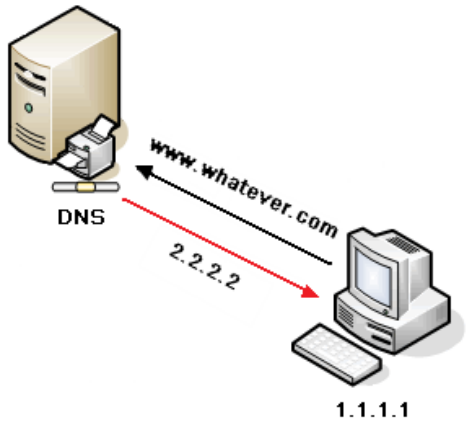
לפני ההסבר - חשוב לציין שלושה עובדות אשר משפיעות מאוד על נושא האבטחה בפרוטוקול:

- בכדי שהכל ירוץ במהירות, פרוטוקול ה-DNS רץ על-גבי UDP כברירת מחדל, דבר שמצד אחד מזכה אותו במהירות, אך גם גורם לצרות כשמדובר באבטחה ואמינות.
- בכדי לסדר יותר את עניין האמינות, שרת ה-DNS בוחר בפורט ראנדומאלי שממנו הוא שולח את המידע, ורק מהפורט הזה יהיה אפשר להחזיר לו את המידע, אך את המידע השולח יקבל, כברירת מחדל, בפורט 53.
- בכל תשאול, שרת ה-DNS מחולל ID מיוחד, שרק בעזרת צירופו לתשובה, הוא יוכל לדעת כי אכן מדובר בשרת שאותו הוא תשאל.

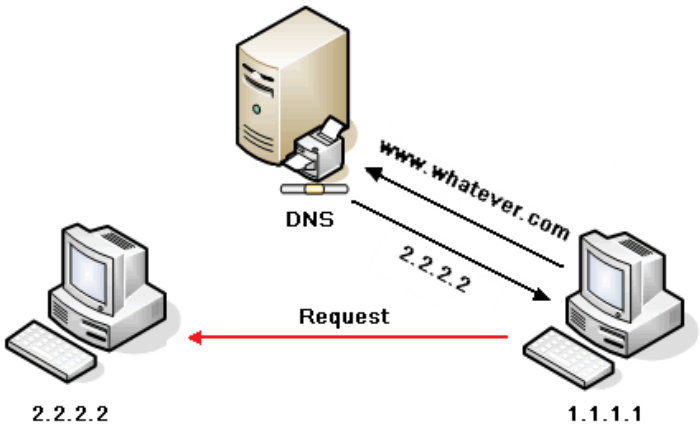
שלב ראשון - המחשב (1.1.1.1) מקבל כתובת מילולית (www.whatever.com) וניגש לשרת ה-DNS לתרגם אותה לכתובת נומרית בכדי לדעת לאיפה לשלוח את הבקשה:



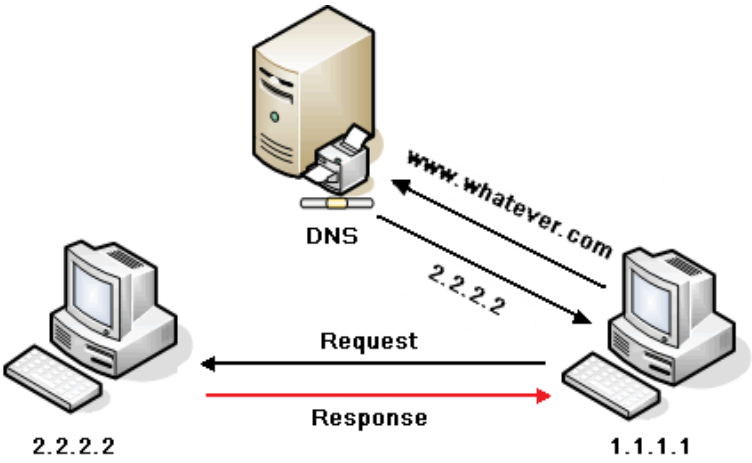
שלב שני - שרת ה-DNS בודק ברשימה שלו לאיזה כתובת נומרית (IP) שייכת הכתובת שהוא קיבל, ומחזיר את התוצאה (2.2.2.2) לשואל (1.1.1.1) :



