**Cyber Lab – Kaminsky Attack**

**מגישים**:  
**אוריאל שפירא – 314779745**

**גיא שמעון - 209306513**

הסבר כללי:

במטלה זאת קיבלנו Docker שמכיל 4 מכונות user, seed-attacker, local-dns-server, attacker-ns והתבקשנו לבצע Kaminsky Attack באמצעות הAttacker על ה- local-dns-server

רעיון המתקפה:

המתקפה שנבצע היא מתקפת, Kaminsky שמטרתה לשנות רשומות DNS ולהטעות את שרת ה-DNS המקומי (local-dns-server) כך שהוא ישמור כתובת IP שגויה עבור דומיין מסוים (במקרה הספציפי שלנו [www.example.com](http://www.example.com) ).

התוקף שולח מספר רב של בקשות DNS עם קידומות אקראיות (xxxxx.example.com ) לדומיין היעד, ומקבל תגובות מזויפות משרת הDNS - שבשליטת התוקף. תגובות אלו מנסות להתאים את מזהה הבקשה (Transaction ID) לשאילתות שנשלחו. כאשר אחת מהתגובות המזויפות תואמת, שרת ה-DNS המקומי יכניס את הרשומה המזויפת לזיכרון המטמון (Cache), וכתוצאה מכך יספק בעתיד כתובת שגויה עבור הדומיין המבוקש (כתובת אותה התוקף קבע בתשובה המזויפת לשאילתת ה-DNS ששלח).

מטרת העל של כל מכונה:

**Seed-Attacker –** זו המכונה שמריצה את הקוד. היא שולחת את הבקשות לשרת ה-DNS המקומי יחד עם התגובות המזויפות. זאת בשביל להטעות את השרת ולבצע עליו Cache Poisoning. התפקיד שלה הוא ליצור בקשות DNS עם קידומות אקראיות ולשלוח תגובות מזויפות תוך ניסיון להתאים את ה-Transaction ID של התשובה ל-Transaction ID של הבקשה שנשלחת משרת ה-DNS ל-Authoritative NS (במטרה לקבל IP Resolution).

**NS-Attacker –** זהו שרת ה DNS - המזויף שמספק תגובות DNS מזויפות. השרת משרת את מטרת ההתקפה בכך שהוא עונה לשאילתות ה-DNS של שרת ה-DNS המקומי ומזייף את התשובות על מנת שהשרת המקומי ישמור את הכתובות המזויפות ב-Cache שלו. תפקידו העיקרי הוא לדמות את השרת ה-Authoritative שה-Local-DNS פונה אליו כדי לקבל את התשובות.

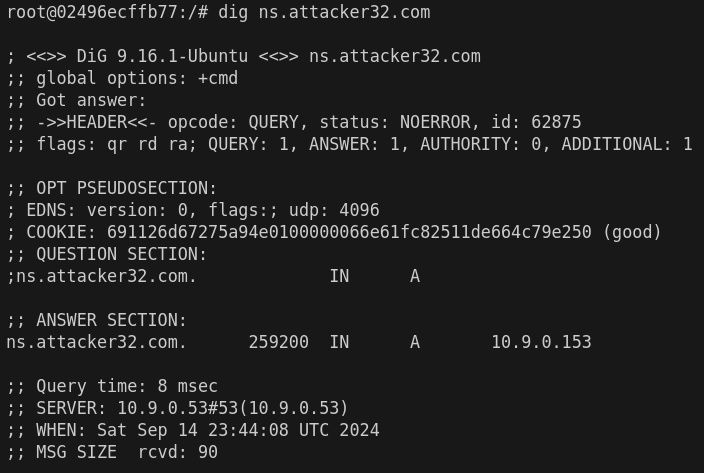
**Local-DNS-server –**זהו היעד של המתקפה. שרת ה-DNS המקומי שבו מתבצע ה-Cache Poisoning. מטרתו לספק IP Resolution לשאילתות אותן הוא מקבל. אם אינו יודע את ה-IP, פונה ל-Authoritative NS. אם יודע את ה-IP, משמע שהכתובת נמצאת ב-Cache שלו. אז אם ההתקפה מצליחה, שרת ה-DNS המקומי יספק למשתמש כתובת IP שגויה של האתר המבוקש (לדוגמה, במקום [www.example.com](http://www.example.com), הוא יחזיר 1.2.3.5 שזו כתובת מזוייפת).

