Finding Name: Insecure Android Manifest Configuration with exported components

Name	Team	Role	Project	Quality Assurance	Is this a re-tested Finding?
Abdulmajeed Alzahrani	PT	Senior Member	Gopher Guardian		

Was this Finding Successful?
Yes

Finding Description

Exported components in Android refer to activities, services, receivers, or providers that are accessible to other applications or system processes. These are defined in the app's AndroidManifest.xml file, often with the android:exported="true" attribute. If not secured properly, these components can pose significant security risks, allowing malicious applications to interact with them. Additionally, other misconfigurations such as android:debuggable="true" and usesCleartextTraffic="true" can further compromise the application's security by enabling unauthorized debugging or exposing sensitive data over insecure connections. Proper configuration and security measures are essential to mitigate these risks

Risk Rating

Impact: medium
Likelihood: Significant

Impact values				
Very Minor	**Minor**	**Significant**	**Major**	**Severe**
Risk that holds little to no impact. Will not cause damage and regular activity can continue.	Risk that holds minor form of impact, but not significant enough to be of threat. Can cause some damage but not enough to impede regular activity.	Risk that holds enough impact to be somewhat of a threat. Will cause damage that can impede regular activity but will be able to run normally.	Risk that holds major impact to be of threat. Will cause damage that will impede regular activity and will not be able to run normally.	Risk that holds severe impact and is a threat. Will cause critical damage that can cease activity to be run.

Likelihood				
Rare	**Unlikely**	**Moderate**	**High**	**Certain**
Event may occur and/or if it did, it	Event could occur occasionally and/or	Event may occur and/or happens.	Event occurs at times and/or	Event is occurring now and/or

happens in specific circumstances.	could happen (at some point)		probably happens a lot.	happens frequently.

Business Impact

Misconfigured exported components can expose sensitive data, allow malicious actions, and compromise the Guardian app's security. This increases the risk of data breaches, regulatory non-compliance, and reputational damage, potentially resulting in financial losses and reduced user trust.

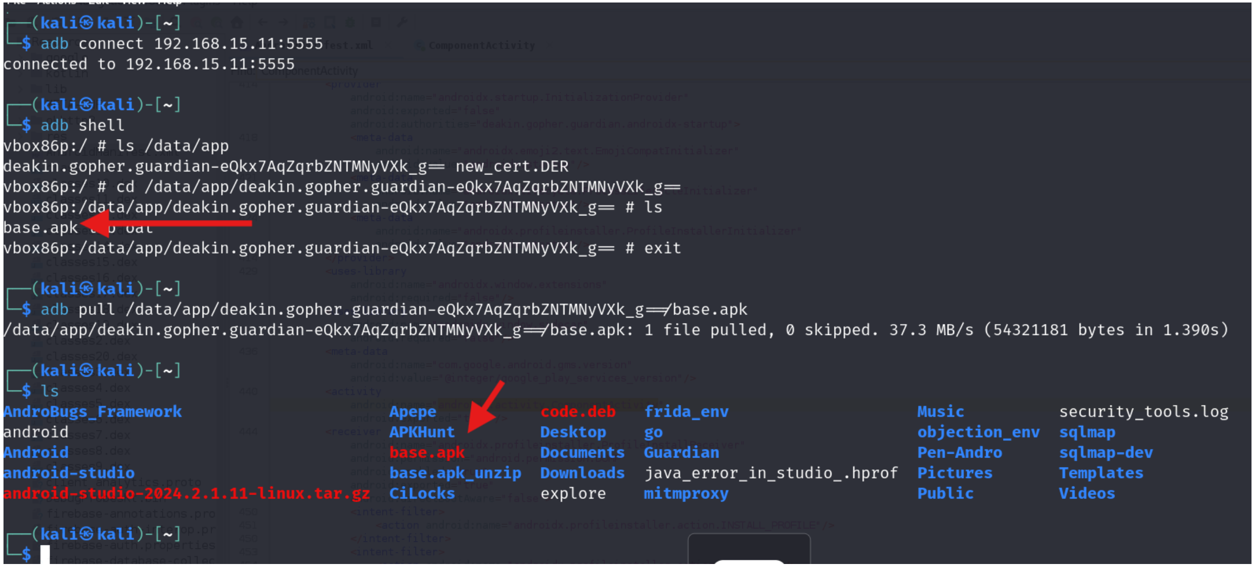
Affected Assets

Guadrian app.