שלב שלישי - השואל ניגש לכתובת שהוא קיבל משרת ה-DNS ומבקש ממנו תוכן:



שלב רביעי - שרת היעד (2.2.2.2) מחזיר את המידע שביקש הלקוח (1.1.1.1):



כך המחשב המבקש (1.1.1.1) מקבל את המידע שהוא ביקש, מבלי לדעת את כתובתו האמיתית של 2.2.2.2. בפרוטוקול ה-DNS ישנם סוגים רבים של רשומות, ובעזרתן אפשר לתשאל את שרת ה-DNS לגבי שירות מסויים, כגון שרת הדוא"ל, שמו של השרת וכו'.

רשומת לדוגמא:

- A- כתובת ה-IP של אותו שרת.
- MX- השרת האחראי על שליחת וקבלת הדוא"ל.
- NS- מידע לגבי אותו דומיין.
- AAAA - כתובת ה-IPv6 של אותו שרת.

ישנן עוד רשומות אך הבנתם את הרעיון הכללי.

במצבים ובהן שרת ה-DNS מבין כי אין ברשותו את המידע שביקשנו (כתובת הדומיין אינה נמצאת ברשימותיו), הוא מתשאל את שרת ה-DNS שמעליו, המכיל כתובות גלובליות יותר, ומבצע תשאול רקורסיבי עד שהוא מגיע לכתובת המדוייקת.

התשאול הרקורסיבי

נניח ותשאלנו את השרת לגבי הכתובת הבאה: **subdomain.whatever.edu.org.il**

לצורך הדוגמא - שרת ה-DNS מגלה שאין לו מושג מה כתובת ה-IP של אותו דומיין, אז הוא מבצע תשאול (בדיוק כמו שאנחנו תשאלנו אותו) לשרתי DNS שמעליו. לדוגמא, הוא יגיע לשרת הראשי (root) שמצויין ע"י נקודה: ".", וישאל אותו האם הוא מכיר את הכתובת המדוברת.

- השרת יגיד לו: "שמע אחי, אין לי מושג, אבל אני מכיר את מי שאחראי על: **.il**"
- המחשב יפנה אל מי שאחראי על ".il", וישאל אותו את אותה השאלה, התשובה שהוא יקבל תהיה דומה- "שמע אחי, אין לי מושג, אבל אני מכיר את מי שאחראי על: **.org.il**"
- כך הלאה, עד שהוא יגיע ל: **www.whatever.edu.org.il** וישאל אותו מה כתובת ה-IP של הסאב-דומיין: **subdomain.whatever.edu.org.il**
- כשהוא יקבל את התשובה הוא ישלח אותה אלינו בכדי שנוכל לבקש/לשלוח מידע מאותו מחשב.

בסופו של דבר, כל שרתי ה-DNS מתנקזים ל-13 שרתים הנקראים- Root Servers, ודרכם עוברות כל חבילות המידע בצורה זו או אחרת.

מנגנון ה- Caching

בכדי לייעל את הרעיון, ישנו מנגנון Caching, כך שלאחר שהשרת השיג כתובת IP של דומיין מסויים שקיים, הוא מוסיף אותו לרשימותיו (ל-Cache), וכך, אם נתשאל אותו עוד יומיים על אותה הכתובת- הוא יוכל לשלוף לנו אותה במהירות מבלי לרוץ שוב את כל הדרך שהוא עשה.

בפעם הראשונה שנבקש מהשרת את ה-IP של הכתובת:

subdomain.whatever.edu.org.il

הוא אכן יתרוצץ בין כל שרתי ה-DNS בדרך לאותו מחשב בכדי להשיג את כתובת ה-IP שלו, אך מעכשיו- בכל פעם שאנו נתשאל את שרת ה-DNS שלנו לגבי אותה כתובת, הוא לא יתרוצץ וינסה לבדוק לבדוק לאיזה כתובת IP אותו דומיין שייך, הוא פשוט ישלוף אותה מהרשימה שלו, וכך יקצר את הזמן שאנו נאלץ לחכות.

