Android Repackaging Attack Lab

Task 1: Obtain an Android App (APK file) and Install It

Step 1: Run *ifconfig* command in Android VM to know its ip address, we know following are its IP address. **Android VM:** 10.0.2.4

```
x86_64:/ $ ifconfig
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope: Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:0 TX bytes:0

eth0      Link encap:Ethernet  HWaddr 08:00:27:c9:e5:b1
          inet addr:10.0.2.4  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fec9:e5b1/64 Scope: Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:34718 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8440 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:51791134 TX bytes:671037

x86_64:/ $
```

Step 2: Check if any device is connected or not by command

adb devices

it returns no device record means no device is connected.

Step 3: Connect our Linux Ubuntu Machine with Android Machine to install app by using command:

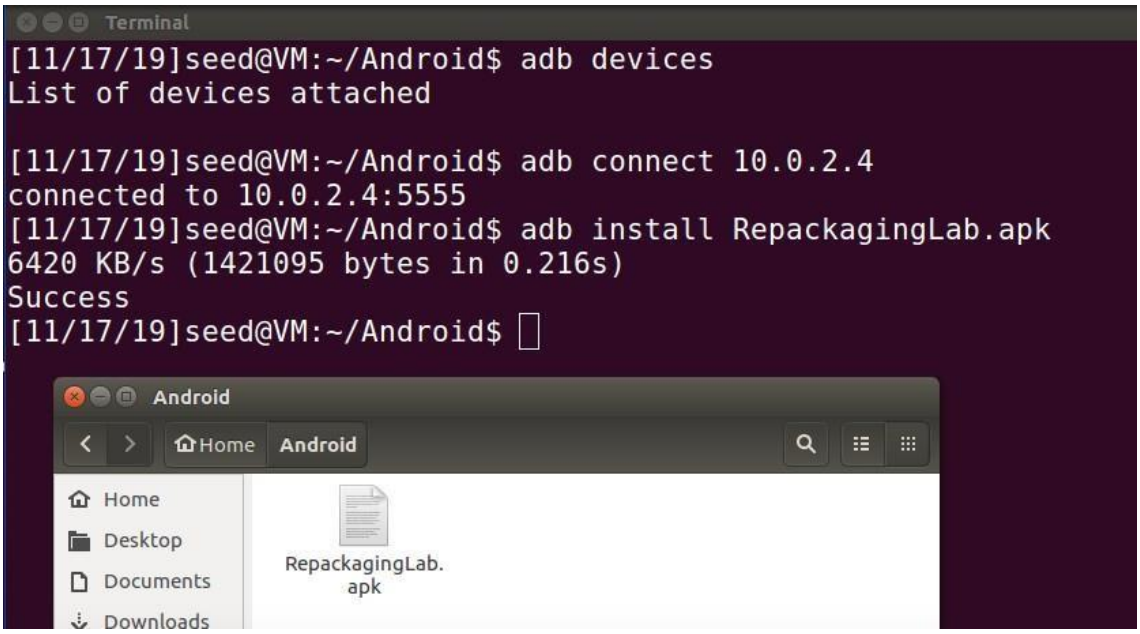
adb connect 10.0.2.4

The output of it is *Connected to 10.0.2.4:555* means our connection between two devices is successfully establish.

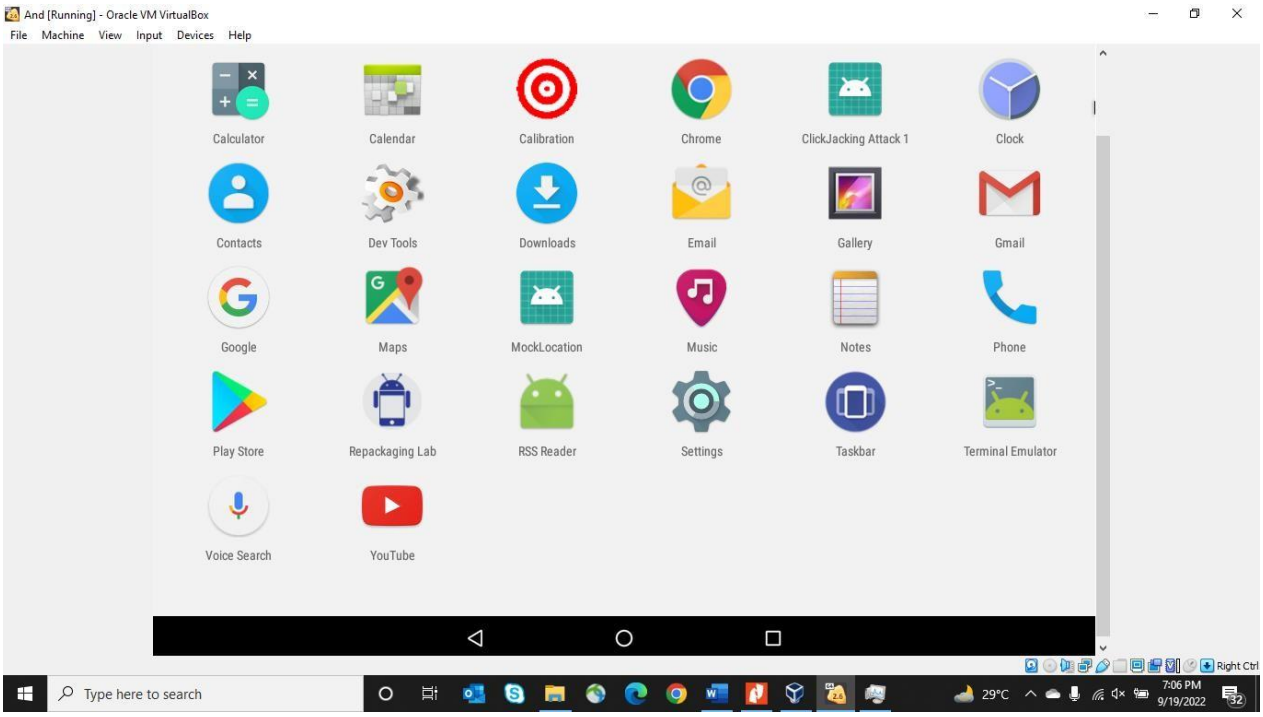
Step 4: To install app **RepackagingLab** on android device remotely we use command:

adb install RepackagingLab.apk

The result shows *success* means successfully installed the apk file in android



Step 4: Swipe up apps menu to confirm that app exist in android device which we installed.

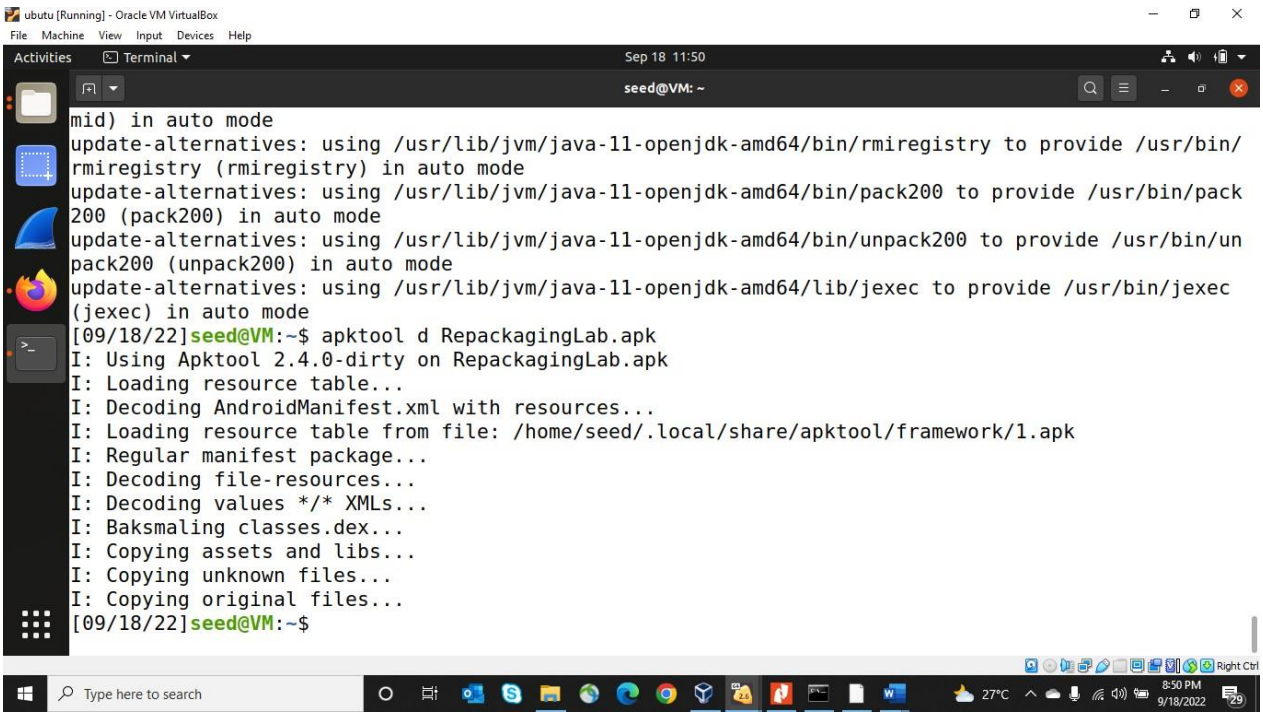


Point to be Noted: We get Repackaging.apk from seeds website and saved them in Android Folder.