**User –** מכונה המדמה משתמש רגיל הפונה לשרת ה-DNS המקומי כדי לבצע שאילתות DNS. מטרת המכונה היא לבדוק האם המתקפה הצליחה על ידי בדיקה איזו כתובת מתקבלת בשאילתת DNS עבור xxxxx.example.com (כאשר מחליפים את xxxxx ברצף אותיות רנדומלי). אם התקבלה כתובת IP מזויפת, משמע שההתקפה צלחה.

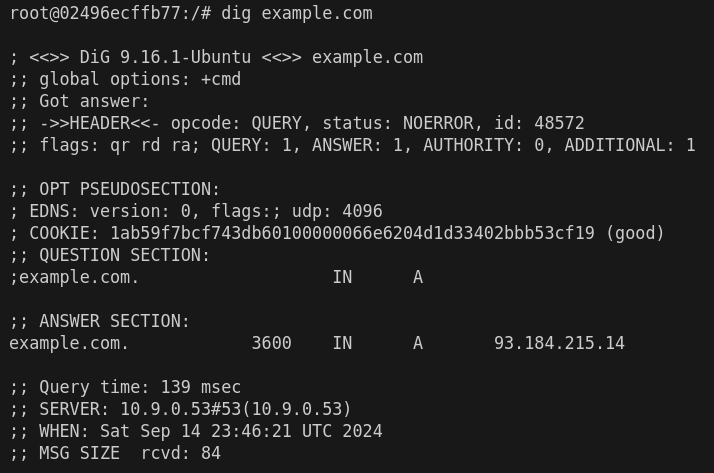
לאחר שהרמנו את כל ה-Container, נבדוק את מה שנדרש:

**2.4: (בדיקת הגדרת הDNS והסביבה)**

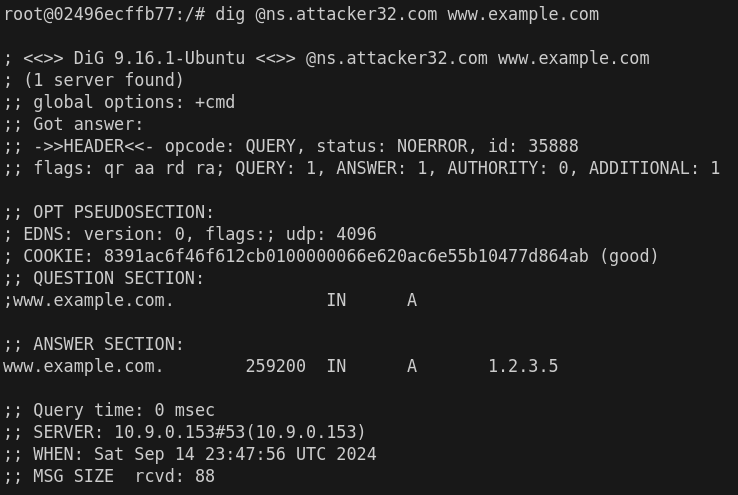
נבצע **dig ns.attacker32.com** ב-User ונקבל שכתובת ה-IP של ה-ns היא 10.9.0.153:



כעת, נריץ dig example.com (באופן רגיל) ונקבל את הכתובת הבאה:

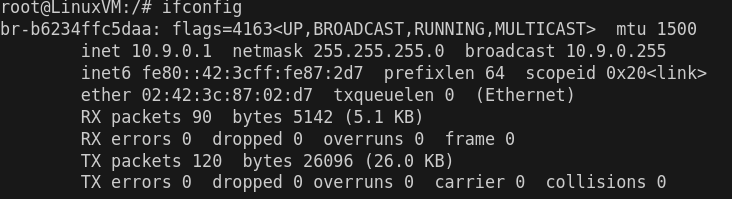


וכאשר נריץ את אותה בקשה דרך ה-attacker ns נקבל כתובת מזויפת:



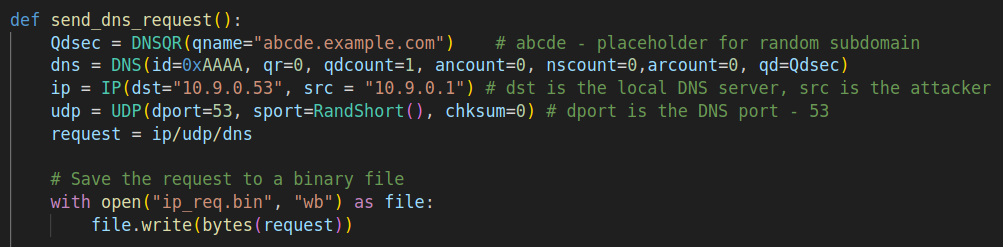
**3.2:**

לצורך בניית פאקטת ה-DNS, הרצנו את הפקודה ifconfig במכונת ה-attacker כדי למצוא את כתובת ה-IP שלו:



באמצעות scapy (יותר קל לבנות חבילה בPython מאשר ב C) ניצור בקשת DNS לכתובת בתצורה XXXXX.example.com.

מטרת הקוד הוא ליצור מעין תבנית לשליחת בקשת DNS מה-Attacker ל-local DNS. כך שבהמשך נשנה את הקידומת של XXXXX.example.com בלבד ונוכל לשלוח בקשות.



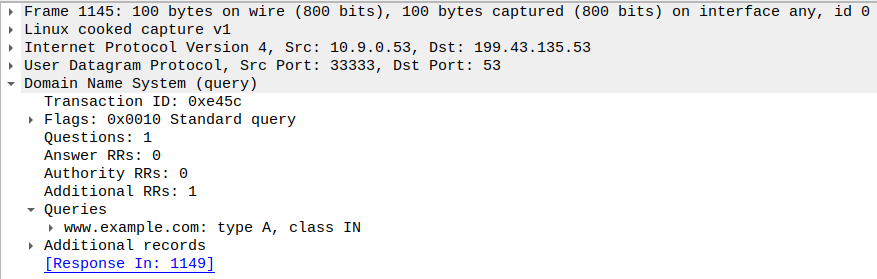
בקטע קוד זה אנו יוצרים שאילתת DNS עבור כתובת מסוימת בתצורת " XXXXX.example.com" ושומרים אותה בקובץ בינארי.

בקוד ב-C נקרא מהקובץ ונשנה רק את החלק של XXXXXX עבור כל שאילתה.

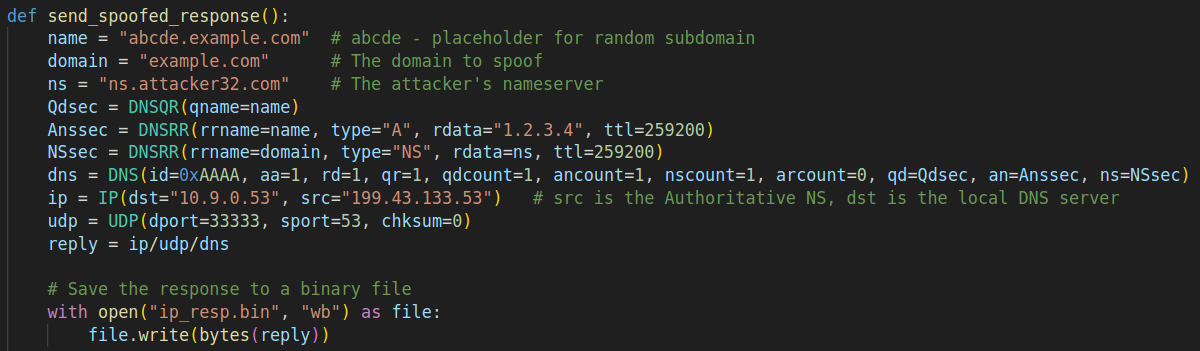
כמו כן, בקוד הזה אנו משתמשים ב-Source Port רנדומלי. אך אין לזה משמעות.

**3.3:**

בחלק זה נרצה לזייף תשובה לבקשת DNS, כך שהתשובה שלנו תחליף את ה-IP האמיתי של האתר www.example.com, ב-IP של ה-Attacker NS. את התשובה הזו נשלח ל-local DNS במטרה שהוא יכניס את התשובה הזו ל-Cache שלו ויענה לשאילתות הבאות עם ה-IP המזויף שניתן לו.



