

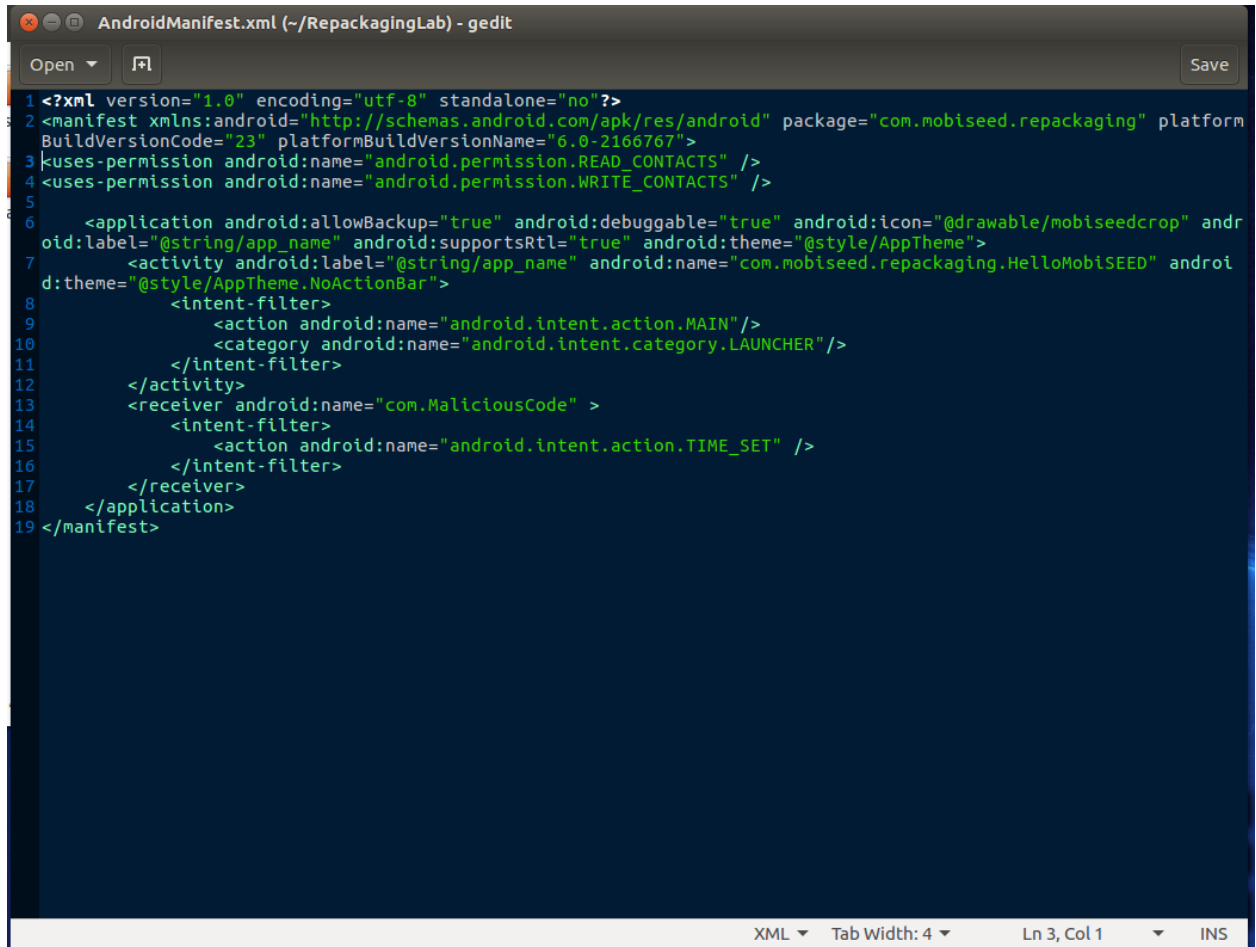


**UNC CHARLOTTE**  
**College of Computing and Informatics**  
Department of Software and Information Systems

Enterprise and Infrastructure Protection  
ITIS 6230

**PROJECT 1 – ANDROID REPACKAGING LAB**  
**SAHIL BHIRUD**

1. Screenshot of the xml file after edition.

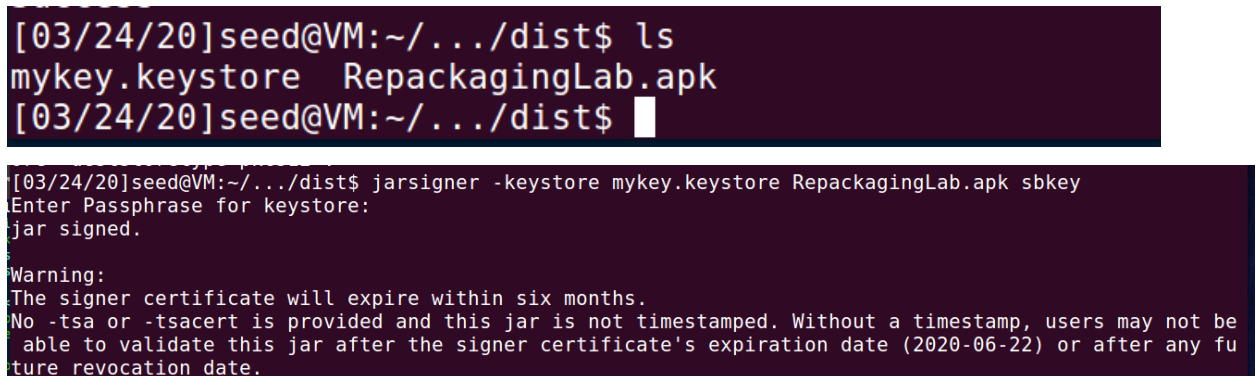


The screenshot shows a gedit editor window titled "AndroidManifest.xml (~/.RepackagingLab) - gedit". The editor contains the following XML code:

```
1 <?xml version="1.0" encoding="utf-8" standalone="no"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android" package="com.mobiseed.repackaging" platform
3 BuildVersionCode="23" platformBuildVersionName="6.0-2166767">
4 <uses-permission android:name="android.permission.READ_CONTACTS" />
5 <uses-permission android:name="android.permission.WRITE_CONTACTS" />
6
7 <application android:allowBackup="true" android:debuggable="true" android:icon="@drawable/mobiseedcrop" andr
8 oid:label="@string/app_name" android:supportsRtl="true" android:theme="@style/AppTheme">
9 <activity android:label="@string/app_name" android:name="com.mobiseed.repackaging.HelloMobiSEED" androi
10 d:theme="@style/AppTheme.NoActionBar">
11 <intent-filter>
12 <action android:name="android.intent.action.MAIN"/>
13 <category android:name="android.intent.category.LAUNCHER"/>
14 </intent-filter>
15 </activity>
16 <receiver android:name="com.MaliciousCode" >
17 <intent-filter>
18 <action android:name="android.intent.action.TIME_SET" />
19 </intent-filter>
20 </receiver>
21 </application>
22 </manifest>
```

The status bar at the bottom indicates "XML", "Tab Width: 4", "Ln 3, Col 1", and "INS".