בכדי לראות את תוכן ה-Caching הנשמר במחשבכם כנסו ל-CMD ושם כתבו:

```
ipconfig /displaydns
```

תקבלו הרבה מאוד בלוקים של פרטי ה-DNS, הבלוקים מורכבים ממספר רשומות.

רשומה לדוגמא:

```
  stun2.l.google.com
----------------------------------------
  Record Name . . . . . : stun2.l.google.com
  Record Type . . . . . : 1
  Time To Live  . . . . : 234
  Data Length . . . . . : 4
  Section . . . . . . . : Answer
  A (Host) Record . . . : 209.85.137.126
```

כמו כן, קיים קובץ אשר בעזרתו אפשר להגדיר למערכת באופן ידני נתוני DNS, הקובץ נמצא:

```
%windir%\system32\drivers\etc
```

ושמו: Hosts

בכדי להוסיף נתונים פשוט כתבו את כתובת ה-IP, רווח (או טאב), ואז את כתובת ה-DNS שאליה אתם רוצים לשייך את כתובת ה-IP.

אם לדוגמא תוסיפו את השורה הבאה בסוף הקובץ:

```
209.85.129.147 www.microsoft.co.il
```

ותעשו PING לכתובת של מיקרוסופט ישראל, המערכת תשלח את הבקשה לכתובת ה-IP שמקבילה אליה בקובץ ה-Hosts, ואתם תגבלו תגובה מגוגל (ה-IP שייחסתם לכתובת של מיקרוסופט ישראל הוא ה-IP של אחד משרתיו של גוגל). אפשר לראות את זה באופן יותר מוחשי, ע"י הפעלת הדפדפן וכניסה לכתובת "www.microsoft.co.il", הדפדפן יציג לכם את עמוד הבית של גוגל, למרות שבשורת ה-URL תראו את הכתובת של מיקרוסופט ישראל. מה יקרה אם מישהו עם כוונות זדוניות, יצליח להגיע לקובץ הזה, ולשנות את כתובת האתר של הבנק שלכם, לכתובת של **אתר מראה** (Mirror Site) של הבנק שלכם, שנמצא בבעלותו של התוקף? כל מידע שתקישו באתר הבנק שלכם, יגיע לידי התוקף. לא חבל?

בכדי לנקות את המידע הנשמר במנגנון ה-Cache של המערכת שלכם, הכנסו שוב ל-CMD וכיתבו את הפקודה הבאה:

```
ipconfig /flushdns
```

שימו לב שאם תיכנסו לכתובת של מיקרוסופט ישראל עדיין תקבלו את האתר של גוגל. למה? כי הקובץ Hosts לא קשור למערכת ה-DNS Cache של מערכת ההפעלה, הוא נוסף אליה.

החולשה

מנגנון ה-Caching נועד להקל עלינו ולהפוך את זרימת הנתונים למהירה יותר, וזה אכן מה שהוא עושה, אך הוא גם יוצר חולשה רצינית בפרוטוקול ה-DNS. מערכת ה-Caching היא מערכת "לומדת", שתשאף להשאר מעודכנת כמה שיותר.

זאת אומרת, כשהיא נתקלת במידע חדש, היא רושמת אותו אצלה בכדי לקצר את התהליך לכשנבקש את אותו המידע בעתיד, ובמקרים ובהם היא תתקל במידע מעודכן (למשל, כתובת של שרת שהתחלפה בעקבות שינוי מיקומו), היא תבין שהיא לא מעודכנת, ותחליף את המידע הישן (הכתובת הקודמת של

השרת, אצלנו) שברשותה, במידע החדש שהיא קיבלה. הרעיון יפה מאוד, אך הוא גם החולשה של הפרוטוקול, המערכת "תלמד" את המידע החדש שהיא קיבלה מבלי לאמת שאכן המידע אותנטי (חוץ מכמובן אותו מספר ID אשר נשלח בכל חבילה), כלומר, שהכתובת החדשה שהיא קיבלה, היא באמת הכתובת המקורית של השרת. דבר המוסיף בעיה הוא השימוש בחבילות UDP.

