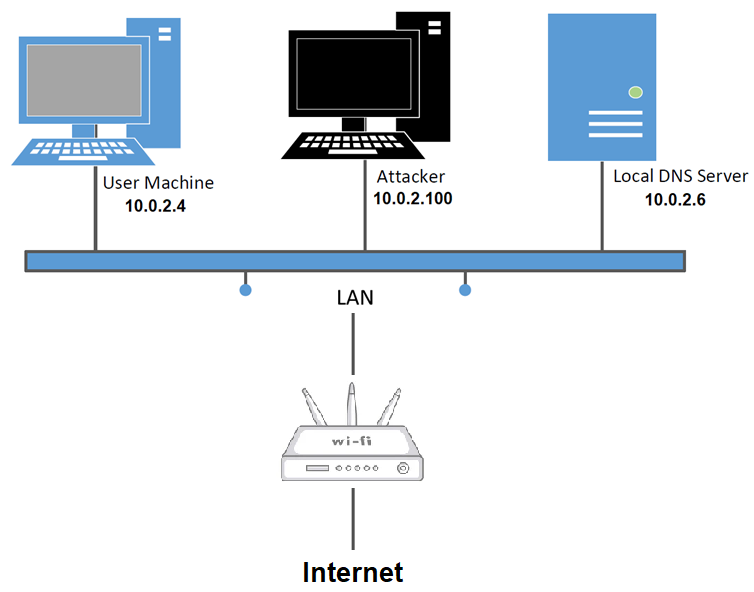
**A03 – Local DNS Attack Lab**

כתובות IP לכל מחשב

|  |  |  |
| --- | --- | --- |
| Name | IP | MAC |
| Attacker | 10.0.2.100 | 08:00:27:73:FC:14 |
| Client | 10.0.2.4 | 08:00:27:FA:2A:A7 |
| Server | 10.0.2.6 | 08:00:27:C3:04:42 |

**Lab Tasks (Part I): Setting Up a Local DNS Server**

התמונה מטה מייצגת את מבנה הרשת המתאים לכל המשימות של Part I



Task 1: Configure the User Machine

* מבוא:
  + תיאור

במשימה זו נרצה להגדיר בCLIENT מי השרת DNS שאליו הוא יפנה את הבקשות ויקבל ממנו את התשובות

* + מטרה

להגדיר שרת DNS לוקאלי שאליו הCLIENT ישלח שאילתות DNS ויקבל מענה לשאילתות האלו.

* + תוצאה מצופה

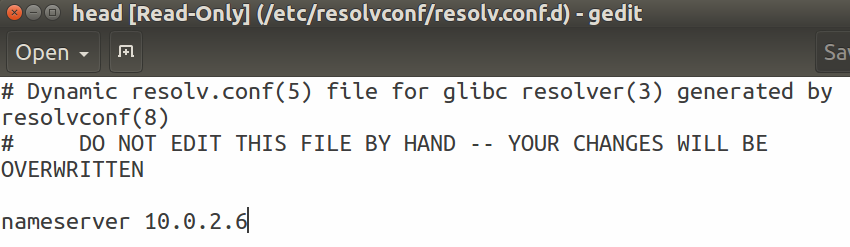
כל בקשות הDNS של הclient ישלחו אל מחשב הביניים שמשמש כ local dns וכל התשובות לשאילתות האלו ישלחו מהlocal dns אל הclient, כלומר הlocal dns יתקשר עם הרשת החיצונית ויעביר את המידע לclient.

* ביצוע המשימה:

תחילה נרצה להגדיר במחשב הCLIENT ששרת הDNS העיקרי שלו יהיה מחשב השרת IP 10.0.2.6 לכן נבצע עריכה לקובץ /etc/resolvconf/resolv.conf.d/head ונוסיף את השורה הבאה:

Nameserver 10.0.2.6

צילום ממחשב הCLIENT



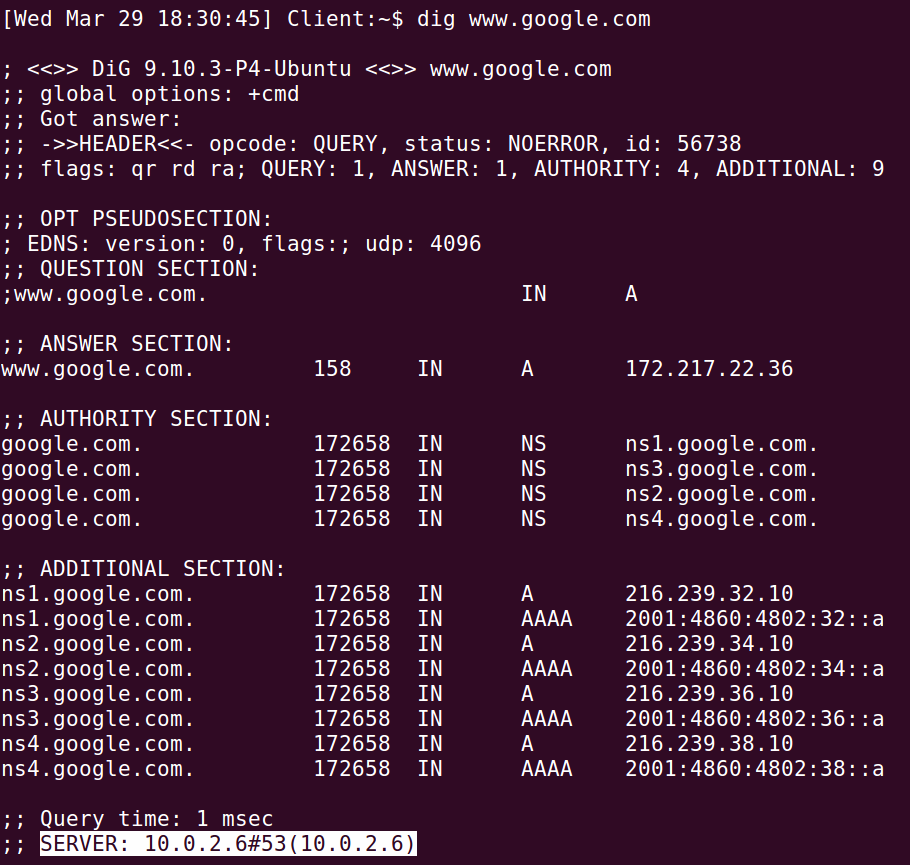
כעת נרשום את הפקודה הבאה בטרמינל:

sudo resolvconf -u

לצורך החלת השינויים בקובץ

כעת נרצה לבדוק במחשב הCLIENT שאכן שרת הDNS דרכו הוא מעביר בקשות הוא השרת שלנו 10.0.2.6

נבצע זאת על ידי הפקודה dig [www.google.com](http://www.google.com)



ניתן לראות בשורה התחתונה המסומנת שאכן הסרבר דרכו עברה הבקשה הוא הסרבר שלנו 10.0.2.6

SERVER מציין את השרת שענה לבקשה שלנו

53 מציין את הפורט שבו השרת מאזין לבקשות שלנו

* סיכום המשימה

הצלחנו לבצע את המשימה, ניתן לראות שהגדרנו את מחשב 10.0.2.6 בתור local dns, וחיברנו את הclient ל10.0.2.6 כך שכל שאילתות הDNS יעברו דרכו

הראינו בעזרת הפקודה DIG שאכן כל השאילתות DNS נשלחות לlocal dns שהגדרנו שהוא 10.0.2.6.

גילינו כיצד להגדיר באופן סטטי שרת DNS שיעביר את שאילתות הDNS של המחשב.

התוצאות התאימו למצופה מאחר והPACKETS של שאילתות הDNS הועברו דרך ה LOCAL DNS שהגדרנו.

לא נתקלנו בבעיות במהלך ביצוע המשימה.

Task 2: Set up a Local DNS Server

* מבוא:
  + תיאור

במשימה זו נרצה להגדיר שרת DNS לוקאלי ולחבר את מחשב הclient אליו כדי שיעביר דרכו את שאילתות הdns query ויקבל ממנו את שאילתות הdns response

* + מטרה

להגדיר שרת DNS לוקאלי שיתווך בין הclient לרשת החיצונית, ולראות שאכן הבקשות של הclient יקבלו מענה דרך שרת הDNS הלוקאלי שהגדרנו

* + תוצאה מצופה

כל בקשות הDNS של הclient ישלחו אל מחשב הביניים שמשמש כ local dns וכל התשובות לשאילתות האלו ישלחו מהlocal dns אל הclient, כלומר הlocal dns יתקשר עם הרשת החיצונית ויעביר את המידע לclient.

* ביצוע המשימה:

**את השלבים הבאים נבצע במחשב השרת 10.0.2.6**

* שלב ראשון Configure the BIND 9 server

נרצה להגדיר את שרת ה- BIND 9 וליצור נתיב לזריקת הCACHE של השרת

נכנס לפי ההוראות במעבדה לקובץ בנתיב: /etc/bind/named.conf.options.

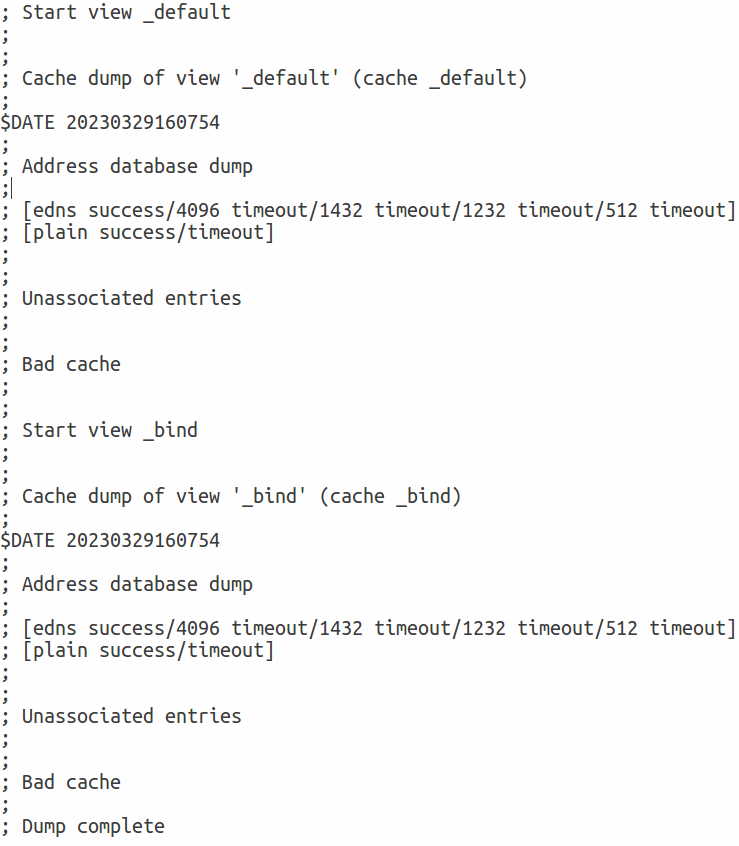
ונוסיף בשורה האחרונה בתוך הבלוק הראשי של option את השורה הבאה:

dump-file "/var/cache/bind/dump.db";

כעת נרצה ליצא את הCACHE של שרת הDNS לקובץ הdump המיועד שלנו על ידי הפקודה הבאה בטרמינל:

sudo rndc dumpdb -cache

וקיבלנו את התוצאה הבאה:



ניתן לראות שהCACHE נזרק לקובץ המיועד שהגדרנו

* שלב שני: Turn off DNSSEC

כעת נרצה לכבות את ההגנה של שרת הDNS מפני SPOOFING ATTACKS, נעשה זאת על ידי הוספת השורה הבאה:

dnssec-enable no;

בקובץ named.conf.options

* שלב שלישי: Start DNS server

כעת נרצה להפעיל את שרת הDNS במחשב השרת 10.0.2.6 על ידי הפקודה הבאה:

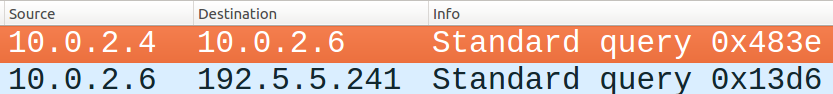
sudo service bind9 restart

את הפקודה נרצה לבצע כל פעם שנשנה את הגדרות השרת כדי לאתחל אותו מחדש או כאשר נרצה להפעיל אותו.

