Local DNS Attack

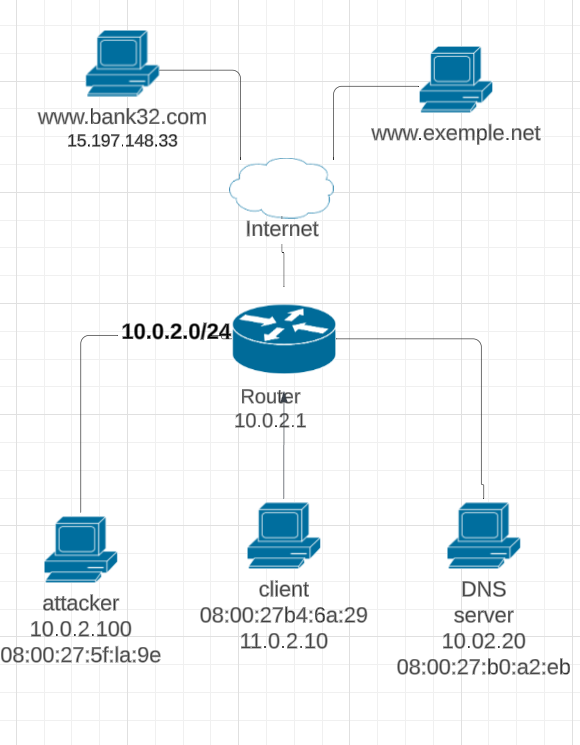
מבוא:

DNS הוא ספר הטלפונים של האינטרנט; זה מתרגם hostnames לכתובות IP (ולהיפך). התרגום הזה הוא באמצעות רזולוציית DNS, מה שקורה מאחורי הקלעים. התקפות DNS מבצעות מניפולציות על תהליך הפתרון הזה בדרכים שונות, מתוך כוונה להפנות משתמשים לא נכון ליעדים חלופיים,

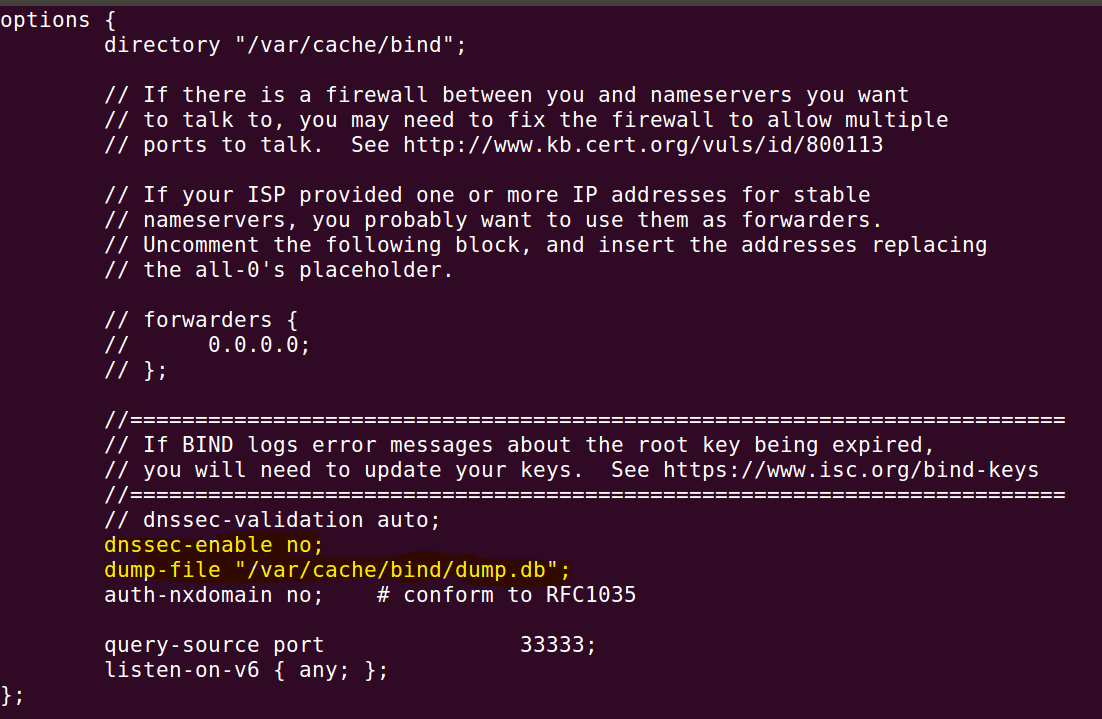
שלעתים קרובות הם זדוניים. מטרת המעבדה הזו היא להבין כיצד פועלות התקפות כאלה.

תחילה נגדיר שרת DNS, ולאחר מכן ננסו התקפות DNS שונות על היעד שהוא גם כן

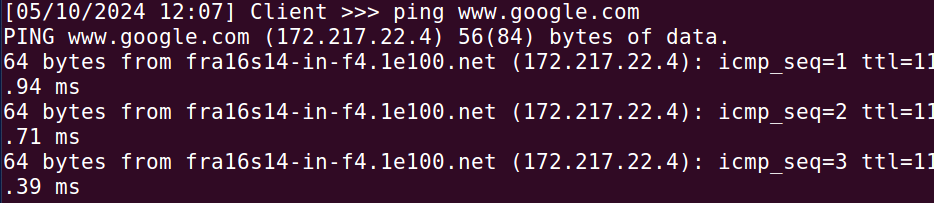
בתוך סביבת המעבדה.