כך, אם בדרך כלשהיא, התוקף יצליח לנחש את מספר ה-ID של ה-Packet שאותו שלח שרת ה-DNS שלנו לשרת ה-DNS מעליו, והוא יהיה מספיק זריז בכדי להחזיר לשרת ה-DNS שלנו תשובה עם מידע שגוי (אך עם ה-ID הנכון), המערכת לא תנסה לאמת שהמידע שהיא קיבלה אמיתי, אלא תסתפק בכך שה-ID נכון, ותעדכן את מנגנון ה-Cache שלה במידע שהתוקף שלח. כך, התוקף יוכל "להרעיל" (ומכאן שם המתקפה) את רשימת ה-DNS שלנו בכתובות מזוייפות, אשר יובילו את הגולש התמים לשרתים אשר מכילים קודים זדוניים, או אתרים פיקטיביים- וכך לגנוב את פרטיו.

דוגמא: אם יצליח התוקף להרעיל את הכתובת: www.Bank.com שהיא במקרה גם כתובת הבנק של הגולש, ויחליף אותה בכתובת ה-IP של שרת אשר מאכסן עליו אתר מראה (Mirror) של אותו הבנק, הגולש לא יוכל לדעת כי אכן מדובר באתר מראה ולא באתר המקורי (כתובת ה-URL תצביע על כתובתו המקורית של הבנק). גם אם הוא ישלח PING לדומיין הבנק, השרת הפיקטיבי יחזיר לו תגובה, ובעצם כל מידע שהוא ישלח לכתובת: www.Bank.com תמיד תגיע לשרת הפיקטיבי, ולאתר הפיקטיבי שנמצא בידי התוקף!

אופן המתקפה

כפי שראינו, בפני התוקף ניצבים שני בעיות עיקריות:

- עליו לדעת מה ה-Source Port שבו השתמש שרת ה-DNS בכדי לתשאל את שרת ה-DNS שמעליו.
- עליו לדעת מה ה-ID שבו שרת ה-DNS השתמש בכדי לתשאל את שרת ה-DNS שמעליו.

ה-Destination Port שאליו התוקף צריך לשלוח את התשובה עם המידע המורעל אינו משתנה. הוא ראנדומלי, אבל קבוע, כך שהדבר מהווה אתגר לא קטן, אך עם זאת, אפשרי לבירור.

הבעיה העקרית היא למצוא את ה-ID שבו השתמש שרת ה-DNS בכדי לתשאל את שרת ה-DNS שמעליו. איך אפשר לעשות את זה? נסביר על ידי הדגמה.

פתחו את תוכנת ה-Sniffer המעודפת עליכם (התוכנה Wireshark מומלצת בחום). תפעילו אותה תחת הפילטר: dns

כנסו ל "CMD", רוקנו את ה-Cache של ה-DNS של המערכת שלכם ע"י:

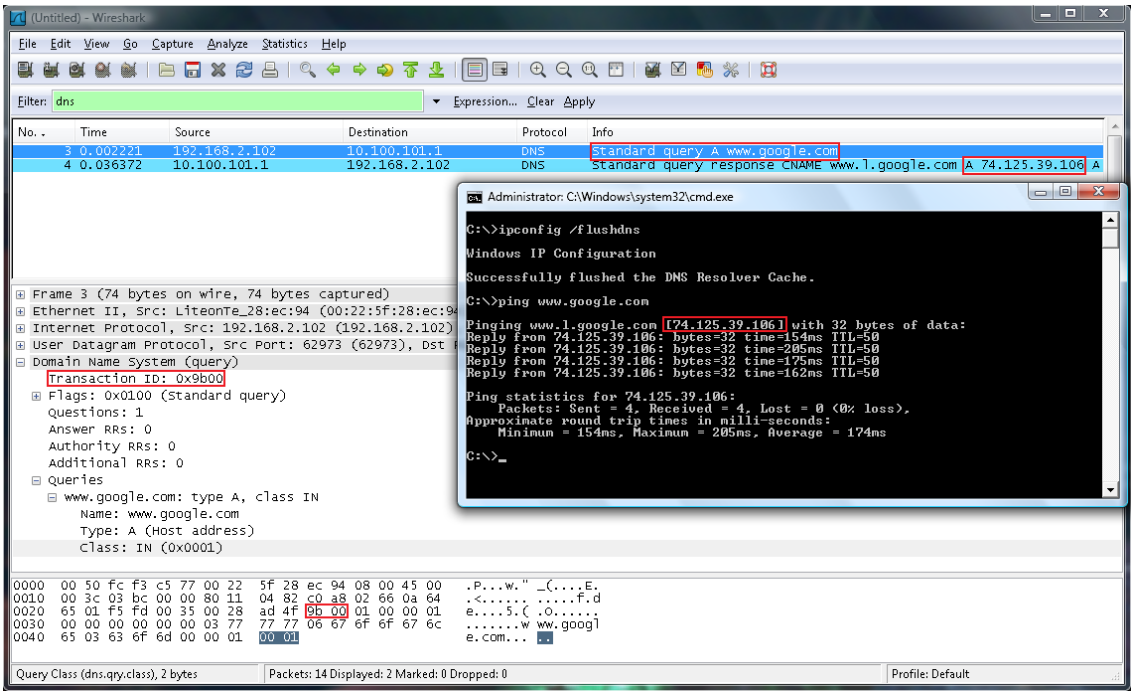
```
Ipconfig /flushdns
```

ושלחו פינג לגוגל:

```
Ping www.google.com
```

כמובן שה-Wireshark יתחיל לצעוק. בגלל שמחקנו את ה-Cache של מערכת ה-DNS שלנו, המערכת נאלצה לברר מי זה google.com, אז היא תשאלה את שרת ה-DNS שמעליכם.

ניגש ל-Wireshark, ונחפש את חבילת ה "Standard Query A" הראשונה שנשלחה, והסתכלו על הפרטים שלה. בהתחלה תראו את ה-Headers של ה-IP, אחריו את ה-Headers של ה-UDP, ואחריהם- את ה-"Transaction ID", וזה בדיוק מה שהתוקף שלנו צריך לנחש. אצלי הערך הוא: 0x9B00, שבדצימאלית זה בדיוק: 39680.



<u>שימו לב</u>: הדבר היחידי שאחראי על אימות אמינותו של ה-Packet בפרוטוקול ה-DNS, הוא מספר בגודל 2 bytes, **זאת אומרת שמספר ה-ID שמאמת את ה-Packet יכול לנוע בין 0 ל-65535!**

בעזרת מתקפת Brute-Force פשוטה (ע"י רשת של זומבים בינונית, למשל), אפשר לעלות על כל הצירופים האפשריים בפרק זמן קצר יחסית.

חלון הזמן של התוקף: על התוקף למצוא את מספר ה-ID לפני ששרת ה-NS של ה-DNS שמעליו, יענה לשרת המותקף. כי אם שרת ה-DNS שמעליו יענה לשרת ה-DNS המתשאל- הוא כבר קיבל תשובה, ואין לו שום סיבה לשאול אותה שוב (בפעם הבאה ששרת ה-DNS ישאל שוב לגבי אותו דומיין, זה יהיה רק לאחר שיפוג הזמן הקצוב ב-TTL שהוגדר ב-Packet). במצב רגיל, לשרתי ה-DNS, לוקח פחות משניה לתקשר ביניהם, וליידע אחד את השני מה כתובת ה-IP של דומיין מסויים, וזה הרבה פחות מהזמן שלוקח לתוקף לנחש את ה-ID המקורי שנשלח ב-Packet.

אז איך בכל זאת יכול התוקף לבצע מתקפה שכזאת בהצלחה? פשוט מאוד- בזמן שחצי מצבא הזומבים שברשותו מנסה לנחש נכונה את מספר ה-ID, החצי השני מבצע מתקפת DDoS / DoS על שרת ה-DNS שאמור להחזיר את התשובה לשרת ה-DNS המתשאל!