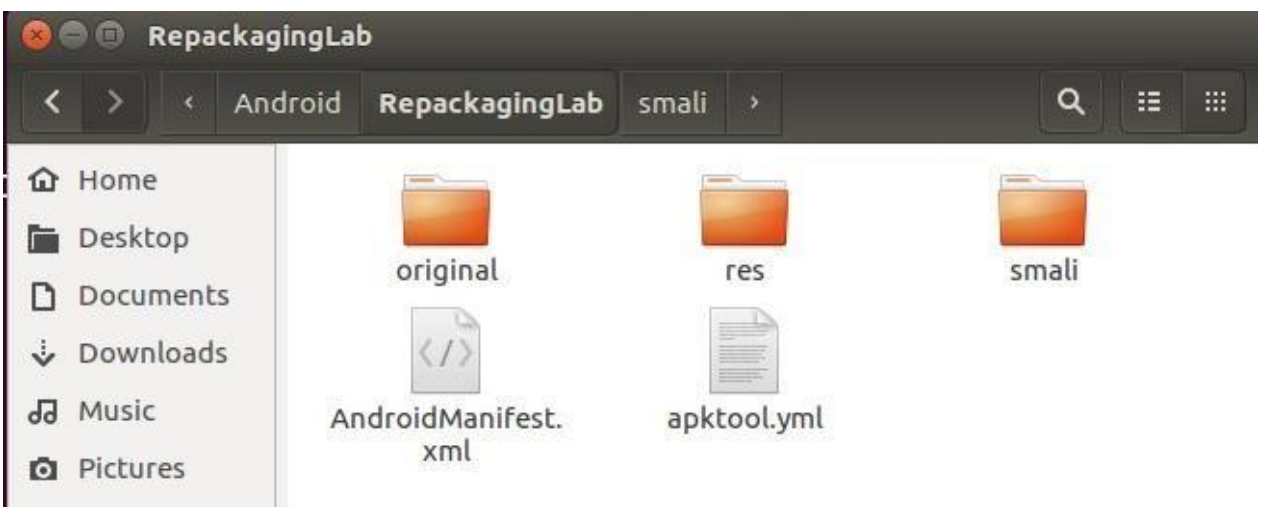
Task 2: Disassemble Android App

Step 1: Disassemble Android apk to inject malicious code inside it by using command:

apktool d RepackagingLab.apk



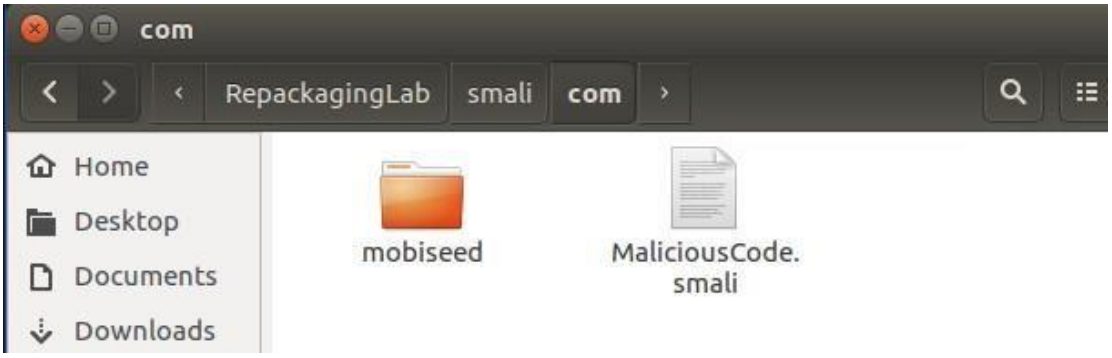
Step 2: Open *RepackagingLab* and we can see a following files.



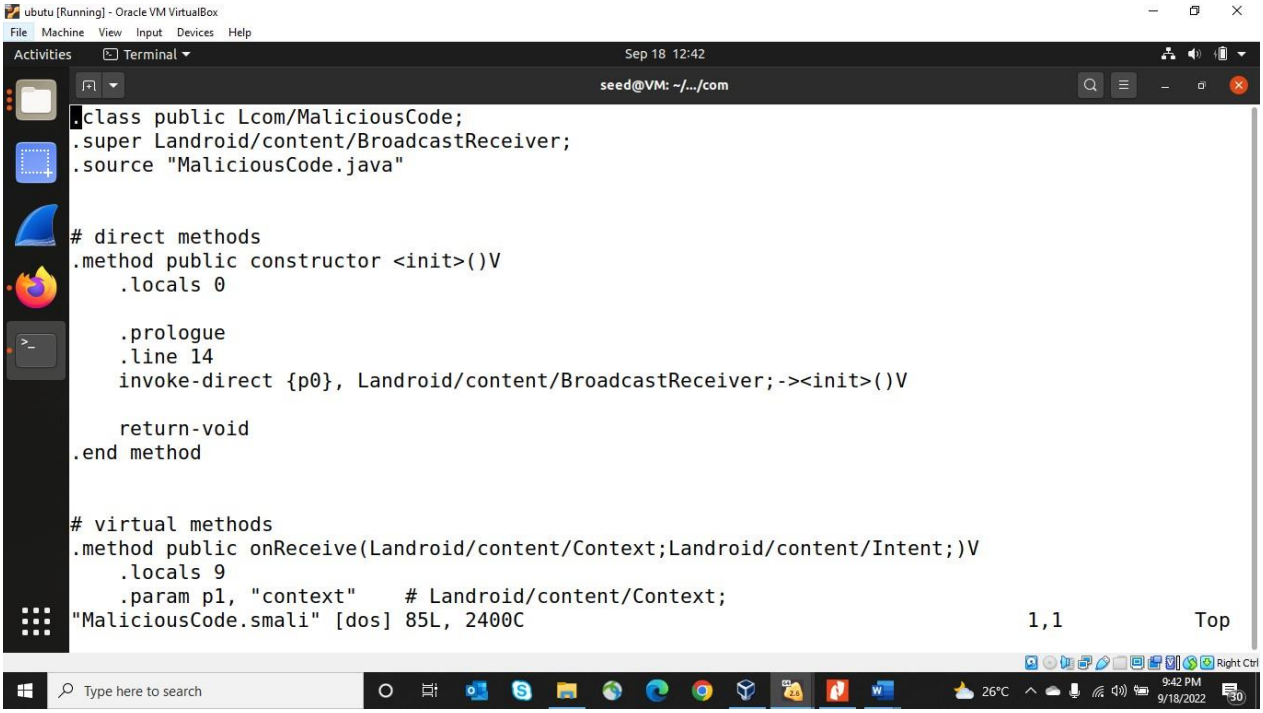
Point to be Noted: *smali code* can be understood by humans