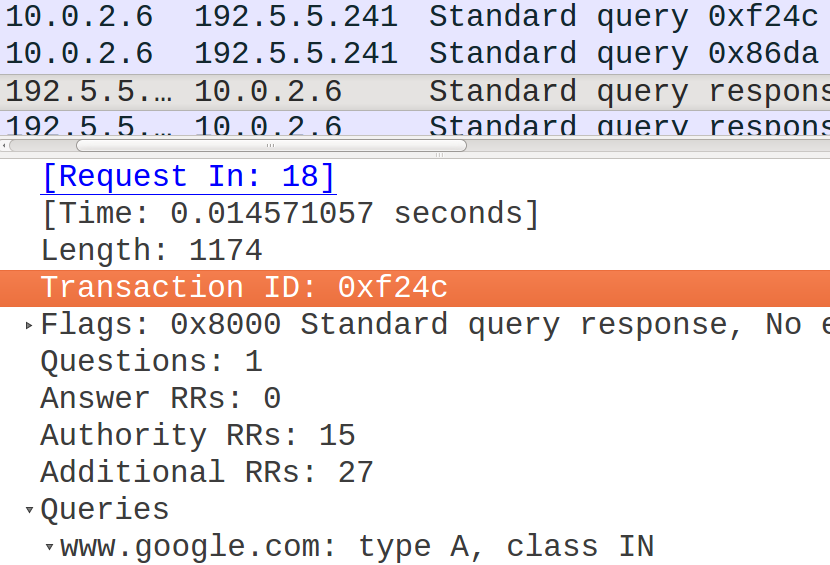
* שלב רביעי: Use the DNS server

נשלח PING מהCLIENT לשרת GOOGLE

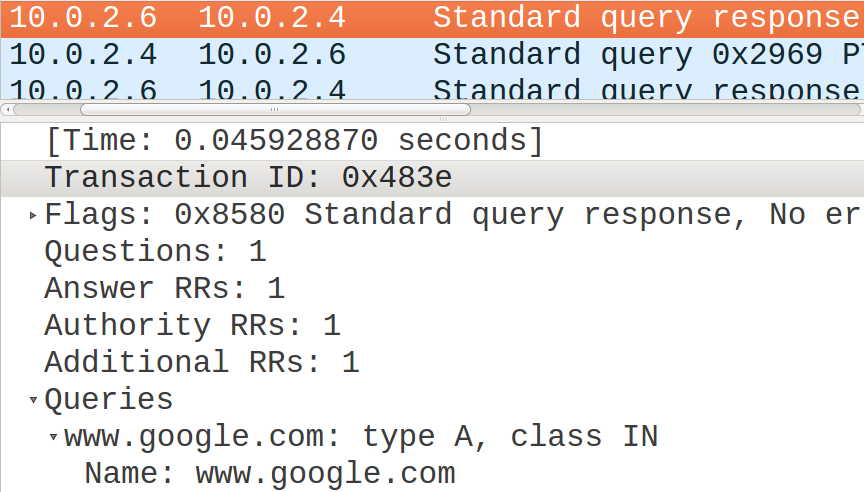
נפתח WIRESHARK במחשב התוקף ונתבונן בPACKETS שמועברים



ניתן לראות שהCLIENT שלח DNS QUERY ל LOCAL DNS 10.0.2.6 עם transaction id = 0x483e וה LOCAL DNS שלח DNS QUERY לשרת של GOOGLE



ניתן לראות שגוגל החזיר DNS RESPONSE ל LOCAL DNS והLOCAL DNS החזיר את התשובה לCLIENT לפי התמונה הבאה:



ניתן לראות שה transaction id = 0x483e גם ב dns query שנשלח מהCLIENT לlocal dns וגם ב dns response שנשלח חזרה מהlocal dns לCLIENT.

* סיכום המשימה

הצלחנו לבצע את המשימה, ניתן לראות שהגדרנו את מחשב 10.0.2.6 בתור local dns, והפעלנו עליו את שרת הbind9 והגדרנו שהשרת יקבל את כל בקשות ה DNS של הCLIENT ויענה לבקשותיו.

הראינו זאת על ידי שליחת ping לגוגל מהclient כך שהPACKETS נשלחו מ10.0.2.4 ל10.0.2.6 ומשם ליעד שהוא גוגל 8.8.8.8 ובחזרה נשלחו מהיעד 8.8.8.8 ל10.0.2.6 ורק לאחר מכן ל10.0.2.4

גילינו כיצד להגדיר באופן סטטי שרת DNS שיעביר את שאילתות הDNS של המחשב, גילינו שקיים שרת מובנה במערכת שהתקנו בשם bind9 אשר מאפשר יצירת כתובות חדשות בצורה קלה ונוחה יותר.

התוצאות התאימו למצופה מאחר והPACKETS של שאילתות הDNS הועברו דרך ה LOCAL DNS שהגדרנו.

לא נתקלנו בבעיות במהלך ביצוע המשימה.

Task 3: Host a Zone in the Local DNS Server

* מבוא:
  + תיאור

במשימה זו נרצה להגדיר ZONE חדש כלומר כתובת חדשה שאנחנו ניצור והנתונים שלה יאוחסנו על השרת DNS הלוקאלי

* + מטרה

להגדיר Zone עם כתובות IP שנבחר וכתובת לכל HOSTNAME שנבחר ולשמור את הנתונים על הlocal dns

* + תוצאה מצופה

כאשר נבצע dig לכתובת שיצרנו נקבל את כל הנתונים שהגדרנו בZONE על הכתובת.

* ביצוע המשימה:
* שלב ראשון: Create zones

נרצה להגדיר כתובת IP לאתר לדוגמא שישמש אותנו בהמשך

כדי לבצע זאת נכנס לקובץ בנתיב /etc/bind/named.conf.local ונרצה להגדיר את הZONE הבא:

zone "example.com" {

type master;

file "/etc/bind/example.com.db";

};

zone "0.168.192.in-addr.arpa" {

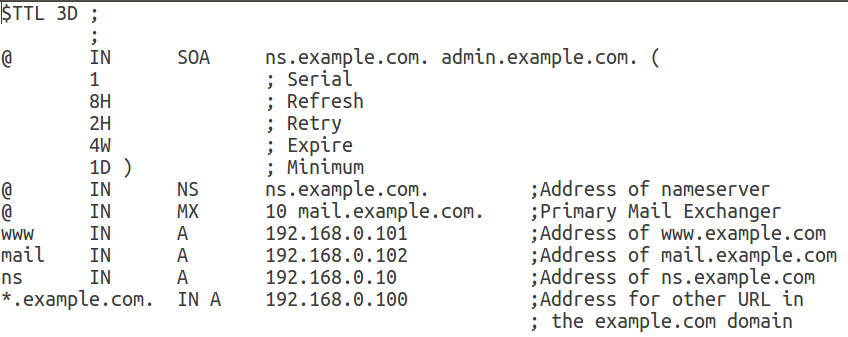
type master;

file "/etc/bind/192.168.0.db";

};

* שלב שני: Setup the forward lookup zone file

ניצור קובץ בשם example.com.db בנתיב /etc/bind/ ובקובץ נרשום את הנתונים הבאים:



הסבר הקוד:

TTL 3D – הכתובת תישאר בזיכרון הCACHE 3 ימים

SOA – הגדרת start of authority כלומר נרצה להגדיר את השרת של האתר וכתובת האימייל של איש הקשר האחראי

REFRESH (8 שעות) מציין באיזו תדירות שרת DNS משני צריך לבדוק עם שרת ה-DNS הראשי אם יש שינויים בקובץ ה ZONE.

RETRY (שעתיים) מציין כמה זמן שרת DNS משני צריך להמתין לפני ניסיון חוזר להתחבר לשרת ה-DNS הראשי לאחר ניסיון כושל.

EXPIRE (4 שבועות) מציין כמה זמן שרת DNS משני צריך להמשיך להגיש נתונים לא עדכניים ללקוחות אם הוא אינו מסוגל ליצור קשר עם שרת ה-DNS הראשי. לאחר 4 שבועות ללא הצלחה ביצירת קשר, הנתונים בשרת ה-DNS המשני נחשבים כלא חוקיים ואין להציג אותם יותר.

10 mail.example.com מגדיר את שרת חילופי המיילים כאשר 10 מציין את עדיפות השרת שיהיה הראשי במידה וקיימים שרתי מיילים נוספים.

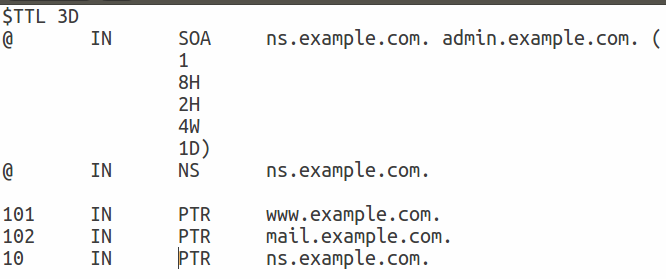
(עדיפות נמוכה יותר אומרת שנרצה להשתמש בשרת יותר כאשר 10 הכי נמוך).

"\*.example.com." ממפה כל תת-דומיין של example.com שאין לו רשומה ספציפית לכתובת ה-IP 192.168.0.100.