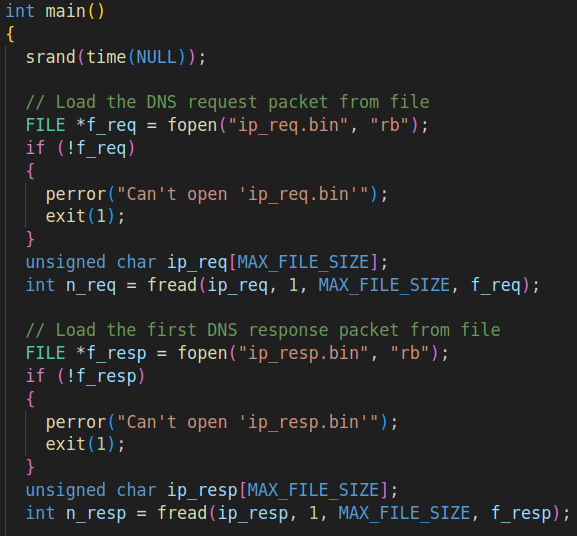
נשים לב שה-IP של ה-Authoritative Nameserver שה-local DNS פונה אליו כדי לקבל IP Resolution על [www.example.com](http://www.example.com) הוא: 199.43.135.53.



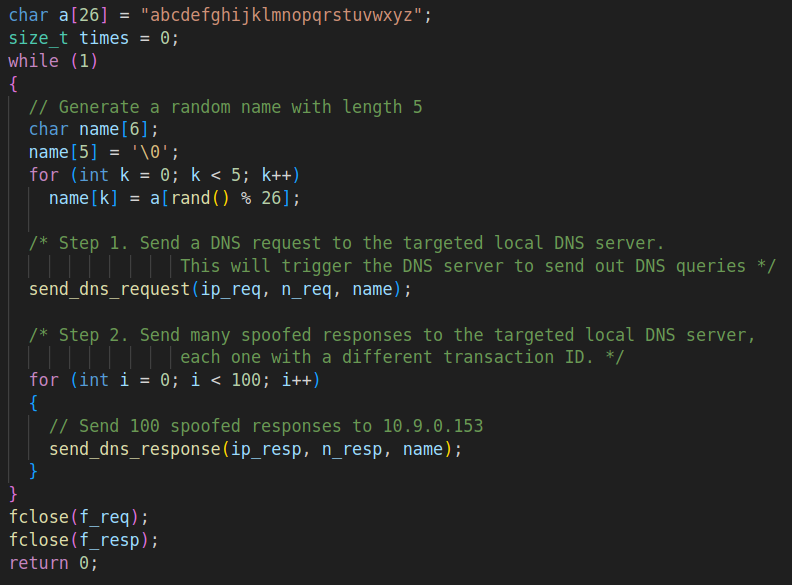
באמצעות הפרטים ממקודם, נבנה תשובת DNS מזויפת, עם פרטי ה-IP המתאימים (כך שהתשובה מגיעה מה-Authoritative NS ויוצאת אל ה-local DNS). והפורטים מתאימים גם כן לבקשה (כמו כן, ה-dport קבוע על 33333 כמו שנטען במטלה).

**3.4:**

באמצעות הקבצים שיצרנו בעזרת הקוד שכתבנו ב-Python, נבצע את המתקפה:



ראשית, נקרא את תוכן הקבצים שיצרנו ב-Python. הם יהוו תבנית לבקשות ותגובות ה-DNS שנשלח בהמשך.



ניצור תחילית בת 5 אותיות רנדומליות ונשלח בקשה לכתובת XXXXX.example.com כאשר XXXXX היא התחילית שיצרנו.

מיד לאחר מכן נשלח 100 תשובות DNS כאשר כל תשובה מכילה transaction id שונה. בתקווה שאחת מהתשובות תכיל את ה-transaction id המתאים.