Task 3: Inject Malicious Code

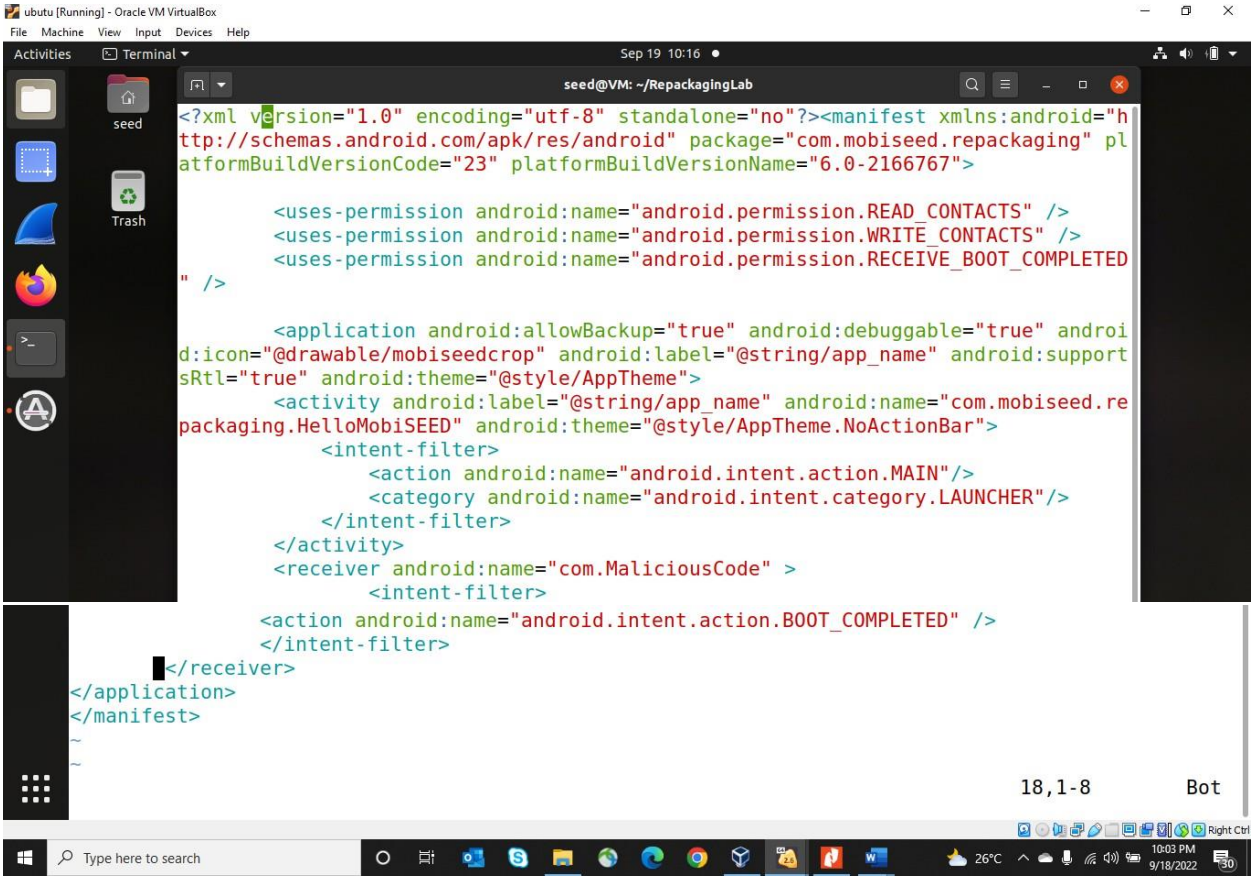
Step 1: Put *Malicious Code smali* in com folder



Below is code inside malicious file



Step 2: Insert permissions to read or write contacts, broadcast receiver to trigger command