* שלב שלישי: Set up the reverse lookup zone file

נרצה להגדיר קובץ לכיוון ההפוך כלומר להגדיר מעבר מIP של השרת אל כתובת [www.example.com](http://www.example.com)

נכנס לנתיב /etc/bind/ ונפתח קובץ חדש בשם 192.168.0.db ונרשום בתוכו את הקוד הבא:



כעת הגדרנו שהIP 192.168.0 ייתן את הכתובת example.com מאחר והserial zone בשניהם זהה ושווה ל1.

PTR = POINTER

* שלב רביעי: Restart the BIND server and test

כעת נבצע ריסטארט לbind9

ונשאל ממחשב הלקוח את שרת הDNS מה הip של האתר example.com



ניתן לראות שקיבלנו משרת הDNS שהוא 10.0.2.6 (השרת הלוקאלי שהגדרנו) תשובה שהIP של האתר הוא 192.168.0.101 כפי שהגדרנו קודם לכן.

* סיכום המשימה

הצלחנו לבצע את המשימה, ניתן לראות שהגדרנו ZONE חדש בשרת 10.0.2.6 בשם [www.example.com](http://www.example.com) עם כתובת IP שבחרנו.

הוכחנו זאת על ידי כך שביקשנו משרת הlocal dns את כתובת הIP של השרת בעזרת הפקודה DIG וקיבלנו את הIP שהגדרנו.

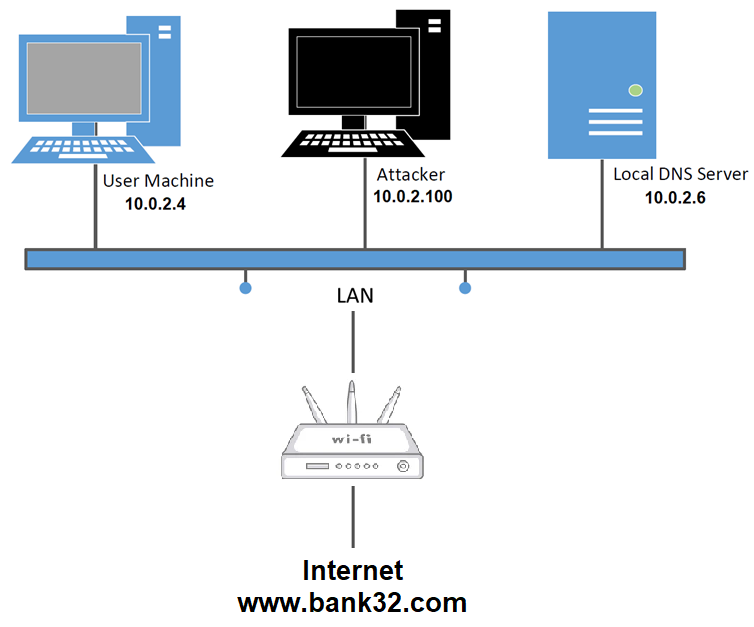
גילינו כיצד להגדיר כתובת שרת חדש משלנו עם IP אותו אנחנו בוחרים.

התוצאות התאימו למצופה מאחר וכתובת האתר נתנה את הIP שהגדרנו לה.

נתקלנו בבעיה כאשר במעבדה היה רשום להגדיר את הZONE בקובץ etc/bind/named.conf. אך בקובץ היה רשום בהערה להגדיר ZONES חדשים בקובץ etc/bind/named.conf.login, מאחר ולא ידענו מה נכון יותר לבצע ניסינו להגדיר בשני הקבצים בנפרד ובהתחלה לא עבד לנו כאשר הגדרנו בקובץ שנאמר במטלה, אך כן עבד לנו כאשר הגדרנו בקובץ השני לפי הערה הרשומה.

**Lab Tasks (Part II): Attacks on DNS**

Task 4: Modifying the Host File

* מבוא:
  + תיאור

במשימה זו נרצה להגדיר כתובת IP סטטית לאתר ברשת החיצונית

* + מטרה

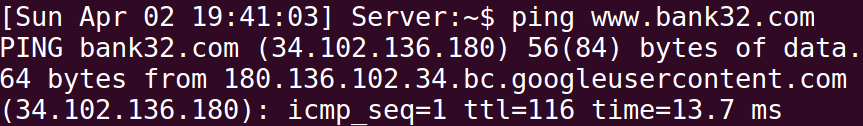
הCLIENT יוכל להתחבר לאתר [www.bank32.com](http://www.bank32.com) ללא שליחת שאילתת DNS מאחר וכתובת הIP שלו מוגדרת באופן סטטי על המחשב.

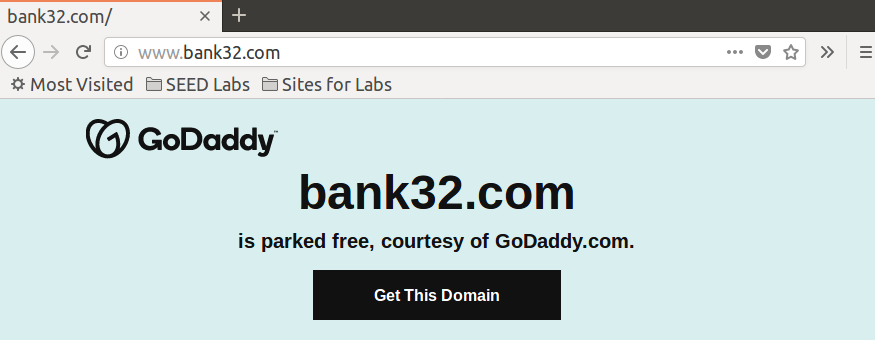
* + תוצאה מצופה

כאשר נרצה להתחבר לאתר לאחר הגדרת כתובת IP שונה, לא נתחבר לאתר המקורי אלא לאתר של הIP החלופי אותו הגדרנו באופן סטטי, בנוסף אם נשלח PING לכתובת אז נראה שהוא שולח לIP שהגדרנו.

* ביצוע המשימה:

תחילה נבדוק מה האייפי שמתקבל כאשר נשלח פינג לכתובת [www.bank32.com](http://www.bank32.com) ונבדוק שהאתר נפתח בדפדפן





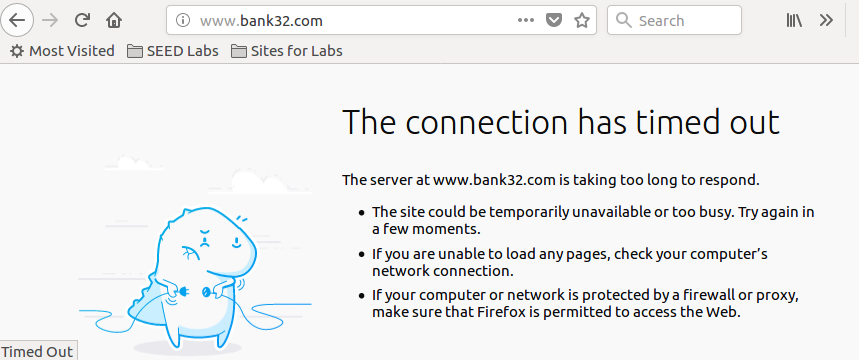
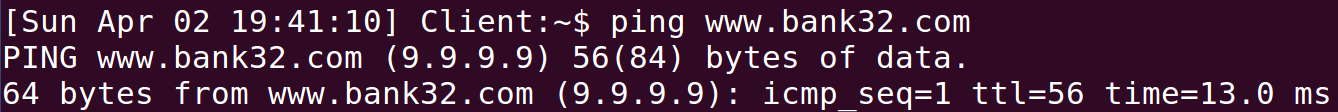
ניתן לראות שהIP של הכתובת הוא 34.102.136.180, והאתר קיים בכתובת הזאת ולכן נפתח בדפדפן

כעת נגדיר בCLIENT כתובת IP סטטית בקובץ הHOSTS בתיקיית ETC

על ידי הוספת השורה הבאה:



כעת כתובת הIP של האתר מוגדרת להיות 9.9.9.9

נשלח פינג לאתר מהCLIENT ונבדוק לאיזה כתובת IP הפינג נשלח ובנוסף ננסה להיכנס לאתר דרך הדפדפן

ניתן לראות שהפינג נשלח לכתובת הסטטית שהגדרנו 9.9.9.9

ושלא ניתן לגשת לאתר מאחר והוא לא בקיים בכתובת הזאת.

* סיכום המשימה

הצלחנו לבצע את המשימה, ניתן לראות שהגדרנו IP סטטי לכתובת [www.bank32.com](http://www.bank32.com) בCLIENT כך הIP יהיה 9.9.9.9

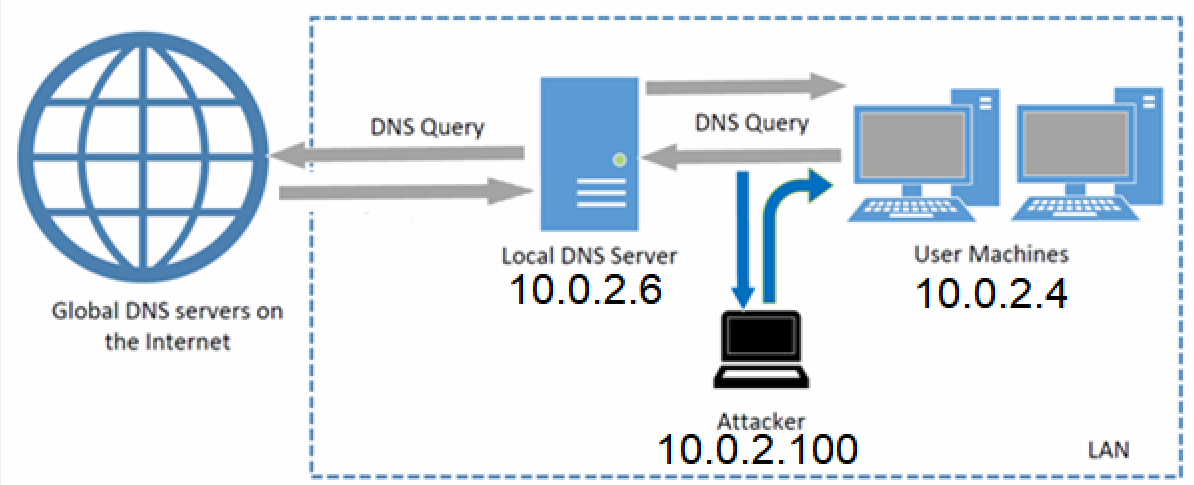
הוכחנו זאת על ידי כך ששלחנו ping לכתובת האתר לפני ביצוע החלפת הIP ואחרי וראינו שכתובת הIP של האתר השתנתה לכתובת אותה בחרנו 9.9.9.9.

גילינו כיצד להגדיר כתובת IP סטטית לאתר שנרצה.

התוצאות התאימו למצופה מאחר וכתובת האתר נתנה את הIP שהגדרנו באופן סטטי וכל שליחת שאילתת DNS או PING לכתובת נתנה את הIP אותו הגדרנו.