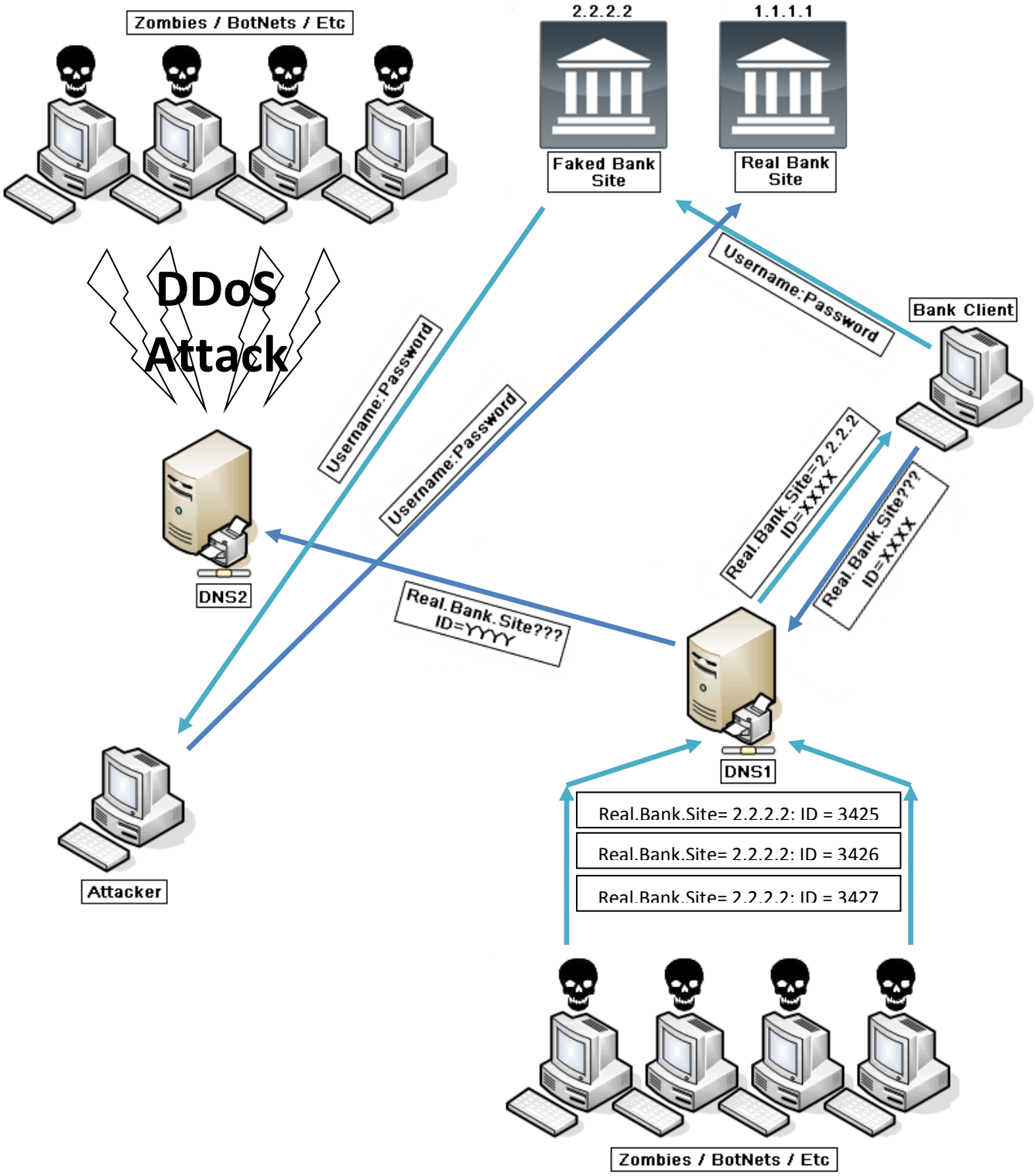
סיכום אופן המתקפה

כמו שאמרנו, הבעיה הרצינית בביצוע המתקפה על מחשבים מחוץ לרשת המקומית שלנו, על ארגונים או על אתרים ספציפים (אתרים המתעסקים בכספים, כגון Ebay, Paypal, אחרי בנקים גדולים וכו', אתרים לעדכוני חבילות אבטחה במוצרים נפוצים, כגון שרתי עדכונים למערכות אבטחה, שרתי עדכונים לדפדפנים נפוצים, עדכוני לשרתים וכו') היא למצוא את מספר ה-ID. וכמו שאמרנו, בידיים ריקות, לתוקף אין יותר מדי סיכוי. אך אם התוקף יבצע מספר פעולות בפרק זמן קצר- סיכוייו יעלו.

על התוקף לבצע בו זמנית:

- מתקפת מניעת שירות על שרת ה-DNS הנמצא "מעל" שרת ה-DNS המותקף, וע"י מתקפה זו- להפיל, או להאט אותו, וכך להרוויח זמן יקר.
- תשאול מאסיבי של שרת ה-DNS המותקף, לכתובת אותה הוא מעוניין להרעיל, וכך לגרום לשרת ה-DNS לשלוח מספר רב של תשאולים, כך ששרת ה-DNS ישתמש במספר רב יותר של חבילות המכילות מספרי ID שונים- מה שיקל על התוקף בניחוש (ככל ששרת ה-DNS שלח יותר חבילות, כך הסיכוי לנחש ID אחד גדל).
- שליחה מאסיבית של תשובה אחת מורעלת, כך שבכל שליחה מספר ה-ID שונה (ביצוע Brute Force - ניחוש שיטתי של מספר ה-ID).

ככל שהתוקף יפעל במסיביות רבה יותר, בכל אחד מהסעיפים, כך סיכוייו לבצע את המתקפה הזאת בהצלחה- גדלים. לכן, ככל שלרשות התוקף יעמדו מספר רב יותר של מחשבים- כך יגדלו סיכוייו להשלים את המתקפה. פירוט סכמתי של המתקפה, להבהרה ויזואלית של אופן המתקפה בעמוד הבא.