2. Screenshot of the repackaging command execution procedure and the date/time of the new packaging file



The screenshot shows a terminal window with the following commands and output:

```
[03/24/20]seed@VM:~/../dist$ ls
mykey.keystore  RepackagingLab.apk
[03/24/20]seed@VM:~/../dist$

[03/24/20]seed@VM:~/../dist$ jarsigner -keystore mykey.keystore RepackagingLab.apk sbkey
Enter Passphrase for keystore:
jar signed.

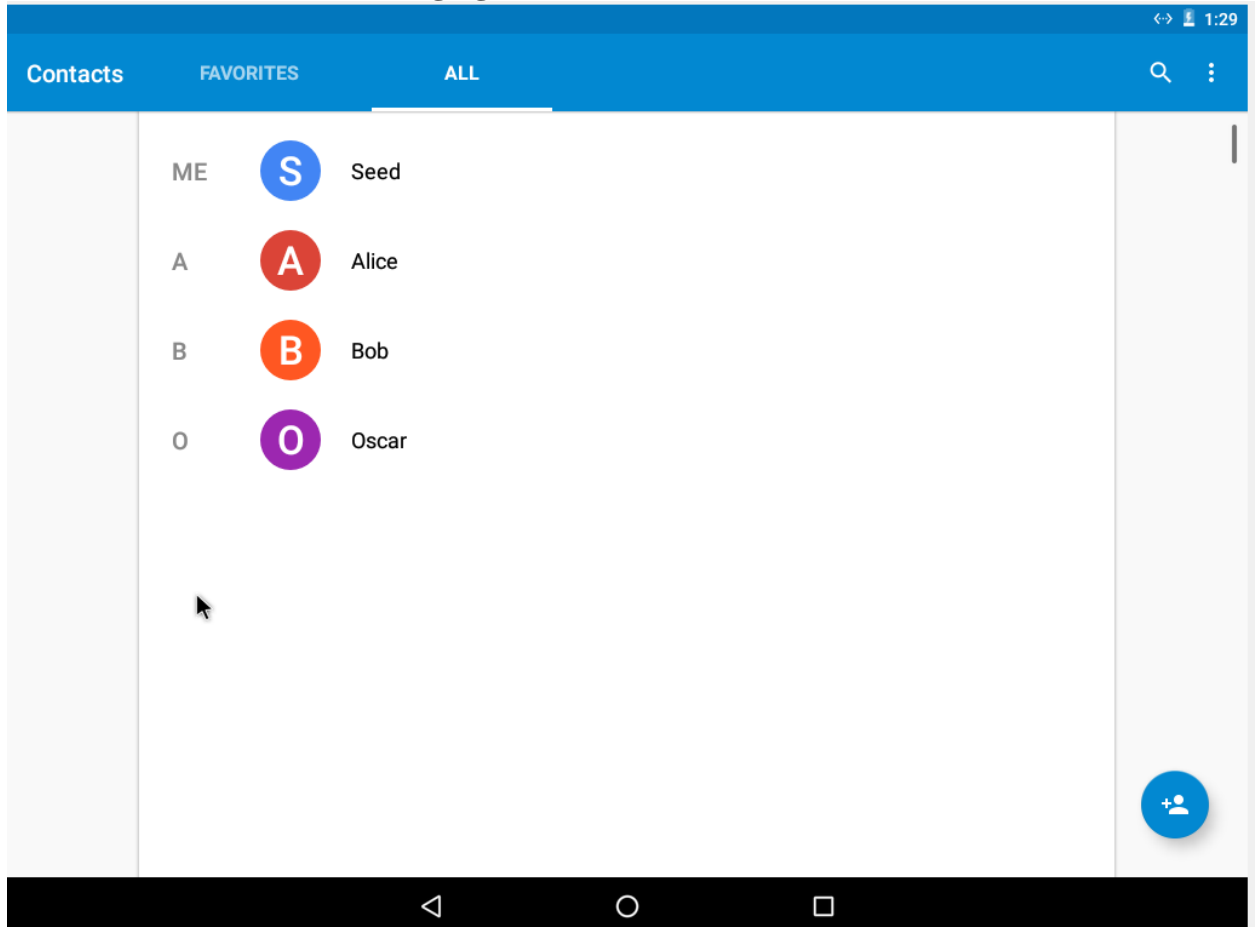
Warning:
The signer certificate will expire within six months.
No -tsa or -tsacert is provided and this jar is not timestamped. Without a timestamp, users may not be