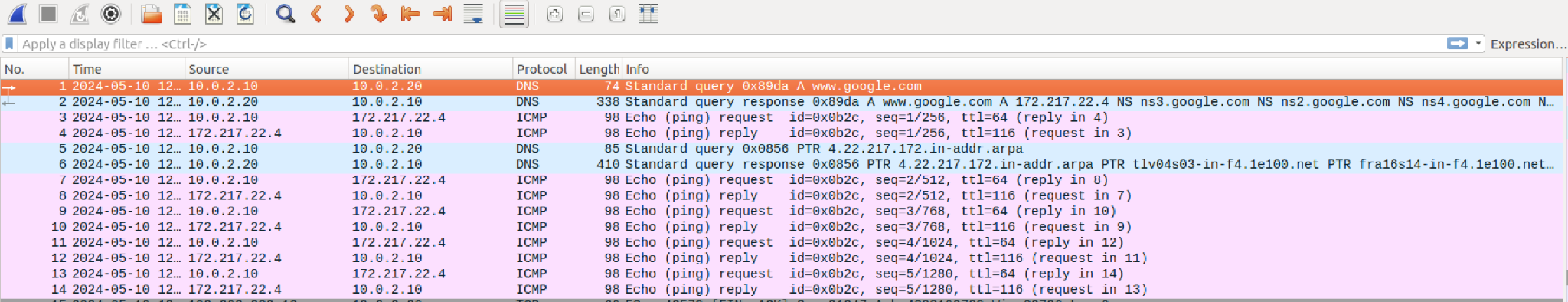


Config dns server

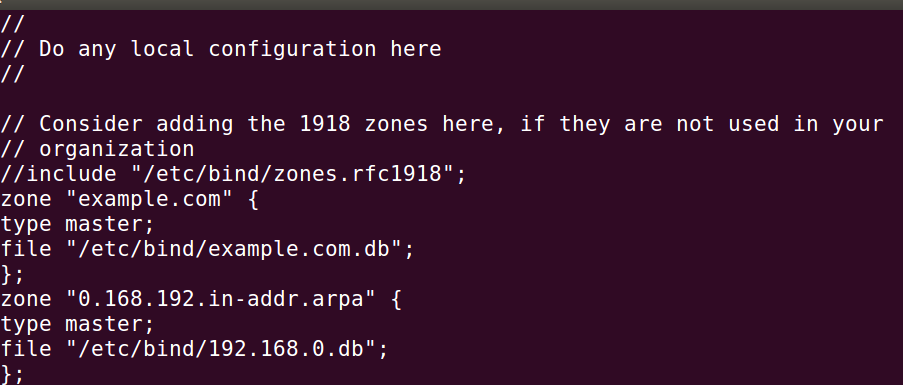




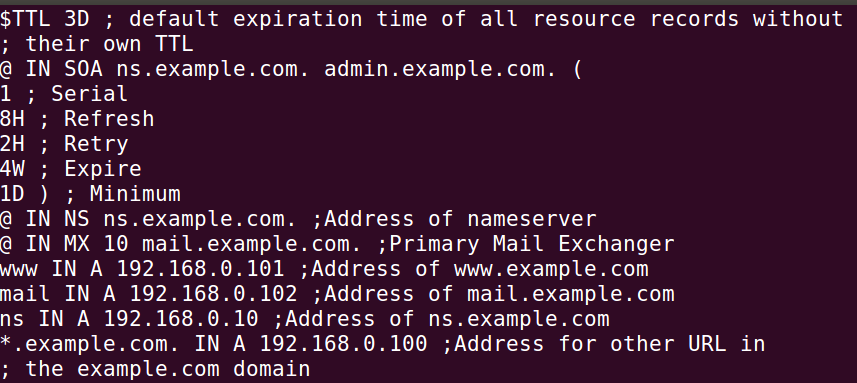




Adding zones at etc/bind/named.conf.local



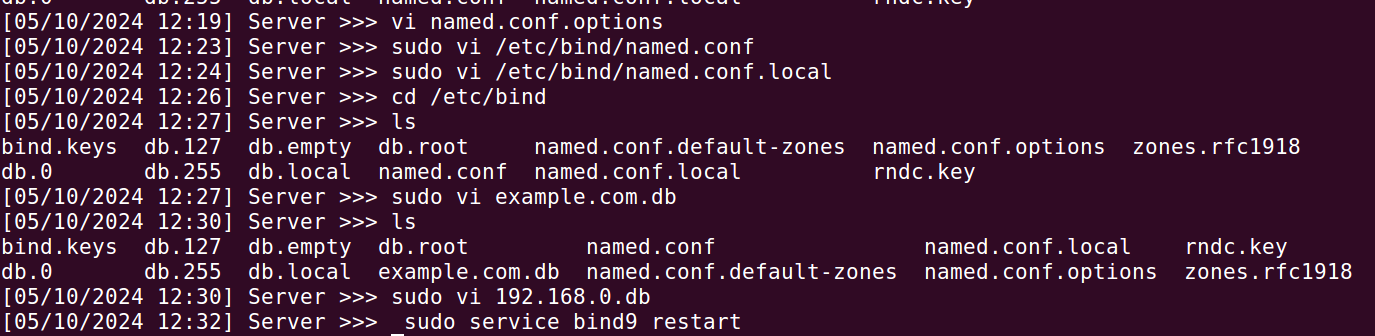
Step 2: Setup the forward lookup zone file



Step 3: Set up the reverse lookup zone file.



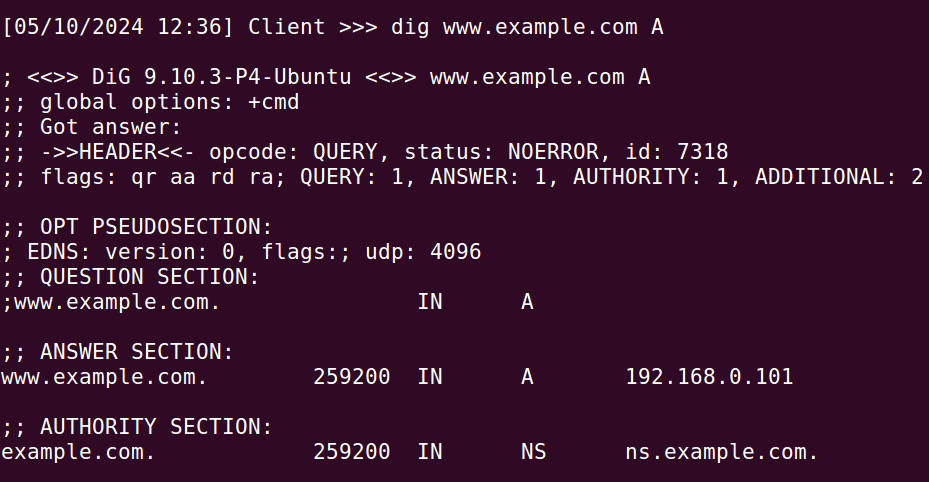
כל זה נעשה תחת ה server, כלומר ה DNS ניתן לראות את ה Terminal שלו:

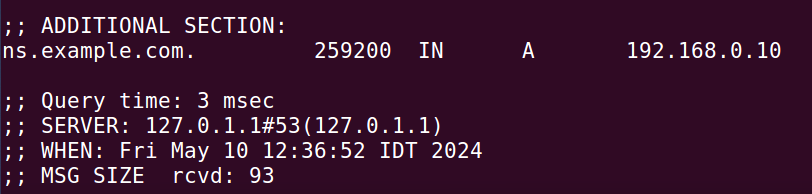


to find the IPv4 address of www.example.com, you would use:

dig www.example.com A







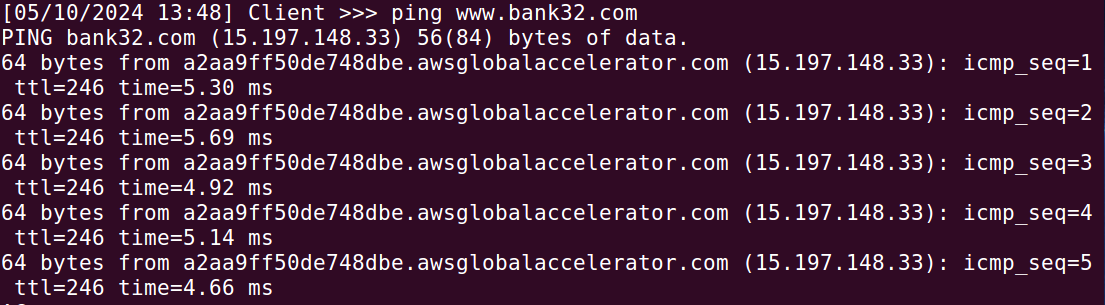
**Task 4: Modifying the Host File**

מבוא:

במטלה זו ננסה לראות את ההשפעה של שינוי בקובץ HOSTS של ה client לערך מסויים ולראות איך הוא מגיב לכך .

