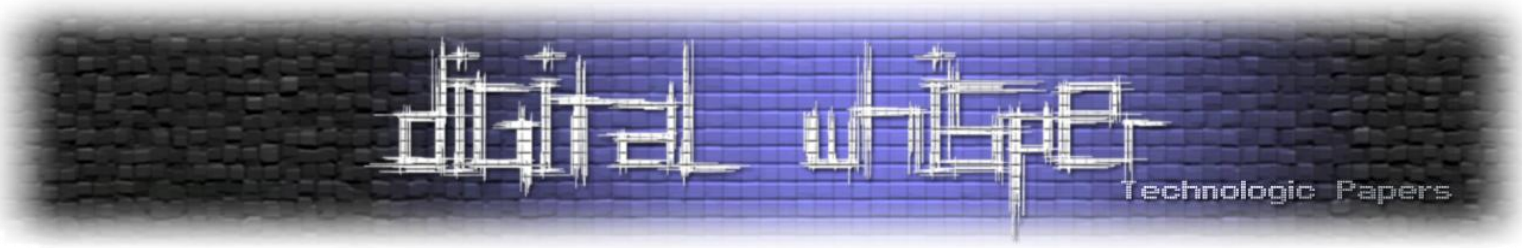

DNS Cache Snooping

מאת: עוז אליסיאן

הקדמה

DNS Cache snooping מתאר מצב שבו שרת DNS מתושאל על ידי גורם מסויים (בדרך כלל זדוני) בכדי "לחטט" בו, ולדעת האם השרת מחזיק ברשומה מסויימת בתוך המטמון (Cache). על ידי כך, אותו גורם יוכל להסיק האם בעל השרת או המשתמשים בו ביקרו באותו אתר (אשר מאוחסן כרשומה) ואף להסיק מתי.

במאמר זה נתאר איך נושא זה מתבצע. ידע מומלץ קודם הוא הכרות עם נושא ה-DNS - השרת והפרוטוקול. מאמר נוסף שמומלץ לקריאה לפני מאמר זה הוא המאמר המעולה של אפיק קסטיאל (cp77fk4r): DNS Cache Poisoning המסביר, בין היתר, באופן תיאורתי על הפרוטוקול.

איך עובד DNS Cache snooping

מנגנון ה-Cache בשרתי DNS

מנגון ה-Cache אשר קיים ברב שרתי ה-DNS הוא דבר יחסית סטנדרטי: אם נעשתה בקשה לגבי רשומה מסויימת, ושרת ה-DNS בסופו של דבר ענה לאותו גורם אשר ביצע את השאילתה (על ידי קבלת הרשומה משרת DNS הממוקם מעליו), השרת ישמור את הרשומה במנגון ה-Cache שלו, זאת בכדי שהוא לא יצטרך שוב לשאול את השרת שמעליו לגבי אותה רשומה, ויוכל לשרת את אותו גורם או גורם נוסף במהירות רבה יותר ובפחות עבודה אם יבקש שוב את אותו המידע.

שדה שחשוב להכיר הוא **שדה ה-TTL (בקיצור: Time To Live)** בשרת DNS - (חשוב לא להתבלבל עם TTL של חבילות IP). TTL עבור DNS, נועד בשביל לדעת כמה זמן יש להחזיק רשומה מסויימת במנגון ה-Cache. כאשר כל שרת DNS, האחראי על "Authoritative Nameserver", מחזיר תשובה, הוא יכול להגדיר TTL שונה אשר ינתן יחד עם התשובה לאותה בקשה. ה-TTL הנפוץ ביותר הוא בדרך

כלל 86,400 שניות, שהן 24 שעות. חשוב לציין בנוסף שגם המחשב עצמו של המשתמש הסופי מחזיק לרוב ברשומה בכדי למנוע מעצמו אפילו לתשאל את שרת ה- DNS מלכתחילה ולהקל עליו.

תשאול שרתי DNS

לנו, כמשתמשים של שרת DNS, עומדות בפנינו 2 סוגי שאילתות:

- Recursive Query
- Iterative Query

Recursive Query - או בעברית, שאילתה רקורסיבית. נתמכת על ידי מרבית שרתי ה-DNS. ובעצם מהווה את ההתנהגות הרגילה, הסטנדרטית והמוכרת לנו. לדוגמה, נניח כי משתמש מסויים מעוניין לגלוש באתר www.digitalwhisper.co.il. המחשב מתשאל את שרת ה-DNS שלו בשאילתה רקורסיבית מהו ה-A Record של האתר www.digitalwhisper.co.il.