able to validate this jar after the signer certificate's expiration date (2020-06-22) or after any fu
ture revocation date.
```

```
[03/24/20]seed@VM:~/../dist$ adb connect 10.0.2.13
connected to 10.0.2.13:5555
[03/24/20]seed@VM:~/../dist$ adb install RepackagingLab.apk
7217 KB/s (1427433 bytes in 0.193s)
Success
[03/24/20]seed@VM:~/../dist$
```

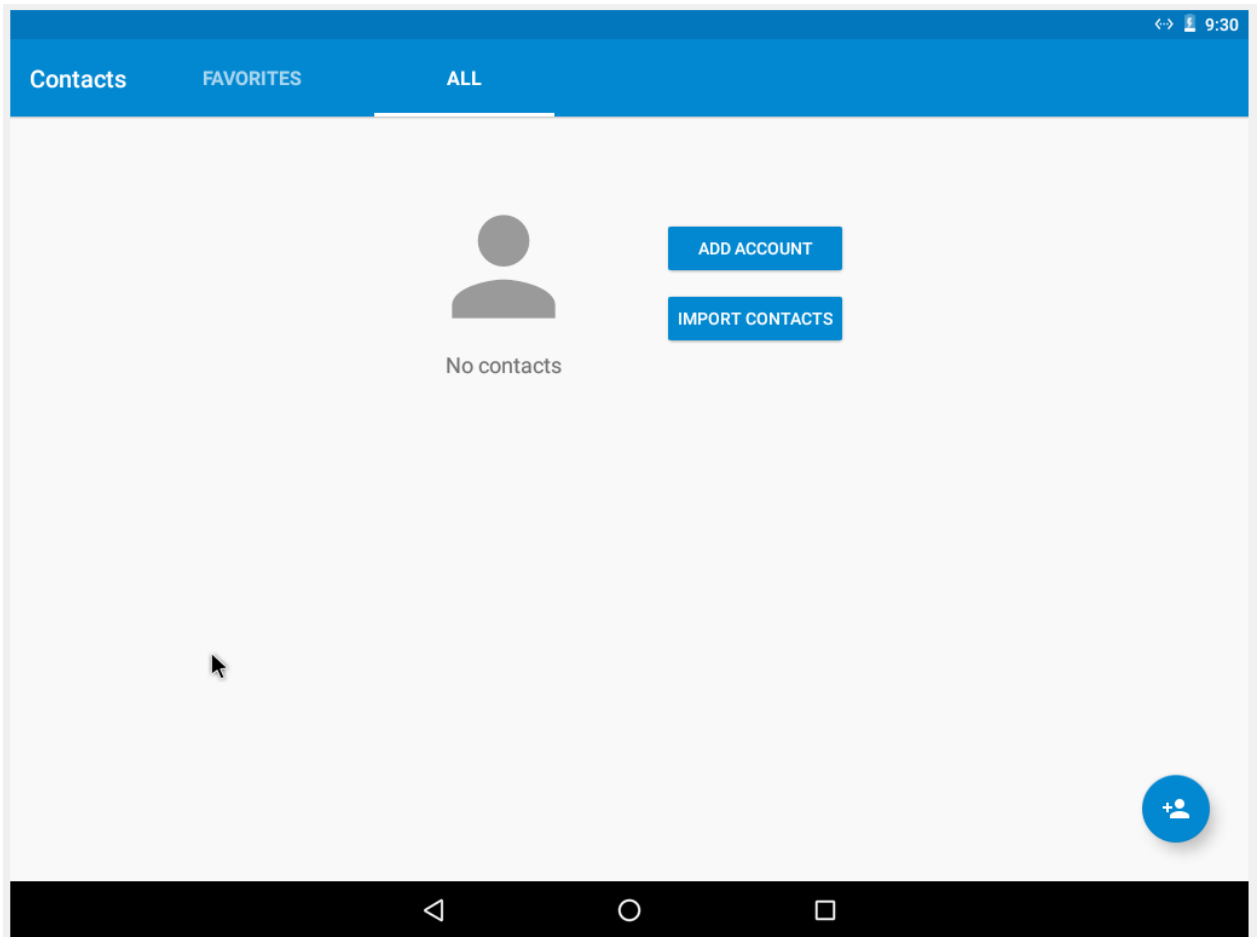
S

```
Terminal
[03/25/20]seed@VM:~/../dist$ ls -l
total 1400
-rw-rw-r-- 1 seed seed 1962 Mar 24 13:32 mykey.keystore
-rw-rw-r-- 1 seed seed 1427433 Mar 24 13:32 RepackagingLab.apk
[03/25/20]seed@VM:~/../dist$
```