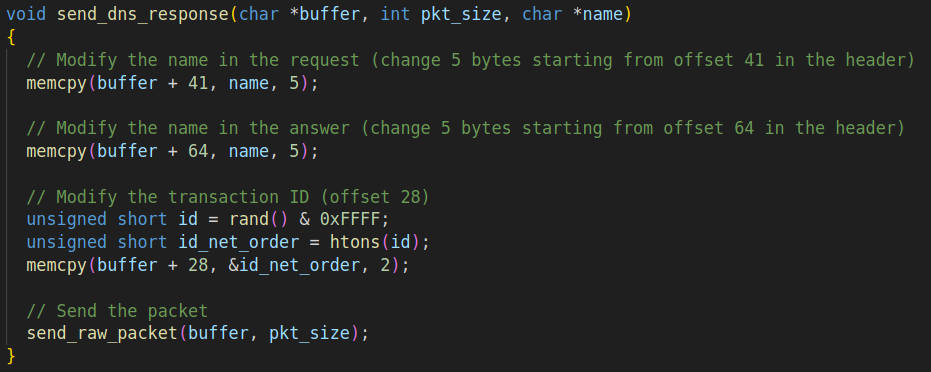
כעת נעבור להסביר את הקוד של השליחת בקשה ושליחת תשובת DNS:

A computer screen with text

Description automatically generated

קטע קוד זה משנה את התחילית של ה-buffer במיקום 41 בייטים מההתחלה לשם אותו שלחנו כארגומנט לפונקציה. ולאחר מכן שולח את הבקשה באמצעות פונקציית send\_raw\_packet.

בעצם הפונקציה משנה את החלק שדיברנו עליו קודם, של XXXXX.example.com ל-“name”.example.com כאשר name זהו צירוף בן 5 אותיות שנבחרו רנדומלית קודם לכן.



ניזכר שתשובת DNS מכילה גם את הבקשה אליה היא עונה.

לכן, נצטרך לשנות את התחילית המתאימה גם במיקום של הבקשה ב-Header וגם במיקום של התשובה ב-Header.

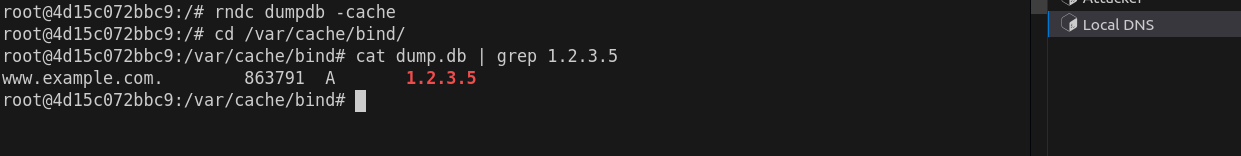
לאחר מכן, נגדיר transaction id רנדומלי – נבצע AND עם 0xFFFF שזהו הערך המקסימלי שיכול להיות ל-transaction id. כך שכל ערך שיחרוג מזה יאופס.

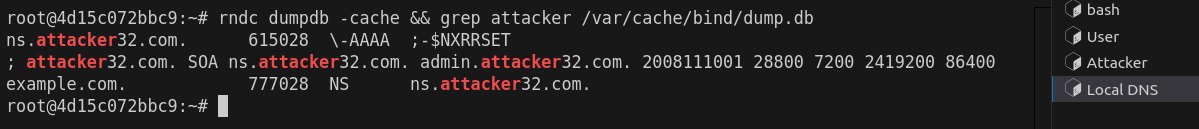
נשנה את ה-transaction id בפאקטה ונשלח אותה באמצעות פונקציית send\_raw\_packet.

כעת נבצע את ההתקפה.

לאחר מס' נסיונות של ה-Attacker לבצע Cache Poisoning, אנחנו יכולים לראות בתמונות הבאות שהוא אכן הצליח.

התחברנו ל-Local DNS וראינו שב-Cache שלו, תחת [www.example.com](http://www.example.com) מופיע ה-IP שהגדרנו.