לא נתקלנו בבעיות במהלך ביצוע המשימה.

Task 5: Directly Spoofing Response to User

* מבוא:
  + תיאור

במשימה זו נרצה לבצע זיוף מענה לשאילתת בקשת DNS ולענות לפני שרת הDNS המקורי

* + מטרה

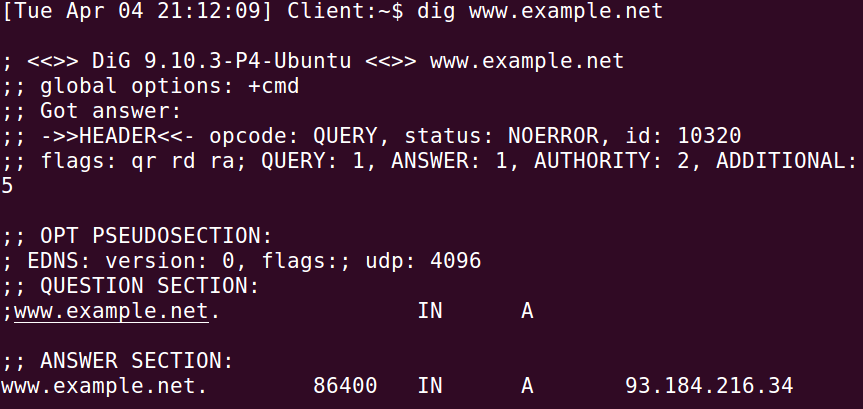
הCLIENT יקבל מענה מהATTACKER לdns query לפני ששרת הDNS המקורי יענה ובכך תתבצע הרעלה לכתובת הIP של האתר אליו הCLIENT רוצה להיכנס והוא יכנס לאתר זדוני.

* + תוצאה מצופה

לאחר ההרעלה נבצע dig ונרצה לוודא שה dns resolver קיבל את הIP המזויף לאתר שאליו הCLIENT ניסה לגשת.

* ביצוע המשימה:

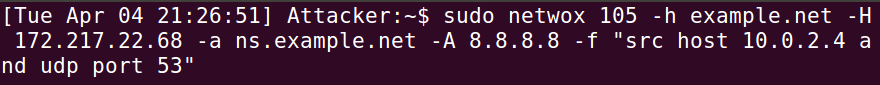
תחילה נבצע dig ממחשב הclient לאתר [www.example.net](http://www.example.net) כדי לראות מה IP האתר



ניתן לראות שהIP הוא: 93.184.216.34

כעת נרצה לנקות את זיכרון הcache של שרת הDNS הלוקאלי שלנו על ידי הפקודה הבאה:



נרצה להרעיל את הCLIENT ולגרום לו להיכנס לIP אחר כאשר הוא נכנס לכתובת [www.example.net](http://www.example.net), נרצה לגרום לו לשלוח בקשת DNS לכתובת של גוגל 172.217.22.68

השתמשנו בכלי netwox 105 אשר מאזין לפאקטות dns query שנשלחו מIP 10.0.2.4 לפורט 53 ומזייף dns response ועונה לפני שרת הdns אליו נשלחה הבקשה עם הנתונים אותם נרצה.

h- – מציין את שם האתר לו נרצה לשנות את הIP

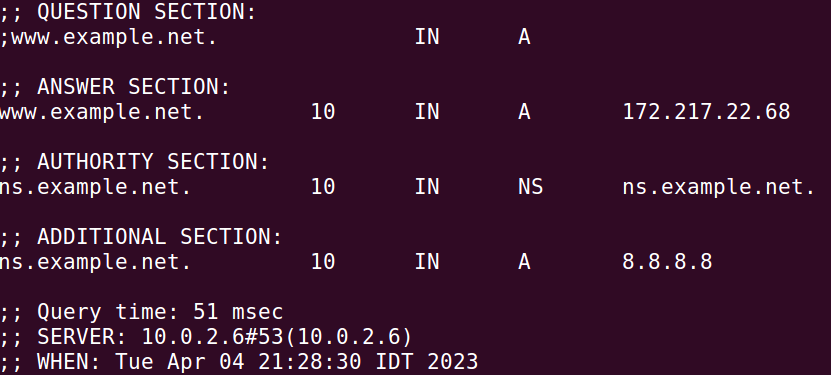
-H – מציין את הIP אליו נרצה לשנות

-a – מציין את שם השרת DNS של האתר אליו נפנה

-A – מציין את הIP של שרת הDNS אליו נפנה

-f – מציין את הפילטר לפיו נרצה להאזין לפאקטות

כעת נשלח שאילתת DNS query מהclient עבור כתובת IP לאתר [www.example.net](http://www.example.net) בעזרת הפקודה dig



ניתן לראות שכתובת הIP של האתר השתנתה מהמקור שהיה לפני כן לIP של גוגל 172.217.22.68 ושרת הauthority השתנה גם כן לשרת של גוגל 8.8.8.8

* נדרש לבצע ניקוי לCACHE של השרת מאחר והשרת מקבל את שאילתת הDNS של הCLIENT ומעביר אותה לשרת DNS חיצוני לבדיקת הכתובת, בו זמנית הATTACKER מחזיר תשובה ראשון עם הIP המזויף אותו רצינו ולכן הCLIENT מציג את התשובה הרצויה, אך לאחר מכן הCLIENT מקבל תשובה נוספת מהשרת DNS המקורי שאומר לו מה הכתובת האמיתית של האתר ולכן מתבצע עדכון בCACHE של השרת ואם נבצע dig נוסף ללא ניקוי הזיכרון נקבל את הנתונים האמיתיים ולא המזויפים.
* סיכום משימה

הצלחנו לבצע את המשימה, שתלנו כתובת IP שונה מהאמיתית בזיכרון הlocal dns על ידי נתינת מענה לשאילתת dns query של השרת הלוקאלי לפני קבלת המענה משרתי DNS חיצוניים

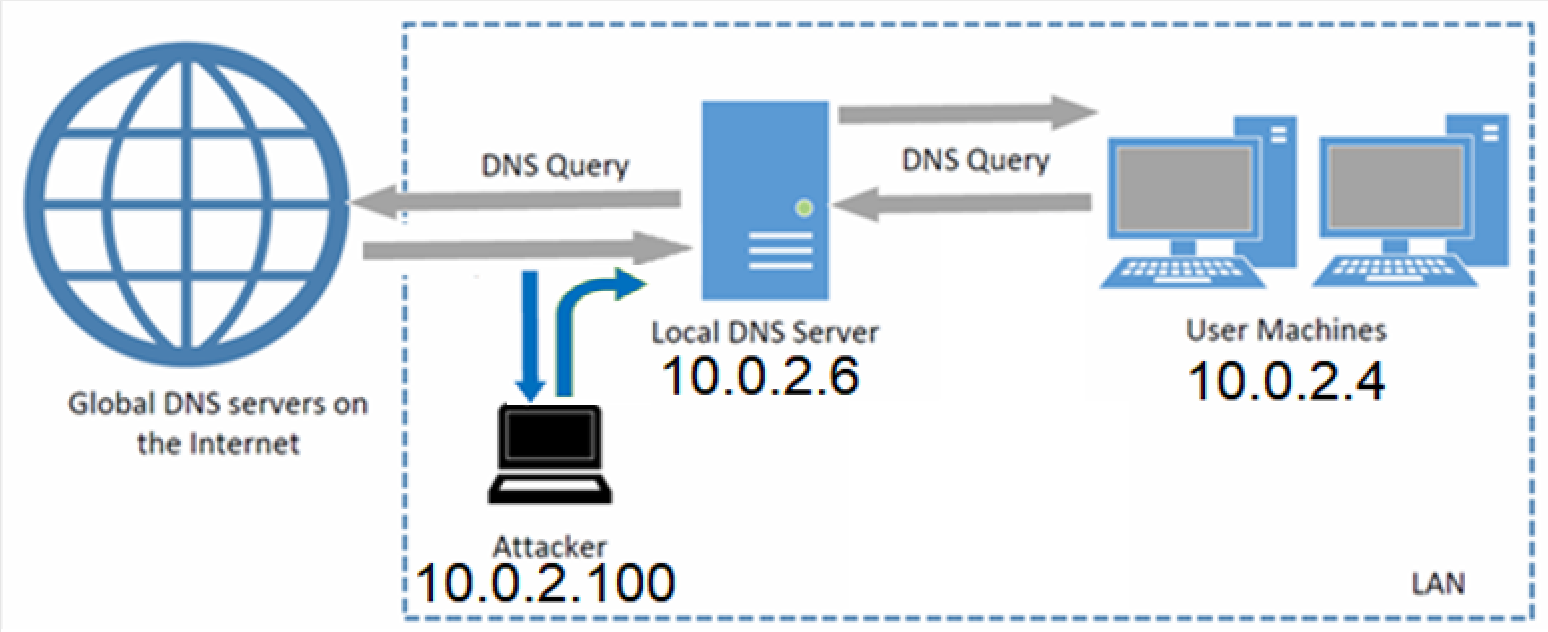
הוכחנו זאת על ידי כך שבעת קבלת מענה של dig קיבלנו לפני ההרעלה את הIP האמיתי ולאחר ההרעלה את הIP המזויף, וגם בעצירת ביצוע המתקפה בעזרת כלי הNETWOX עדיין כתובת הIP המזויפת נשמרה בזיכרון של הlocal dns.

גילינו כיצד להרעיל זיכרון של local dns ולהשתיל בו כתובת IP מזויפת למשך זמן שנבחר ובכך למנוע ממנו לפנות לשרתי DNS חיצוניים, בנוסף גילינו מדוע נדרש להגדיר raw בspoof ip.

התוצאות התאימו למצופה מאחר וכתובת האתר נתנה את הIP שהטמענו בשאילתת הdns response המזויפת ונשמרה בזיכרון למשך הזמן שהקצנו.

לא נתקלנו בבעיות במהלך ביצוע המשימה.

Task 6: DNS Cache Poisoning Attack

* מבוא:
  + תיאור

במשימה זו נרצה לבצע זיוף מענה לשאילתת בקשת DNS שתענה לפני שרת הDNS המקורי ולגרום לשרת הlocal dns לשמור את המענה בזיכרון הcache.