A Record מהווה את התרגום מ-"Host Name" ל-IPv4. שרת DNS אמור להחזיר את כתובת ה-IP של האתר המבוקש בכדי שהמחשב יוכל לתקשר עימו.

פה בעצם נכנס עניין השאילתה הרקורסיבית: אם שרת ה-DNS, מחזיק כבר ברשומה (עקב שמירתה ב-Cache - משאילתה שנעשתה לפני כן) הוא יענה ישר "הנה קח את כתובת ה-IP של האתר www.digitalwhisper.co.il".

אם שרת ה-DNS אינו מחזיק ברשומה, (עקב זה ששאילתה לגבי האתר לא נעשתה לפני כן, או שזמן ה-TTL נגמר וכתוצאה מכך הרשומה נמחקה), והוא תומך בשאילתה רקורסיבית, יתחיל לבצע את התהליך הרקורסיבי הבא:

הוא יפנה אל שרת ה-DNS אשר מוגדר לו לקבלת המידע המבוקש. אם השרת הוא שרת DNS פנימי, אז ייתכן מאוד שברירת המחדל תהיה שרת ה-DNS של הספקית (ISP). אם זה שרת ציבורי, אז הוא ישאל את שרת ה-DNS המוגדר לו או אולי אפילו יתשאל את ה-Root Name Servers. אשר נמצאים ברמה הגבוהה בהיררכיה של שרתי ה-DNS ואמורים לספק את התשובות של שרתי "מפתח" (למשל מיהו שרת ה-DNS אשר מחזיק במידע לגבי הדומיין הישראלי co.il) ומשם הלאה.

אם השרת ה-DNS המתושאל אינו מחזיק ברשומה, הוא ילך וישאל את השרת ה-DNS המוגדר לו באותו האופן, יקבל את התשובה, ויחזיר את התשובה אל המחשב\השרת הקודת אשר ביצע את השאילתה.

Iterative Query - שאילתה איטרטיבית זהו סוג שאילתה המגדיר את הביט של הדגל הרקורסיבי כ-0.

נחזור שוב לדוגמא: נניח כי משתמש מסויים מעוניין לגלוש ל-DigitalWhisper, הוא שואל את שרת ה-DNS שלו, לגבי האתר www.digitalwhisper.co.il, והוא אינו מחזיק ברשומה במנגון ה-Cache. הוא יענה תשובה פשוטה: "אינני יודע, הנה שרת או כמה שרתים שמורשים/יכולים לענות לך על כך" כאשר בדרך כלל יצביע על השרת אשר מחזיק במידע לגבי הדומיין co.il.

היכן הבעיה?

כשנראה את הבעיה נוכל לתת דוגמא איפה הבעיה הזאת יכולה לשמש אותנו כנגד שרתי **DNS ציבוריים**, אבל הדבר המעניין באמת הוא שרתי ה-**DNS פנימיים בארגון**.

במקרים רבים, שרתים פרטיים (פנים-ארגוניים), אשר אמורים לשרת רק את משתמשי הארגון ולעזור בתשובות מהירות במקום שמחשבים פנים ארגוניים ילכו וישאלו שרתים מרוחקים, **עונים גם הם לשאילתות אשר מגיעות/מקורן ממחשבים חיצוניים לארגון.**

שרתים פנים-ארגוניים לא נועדו בכדי לענות לגורמיים חיצוניים. מטרתם אינה להוות שרתי DNS ציבוריים, אלא לשרת רק את הארגון עצמו ולתת לו את המהירות המקסימלית. המון מוצרי "הכל-כלול" (מוצר הכולל בתוכו IPS/NAT/AV/IDS וכו') מכילים שירות DNS, ובמקום לשרת רק את המחשבים הפניימים גם משרתים גורמים חיצוניים בלי ידיעה על כך – וכך, על-ידי תשאול אותם שרתים, גורמים חיצוניים לארגון, יוכלו להסיק האם בעל השרת או משתמשים בו, ביקרו באותו ואפילו להסיק מתי. מידע זה יכול להוביל לגילוי מידע וסטטיסטיקה לגבי יצרן / בנק / ספק אינטרנט / שותפים עסקיים ואפילו דברים מביכים או פרטיים.

