**מטלה 2:**

שלב ראשוני עברנו על המסווג ניסינו להבין איזה פיצ'רים הוא מחפש, מהן נק' החולשה שלו על מנת שנוכל לבנות התקפה מתאימה ,

המסווג שלנו עובד על FM Factorization Machine)) הוא מסתכל על 7 סוגי פיצ'רים מקוד המקור ומקובץ המניפסט, מתוכם ארבעה סוגים מופקים מקובץ המניפסט של אפליקציה ושלושה סוגי פיצ'רים נוספים מקוד המקור של האפליקציה שנפרקה.

-רכיבי האפליקציה

- תכונות חומרה

- מסנן כוונות

- הרשאות

- ממשקי API מוגבלים

-ממשקי API חשודים

- הרשאות בשימוש

**חולשות במכונה:**

1. המכונה מחפשת אחרי סט פיצ'רים מוגדר מראש ולכן אם נשנה אותם או שתתקבל תוכנה זדונית שלא משתמש בהם המסווג לא יזהה את התוכנה.
2. סוג המסווג הוא סטטי ולכן שינויים דינאמיים הוא לא יצליח לגלות
3. חולשה נוספת יכולה להיות כאשר הקוד מעורפל ואז למסווג אין יכולת לקרוא את הקוד ולהוציא את הפיצ'רים.
4. המכונה יוצאת מנקודת הנחה שלא כל התוקפים יכניסו בדיוק את אותו ווקטור של נתונים(הרשאות/API וכו) שהם לא משתמשים בו ולכן נוכל להוסיף נתונים נוספים על מנת לבלבל את המודל.

בשלב השני אימנו את המודל על 70% אפליקציות טובות וזדוניות , וב30% השארנו לטסט.

חקרנו את האפליקציות מתוך החלק של הטסט וניסינו להבין איזה שינויים נוכל לבצע.

את הלמידה על האפליקציות עשינו ע"י קובץ הdata ואת הביצוע נבצע באפליקציה עצמה.

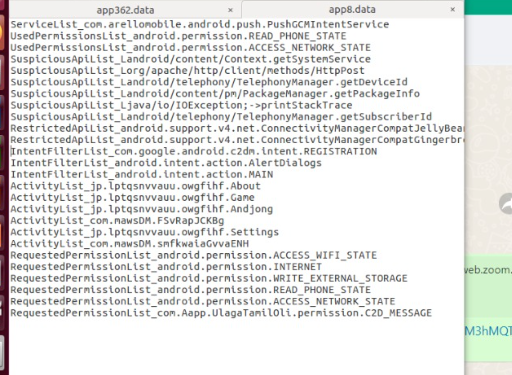
לקחנו פיצ'רים שחוזרים על עצמם באפליקציות טובות שמתבסס על האופן ש FM עובד, ובדקנו מה קורה כאשר אנחנו מוסיפים את הפיצ'רים האלו לאפליקציה זדונית .

לאחר הוספת הפיצ'רים גילינו כי המסווג סיווג את האפליקציה כטובה במקום זדונית והורדנו את רמת הדיוק של המסווג

**אפליקציה זדונית לפני שינוי (מסווג כזדונית):**



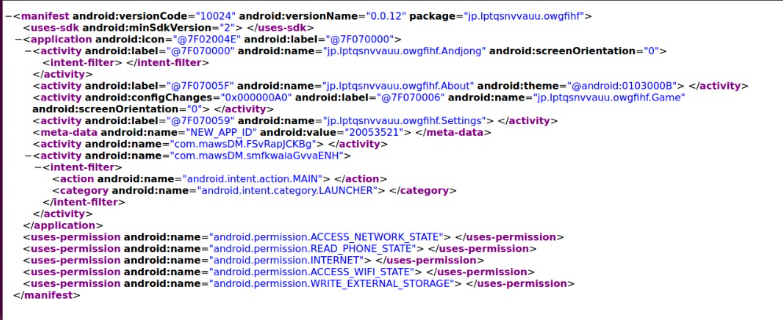
**אפליקציה זדונית לאחר שינוי(מסווג כטוב):**