* + מטרה

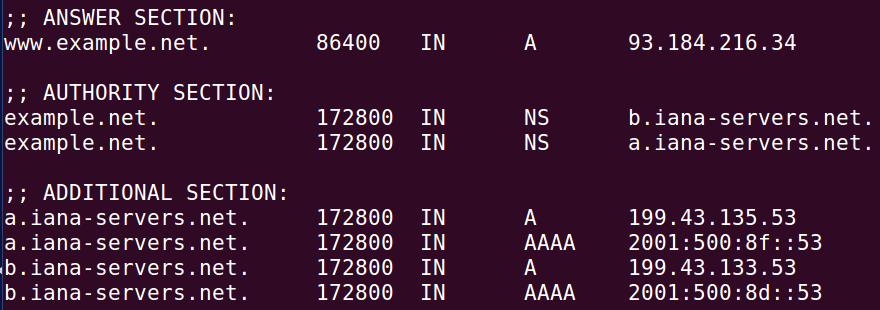
הlocal dns יקבל מענה מהATTACKER לdns query לפני ששרת הDNS המקורי יענה עם כמות זמן לשמירת הIP בזיכרון הCACHE ובכך תתבצע הרעלה לכתובת הIP של האתר אליו נשלחה בקשת dns query, דבר זה יגרום לclient לקבל מענה עם הip המזויף כל עוד הכתובת תישאר בזיכרון הcache של הlocal dns.

* + תוצאה מצופה

לאחר ההרעלה נבצע dig ונרצה לוודא שהכתובת הIP המזויפת מתקבלת במהלך הזמן שהוקצה לו להישאר בזיכרון הlocal dns ללא צורך בביצוע הרעלה נוספת.

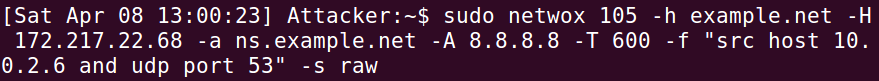
* ביצוע המשימה:

תחילה נבצע ניקוי לזיכרון הcache בשרת הlocal dns 10.0.2.6 על ידי הפקודה sudo rndc flush

נשלח dig מהclient כדי לראות שכעת הוא מקבל מענה עם הIP המקורי של השרת [www.example.com](http://www.example.com)

ניתן לראות שהIP כעת הוא 93.184.216.34

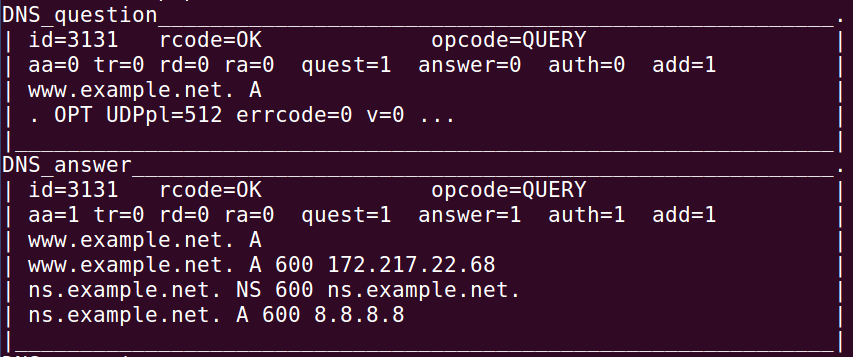
ננקה שוב את זיכרון הCACHE ב local dns ונבצע הרעלה לשרת

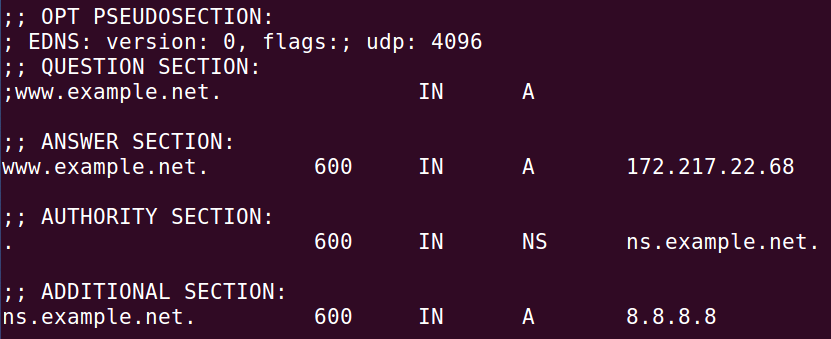
על ידי הרצת הפקודה הבאה מהATTACKER

הוספנו הגדרה לכמה זמן נרצה להשאיר את הdns response בזיכרון הcache של הסרבר על ידי הפרמטר -T (TTL) שאותו בחרנו ל600 שניות

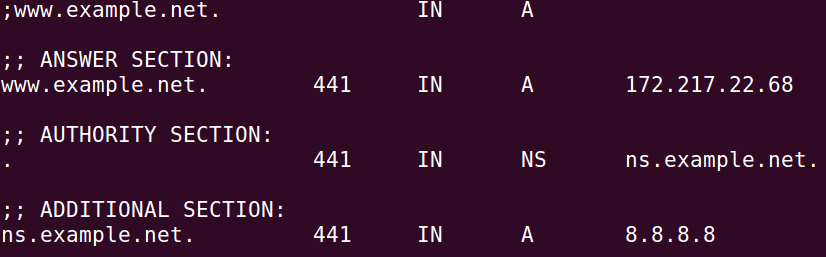
בנוסף נדרש להגדיר בspoof ip (-s) את המונח raw כדי למנוע מהכלי netwox 105 לחכות למענה של הכתובת MAC כדי לזייף אותה מאחר ואנחנו לא יושבים על אותה רשת עם שרת הDNS של האתר ולכן הכתובת MAC לא רלוונטית ואם הכלי יחכה לתשובת ARP היא לא תגיע, ובינתיים ההרעלה תתעכב ולא תתבצע ושרת הDNS המקורי יספיק לענות לפנינו.

כעת נשלח dig מהclient לאתר [www.example.net](http://www.example.net)

**תמונה מהATTACKER לזיוף מענה מהשרת הלוקאלי**

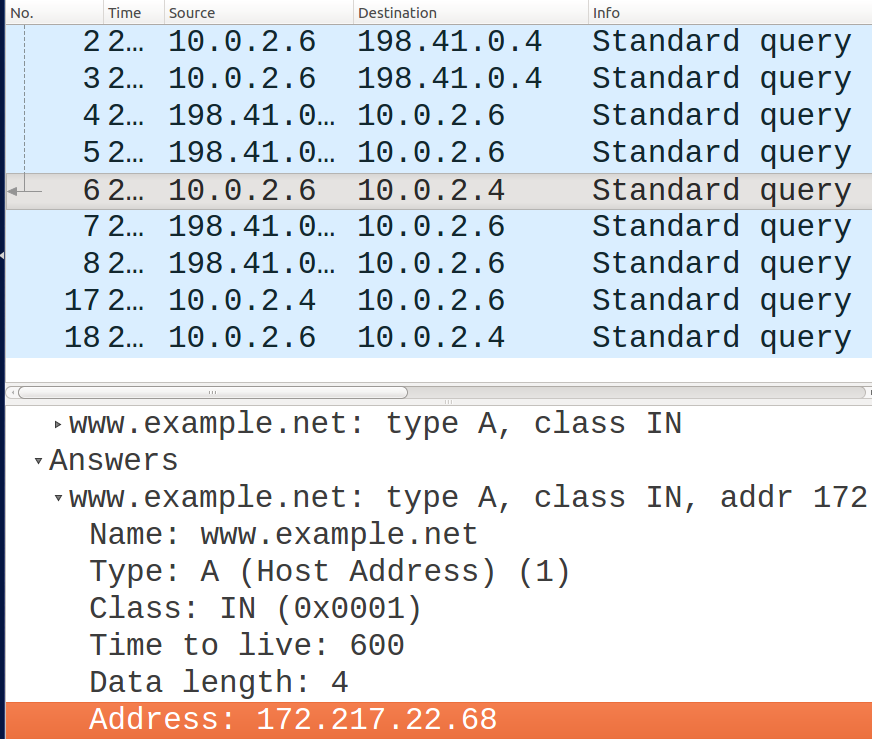
**תמונה מהCLIENT למענה לdig**

ניתן לראות שהזיוף הצליח וכעת כתובת האתר השתנתה ונשארת בזיכרון למשך 600 שניות

כעת ביטלנו את כלי הnetwox ושלחנו בקשת dig נוספת מהclient כדי לראות שאכן זכרון הcache שומר את הכתובת המזויפת וקיבלנו מענה דומה עם עדכון לTTL שנשאר

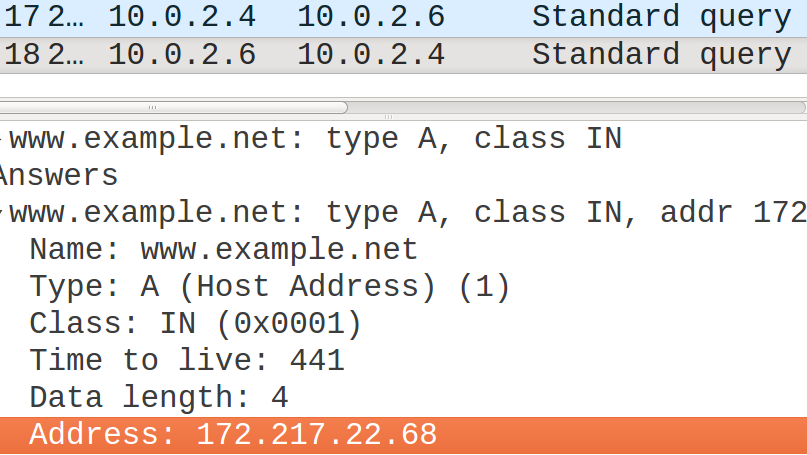
ניתן לראות שהזמן הנותר בזיכרון הוא 441 שניות

בדקנו במהלך ביצוע הזיוף ולאחר ביצוע הזיוף את הPACKETS שנשלחו ברשת בעזרת WIRESHARK וראינו את התעבורה הבאה:



ניתן לראות בpacket המסומן את התשובה שקיבל הclient מהlocal dns בזמן הזיוף, התשובות שהlocal dns קיבל מהdns ברשת לא נשמרו בזיכרון מאחר והוא קיבל מענה נוסף לפני כן בעזרת הכלי netwox.

כעת אספנו את הpackets שעברו לאחר ביטול כלי הnetwox



ניתן לראות שהפעם שרת הlocal dns לא פנה לשרתי dns חיצוניים מאחר והתשובה כבר הייתה אצלו בזיכרון הcache והוא יכל לשלוף אותה ללא צורך בבדיקה מי הIP של הכתובת

ולפי השורה המסומנת הlocal dns שלח את הכתובת המזויפת ששתלנו בזיכרון.

* סיכום המשימה

הצלחנו לבצע את המשימה, שתלנו כתובת IP שונה מהאמיתית בזיכרון הlocal dns על ידי נתינת מענה לשאילתת dns query של השרת הלוקאלי לפני קבלת המענה משרתי DNS חיצוניים

הוכחנו זאת על ידי כך שבעת קבלת מענה של dig קיבלנו לפני ההרעלה את הIP האמיתי ולאחר ההרעלה את הIP המזויף, וגם בעצירת ביצוע המתקפה בעזרת כלי הNETWOX עדיין כתובת הIP המזויפת נשמרה בזיכרון של הlocal dns.