Zombies / BotNets / Etc
2.2.2.2
1.1.1.1
Faked Bank Site
Real Bank Site
DDoS Attack
Username:Password
Bank Client
Username:Password
Username:Password
Real.Bank.Site=2.2.2.2
ID=XXXX
Real.Bank.Site???
ID=XXXX
DNS2
Real.Bank.Site???
ID=YYYY
DNS1
Attacker
Real.Bank.Site= 2.2.2.2: ID = 3425
Real.Bank.Site= 2.2.2.2: ID = 3426
Real.Bank.Site= 2.2.2.2: ID = 3427
Zombies / BotNets / Etc

- **שלב ראשון** - לקוח הבנק (Bank Client) שואל את שרת ה-DNS שלו (DNS1) מה כתובת ה-IP של Real.Bank.Site, הוא משתמש ב-Transaction ID שלא ידוע לתוקף.
- **שלב שני** - שרת DNS1 מגלה כי הוא אינו בעל המידע הדרוש ולכן הוא מתשאל את שרת DNS2.
- **שלב שלישי** - התוקף (Attacker) מגייס זומבים, ומבצע מתקפת מניעת שירות (DDoS) לשרת DNS2 בכדי שהוא לא יוכל לשרת את DNS1.
- **שלב רביעי** - התוקף (Attacker) מגייס עוד זומבים, ומבצע מתקפת Brute-Force על DNS1 עם תשובות (מזוייפות) מ-DNS2. התשובות מכילות מידע כוזב לגבי כתובתו האמיתית של Real.Bank.Site ומפנות לכתובת ה-IP של Faked.Bank.Site - אתר מראה של הבנק אשר נמצא ברשותו של התוקף (Attacker).

התוקף חייב לבצע Brute-Force לערך ה-ID ששרת DNS1 השתמש בו, בכדי לתשאל את DNS2, ורק אם הוא יצליח לגלות את אותו מספר סודי- הוא יצליח לבצע את המתקפה בהצלחה.