### 3. Contact list before reboot/changing date and time:



Contact list after reboot/changing date and time:



### Answer 1:

Yes, I do think that an attacker can insert or substitute people and their phone numbers in the contact list using Android Repackaging. The attacker can add a new component and edit the smali code and add code which will allow him to remotely edit the contacts app and then repackage it and make it available for download. According to me, the attacker will also require network access which will help him in accessing the device from a remote location. To do so, he must edit the **AndroidManifest.xml** file and add a line **<uses-permission android:name="android.permission.INTERNET"/>**.

### Answer 2:

I don't think that prevention of self-signing can completely eliminate the threat of malicious apps. Even if self-signing is discarded, there are other ways through which malicious apps can deceive their way on a victim's device. According to the researchers at Bitdefender Labs, there specific

techniques which are used by attackers to bypass security filters implemented by Google. Some of those are:

- **Logic encryption and obfuscated coding** in which the developers exclude the app's main logic and import it from a remote executable dynamic library when the app is run on the phone.
- **Open-source utility libraries** such as Dropbox are used by attackers to pull and run jobs in the background instead of an Android API.
- **Clean SDK's (initially)** and then later replacing them with the ones with malicious code. This is generally done in the form of periodic updates and changing configuration of apps.

### **Answer 3:**

#### **Advantages of taking control of a cell phone:**

Mobile phones can be used to control various IoT devices such as smart thermostats, motion activated security cameras, security system, lighting, smart keys, etc. If the phone is compromised, then the attacker will get access to these devices through which he can bypass the security system of the house and easily break in and since almost all personal information is stored on a person's cell phone, the attacker also gets hands on the victim's Personally Identifiable Information (PII).

#### **Disadvantages of taking control of a cell phone:**

Since cell phones are constantly used by individuals, there is a higher chance that they will be able to detect unusual behavior of their phones. It won't require them much time to find out what and where is the problem. Thus, making the attack unsuccessful.

#### **Advantages of taking control of a Smart Hub:**

If an attacker wants to compromise a smart home then, the best place to attack will be the smart hub. Smart Hub is a device which controls communication between all the smart home components which also means it is also the single point of failure for the smart home network. Taking the smart hub down will result in taking over control of the entire house.

#### **Disadvantages of taking control of a Smart Hub:**

If the victim is constantly changing passwords, then it will be difficult for the attacker to break into the smart hub. Furthermore, if the user has disabled unused features, enabled two-step authentication and kept his software updated with all the security patches then it will limit attacking places for the attacker, keeping the smart hub secure.

**REFERENCES:**

1. <https://www.bitdefender.com/files/News/CaseStudies/study/290/Bitdefender-WhitePaper-Android-Dozens-of-Apps-Still-Dodging-Googles-Vetting-System.pdf>