Task 4: Repack Android App with Malicious Code

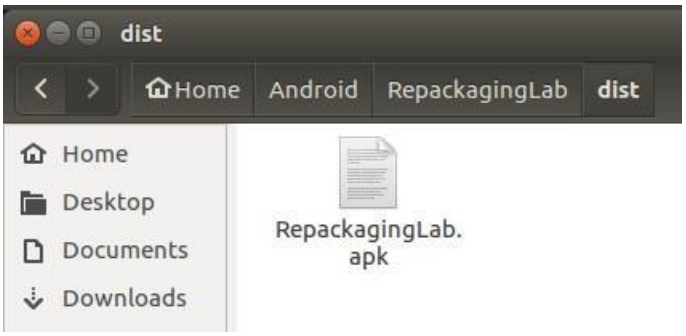
Step 1: After inserting malicious file we rebuild apk using command:

apktool b RepackagingLab

```
[09/18/22]seed@VM:~/RepackagingLab$ ls
AndroidManifest.xml  AndroidManifest.xml-backuo  apktool.yml  original  res  smali
[09/18/22]seed@VM:~/RepackagingLab$ vi AndroidManifest.xml

[1]+  Stopped                 vi AndroidManifest.xml
[09/18/22]seed@VM:~/RepackagingLab$ vi AndroidManifest.xml
[09/18/22]seed@VM:~/RepackagingLab$ cd ..
[09/18/22]seed@VM:~$ ls
Desktop    Downloads  Pictures  RepackagingLab      RepackagingLab.apk.zip  Videos
Documents  Music      Public    RepackagingLab.apk  Templates
[09/18/22]seed@VM:~$ apktool b RepackagingLab
I: Using Apktool 2.4.0-dirty
I: Checking whether sources has changed...
I: Smaling smali folder into classes.dex...
I: Checking whether resources has changed...
I: Building resources...
W: aapt: brut.common.BrutException: brut.common.BrutException: Could not extract resource: /prebui
lt/linux/aapt_64 (defaulting to $PATH binary)
I: Building apk file...
I: Copying unknown files/dir...
I: Built apk...
[09/18/22]seed@VM:~$
```

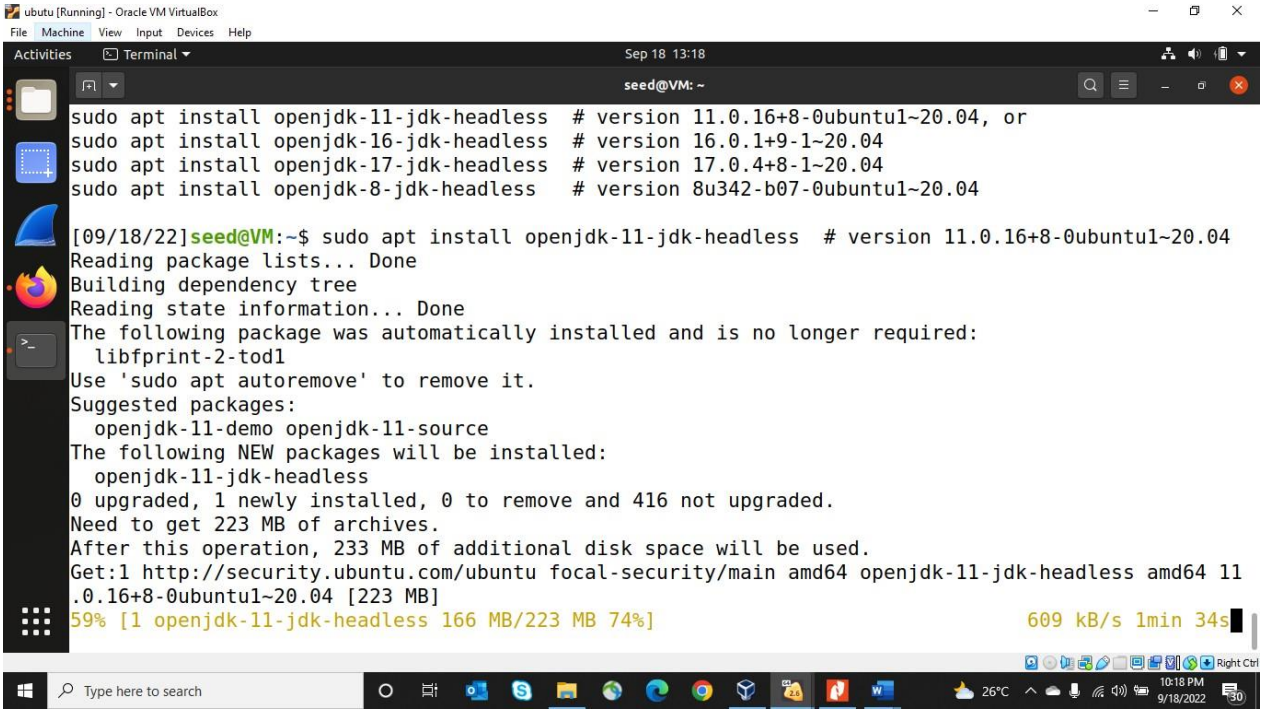
We get RepackagingLab.apk file in dist folder of RepackagingLab