גילינו כיצד להרעיל זיכרון של local dns ולהשתיל בו כתובת IP מזויפת למשך זמן שנבחר ובכך למנוע ממנו לפנות לשרתי DNS חיצוניים, בנוסף גילינו מדוע נדרש להגדיר raw בspoof ip.

התוצאות התאימו למצופה מאחר וכתובת האתר נתנה את הIP שהטמענו בשאילתת הdns response המזויפת ונשמרה בזיכרון למשך הזמן שהקצנו.

לא נתקלנו בבעיות במהלך ביצוע המשימה.

Task 7: DNS Cache Poisoning: Targeting the Authority Section

* מבוא:
  + תיאור

במשימה זו נרצה לבצע זיוף מענה לשאילתת בקשת DNS ולגרום לשרת הlocal dns לשמור את המענה בזיכרון הcache עם שרת Authority שונה מהמקור.

* + מטרה

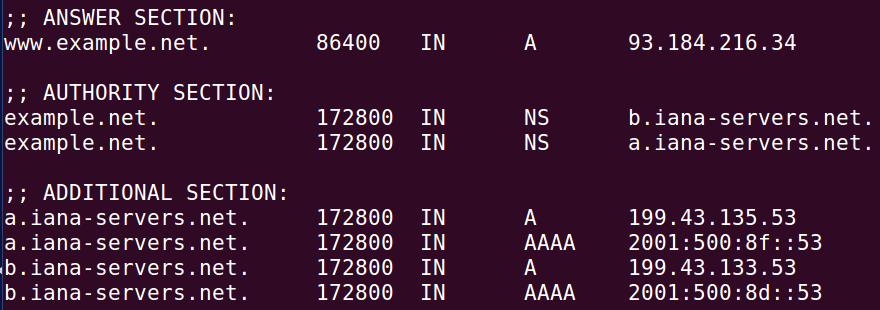
להרעיל את שרת הlocal dns כך שעבור כל בקשות עם סיומתdomain example.net יועברו לauthority שאנחנו נבחר במקום המקור.

* + תוצאה מצופה

לאחר ההרעלה נבצע dig ונוודא שכתובת הauthority של הכתובת תהיה הכתובת המזויפת שבחרנו.

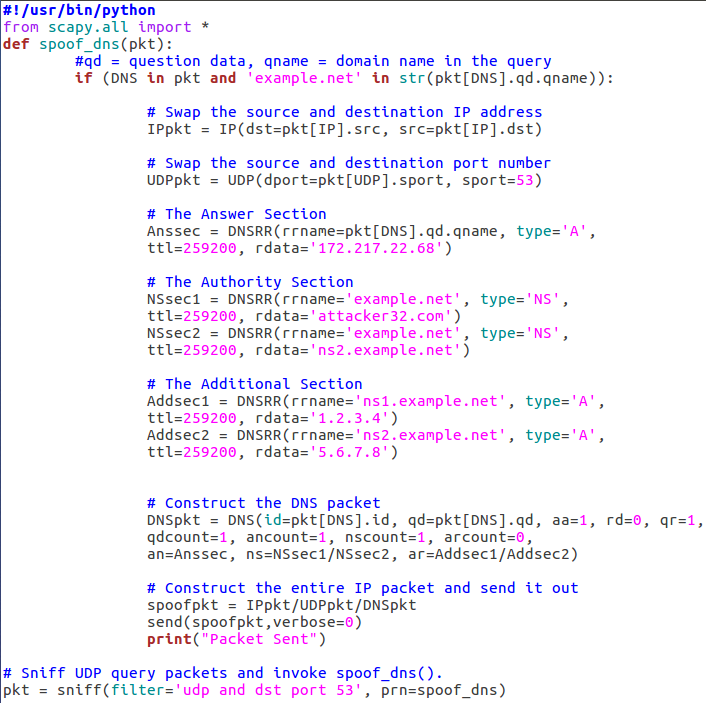
* ביצוע המשימה:

תחילה נבצע ניקוי לזיכרון הcache בשרת הlocal dns 10.0.2.6 על ידי הפקודה sudo rndc flush

נשלח dig מהclient כדי לראות שכעת הוא מקבל מענה עם הIP המקורי של השרת [www.example.com](http://www.example.com)

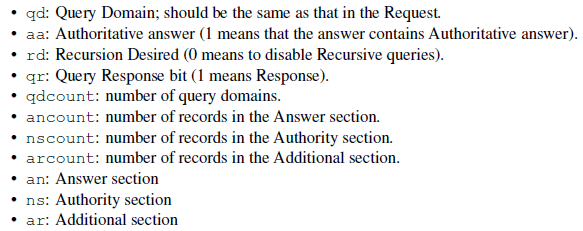
ניתן לראות שהIP כעת הוא 93.184.216.34

ננקה שוב את זיכרון הCACHE ב local dns ונבצע הרעלה לשרת

נכתוב את הקוד הבא בscappy לצורך האזנה לפאקטות DNS ושליחת מענה מזויף מהattacker:

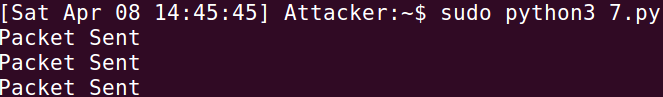
בקוד ביצענו האזנה לפאקטות הנשלחות בUDP עם פורט 53 כלומר DNS, כאשר מתקבלת פאקט ברשת פונקציית הזיוף שכתבנו תענה לdns query שנשלח במידה והוא מכיל שאלה על הכתובת [example.net](http://www.example.net)

פירוט קצר על יצירת dns packet:



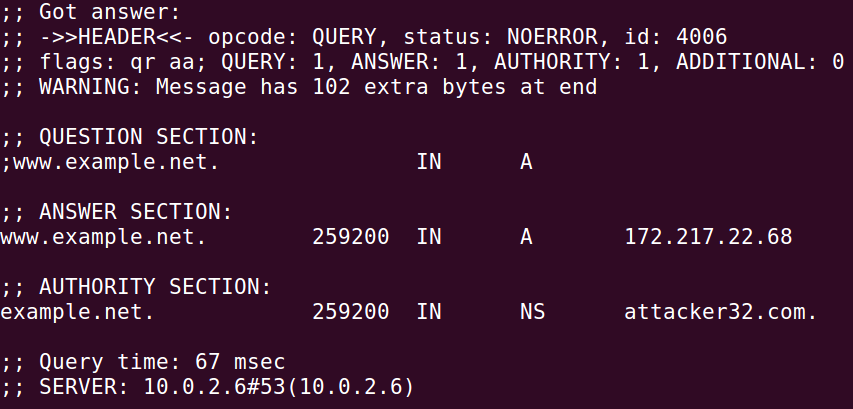
מאחר והתבקשנו להוסיף רק שורת authority בזיכרון הCACHE נרצה לשלוח רק שורה אחת ולכן ancount = 1(answer) , nscount = 1 (authority) arcount = 0 (additional)

נריץ את הקוד על הATTACKET ונבצע dig מהclient



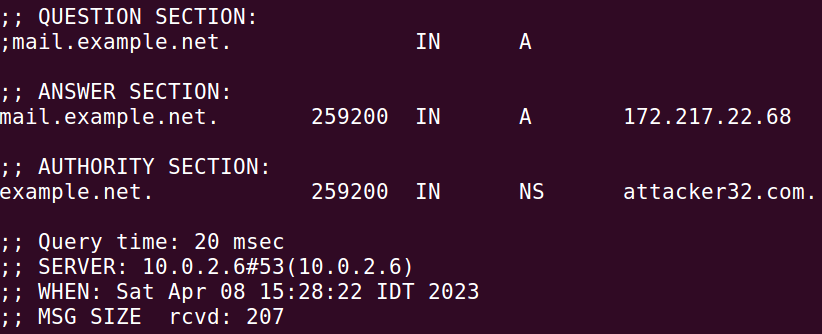
ניתן לראות שהפאקט נשלח מהתוקף

תמונה מהCLIENT לאחר ביצוע dig



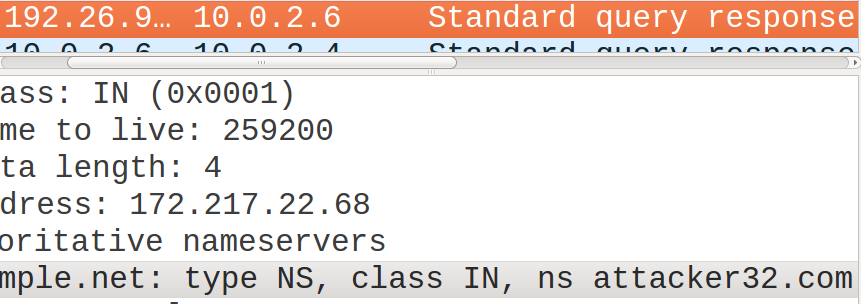
ניתן לראות שהclient קיבל את הנתונים המזויפים אותם רצינו והוגדר לו ששרת הauthority של כתובת [www.example.net](http://www.example.net) הוא attacker32.com

תמונה מהclient לאחר ביצוע dig גם לכתובת mail.example.net



ניתן לראות שהclient קיבל את הנתונים המזויפים אותם רצינו והוגדר לו ששרת הauthority של כתובת [mail.example.net](http://www.example.net) הוא attacker32.com

תיעדנו את הPACKETS שעוברו על הרשת במהלך ביצוע ההתקפה מהATTACKER



ניתן לראות את אחת הפאקטות שזייפנו ובה את כתובת הauthority שרצינו attacker32.com

* סיכום המשימה

הצלחנו לבצע את המשימה, שתלנו כתובת AUTHORITY שונה בזיכרון הcache של שרת ה- local dns ובכך כל בקשה על כתובות הקשורות לexample.net תשלח לשרת dns אותו הגדרנו במקום למקור.

הוכחנו זאת על ידי כך שבעת ביצוע dig מהclient הראנו ששרת הauthority המוגדר השתנה מהמקור לשרת שרצינו attacker32.com.

גילינו כיצד לבצע הגדרה של שרת DNS אחר לכתובת מסוימת ועל ידי כך להרעיל את כל הבקשות הנשלחות על אותה הכתובת ולא רק עבור שרת ספציפי.  
התוצאות התאימו למצופה מאחר וכתובת הAUTHORITY השתנתה לכתובת שרצינו והשאילתות נשלחות אליה.

לא נתקלנו בבעיות במהלך ביצוע המשימה.

Task 8: Targeting Another Domain

* מבוא:
  + תיאור

במשימה זו נרצה לבצע זיוף מענה לשאילתת בקשת DNS ולגרום לשרת הlocal dns לשמור את המענה בזיכרון הcache עם שרת Authority שונה מהמקור לכתובת נוספת שלא התבקשנו לתת מענה עליה.