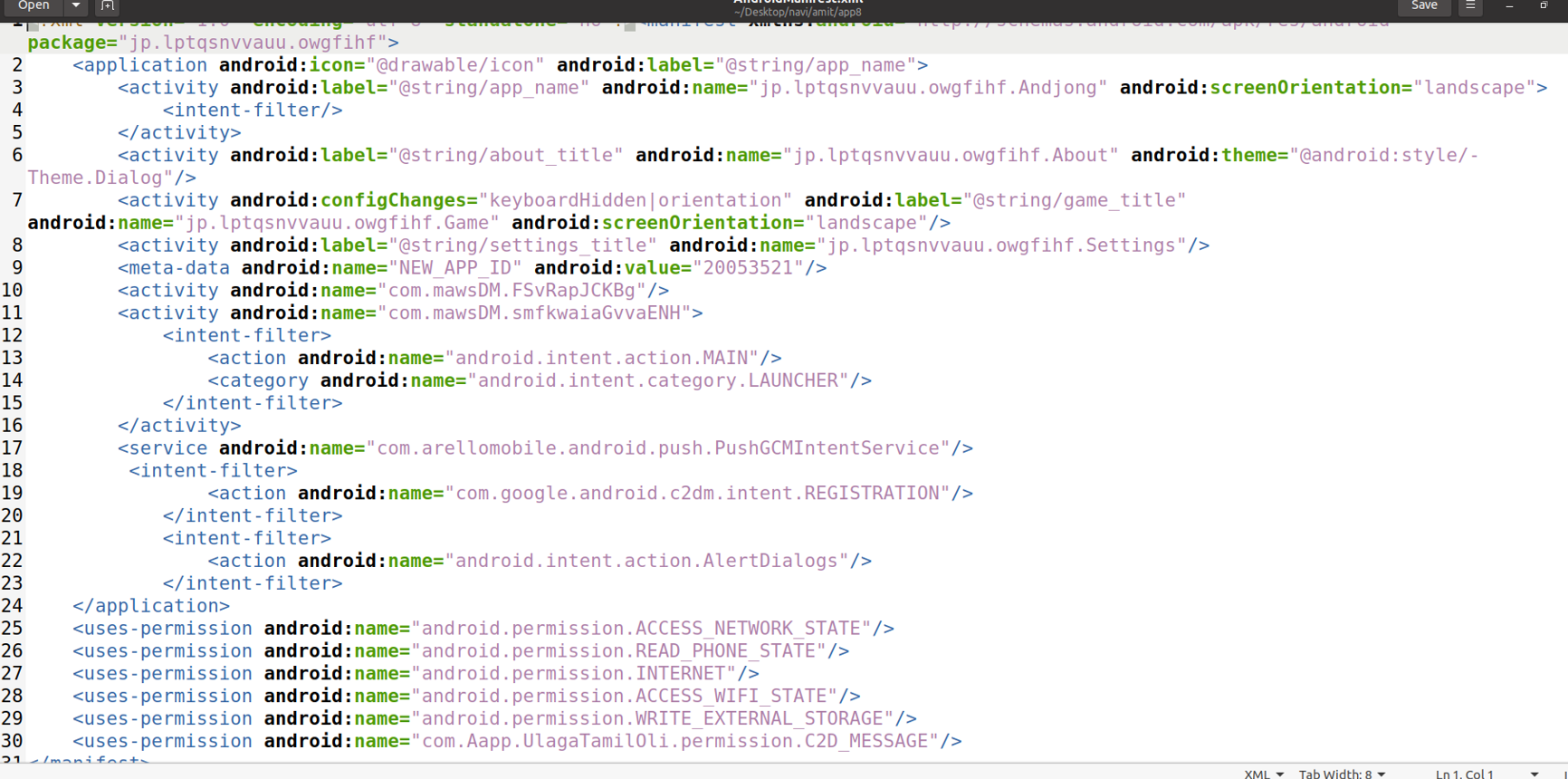
את השינויים האלו נבצע באפליקציה עצמה.

בשלב ראשון הוספנו את הרשאות בקובץ manifest

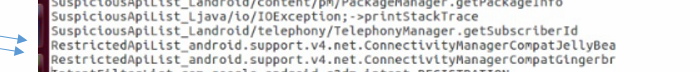
לפני שינוי



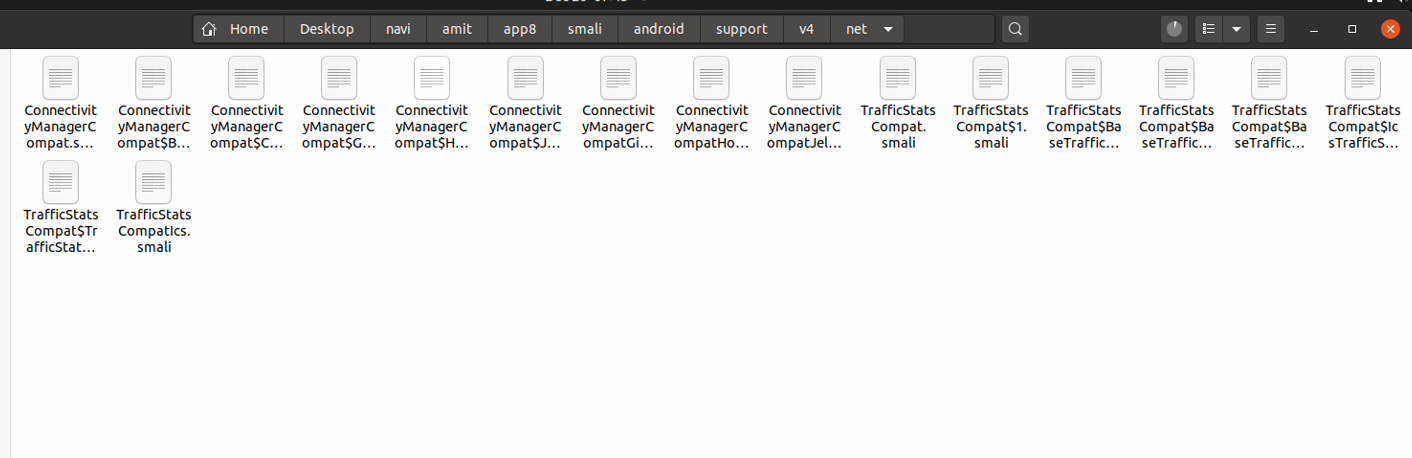
לאחר שינוי:



מהקובץ data ראינו שהוא מוסיף את שני הדברים הבאים:



ולכן הוספנו את המידע הבא לאפליקציה הזדונית בקובץ smali



חתמנו את האפליקציה הזדונית לאחר השינויים והרצנו את המסווג, קיבלנו שהמסווג סיווג את האפליקציה כטובה.

**מסקנות:**

בשלב הבא של המטלה נרצה לממש את התקפה בשאר האפליקציות לפי האפליקציות הטובות שיש לנו בטסט.