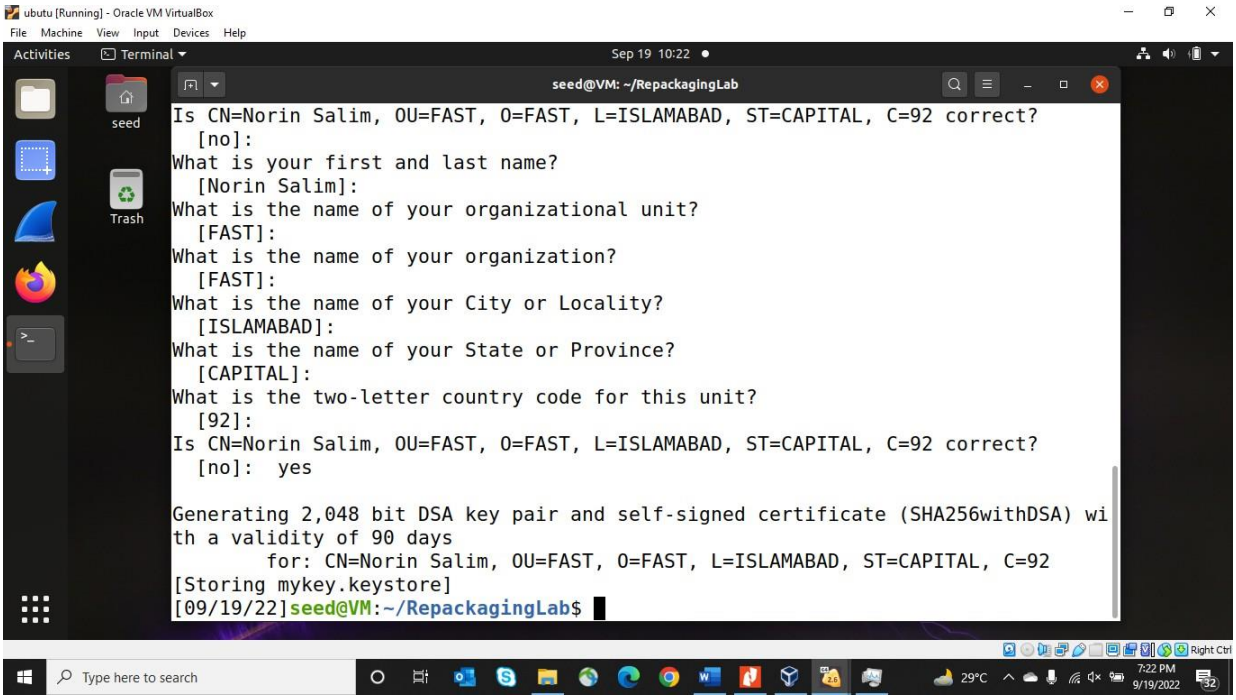
Step 2: Create public and private key with *keytool* command

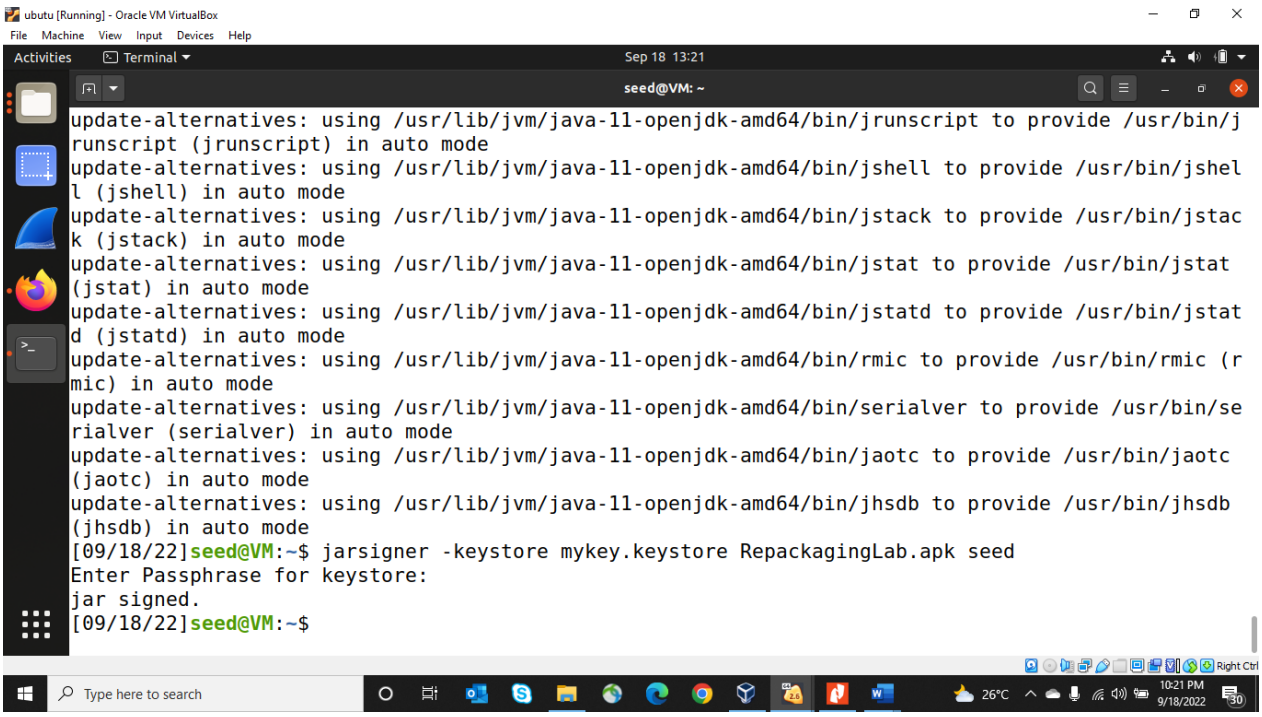
```
[11/17/19]seed@VM:~/Android$ keytool -alias RepackagingLab -genkey -v -keystore
mykey.keystore
Enter keystore password:
Keystore password is too short - must be at least 6 characters
Enter keystore password:
Re-enter new password:
```