* + מטרה

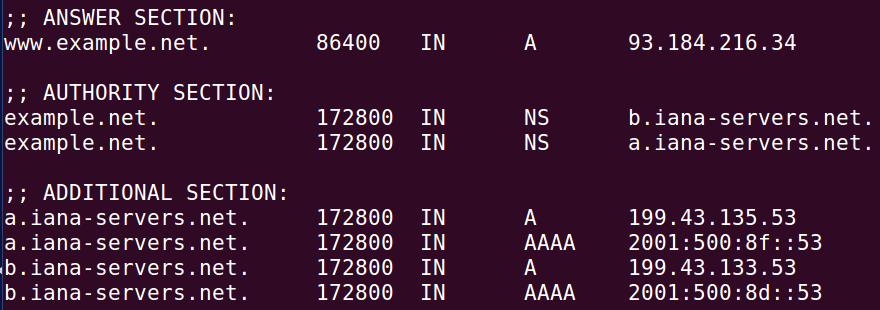
להרעיל את שרת הlocal dns כך שעבור כל בקשות עם סיומתdomain example.net יועברו לauthority שאנחנו נבחר במקום המקור וביחד עם שינוי הauthority לכתובת המבוקשת נבצע שינוי לauthority של כתובת נוספת אותה נרצה כגון google.com

* + תוצאה מצופה

לאחר ההרעלה נבצע dig ונוודא שכתובת הauthority של הכתובת תהיה הכתובת המזויפת שבחרנו ונקבל שורה נוספת על הכתובת של google.com עם authority מזויף, בנוסף נרצה שהכתובת תשמר בCAHCE של ה.local dns

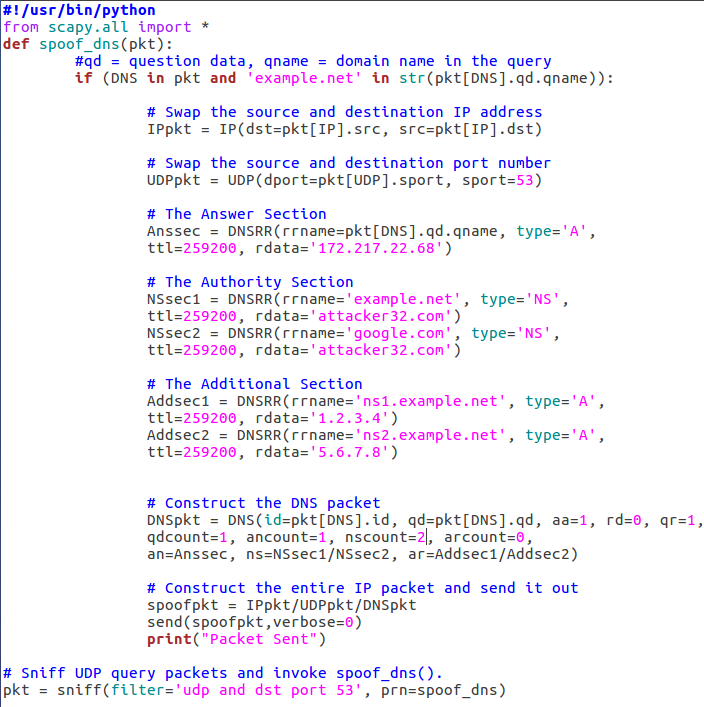
* ביצוע המשימה:

תחילה נבצע ניקוי לזיכרון הcache בשרת הlocal dns 10.0.2.6 על ידי הפקודה sudo rndc flush

נשלח dig מהclient כדי לראות שכעת הוא מקבל מענה עם הIP המקורי של השרת [www.example.com](http://www.example.com)

ניתן לראות שהIP כעת הוא 93.184.216.34

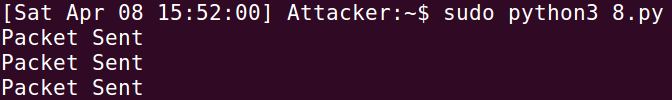
ננקה שוב את זיכרון הCACHE ב local dns ונבצע הרעלה לשרת

נכתוב את הקוד הבא בscappy לצורך האזנה לפאקטות DNS ושליחת מענה מזויף מהattacker: 

בקוד ביצענו האזנה לפאקטות הנשלחות בUDP עם פורט 53 כלומר DNS, כאשר מתקבלת פאקט ברשת פונקציית הזיוף שכתבנו תענה לdns query שנשלח במידה והוא מכיל שאלה על הכתובת [example.net](http://www.example.net)

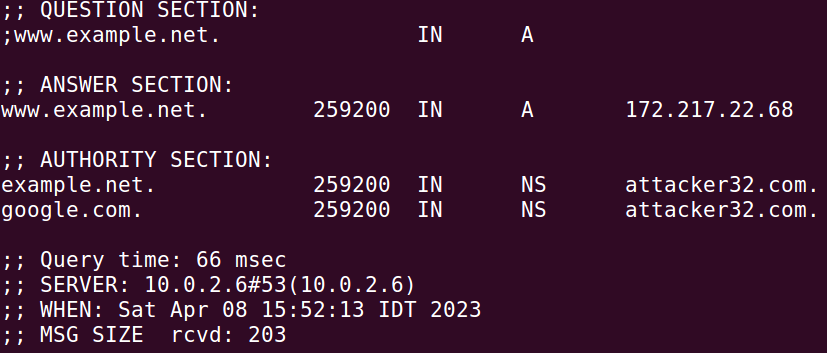
מאחר והתבקשנו להוסיף רק 2 שורות authority בזיכרון הCACHE נרצה לשלוח רק 2 שורות ולכן ancount = 1(answer) , nscount = (authority) arcount = 0 (additional)

נריץ את הקוד על הATTACKET ונבצע dig מהclient



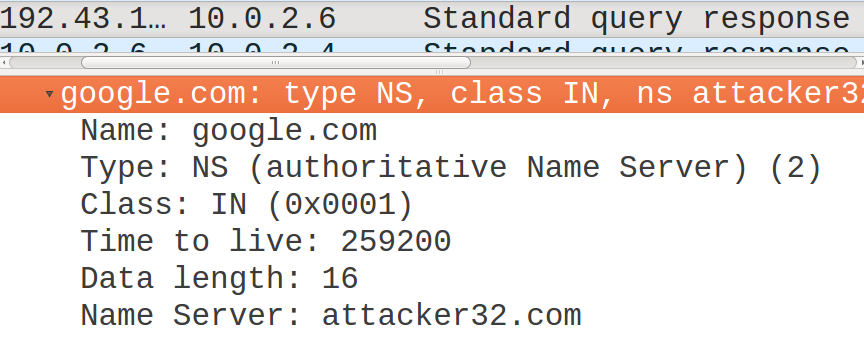
ניתן לראות שהפאקט נשלח מהתוקף

תמונה מהCLIENT לאחר ביצוע dig



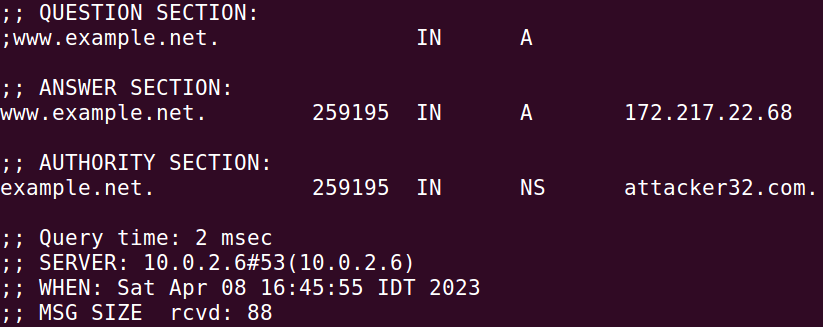
ניתן לראות שהclient קיבל את הנתונים המזויפים אותם רצינו והוגדר לו ששרת הauthority של כתובת [www.example.net](http://www.example.net) הוא attacker32.com וגם של הכתובת google.com

תיעדנו את הPACKETS שעוברו על הרשת במהלך ביצוע ההתקפה מהATTACKER



ניתן לראות את אחת הפאקטות שזייפנו ובה את כתובת הauthority שרצינו attacker32.com עבור google.com

לאחר סיום המתקפה בדקנו מה נשמר בזיכרון על ידי שליחת dig פעם נוספת מהclient



ניתן לראות שהמידע הנוסף אותו שלחנו על הauthority של google.com נמחק ולא נשמר בזיכרון הcache.

לאחר חקירה מדוע המידע הנוסף לא נשמר בזיכרון הcache של הlocal dns עלתה הסבירה הבאה:

מאחר ואנו מבצעים זיוף לפאקט ולא תוקפים את זיכרון הcache אז לא מתבצעת שמירה בזיכרון לחלקים שלא נדרשו בשאילתה.

* סיכום המשימה

הצלחנו לבצע את המשימה באופן חלקי, שתלנו כתובת AUTHORITY שונה בתשובה לשרת ה- local dns עבור הכתובת google.com כאשר נשלחה בקשה לכתובת אחרת שהיא example.net.

אך המידע לא נשמר בזיכרון הCACHE והצגנו סבירה לסיבה.

הוכחנו את החלק שהצליח על ידי כך שבעת ביצוע dig מהclient הראנו ששרת הauthority המוגדר השתנה מהמקור לשרת שרצינו attacker32.com עבור הכתובת google.com.

גילינו כיצד לבצע הרעלה לכתובת נוספת כאשר אנו מזייפים מענה לשאילתת dns query על כתובת אחרת.  
התוצאות התאימו למצופה באופן חלקי מאחר וכתובת הAUTHORITY השתנתה לכתובת שרצינו עבור google.com כאשר נשלחה שאילתה עבור כתובת example.net.

נתקלנו בבעיה מדוע המידע לא נשמר בזיכרון הCACHE של הlocal dns.

Task 9: Targeting the Additional Section

* מבוא:
  + תיאור

במשימה זו נרצה לבצע זיוף מענה לשאילתת בקשת DNS ולגרום לשרת הlocal dns לשמור את המענה בזיכרון הcache עם additional data שלא התבקשנו לתת.

* + מטרה

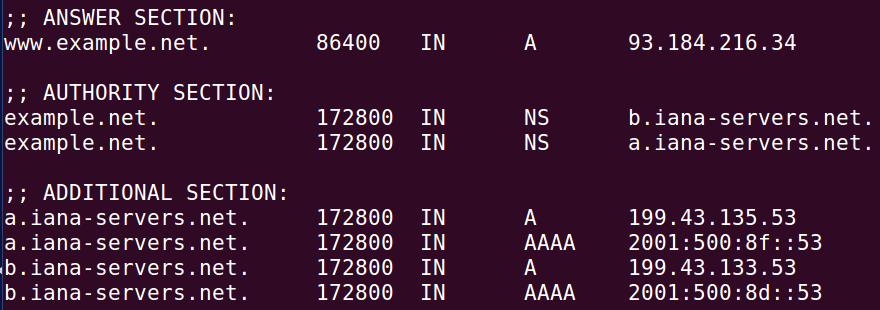
להרעיל את שרת הlocal dns עם מידע נוסף שאותו הוא לא ביקש (למשל IP שונה לאתרים attacker32 ו facebook)

* + תוצאה מצופה

לאחר ההרעלה נבצע dig ונוודא שהמידע הנוסף מופיע בתשובה של הבקשה, ונרצה לראות שהמידע הנוסף נשמר בזיכרון הCACHE של הlocal dns.