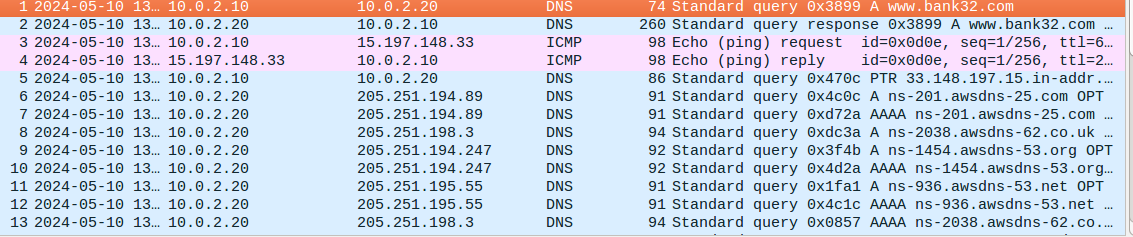
תחילה נעשה PING מה client אל www.bank32.com על מנת לראות את הכתובת האמיתי של אתר זה. לאחר מכאן נניח שפרצנו לclient ונוסיף לו בקובת HOSTS את השורה החדשה עם כתובת זו. ונוכל לוודא שצלחנו בכך שכאשר נשלח שוב PING אל [www.bank32.com](http://www.bank32.com) ה PING ישלח אל 1.1.1.1

לפני השינוי של הקובץ:



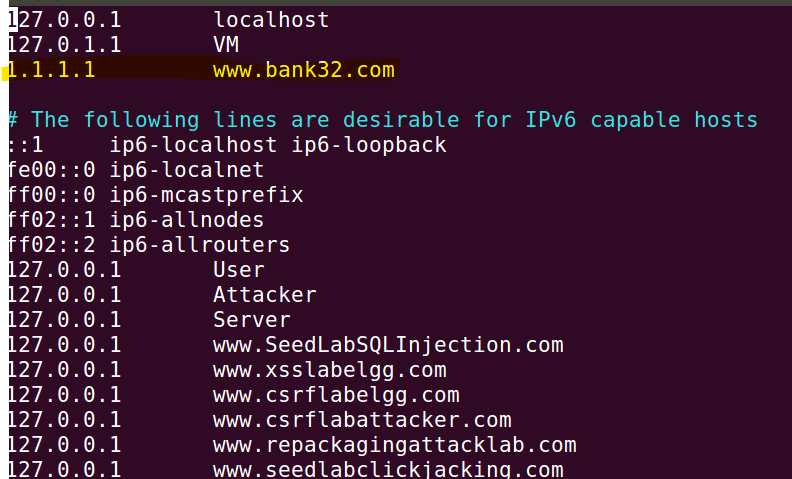
אפשר לראות מהתמונה ש [www.bank32.com](http://www.bank32.com) יושב בכתובת IP של 15.197.148.33

ניראה את ה PING ב whireshark



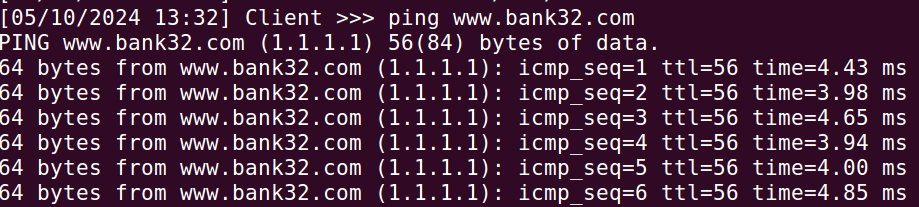
אפשר לראות שה PING אכן נשלח אל הכתובת 15.197.148.33 והוא מחזיר תגובה

לאחר השינוי של הקובץ - לפי הנחת המטלה



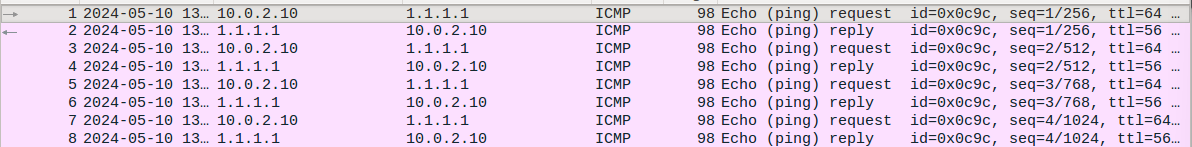
אםשר לראות שהוספנו את כתובת IP של [www.bank32.com](http://www.bank32.com) .1.1.1.1

לאחר השינוי נשלח שוב PING אל [www.bank32.com](http://www.bank32.com) ונריאה לאיזה IP הוא שולח אותו



אפשר לראות ששלחנו PING אל [www.bank32.com](http://www.bank32.com) ולא נשלח אל הכתובת המקורית 15.197.148.33

נסתכל ב wireshark



אפשר לראות שה PING אכן נשלח אל 1.1.1.1 הכתובת שאכן הזנו לו בקובץ הHOSTS

סיכום:

ניתן לראות שהצלחנו במטלה לפי זה ששלחנו פינג אל הכתובת וראינו ב wireshark שהפינג אכן נשלח אל הכתובת שהזנו לו בקובץ הHOSTS.

גילינו שכאשר אנחנו מצליחים לקבל גישה ישירה למחשב הקורבן, לפגוע בו למעשה הופך להיות כלכך קל.

התוצאות שקיבלנו אכן תואמות את הציפיות שלנו מכיוון שכאשר הצלחנו להוסיף את השורה שרצינו בקובץ הHOSTS אז הקורבן קודם יבדוק שם לפני שהוא יבצע שאילתת DNS ולכן הצלחנו לשלוח אותו לכתובת הIP שרצינו.

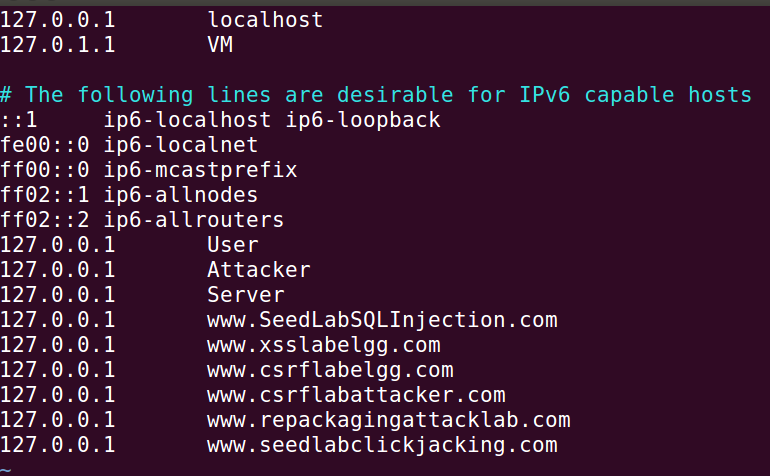
**Task 5: Directly Spoofing Response to User**

מבוא:

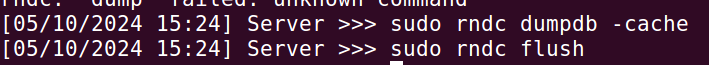
במשימה זו המטרה היא לענות לשאילתת ה DNS של ה client לפני שה local DNS כלומר ה server יענה לו, זה אפשרי כיוון שזאת הפעם הראשונה שהserver צריך להחזיר תשובה על כתובת זו ולכן הוא צריך לבצע את כלל התהליך של שאילתת DNS.