Step 3: Download openjdk to run jarsigner



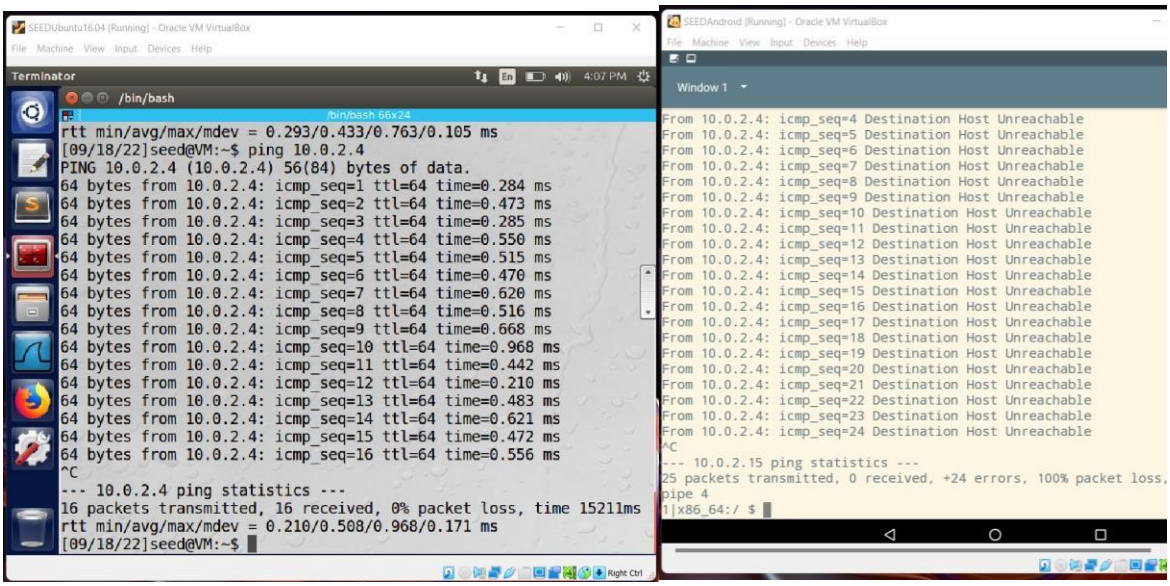
Step 4: Sign apk using previously created keys with jarsigner





Task 5: Install the Repackaged App and Trigger the Malicious Code

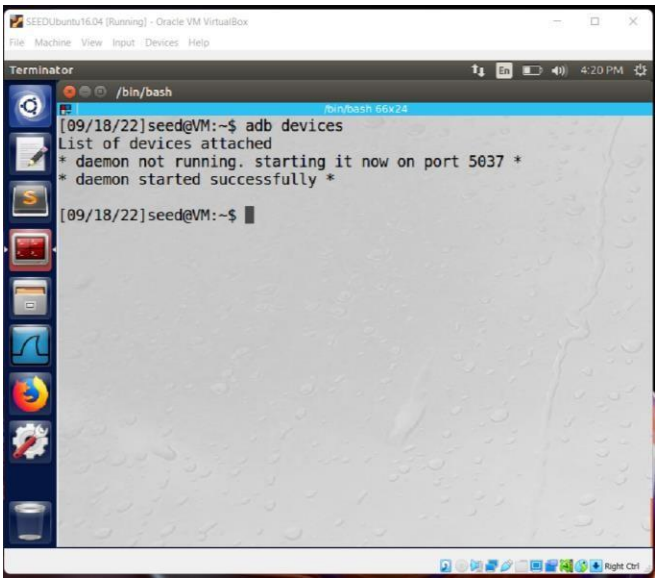
Step 1: Ping from both devices to check still devices are connected and result is false