Evidence

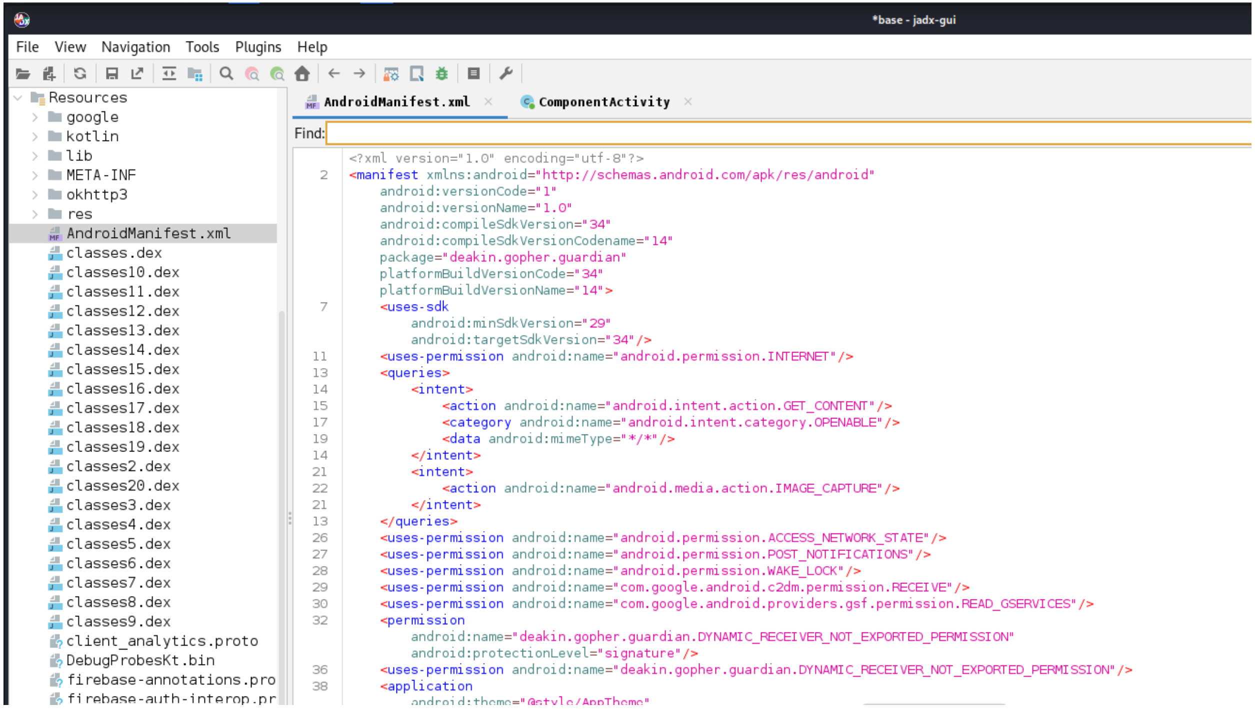
Provide a step by step guide on how to reproduce the vulnerability with screenshots