הדגמה

בכדי לבצע את המתקפה הנ"ל ניתן להשתמש במספר רב של כלים, ישנם אפילו שירותי אינטרנט המאפשרים לתשאל דרכם את היעדים שלנו, במאמר הנ"ל, אשתמש בכלי "dig" על מנת לבצע שאילתה לשרת ה-DNS ממחשב מרוחק אשר נמצא ב-"External Network" ונראה גם איך מחשב פנימי (מאחורי שרת ה-DNS) גולש לאתר www.linkedin.com ומהי ההשפעה על תוצאת השאילתה שנבצע באמצעות הכלי "dig".

כמה מילים על הכלי הנ"ל: אפשר לומר שכלי זה הוא המחליף של "nslookup" ומאפשר יכולות רבות יותר, הכלי קיים גם ל-Windows וגם ל-Liunx.

אופן השימוש בו:

המחשב ה-"תוקף" אשר נמצא ברשת האינטרנט (וכמובן, מחוץ לרשת האירגונית), מבצע שאילתה לא רקורסיבית (= איטרטיבית), כאשר היעד הוא שרת ה-DNS של הארגון, הוא שואל מהי כתובת ה-IP, של האתר www.linkedin.com כמו שאמרנו בקשת - A Record:

```
C:\dig>dig @[redacted] www.linkedin.com A +norecurse

; <<>> DiG 9.3.2 <<>> @[redacted] www.linkedin.com A +norecurse
; (1 server found)
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 1352
;; flags: qr ra; QUERY: 1, ANSWER: 0, AUTHORITY: 6, ADDITIONAL: 8

;; QUESTION SECTION:
;www.linkedin.com.              IN      A

;; AUTHORITY SECTION:
linkedin.com.           50091   IN      NS      pdns4.ultradns.org.
linkedin.com.           50091   IN      NS      pdns3.ultradns.org.
linkedin.com.           50091   IN      NS      pdns5.ultradns.info.
linkedin.com.           50091   IN      NS      pdns2.ultradns.net.
linkedin.com.           50091   IN      NS      pdns1.ultradns.net.
linkedin.com.           50091   IN      NS      pdns6.ultradns.co.uk.

;; ADDITIONAL SECTION:
pdns1.ultradns.net.     50028   IN      A       204.74.108.1
pdns1.ultradns.net.     52232   IN      AAAA    2001:502:f3ff::1
pdns2.ultradns.net.     50130   IN      A       204.74.109.1
pdns3.ultradns.org.     49996   IN      A       199.7.68.1
pdns4.ultradns.org.     49996   IN      A       199.7.69.1
pdns4.ultradns.org.     51918   IN      AAAA    2001:502:4612::1
pdns5.ultradns.info.    49996   IN      A       204.74.114.1
pdns6.ultradns.co.uk.   49996   IN      A       204.74.115.1

;; Query time: 12 msec
;; SERVER: [redacted]#53([redacted])
;; WHEN: Tue Sep 27 02:44:02 2011
;; MSG SIZE  rcvd: 357
```

תחביר הקלט הולך כך:

```
dig @<DNS_IP> <Request_Site> A +norecurse
```

- A בשביל בקשת רשומת ה-IP בלבד.
- +norecurse עבור שאילתה לא רקורסיבית (מסמנים זאת מפני ש-dig מבצע ברב השאילתות בקשה רקורסיבית כברירת מחדל).

הפלט:

- ANSWER=0 - שרת ה-DNS ענה שהוא אינו מחזיק ברשומה שביקשנו.
- AUTHORITY=6 - כמספר השרתים "המורשים" אשר יכולים "לעזור".

שימו לב שאם היינו שולחים בקשה רקורסיבית אז היינו גורמים לרשומה להיכנס לתוך ה-Cache ואם היינו מבצעים שאילתה נוספת היינו מקבלים פלט שהרשומה נמצאת (כביכול מישהו ביקש אותה קודם לכן) אבל בפועל אנו גרמנו. **באמצעות שאילתה איטרטיבית אנחנו מונעים מצב זה** ויכולים להיות בטוחים שהרשומה אינה מאוחסנת באמצעות מספר חוזר של שאילתות בכדי לוודא את אמינות המידע.