את התקיפה נעשה באמצעות פעולת netwox 105 - נעשה Spoofing לשאילתא ואז נחזיר ל client תשובה שהיא בעצם Spoof Response ובכך נגרום לו לפנות לכתובת הIP שאנחנו רוצים שיפנה אליה במקום הכתובת שהוא באמת ביקש. נוכל לוודא שהצלחנו במטלה בעזרת בכך שנראה שהclient שולח PING אל האתר בדיוק בכתובת שאנחנו הזנו לו ב Spoof Response דרך ה wireshark

תחילה נמחק את השינוי שהוספנו בטבלה של ה client:

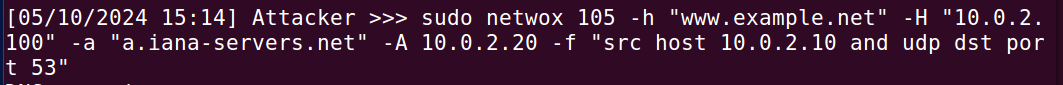


אפשר לראות שעכשיו [www.bank32.com](http://www.bank32.com) אינו מופיע בקובץ הHOSTS של ה client ולכן על מנת לבצע PING יצטרך לשאול אתה local DNS server (במקרה שלנו זה ה server)על מנת לדעת לאיזה IP לשלוח את PING

בנוסף נמחק גם את ה cache בצד של ה server 

נעשה זאת כדי להתחיל ממצב שבו הכל נקי - כלומר פעם ראשונה שניגשים לשרת

נשתמש בפקודת netwox105 על מנת לבצע את המתקפה



-h - The domain of the example server

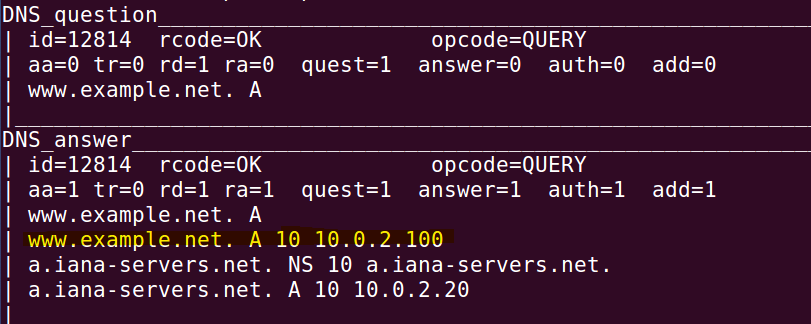
-H - The attacker machine (10.0.2.100)

-a - The authoritative server (a.iana-servers.net)

-A - The server ip (10.0.2.20)

-f - The filter to sniff only src host 10.0.2.10 with udp packet on dst port 53

התשובה ל PING שנשלח מה client ל www.example.net.com



אפשר לראות ש attackr אכן תפס את שאילתת הבקשה של ה client אל ה server ושלח לפניו תשובה מזויפת שבה הכתובת IP של האתר [www.example.net](http://www.example.net) היא 10.0.2.100

סיכום:

ניתן לראות שהצלחנו במטלה לפי זה ששלחנו פינג מהCLIENT אל כתובת האתר [www.example.net](http://www.example.net) וראינו דרך הwireshark שהפינג אכן נשלח לכתובת המזוייפת שהחזרנו לCLENT בשאילתת הDNS שלו.

התוצאה שקיבלנו אכן מתאימה לציפיות שלנו וחווינו קצת בעיות בלהבין כל פרמטר בפונקציית הNETWOX אבל התמודדנו עם זה בכך שחקרנו על פונקציה זו באינטרנט וגילינו בדיוק מה כל פרמטר מייצג.

**Task 6: DNS Cache Poisoning Attack**

מבוא:

משימה זו דומה מאוד למשימה הקודמת אך הפעם במקום שננסה להחזיר תשובה לשאילתת DNS של הCLIENT ננסה לבצע Cache Poisoning על ה local DNS שלו מכיוון של לבצע Spoofing Response על כל בקשת DNS של ה client זה לא יעיל נוכל לטווח הארוך לבצע DNS Cache Poisoning על ה loacl DNS שלו על מנת שיחזיר לו כל פעם את הIP המזוייף שהחדרנו לו פעם אחת עד שהזמן שהכתובת נשמר בCache יגמר.

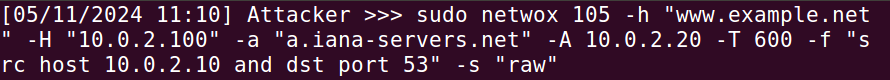
את ההחדרה הזו שוב נבצע בעזרת netwox 105 אך הפעם נשנה מעט את הפרמטרים כך שיתאימו לכך שביצוע ה Spoofing Response יתאים לקורבן שהפעם הוא ה server נוכל לדעת שהצלחנו במשימה בכך שנבצע שאילתת DNS מה client ונקבל את ה Spoofing Response שהחזרנו אל ה local DNS שלו.

תחילה ננקה את ה cache בצד של השרת



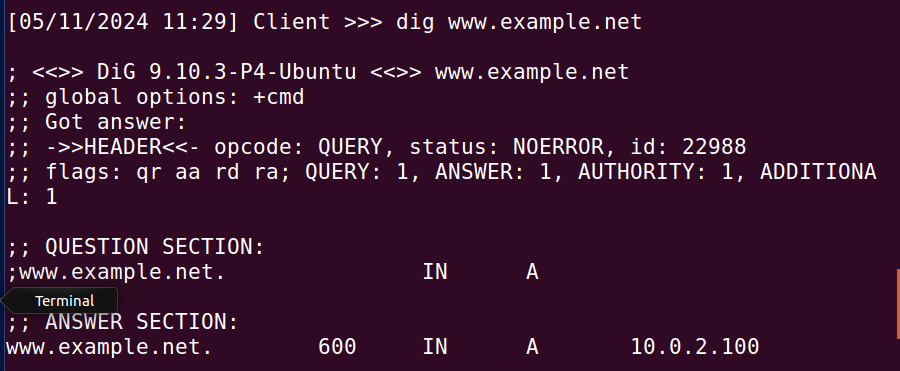
נעשה זאת כדי להתחיל ממצב שבו הכל נקי - כלומר ה server צריך לבקש מה local DNS שלו את IP של האתר על מנת שנוכל לבצע עליו את ה Cache Poisoning

נריץ את המתקפה על ידי 105 netwox

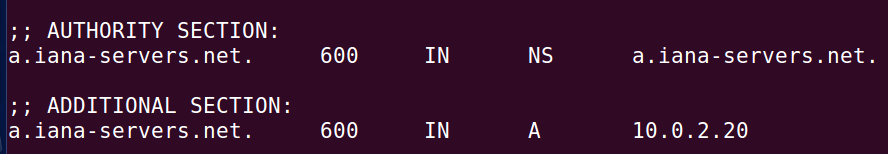


נשתמש באותה פקודה כמו במשימה הקודם אך הפעם קורבן יהיה ה server ונענה לו בתור ה DNS שלו.