Step 1. Connect using abd and pull the application to our machine



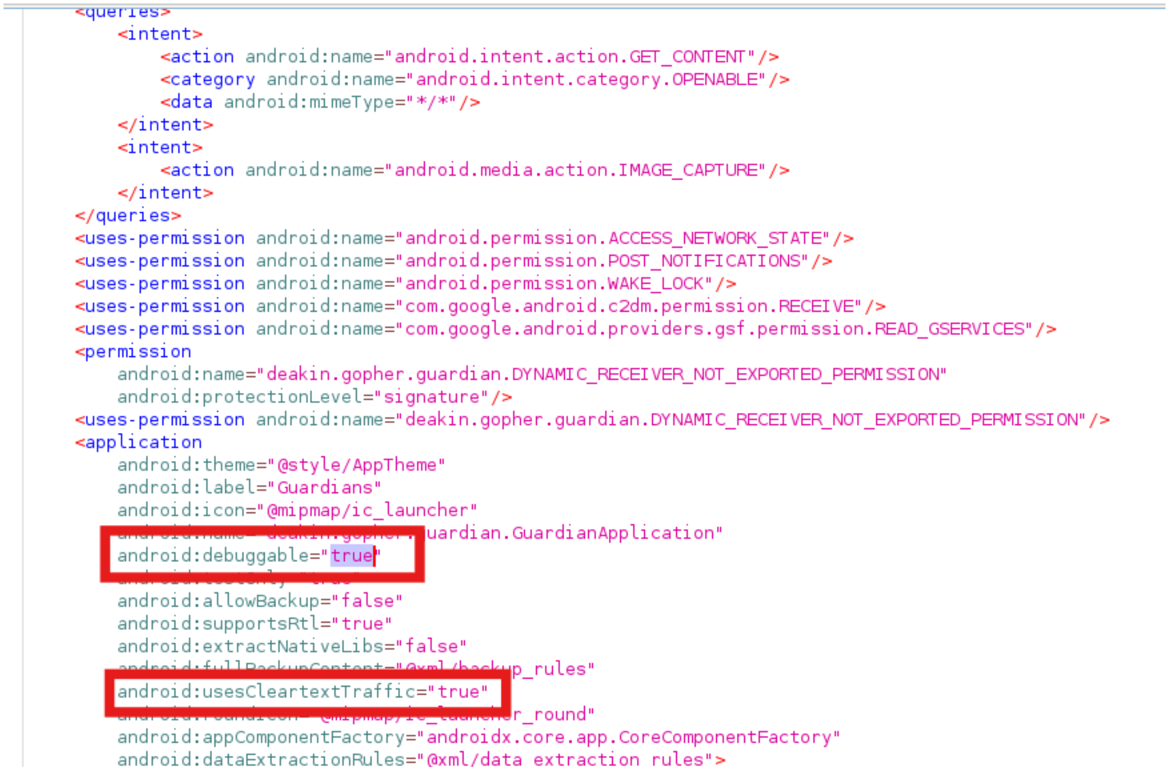
```
┌──(kali㉿kali)-[~]
└─$ adb connect 192.168.15.11:5555
connected to 192.168.15.11:5555

┌──(kali㉿kali)-[~]
└─$ adb shell
vbox86p:/ # ls /data/app
deakin.gopher.guardian-eQkx7AqZqrbZNTMNyVXk_g== new_cert.DER
vbox86p:/ # cd /data/app/deakin.gopher.guardian-eQkx7AqZqrbZNTMNyVXk_g==
vbox86p:/data/app/deakin.gopher.guardian-eQkx7AqZqrbZNTMNyVXk_g== # ls
base.apk lib oat
vbox86p:/data/app/deakin.gopher.guardian-eQkx7AqZqrbZNTMNyVXk_g== # exit

┌──(kali㉿kali)-[~]
└─$ adb pull /data/app/deakin.gopher.guardian-eQkx7AqZqrbZNTMNyVXk_g==/base.apk
/data/app/deakin.gopher.guardian-eQkx7AqZqrbZNTMNyVXk_g==/base.apk: 1 file pulled, 0 skipped. 37.3 MB/s (54321181 bytes in 1.390s)

┌──(kali㉿kali)-[~]
└─$ ls
AndroBugs_Framework                 Apepe            code.deb   frida_env                      Music          security_tools.log
android                             APKHunt          Desktop    go                             objection_env  sqlmap
Android                             base.apk         Documents  Guardian                       Pen-Andro      sqlmap-dev
android-studio                      base.apk_unzip   Downloads  java_error_in_studio_.hprof    Pictures       Templates
android-studio-2024.2.1.11-linux.tar.gz  CiLocks     explore    mitmproxy                      Public         Videos

┌──(kali㉿kali)-[~]
└─$ 
```