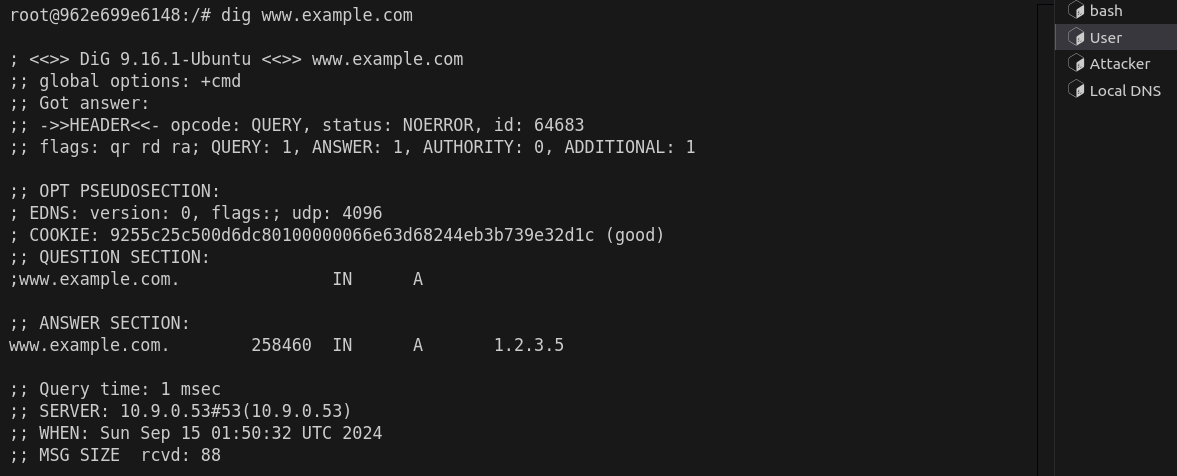




כמו כן, ניתן לראות בשורה האחרונה ש-example.com מופנה ל-ns.attacker32.com. שזו הייתה מטרת כל ההתקפה.

**3.5:**

כעת נראה דרך ה-User שהמתקפה אכן פעלה:



A screenshot of a computer

Description automatically generated

ב-2 התמונות המצורפות ניתן לראות שאנו מבצעים dig ל-www.example.com באופן רגיל ודרך ה-Attacker NS, ושניהם מחזירים את אותו IP עבור האתר.

משמע שהצלחנו לגרום ל-Local DNS לשמור את הכתובת אותה ה-Attacker NS מחזיר.

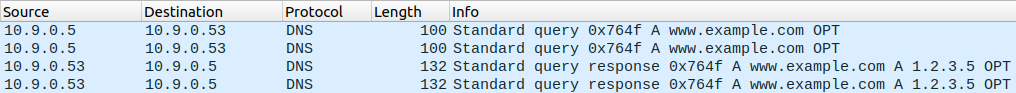
**הקלטות Wireshark:**

A screenshot of a computer

Description automatically generated

לפני שההתקפה הצליחה, ניתן לראות בתמונה דוגמה לכך שה-User מבצע שאילתת DNS על “bvnsi.example.com”, ומיד לאחר מכן נשלחות 100 DNS Responses. כל אחת עם transaction id שונה (מסומן באדום).

לאחר שהצלחנו לבצע Cache Poisoning:



ניתן לראות שכאשר אנו מבצעים שאילתה עבור [www.example.com](http://www.example.com), השאילתה אכן נשלחת ל-10.9.0.53 שזו הכתובת של ה-Local DNS. והוא מחזיר תשובה שהיא כתובת ה-IP שהגדרנו (1.2.3.5).

**הגנה**:

להלן כמה אופציות להגנות אפשריות כנגד מתקפה זאת:

1. כפי שצוין בקובץ, DNSSEC הוא פתרון יעיל למתקפה בכך שכל תשובת DNS חתומה עם מפתח פרטי שמוחזק על ידי שרת ה-Authoritative בנוסף הלקוח מקבל גם את החתימה עם המידע המבוקש

בכדי לאמת את החתימה שרת ה-DNS המקומי מבקש את המפתח הציבורי של ה-Authoritative DNS. אם חתימה זאת תואמת את המפתח הציבורי השרת יודע כי המידע לא שונה ויכול להוסיף את התשובה ל cache שלו.

1. ניתן להגביל את כמות מספרי התשובות שיתקבלו עבור כל שאילתת DNS ובך אנחנו מקטינים את הסיכוי של התוקף לפגוע ב Transaction ID .