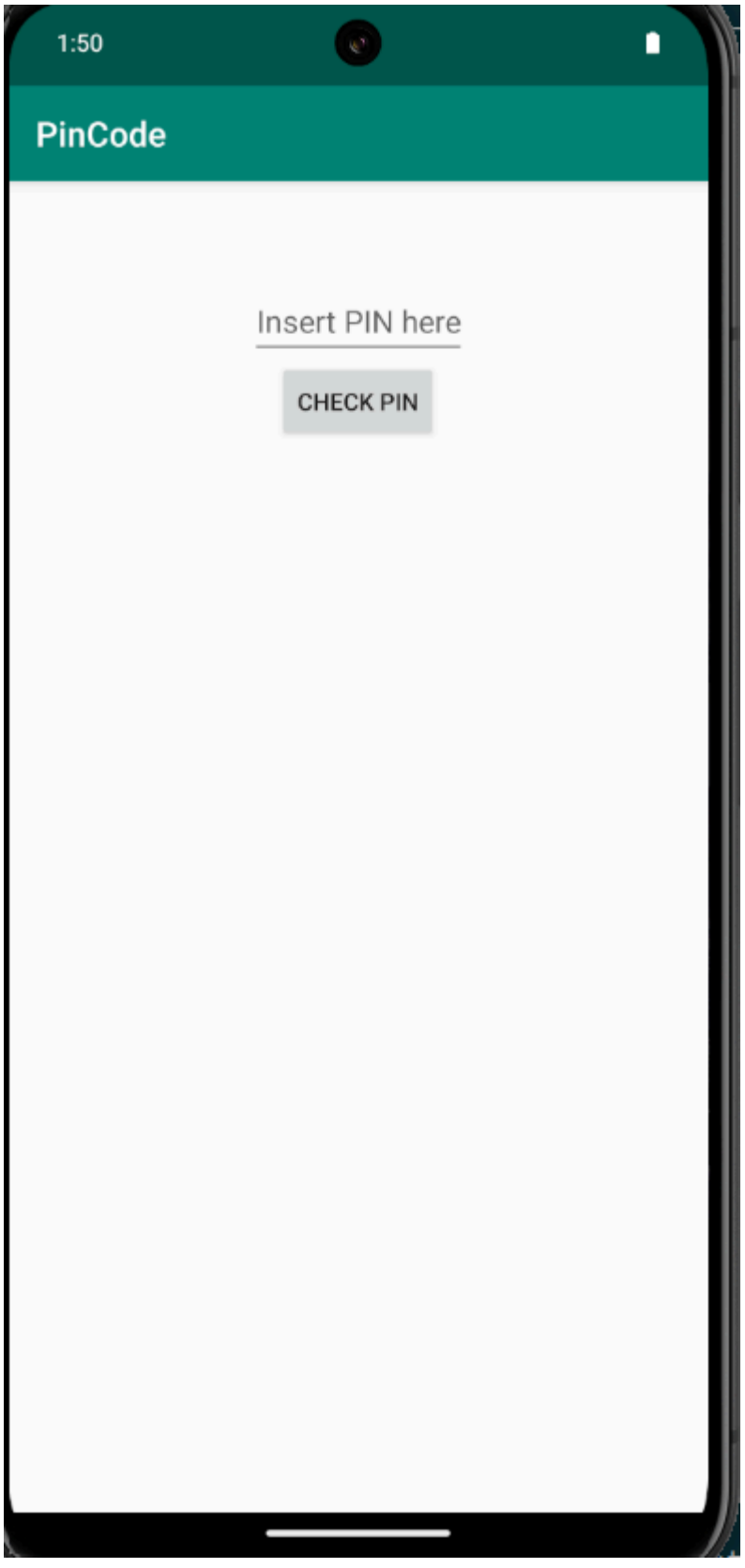
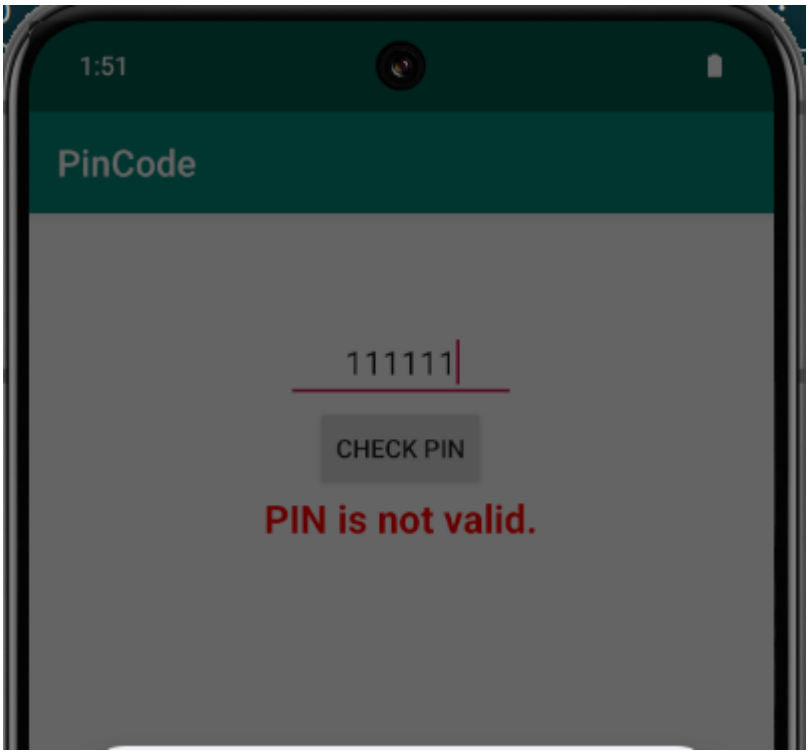
2-PinCode - reverse engineering

ده تاني تحدي معانا وهو ان احنا علشان نجيب flag لازم يكون pin صحيح واحنا مش عارفينه فلازم نروح نحلل app ونشوف ازاي pin بيتعملوا generation .....


1:50
PinCode
Insert PIN here
CHECK PIN



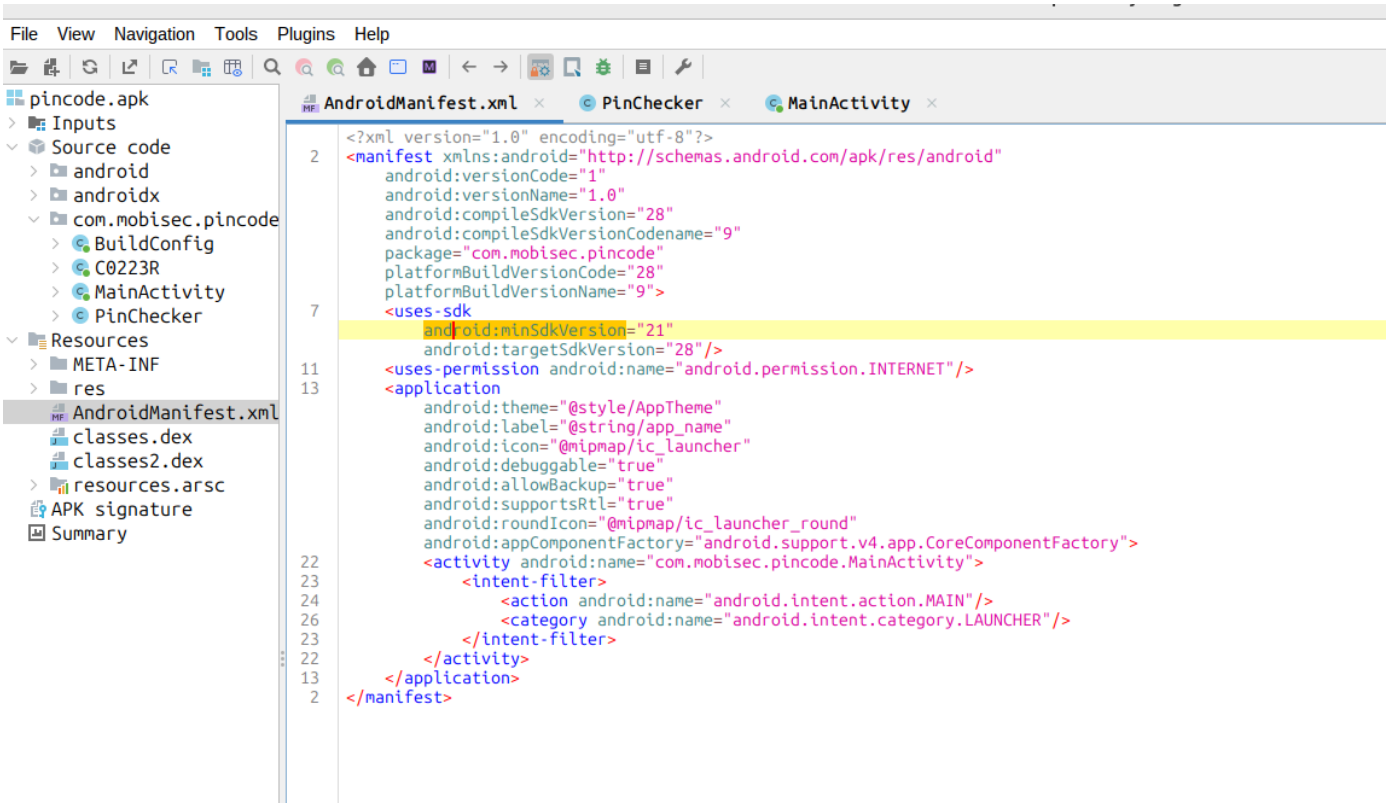
1-open app with jadx-gui tool : decompile دي عبارة عن اداة بتحلل التطبيق من غير ما نعمله

```
jadx-gui pincode.apk
```

اول خاجة لازم نتاكد منها هو ملف ManiFest.xml ؟ علشان ممكن نستغله ان نحاول نعمل bypass activity or hack content provider or hacking broadcast receaiver or

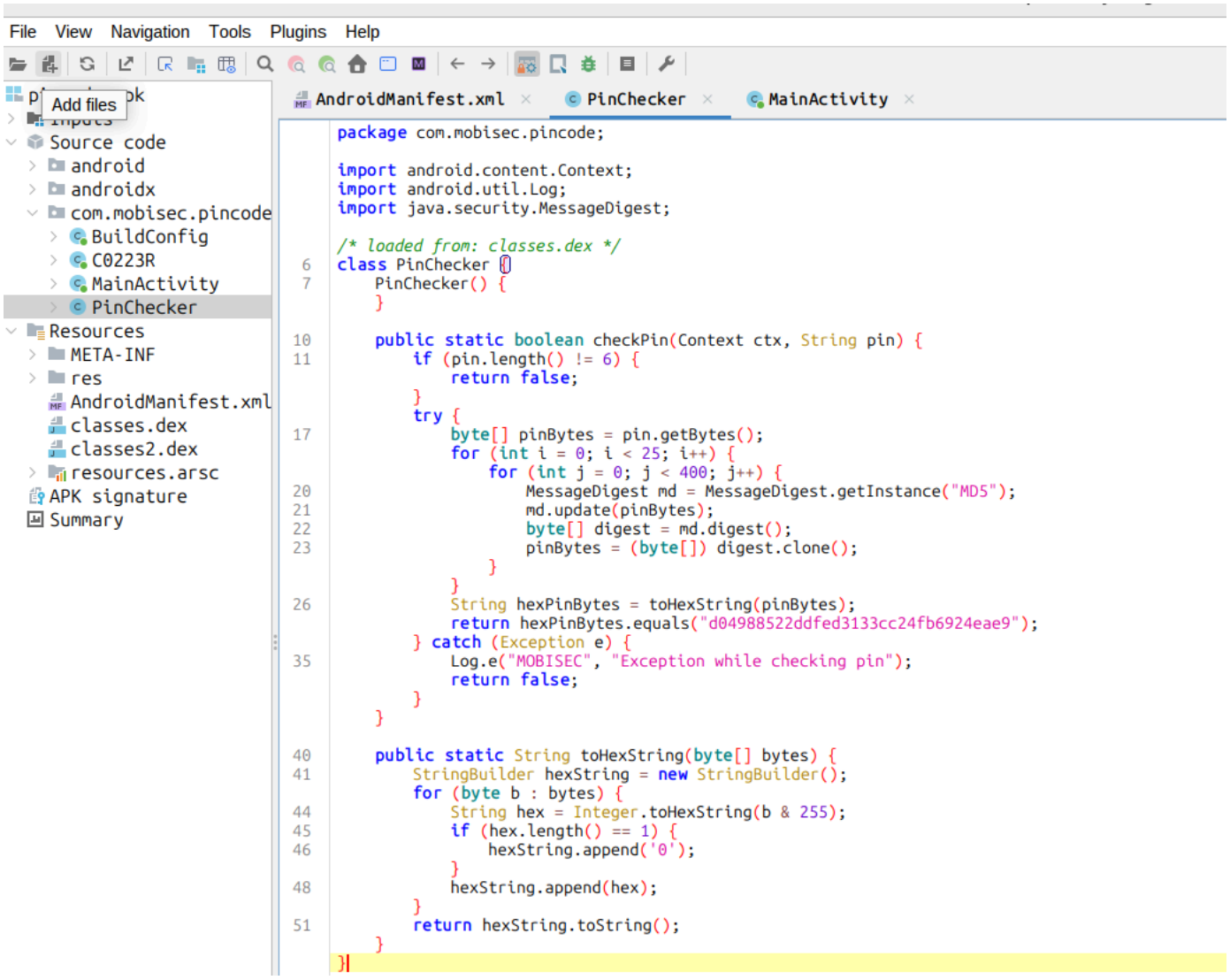
2-open ManiFest.xml

بس مش هنلاقي في اي حاجة مهمة ممكن نستغلها



3- open PinChecker

وده عبارة عن code اللي بيعمل generate PinCode



analyze code java

1- code length must be = 6

```
        if (pin.length() != 6) {
            return false;
        }
```

2- encode the PinCod

- convert code into byte --> pin.getBytes()
- use for loop (25 * 400) times
- identify the hash type --> MessageDigest.getInstance("MD5");
- update PinByte --> كبير ان هي تجزءه byte دي بستسخدم لو حجم
- apply md5 hash for PinByte --> digest.clone()
- get a clone after encode --> (byte[]) digest.clone()

الخلاصة هنا هو بيستخدم for loop 400*25 مرة وكل مرة بيعمل encode by md5 hash for new value يعني كل hash value يروح يعملها hash تاني وهكذا لحد ما يخرج من for loop

بعد كده اخر قيمة لل hash هنحولها ل hex باستخدام الفانكشن اللي هو عملها create ويقارنها مع القيمة النهاءية اللي هي d04988522ddfed3133cc24fb6924eae9

```
        try {
            byte[] pinBytes = pin.getBytes();
            for (int i = 0; i < 25; i++) {
                for (int j = 0; j < 400; j++) {
                    MessageDigest md = MessageDigest.getInstance("MD5");
                    md.update(pinBytes);
                    byte[] digest = md.digest();
                    pinBytes = (byte[]) digest.clone();
                }
            }
            String hexPinBytes = toHexString(pinBytes);
            return hexPinBytes.equals("d04988522ddfed3133cc24fb6924eae9");
        }
```

دلوقتي بقي عرفنا ان قيمة hash لل PinCode الصح هي دي d04988522ddfed3133cc24fb6924eae9

3-analyze toHexString function

هنا هو بياخد قمية byte وبيمشي علي كل byte ويحولها ل hex

```
public static String toHexString(byte[] bytes) {
        StringBuilder hexString = new StringBuilder();
        for (byte b : bytes) {
            String hex = Integer.toHexString(b & 255);
            if (hex.length() == 1) {
                hexString.append('0');
            }
            hexString.append(hex);
        }
        return hexString.toString();
    }
```

هنا بقي خلينا نفهم method دي : دي بتاخد byte وتحوله ل integer وبعد كده بتقارنه مع 255 اللي هو علار عن 11111111 وتعمل and operation علشان لو دخلنا قيمة سالبة يحولها وهكذا

```
String hex = Integer.toHexString(b & 255)
```