נבצע מה client פעולת dig אל www.example.net



אפשר לראות כי ה IP ש client קיבל בחזרה מה server הוא ה10.0.2.100 כלומר הצלחנו להחזיר תשובה אל ה server לפני ה DNS שלו משמע הצלחנו במתקפה



סיכום:

ניתן לראות שהצלחנו במתקפה בכך שביצענו

מה client פעולת dig אל [www.example.net](http://www.example.net) וכתובת הIP שקיבלנו בחזרה מהSERVER היא כתובת הIP שאנו החדרנו לCache של הSERVER.

תוצאות אלו שקיבלנו תואמות את מה שציפינו שנקבל מכיוון שכאשר שרת הDNS מקבל שאילתה לכתובת מסויימת שאינה נמצאת אצלו בCache הוא צריך להעביר את השאילתה לשרתים הבאים בתור על מנת לקבל תשובה וזה בדיוק הזמן בו אנחנו מסניפים את השאילתה ומחזירים תשובה מזוייפת משלנו.

**Task 7: DNS Cache Poisoning: Targeting the Authority Section**

מבוא:

במשימה זו הפעם נרצה לבצע DNS Cache Poisoning לא רק על כתובת IP בודדת מכיוון שזה לא יעיל להריץ את המתקפה על כל כתובת בנפרד בדומיין. לשם כך נרצה לבצע Spoof Response על כל מקטע של כתובות שתחת ה domain של example.net ובכך כל בקשה שה client יבצע אל ה local DNS שתחת ה domain הזה ה local DNS שלו יחזיר לו את ה Spoof Response שאנחנו הגדרנו לו מראש לדוגמה attacker32.com

נוכל לדעת שהצלחנו במשימה דרך ה wireshark בכך שה client ישאל את הlocal DNS שלו כתובת IP תחת הדומיין example.net וניראה אם התשובה שוחזרה לו היא attacker32.com כמו ששהגדרנו במתקםה

נכתוב קוד עם scapy שעושה sniffing וומחפש packet שמתאים לתנאים ומחזירSpoof Response אל localDNS



בקוד שכתבנו פילטרנו כך שנזייף רק בקשות DNS המיודעות אל www.example.com החלפנו את הפורט יעד ומקור ואותו דבר עשינו לכתובת ה IP על מנת שנוכל לזייף את ה UDP packet, לאחר מכאן הוספנו רשומות של NS והגדרנו להן IP

ולבסוף שלחנו את ה packet

Nscount =1 – Authority section

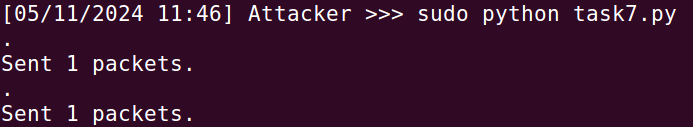
arcount=0 – disable the aditional response

ננקה את ה server cahhe



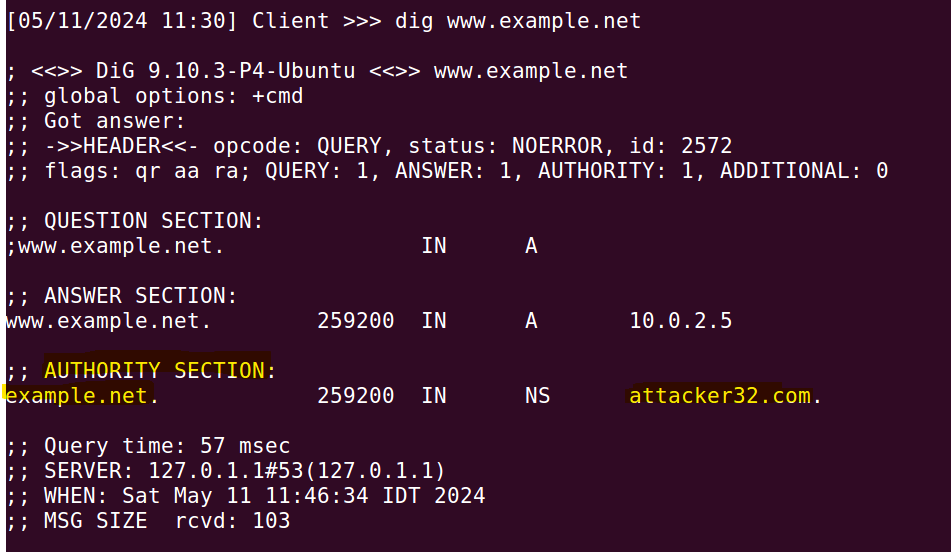
על מנת שהוא ישלח שאילתא ל Authority ואנחנו נוכל להחזיר לו Spoof Response

נריץ את הקוד שכתבנו:



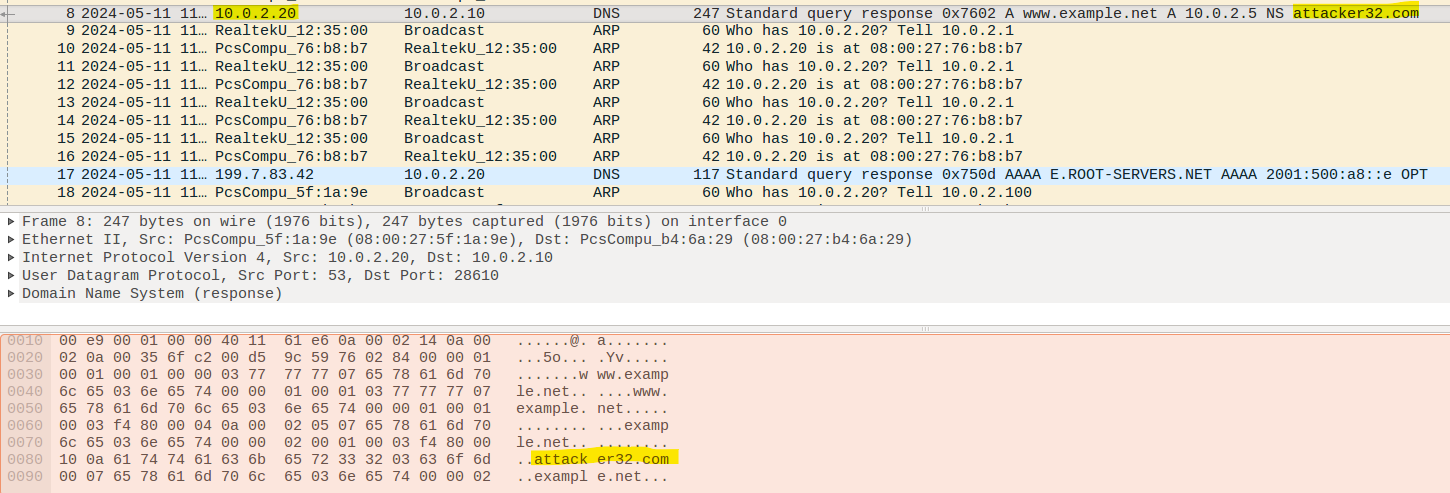
אפשר לראות שהקוד רץ בהצלחה וכעת צריך לבדוק את תוצאות המתקפה

נבצע פקודת dig על [www.example.net](http://www.example.net) לאחר שהפעלנו את ה קוד בצד של התוקף.



אפשר לראות כי כל ה domin של example.net נמצא עכשיו תחת attacker32.com משמע הצלחנו להרעיל את ה local DNS

נבדוק ב wireshark מאיפה הגיעה תשובה לשאילתת ה DNS של ה client:



כפי שניתן לראות ב wireshark ה url של attacker32.com הגיע מ ה local DNS שלנו שהוא ה server שה IP שלו 10.0.2.20 .

סיכום:

ניתן לראות שהצלחנו במתקפה מכיוון שבדקנו בwireshark מאיפה (מאיזה דומיין) הגיעה תשובה לשאילתת ה DNS של ה client וגילינו במתקפה זו שניתן לבצע הרעלה לא רק על כתובת ספציפית אלא על כל הדומיין.

**Task 8: Targeting Another Domain**

מבוא:

בהתקפה הקודמת, הצלחנו להרעיל את המטמון של שרת ה-DNS המקומי, אז attacker32.com הופך לשרת השמות עבור הדומיין example.com. בהשראת הצלחה זו, ברצוננו להאריך השפעתו על תחום אחר. כלומר, בתגובה המזויפת שהופעלה על ידי שאילתה עבור www.example.net,

ברצוננו להוסיף ערך נוסף בAuthority section, אז attacker32.com

משמש גם כשרת השמות עבור google.com.

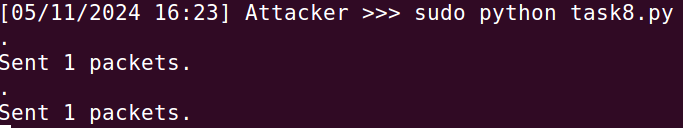
תחילה ננקה את ה cache בשרת dns.



נכתוב קוד זהה לקוד הקודם של הסעיף הקודם, רק שהפעם נוסיף ב Authority section את google.com שילקח מ attacker32.com. נוסף על כך נשנה את ה nscount=2 על מנת שנתייחס גם ל NSsec1 וגם ל NSsec2.

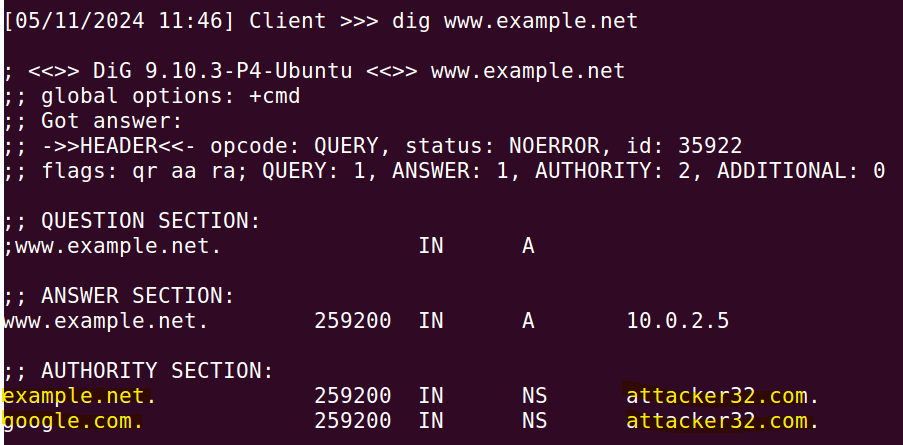


בקוד שכתבנו פילטרנו כך שנזייף רק בקשות DNS המיודעות אל www.example.com החלפנו את הפורט יעד ומקור ואותו דבר עשינו לכתובת ה IP על מנת שנוכל לזייף את ה UDP packet לאחר מכאן הוספנו רשומות של NS של google.com ושל example.com והגדרנו אותם תחת הדומיין של attacker32.com

כעת נפעיל את הקוד שלנו שיבצע sniffing עד שיקבל packet של udp עם dst port 53 ועם בקשת DNS בתוכו לכתובת [www.example.net](http://www.example.net), רק כאשר נמצא packet כזה (כפי שניתן לראות בהתניות בקוד למעלה) ישלח packet מתאים בתגובה. (השליחה שרואים בתמונה היא לאחר הרצת ה dig לכתובת של example.net. 

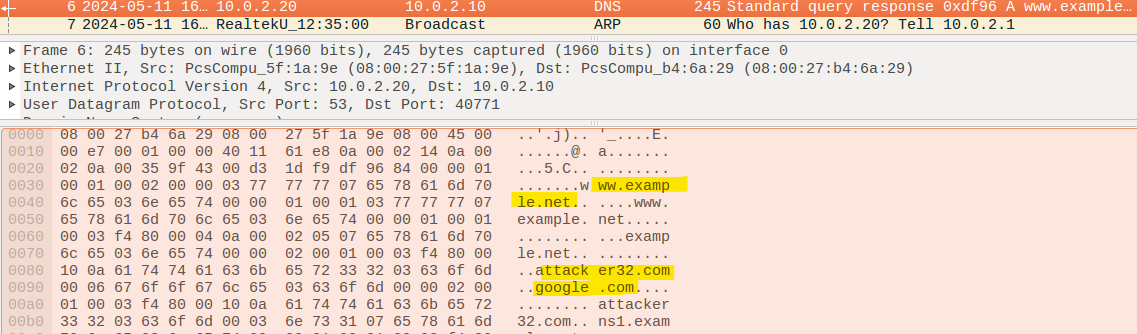
אפשר לראות הקוד רק בהצלחה

לאחר שליחת ה dig ל [www.example.net](http://www.example.net) כפי שניתן לראות קיבלתי ב Authority section גם את של example.net וגם את google.com



אפשר לראות שאכן הצלכנו להכניס את ה example.net ואת google.com תחת הדומיין של attacler32.com

כפי שניתן לראות ב wireshark נשלחה packet של DNS מהשרת DNS אל ה client עם google.com ועם example.net.



סיכום:

ניתן לראות שהצלחנו במתקפה מכיוון שראינו בwireshark שאכן נשלחה packet של DNS מהשרת DNS אל ה client עם google.com ועם example.net.

**Task 9: Targeting the Additional Section**

מבוא:

ב Replies DNS, יש סעיף שנקרא מדור נוסף, המשמש לספק מידע נוסף.

בפועל, הוא משמש בעיקר כדי לספק כתובות IP עבור שמות מארחים מסוימים, במיוחד עבור אלה המופיעים בסעיף הרשות. המטרה של משימה זו היא לזייף כמה ערכים בסעיף זה ולראות אם הם יישמרו בהצלחה על ידי שרת ה-DNS המקומי היעד. בפרט, כאשר מגיבים לשאילתה עבור www.example.net, אנו מוסיפים את הערכים הבאים בתשובה המזויפת, בנוסף לערכים בסעיף התשובות.

תחילה נמחק את ה cache:

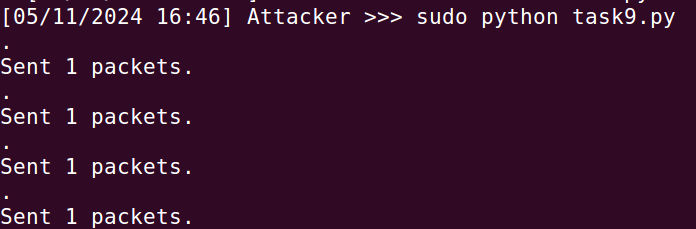




בקוד שכתבנו פילטרנו כך שנזייף רק בקשות DNS המיודעות אל www.example.com החלפנו את הפורט יעד ומקור ואותו דבר עשינו לכותבת ה IP על מנת שנוכל לזייף את ה UDP packet לאחר מכאן הוספנו רשומות של NS וגדרנו אותם תחת הדומיין של attacker32.com ובנוסף הגדרנו להם IP שאנחנו החלטנו

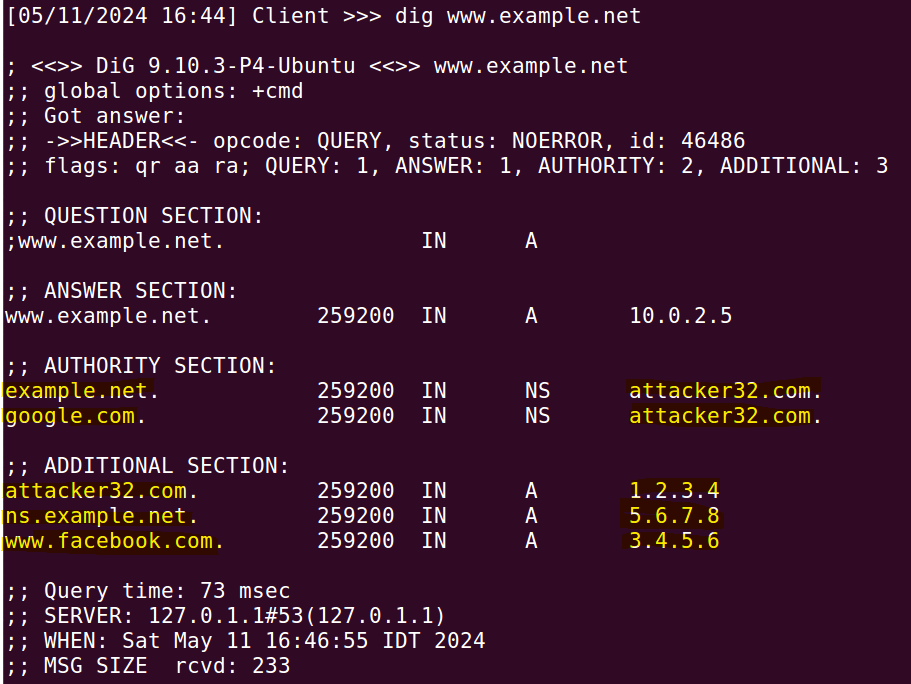
בנוסף עשינו inject לכתובת של [www.facebook.com](http://www.facebook.com) והגדרנו אותה גם תחת האותו דומיין ונתנו לו IP מזוייף  
ולבסוף שלחנו את ה packet

הרצנו את הקדו שכתבנו



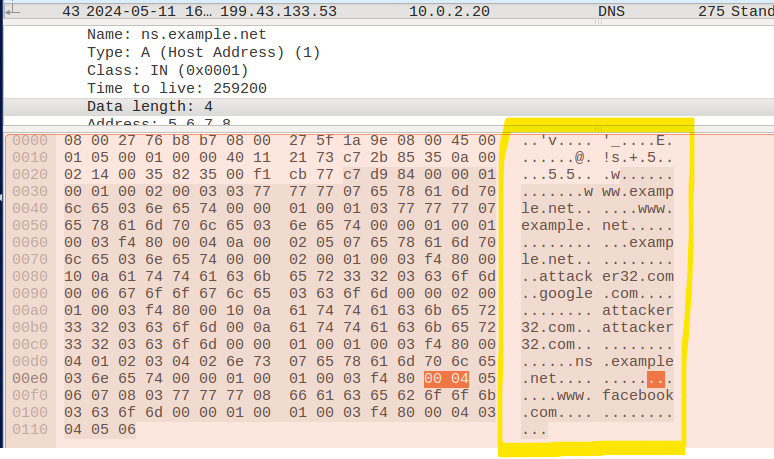
אפשר לראות שהקוד רץ בהצלחה

כעת נבדוק מהצד של ה client:



אפשר לראות כי אכן google.com ו example.com נכנסו תחת הדומיין של attacker32.co, והכתובות IP שהגדרנו לשאר ה דומיינים זוייפו לכתונות שהגדרנו בקוד למעלה

נסתכל עכשיו דרך ה wireshark:



אפשר לראות כי packet אכן נשלח והגיע אל ה DNS server שלנו

סיכום:

הצלחנו לשלוח תשובת DNS מזויפת הכוללת רשומות נוספות.

הוכחנו זאת באמצעות מעקב אחרי הרשומות שנשמרו במטמון ה-DNS של השרת המקומי ובדיקת כתובות ה-IP השמורות.

גילינו שרשומות ה-Additional הקשורות לרשומות ה-Authority (attacker32.com ו-ns.example.net) נשמרו במטמון, בעוד שרשומות לא קשורות (www.facebook.com) לא נשמרו. למדנו שהשרת המקומי מתייחס לרלוונטיות הרשומות בהקשר לשאילתא המקורית.

זה מתאים לתיאוריה ששרת DNS ישמור במטמון רשומות Additional רק אם הן קשורות ישירות לרשומות ה-Authority, כפי שמצופה משרת DNS שמתפקד כראוי.

הבעיה העיקרית הייתה לוודא שהרשומות המזויפות נכנסות למטמון ה-DNS. התמודדנו עם זה על ידי בדיקות חוזרות ונשנות ושימוש בכלים לניטור המטמון כדי לאשר שהרשומות הרלוונטיות נשמרות.

**סיכום כללי של המעבדה:**

הצלחת המתקפה:

הצלחנו להחדיר רשומות DNS מזויפות למטמון של השרת המקומי. ראינו זאת כאשר שלחנו פינג לכתובת והפינג נשלח לכתובת המזויפת שהזנו בקובץ ה-HOSTS או החזרנו בתשובת ה-DNS. ב-Wireshark ראינו שהשאילתות נענו על ידי הכתובות המזויפות שהכנסנו.

הוכחת ההצלחה:

הוכחנו את ההצלחה על ידי שימוש בכלים לניטור כמו Wireshark שראו את הכתובות המזויפות שנענו, וביצוע שאילתות DNS כמו dig שראו שהכתובות שהחזרנו אכן נענות מהמטמון של השרת המקומי.

מה גילינו ולמדנו:

גילינו שהוספת רשומות Additional הקשורות ישירות לרשומות ה-Authority נשמרות במטמון ה-DNS, בעוד שרשומות שאינן קשורות אינן נשמרות. למדנו שהשרת המקומי שומר רק רשומות רלוונטיות לשאילתא המקורית במטמון.

התאמה לתיאוריה:

התוצאות שקיבלנו תואמות את הציפיות ואת התיאוריה ששרת DNS ישמור במטמון רשומות Additional רק אם הן קשורות לרשומות ה-Authority. זה מצביע על תפקוד נכון של שרת ה-DNS בהקשר זה.

בעיות ופתרונות:

התמודדנו עם בעיות בהבנת הפרמטרים השונים בפונקציות הרשת ובתהליך ההתקפה. פתרנו זאת באמצעות חקירה ולימוד מעמיק של הפונקציות והפרמטרים דרך מקורות מידע באינטרנט.

רפלקציה ותובנות נוספות:

המעבדה חידדה את הבנתנו לגבי מתקפות DNS Spoofing ואת החשיבות של הבטחת שרתי DNS מפני מתקפות כאלה. גילינו שאם יש גישה ישירה למחשב הקורבן, הפגיעה בו הופכת לקלה מאוד. אחת הדרכים להגן על מערכות מפני מתקפות כאלה היא על ידי שימוש ב-DNSSEC (DNS Security Extensions), שמוסיף שכבת אבטחה על ידי חתימה דיגיטלית של רשומות DNS, ובכך מונע שינוי בלתי מורשה של רשומות ה-DNS.

המלצות להמשך מחקר:

נכון להמשיך ולחקור כלים וטכניקות להגנה על מערכות DNS, כמו DNSSEC, וללמוד על מתקפות מתקדמות יותר כמו מתקפות Cache Poisoning מורכבות. בנוסף, ניתן לבחון כלים חדשים להגנה על רשתות ושרתי DNS, ולהעמיק בחקר התקפות שבוצעו בפועל וההגנות שפותחו בתגובה אליהן.

כלי מתקדם:

אחד הכלים החדשניים בתחום ההגנה על DNS הוא Pi-hole, שמספק פתרון מבוסס DNS להחסמת פרסומות ומעקב, ומסייע גם במניעת גישה לאתרים מזיקים על ידי שימוש ברשימות חסימה שמתעדכנות באופן תדיר.

סיכום כללי:

המעבדה הייתה מוצלחת והצלחנו להחדיר רשומות מזויפות לשרת ה-DNS המקומי. למדנו על תהליך מתקפות DNS Spoofing, הדרכים להחדיר רשומות מזויפות, והחשיבות של הגנות מתאימות כמו DNSSEC. המעבדה חידדה את המודעות לסכנות האפשריות ולחשיבות ההגנה המתאימה על תשתיות רשת ו-DNS.

**הסבר מפורט על Pi-hole:**

הגנה מפני מתקפת DNS Spoofing באמצעות Pi-hole:

1. חסימת דומיינים זדוניים:

Pi-hole משתמש ברשימות חסימה שמתעדכנות באופן שוטף וכוללות דומיינים הידועים כזדוניים או מפיצי תוכנות זדוניות. במתקפת DNS Spoofing, התוקף מנסה להוסיף רשומות מזויפות עבור דומיינים מסוימים, כמו attacker32.com או ns.example.net. Pi-hole יכול לזהות ולחסום דומיינים זדוניים אלו מראש, וכך למנוע את האפשרות שהמכשירים ברשת ינסו לגשת אליהם.

2. שמירה על שלמות השאילתות:

Pi-hole פועל כשרת DNS מקומי ומסנן את כל השאילתות היוצאות. בכך הוא מונע את האפשרות ששאילתות DNS יועברו לשרתים זדוניים או מזויפים שהתווספו למטמון ה-DNS של השרת המקומי. Pi-hole מבטיח שהשאילתות ינותבו רק לשרתים בטוחים ומהימנים, תוך כדי מניעת שינוי של תשובות ה-DNS על ידי תוקפים.

3. ניטור וניהול השאילתות:

Pi-hole מספק ממשק ניהול שמאפשר לנטר בזמן אמת את כל השאילתות המתבצעות ברשת. במידה ומתקפת DNS Spoofing מתרחשת, מנהל הרשת יכול לזהות במהירות שאילתות חשודות או חריגות ולנקוט בפעולות מתאימות. זה כולל חסימת דומיינים נוספים או התאמת הגדרות האבטחה ב-Pi-hole.

4. שילוב עם DNS-over-HTTPS (DoH) ו-DNS-over-TLS (DoT):

Pi-hole תומך ב-DNS-over-HTTPS (DoH) ו-DNS-over-TLS (DoT), המוסיפים שכבת אבטחה על ידי הצפנת שאילתות ה-DNS. זה מונע מהתוקפים ליירט ולשנות את התשובות לשאילתות ה-DNS, מכיוון שהשאילתות מוצפנות ואינן ניתנות לשינוי בקלות.

דוגמה ספציפית להגנה:

נניח שבוצעה מתקפת DNS Spoofing במטרה להפנות את המשתמשים לאתר מזויף על ידי שינוי הכתובת של www.example.net לכתובת IP זדונית. אם Pi-hole מותקן ומשתמש ברשימות חסימה מעודכנות, הוא יוכל לזהות שהכתובת attacker32.com או ns.example.net הן דומיינים זדוניים ולחסום את השאילתות אליהם.

במקרה כזה, כל שאילתת DNS לכתובת www.example.net תיבדק מול רשימות החסימה של Pi-hole, ואם היא זדונית היא תיחסם מיד. כך, Pi-hole ימנע מהמשתמשים גישה לכתובות ה-IP המזויפות ויבטיח שהם יגיעו רק לאתרים לגיטימיים.

**סיכום:**

Pi-hole מספק שכבת הגנה משמעותית כנגד מתקפות DNS Spoofing על ידי חסימת דומיינים זדוניים, ניטור וניהול השאילתות, והצפנת התעבורה. זה מונע מהתוקפים לשנות את תשובות ה-DNS ומבטיח שהמשתמשים יפנו רק לכתובות IP מהימנות. באמצעות כלים אלו, Pi-hole יכול להגן ביעילות על הרשת המקומית מפני מתקפות DNS Spoofing.