Ex140 - MobileAppTest

Simon Tobon

2021-07-08

Contents

Technical Report

Introduction

For this exercise we were tasked to reverse engineer a Java Android Application and exfiltrate sensitive data. This was achieved with jadx and some decryption.

Finding: Description of finding

Vulnerability Description

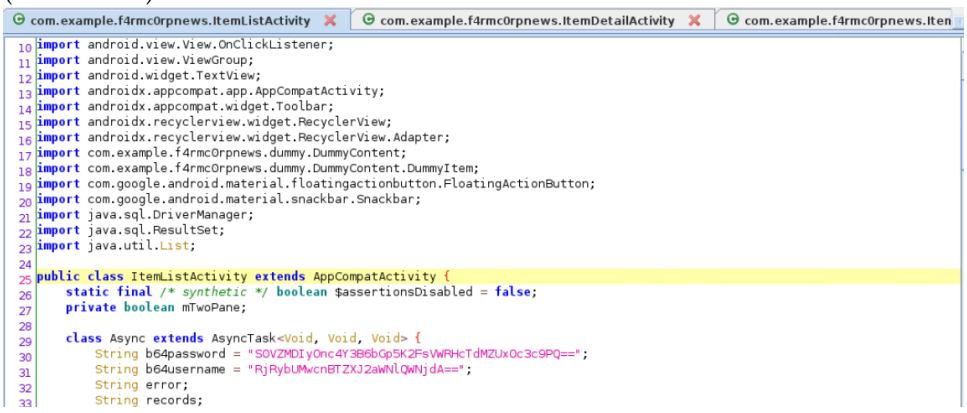
The risk arises from not following the OWASP recommendations for mobile app development. In this particular case, there was crucial data that should not have been included in the application source which led to easy exploitation and a breach of privacy. Specifically, login credentials were included in the source (not raw, but, encrypted).

Confirmation method

So to begin, I navigated to the APK file on F4rmC0rp's domain. This was downloaded and placed on the Kali desktop, from there I used Jadx to examine the apk's source. This was done by running the following command: **/usr/share/jadx/bin/jadx-gui /home/kali/Desktop/F4rmC0rp.apk.**

I then inspected the code in greater detail, the file of interest was **com.example.f4rmc0rpnews.ItemListActivity.java.**

In this file we can see that there are some credentials written into the source (hard coded)



From the name of the variables I guessed that these credentials were Base 64 encoded. I verified this by using the base64 decoding command on Kali:

- b64password

```
kali@kali:~$ echo -n "S0VZMDIyOnc4Y3B6bGp5K2FsVWRHcTdMZUx0c3c9PQ==" | base64 -d
KEY022:w8cpzljy+alUdGq7LeLtsw==kali@kali:~$
```

- b64username

```
kali@kali:~$ echo -n "RjRybUMwcnBTZXJ2aWNlQWNjdA==" | base64 -d
F4rmC0rpServiceAcctkali@kali:~$
```

From these results we obtained KEY022. This concluded the exercise.

Mitigation or Resolution Strategy

This could be easily avoided if the credentials were not set inside the application source, perhaps they could be set as an Environmental variable or on the backend, also do not use something as trite and trivial as Base 64 encryption, this can be easily decoded.