Solution

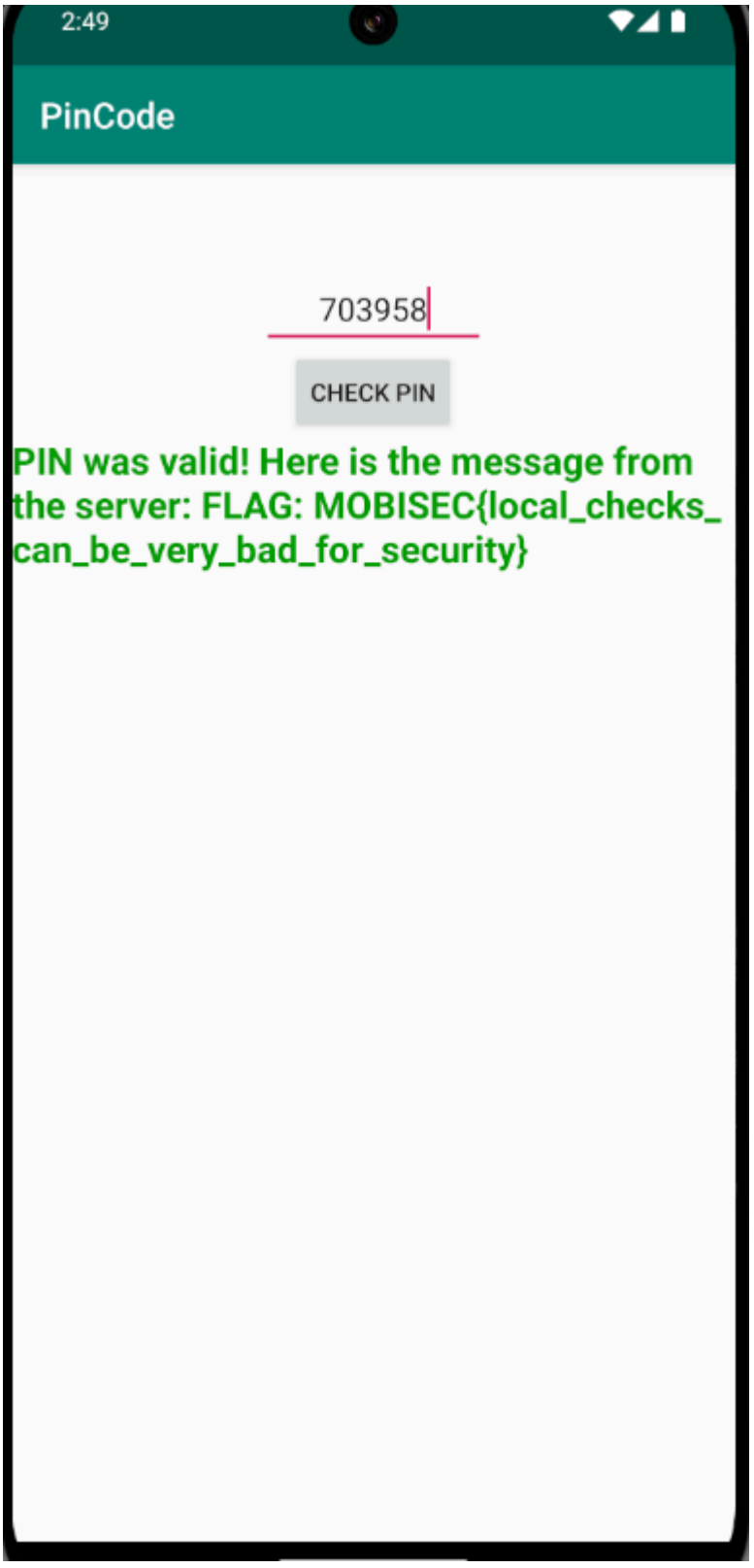
دلوقتي احنا عارفين ان الرقم عبارة عن 6 digit ف احنا ننشيء script ان هو ينشء ارقام من 000000 : 999999 ونعمل نفس الخطوات بتاع تشفير كل رقم زي ما هو عملها وشنوف القيمة اللي هطلع لو بتساوي القيمة دي d04988522ddfed3133cc24fb6924eae9 يبقي فعلا هو ده الرقم الصح

create python script

```
import hashlib
import itertools
result="d04988522ddfed3133cc24fb6924eae9"
for code in itertools.product(range(10),repeat=6):
    pincode=""
    for x in code:
        pincode+=str(x)
    pincode=pincode.encode()
    for i in range(0,400*25):
        md5=hashlib.md5()
        md5.update(pincode)
        pincode=md5.digest()
    if pincode.hex()==result:
        print(code)
        break
```

correct pin is --> **703958**

check pin



Flag is ---> **MOBISEC{local_checks_can_be_very_bad_for_security}**