* ביצוע המשימה:

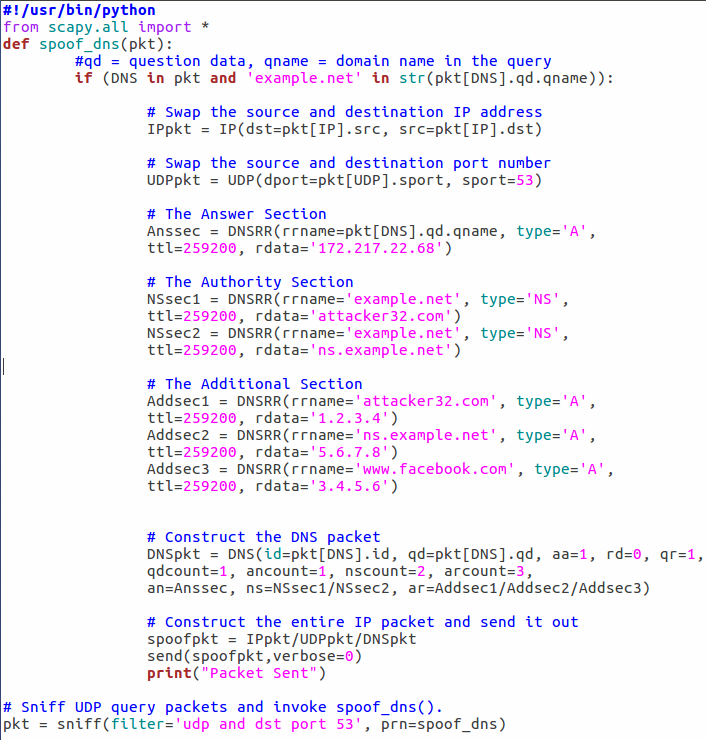
תחילה נבצע ניקוי לזיכרון הcache בשרת הlocal dns 10.0.2.6 על ידי הפקודה sudo rndc flush

נשלח dig מהclient כדי לראות שכעת הוא מקבל מענה עם הIP המקורי של השרת [www.example.com](http://www.example.com)

ניתן לראות שהIP כעת הוא 93.184.216.34

ננקה שוב את זיכרון הCACHE ב local dns ונבצע הרעלה לשרת

נכתוב את הקוד הבא בscappy לצורך האזנה לפאקטות DNS ושליחת מענה מזויף מהattacker:



בקוד ביצענו האזנה לפאקטות הנשלחות בUDP עם פורט 53 כלומר DNS, כאשר מתקבלת פאקט ברשת פונקציית הזיוף שכתבנו תענה לdns query שנשלח במידה והוא מכיל שאלה על הכתובת [example.net](http://www.example.net)

מאחר והתבקשנו להוסיף 2 שורות authority בזיכרון הCACHE ו3 שורות additional נרצה לשלוח את הקוד עם הפרמטרים:

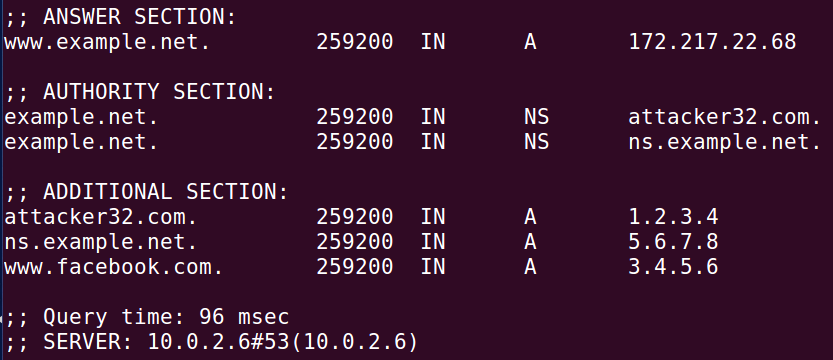
ancount = 1(answer), nscount = 2(authority) arcount = 3 (additional)

נריץ את הקוד על הATTACKET ונבצע dig מהclient



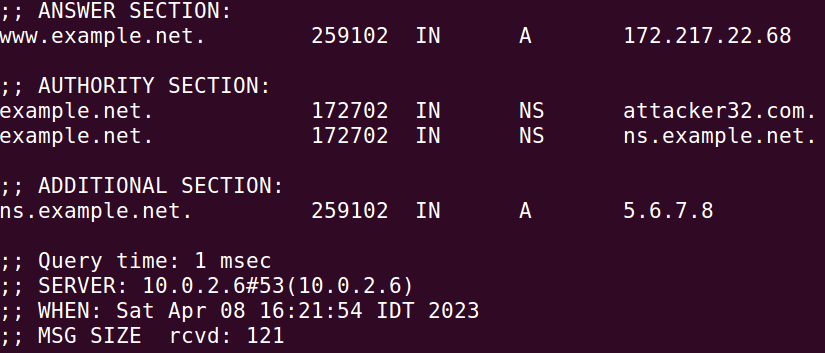
ניתן לראות שהפאקט נשלח מהתוקף

תמונה מהCLIENT לאחר ביצוע dig



ניתן לראות שהclient קיבל את הנתונים המזויפים אותם רצינו

לאחר סיום המתקפה בדקנו מה נשמר בזיכרון על ידי שליחת dig פעם נוספת מהclient



ניתן לראות שהמידע הנוסף אותו שלחנו על attacker32 ו- facebook נמחק ולא נשמר בזיכרון הcache.

לאחר חקירה מדוע המידע הנוסף לא נשמר בזיכרון הcache של הlocal dns עלו שתי סבירות:

1. מאחר ומדובר במידע נוסף שיכול לעזור לclient והוא לא ביקש אותו, זיכרון הCACHE לא שומר אותו כדי למנוע מצב שאולי בוצע זיוף לכתובות הip המוצעות, או שאולי הכתובות שנשלחו לא מעודכנות.
2. מאחר ואנו מבצעים זיוף לפאקט ולא תוקפים את זיכרון הcache אז לא מתבצעת שמירה בזיכרון לחלקים שלא נדרשו בשאילתה.

* סיכום המשימה

הצלחנו לבצע את המשימה באופן חלקי, שתלנו מידע נוסף בתשובה לבקשת DNS QUERY כאשר נשלחה בקשה לכתובת שהיא example.net.

אך המידע לא נשמר בזיכרון הCACHE והצגנו 2 סבירות לסיבה.

הוכחנו את החלק שהצליח על ידי כך שבעת ביצוע dig מהclient הראנו שהמידע הנוסף אותו שלחנו מופיע בתשובה המתקבלת.

גילינו כיצד לבצע ניתן לשלוח מידע נוסף שקרי לבקשות dns query שלא קשור למה שנשאל ובכך לבצע הרעלה רחבה יותר.  
התוצאות התאימו למצופה באופן חלקי מאחר והמידע הנוסף התקבל במענה לשאילתת dns query אך המידע הנוסף לא נשמר בזיכרון הcache של הlocal dns.

נתקלנו בבעיה מדוע המידע לא נשמר בזיכרון הCACHE של הlocal dns.

**סיכום כללי למעבדה**

המשימות במעבדה זו מלמדות כיצד ניתן לתקוף שרת LOCAL DNS.

תחילה קינפגנו שרת DNS לוקאלי, לאחר מכן ביצענו את השלבים הבאים:

* כיצד DNS עובד
* הגדרת שרת הDNS
* איך לבצע התקפת dns cache poisoning
* זיוף תשובות DNS
* האזנה וזיוף לפאקטות של dns query עבור domains מסוימים
* שימוש בscapy ובnetwox 105

לטובת המעבדה נדרשנו ליצר סביבה עם 3 מחשבים: לקוח, שרת DNS לוקאלי, ותוקף.

הגדרנו למשימה הראשונה את שרת הDNS לCLIENT כך שכל הבקשות יעברו דרכו.

למדנו להשתמש בפקודה DIG כדי לחקור את התשובה שקיבלנו.

גילינו כיצד להגדיר באופן סטטי שרת DNS שיעביר שאילתות.

לאחר מכן, הפעלנו הגדרות שרת מובנה bind9 והגדרנו שהשרת יקבל את כל הבקשות של הclient ויענה להן.

וידאנו שהפקודות אכן עוברות דרך השרת שהגדרנו ושהכל עובד כמתוכנן.

הגדרנו ZONE חדש עבור example.com עם כתובת IP שבחרנו.

הצלחנו לשנות הגדרה בזיכרון הcache לIP של אתר כך שהוא יפנה לאתר זדוני אחר.

ביצענו מענה מהיר לשאילתות dns query לפני ששרת הdns המקורי עונה ובכך הרעלנו כתובות והפנו לאתרים אחרים מהנדרש, וגם ביצענו הרעלות לhostnames שונים כגון mail, web וכו'.

שינינו authority והוספנו מידע נוסף לבקשות dns כך שהמידע שרשמנו לא נדרש בכלל ויכולנו להכניס בבקשה לכתובת מסוימת הרעלות לכתובות נוספות.

לא הצלחנו לשמור את המידע הנוסף והauthority ששלחנו לכתובות נוספות בזיכרון הcache של הlocal dns והעלנו השערות לכך שהן:

1. ביצענו זיוף פאקטות ולא התקפה לcache ולכן המידע לא נשמר
2. מאחר ומדובר במידע נוסף שיכול לעזור לclient והוא לא ביקש אותו, זיכרון הCACHE לא שומר אותו כדי למנוע מצב שאולי בוצע זיוף לכתובות הip המוצעות, או שאולי הכתובות שנשלחו לא מעודכנות.

לסיכום,

ביצענו את כל המשימות בהצלחה, למדנו רבות על כיצד לקנפג שרת DNS לוקאלי וכיצד להרעיל את שרת הDNS הלוקאלי או את הקורבן.

**משהו חדשני:**

מצאנו כלי בשם Ettercap, זהו כלי חינמי בקוד פתוח המשמש למתקפות man in the middle.

בכלי זה יש אפשרות ליצור אתר phishing ולבצע dns poisoning לקורבן על ידי הפנייתו לאתר הזדוני שיצרנו.

מצורפת הדגמה שבוצעה על הכלי על ידי ארגון OWASP בקובץ PDF המצורף מטה:

