על מנת לזהות תקיפת ARP-spoofing השתמשנו במספר שיטות:

1. בדיקת כפילות כתובות ב- arp table

לאחר העתקת הטבלה לקובץ f1 [בעזרת הפקודה os.system("arp -a> f1")] הכנסנו לרשימה b את כתובות ה- MAC שבטבלה.

בעזרת הפונקציה set שבונה רשימה ללא איברים כפולים, ובעזרת הפונקציה len שמחזירה אורך של רשימה, בדקנו האם ברשימה b ישנה כתובת MAC החוזרת על עצמה. [if len(set(b)) != len(b)]

2. בדיקה אם יש המון הודעות ARP REPLY יחסית להודעות ARP המתקבלות בזמן מסוים.

ברשימה pkt\_sniff הכנסנו את כל ההודעות מסוג ARP שהתקבלו ולרשימה c הכנסנו רק את הכתובות שהן מסוג arp reply [כלומר op=2].

אם מספר ההודעות מסוג arp-reply שווה ל- 70% או יותר מסך ההודעות [if len(pkt\_sniff) \* 0.7 <= len(c)]  
נסיק שמספר התגובות הוא יחסית גדול למקרה שהכל תקין ולכן ישנה אפשרות סבירה לתקיפה.

3. התקבלו יותר מ-3 הודעות מכתובת IP זהה ב-10 שניות שהקצבנו.

את כתובות ה-IP שהתקבלו מיינו בסדר יורד לפי מספר המופעים של כל כתובת [Counter(c).most\_common(1)[0][1]]. נבדוק אם הכתובת מופיעה יותר מ- 3 פעמים.

4. קבלת חבילה כפולה, כלומר עבור אותה כתובת IP ישנן שתי כתובות MAC שאינן זהות.

הפקודה len(set(zip(c, d))) מחזירה את מספר זוגות ה- IP-MAC הזהים,

והפקודה dict(set(zip(c, d)) מחזירה רק את כתובות ה- IP הזהות, לכן אם הפקודות מחזירות לנו שני מספרים שאינם זהים, נוכל להסיק כי ישנן כתובות IP זהות בעלות כתובות MAC שאין זהות.

5. בדיקה לכל כתובת IP- אם זו הכתובת הנכונה (לפי כתובות הIP השמורות בarp table)

עבור כל כתובת IP, נבדוק בעזרת הפונקציה get\_mac\_address אם כתובת ה-MAC שלו המוצגת בטבלה היא באמת כתובת ה- MAC שלו.

נבדוק כמה מבין הפרמטרים למעלה זיהו בעיה:

אם מצאנו פחות מ-2 בעיות: כנראה שאין התקפה.

אם מצאנו 2-3 הבעיות: זרקנו הערה שהמחשב בסיכון.

אם מצאנו 4-5 הבעיות: זרקנו הערה שהמחשב תחת התקפה, ניגשנו באמצעות פקודת מערכת לטבלת הarp והגדרנו בצורה סטטית את הכתובת הנכונה.

[os.system("arp -s " + attacker\_ip + " " + real\_mac)]

(בשלבים הקודמים שמרנו את כתובת הIP שאנחנו חושדות ששייך לתוקף ואת כתובת הMAC הנכונה.)