Step 2. Decompile the application using jadx



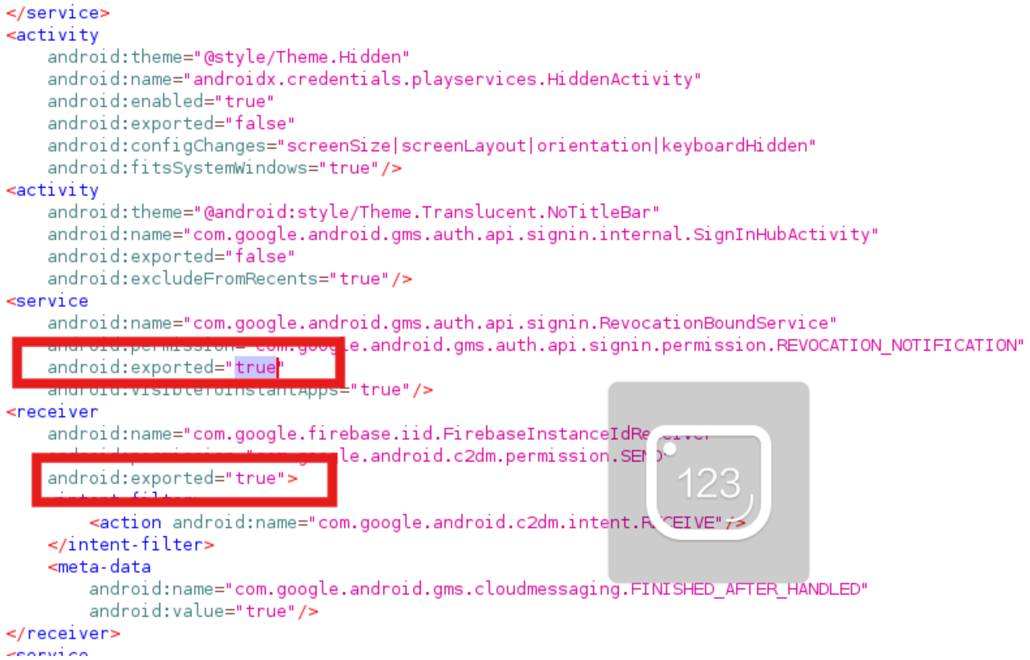
```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    android:versionCode="1"
    android:versionName="1.0"
    android:compileSdkVersion="34"
    android:compileSdkVersionCodename="14"
    package="deakin.gopher.guardian"
    platformBuildVersionCode="34"
    platformBuildVersionName="14">
    <uses-sdk
        android:minSdkVersion="29"
        android:targetSdkVersion="34"/>
    <uses-permission android:name="android.permission.INTERNET"/>
    <queries>
        <intent>
            <action android:name="android.intent.action.GET_CONTENT"/>
            <category android:name="android.intent.category.OPENABLE"/>
            <data android:mimeType="*/*"/>
        </intent>
        <intent>
            <action android:name="android.media.action.IMAGE_CAPTURE"/>
        </intent>
    </queries>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
    <uses-permission android:name="android.permission.POST_NOTIFICATIONS"/>
    <uses-permission android:name="android.permission.WAKE_LOCK"/>
    <uses-permission android:name="com.google.android.c2dm.permission.RECEIVE"/>
    <uses-permission android:name="com.google.android.providers.gsf.permission.READ_GSERVICES"/>
    <permission
        android:name="deakin.gopher.guardian.DYNAMIC_RECEIVER_NOT_EXPORTED_PERMISSION"
        android:protectionLevel="signature"/>
    <uses-permission android:name="deakin.gopher.guardian.DYNAMIC_RECEIVER_NOT_EXPORTED_PERMISSION"/>
    <application
        android:theme="@style/AppTheme"
```