- **שלב חמישי** - שרת DNS1 מקבל אלפי תשובות מהזומבים של התוקף, אחת מהתשובות מכילה את מספר ה-ID הסודי שבו הוא השתמש בכדי לתשאל את DNS2, ולכן הוא מניח כי הבקשה אכן התקבלה ממקור אמין ומשגר את התשובה ללקוח הבנק (Bank Client).
- **שלב שישי** - לקוח הבנק (Bank Client) נכנס לאתר המראה (Faked.Bank.Site) שנמצא תחת חסותו של התוקף בהנחה כי הוא אכן נמצא באתר המקורי של הבנק שלו, מכניס את פרטי ההתחברות (Username:Password), ומקבל הודעת שגיאה כי עקב תקלות אתר הבנק לא עובד (או כל הודעה שהתוקף קובע).
- **שלב שביעי** - אתר המראה (Faked.Bank.Site) שולח את פרטי ההתחברות של לקוח הבנק (Bank Client) לתוקף (Attacker).
- **שלב שמיני** - התוקף מתחבר לאתר המקורי של הבנק (Real.Bank.Site), מכניס את פרטי החשבון שהוא קיבל מאתר המראה שלו (Faked.Bank.Site) ומעביר את כל כספו של לקוח הבנק לחשבון סודי בשוויץ.

עוד נקודה חשובה היא שהתוקף **לא חייב לחכות שמישהו ישלח בקשה לשרת ה-DNS בכדי לנסות לזייף אותה**, התוקף לא צריך לדעת מתי לקוח הבנק מתשאל את שרת ה-DNS, הוא יכול לשלוח בעצמו בקשה לשרת ה-DNS בכל זמן ואז לבצע את המתקפה, ולהמתין ללקוח מזדמן.

הנושא הרבה פחות מעניין ומורכב, אך כאשר מדובר במתקפת DNS Cache Poisoning ברשת מקומית (Intranet), הדבר הופך לפשוט ביותר. ע"י היכולת לבצע מתקפת MITM בעזרת Arp Poisoning , בין הנתקף לבין שרת ה-DNS, אפשר בקלות לקבל את כל חבילות המידע הנשלחות מהנתקף, וע"י ביצוע Packet Manipulating פשוט לחבילות ה-DNS, ושינוי התשובה החוזרת משרת ה-DNS, בקלות אפשר לגרום לנתקף לחשוב שהוא נמצא בכל מקום שרוצים. במקרה כזה, אין שום צורך בניחוש מספר ה-ID שהנתקף השתמש בו, מפני שהוא שולח אותו אלינו. וכשאין צורך לנחש את ה-ID, אין צורך במתקפות DDoS, לא על שולח ה-Request ולא על שולח ה-Response.

עוד נקודה היא, שבמצב כזה התוקף לא חייב להרעיל את מנגנון ה-Cache של שרת ה-DNS, הוא פשוט יכול לכתוב סקריפט שמבצע Match & Replace לחבילות המידע העוברות דרכו, וכך לגנוב מידע.

בעזרת כלים מצויינים כגון Cain & Able, אפשר לבצע מתקפות כאלה ע"י לחיצת כמה קליקים, הכלי אפילו מתוכנן לזהות לבד מידע "מעניין" כגון סיסמאות, Certificates וכו'.

קישורים

הגענו לסוף המאמר. להלן מספר קישורים למאמרים שיכולים לעזור לכם להבין את הנושא באופן עמוק יותר:

- ארוך, אבל שווה לעבור עליו, אפילו בריפרוף, ה-RFC של הפרוטוקול:
  - http://www.ietf.org/rfc/rfc1034.txt
  - http://www.ietf.org/rfc/rfc1035.txt
- הערך בויקיפדיה על הנושא: http://en.wikipedia.org/wiki/Domain_Name_System
- האתר של ה-Root Servers, מכיל שמות, מיקומים, ומידע על כל שרתי ה-Root של האינטרנט שלנו: http://www.root-servers.org/