וכעת, נניח כי מחשב פנים ארגוני אכן ניגש לאתר www.linkedin.com בכדי לבצע את עיסוקו. התוקף שוב חוזר על אותה שאילתה, הפעם ניתן לראות את השוני בפלט שהתקבל.

```
C:\dig>dig @[redacted] www.linkedin.com A +norecurse

; <<>> DiG 9.3.2 <<>> @[redacted] www.linkedin.com A +norecurse
; (1 server found)
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 1638
;; flags: qr ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;www.linkedin.com.              IN      A

;; ANSWER SECTION:
www.linkedin.com.       125     IN      CNAME   la.linkedin.com.
la.linkedin.com.        18      IN      A       216.52.242.80

;; Query time: 3 msec
;; SERVER: [redacted]#53([redacted])
;; WHEN: Tue Sep 27 02:45:51 2011
;; MSG SIZE  rcvd: 79
```

ניתן לראות כי שרת ה-DNS, ענה שהוא אכן מכיר את הרשומה ואינו הפנה אותנו אל שרתים המורשים לעזור לנו. **ניתן לראות זאת ע"י: ANSWER=X > 0 ובנוסף: AUTHORITY=0.**

ניתן גם לראות את שדה ה-TTL, של שני הרשומות שהתקבלו - 125 ו-18.

ניתן להבין ש-Linkedin. מעניקים לתשובה מספר יחסית קטן אשר מקשה על ביצוע סטיסטיקה לגבי זמני הביקור של אותו משתמש/ים.

ניתן לבצע שאילתה לפני כן ולראות מהו מספר ה-TTL הראשוני שהאתר מעניק, ולהשוות אותו עם ה-TTL שהתקבל (חשוב לציין כי המספר מציין "שניות").

קודם לכן, הזכרנו כי יש שימושים לטכניקה הזאת על שרתי DNS ציבוריים. דוגמא קצרה: במהלך הבלאגן שהלך ברשת עם סוני (כאשר דיסק שלהם התקין ללקוחות rootkit ושימש בעצם הכנה לדברים טובים נוספים אשר הגיעו...) החוקר Dan Kaminsky השתמש בטכניקה הזאת בכדי להעריך כמה שרתי DNS "יצרו קשר" עם השרתים אשר היו מעורבים בשליחה וקבלת נתונים עקב התקנתו של ה-rootkit. ההערכה הייתה אז כ-568,200 רשתות שונות.

סיכום

ניתן למנוע שימוש בטכניקה זו באמצעות הגדרות נכונות של שרת ה-DNS, לדוגמא: להגביל תשובות רק למקור אמין, למשל כתובת הרשת הפנימית. טכניקה נוספת היא לחסום מענה לשאילתות איטרטיביות ואז גורם חיצוני לא יכול להיות בטוח האם הוא זה שגרם לשאילתה להכנס ל-Cache או כי הדבר נגרם על-ידי גורם פנימי בארגון.

לשיטה השניה יש חסרון, קיימת דרך לגשש למרות חסימה כזו, שאומרת דבר כזה:

- תחילה, תתשאל לגבי אתר - בין אם הוא מאוחסן ב-Cache או בין אם תגרום לו להיות מאוחסן.
- תתשאל את אותו אתר שוב, ותבדוק כמה זמן לקח לתשובה להגיע. (לצורך הדוגמא, נני כי לקח לו 2 שניות)

- לאחר מכן, תתשאל שוב אתר אחר, ותבדוק כמה זמן לקח לתשובה להגיע. האם הזמן גדול משתי שניות? אם כן סביר להניח כי האתר לא היה קיים ב-Cache, מכיוון ששרת ה-DNS היה צריך לתשאל שרת אחר ובגלל זה לקח לו יותר זמן לענות!

טכניקה זאת מאופשרת באמצעות מנגון הסקריפטינג של Nmap, ובנוסף לאותו סקריפט יש אפשרות גם לבצע שאילתות איטרטיביות למספר רב של אתרים בו זמנית ולבנות מאגר גדול במהירות שמציין מה נמצא ב-Cache כרגע, ומה לא. לא נדגים זאת כאן, אבל ניתן בקלות לקרוא את ה-manual ולהבין כיצד הדבר עובד.

לסיום, טכניקה זאת יכולה לעזור המון באיסוף מידע על היעד ונראה שתתרום רבות בבניית מתקפת Social-Engineering. כי כאשר אנחנו יודעים על התחביבים / עניין / שגרת העובד - נוכל לבנות מתקפה טובה יותר ובעלת סיכויי הצלחה רבים יותר.

מקורות ומידע נוסף

- Dig tool
- Prevent with "simpledns"
- NSE - dns-cache-snoop
- Luis Grangeia - DNS Cache Snooping