Step 3. Explore the source code to find weakness points I find that the application has debbugable attribute set to true which is the first weakness point which allows the attacker to read our application source code ,usescleartexttraffice attribute set to true which is exploitable by man in the middle attack since the app sends clear text traffic



```xml
    <queries>
        <intent>
            <action android:name="android.intent.action.GET_CONTENT"/>
            <category android:name="android.intent.category.OPENABLE"/>
            <data android:mimeType="*/*"/>
        </intent>
        <intent>
            <action android:name="android.media.action.IMAGE_CAPTURE"/>
        </intent>
    </queries>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
    <uses-permission android:name="android.permission.POST_NOTIFICATIONS"/>
    <uses-permission android:name="android.permission.WAKE_LOCK"/>
    <uses-permission android:name="com.google.android.c2dm.permission.RECEIVE"/>
    <uses-permission android:name="com.google.android.providers.gsf.permission.READ_GSERVICES"/>
    <permission
        android:name="deakin.gopher.guardian.DYNAMIC_RECEIVER_NOT_EXPORTED_PERMISSION"
        android:protectionLevel="signature"/>
    <uses-permission android:name="deakin.gopher.guardian.DYNAMIC_RECEIVER_NOT_EXPORTED_PERMISSION"/>
    <application
        android:theme="@style/AppTheme"
        android:label="Guardians"
        android:icon="@mipmap/ic_launcher"
        android:name="deakin.gopher.guardian.GuardianApplication"
        android:debuggable="true"
        android:testOnly="true"
        android:allowBackup="false"
        android:supportsRtl="true"
        android:extractNativeLibs="false"
        android:fullBackupContent="@xml/backup_rules"
        android:usesCleartextTraffic="true"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:appComponentFactory="androidx.core.app.CoreComponentFactory"
        android:dataExtractionRules="@xml/data extraction rules">
```

Step.4. after deep analysis I found that there are exported components(service and receiver) which allow other applications to access our application data.



```
</service>
<activity
    android:theme="@style/Theme.Hidden"
    android:name="androidx.credentials.playservices.HiddenActivity"
    android:enabled="true"
    android:exported="false"
    android:configChanges="screenSize|screenLayout|orientation|keyboardHidden"
    android:fitsSystemWindows="true"/>
<activity
    android:theme="@android:style/Theme.Translucent.NoTitleBar"
    android:name="com.google.android.gms.auth.api.signin.internal.SignInHubActivity"
    android:exported="false"
    android:excludeFromRecents="true"/>
<service
    android:name="com.google.android.gms.auth.api.signin.RevocationBoundService"
    android:permission="com.google.android.gms.auth.api.signin.permission.REVOCATION_NOTIFICATION"
    android:exported="true"
    android:visibleToInstantApps="true"/>
<receiver
    android:name="com.google.firebase.iid.FirebaseInstanceIdReceiver"
    android:permission="com.google.android.c2dm.permission.SEND"
    android:exported="true">
    <intent-filter>
        <action android:name="com.google.android.c2dm.intent.RECEIVE"/>
    </intent-filter>
    <meta-data
        android:name="com.google.android.gms.cloudmessaging.FINISHED_AFTER_HANDLED"
        android:value="true"/>
</receiver>
<service
```

Remediation Advice

1. Use encryption techniques to send the application traffic .
2. Obfuscate the application so the attacker can not find the app source code easily
3. Disable unnecessary exported components

References

ADB
JADX

Contact Details

Abdulmajeed Alzahrani
S223844731@deakin.edu.au