Step 2: To check device connection there is another command:

adb devices

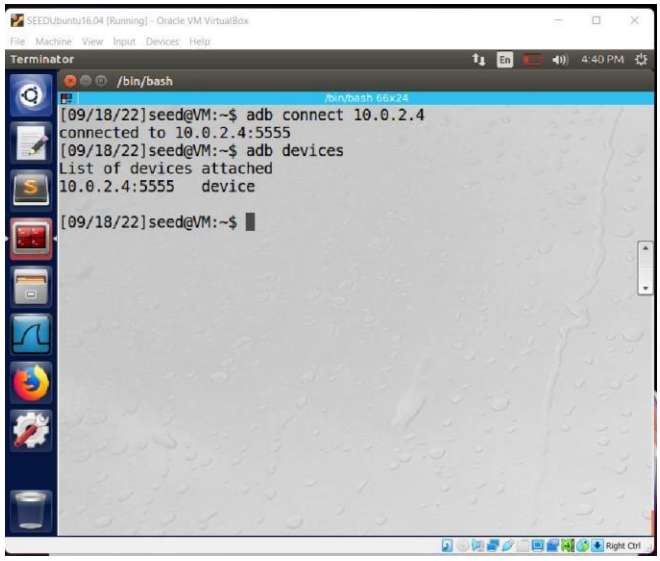
which shows any connected device but it returns nothing means no device is connected with **UBUNTU VM**



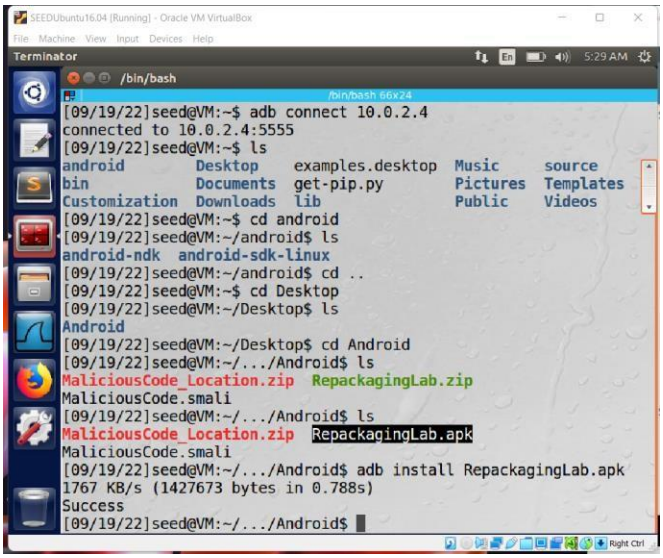
Step 3: Connect android and ubuntu vm by command:

adb connect 10.0.2.4

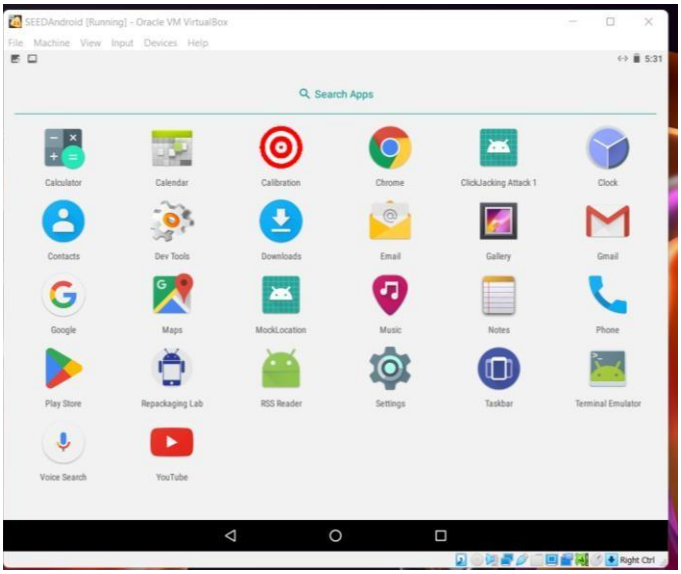
we can confirm connection from result we get list of attached devices



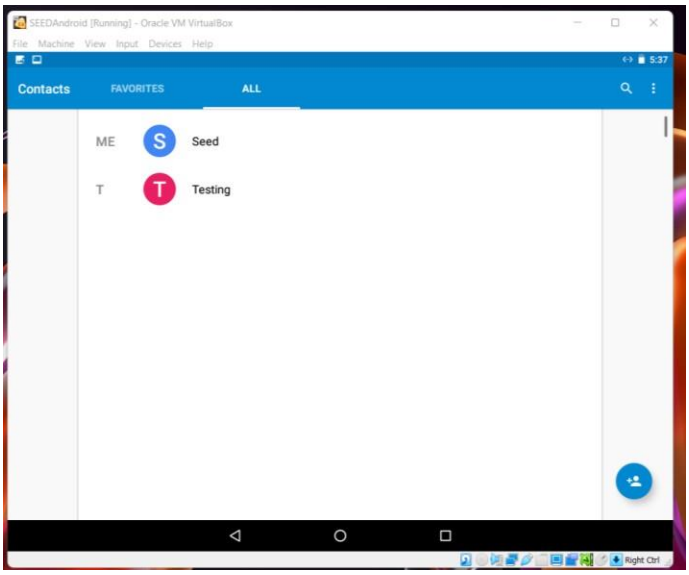
Step 4: Repeat Task 1 to Task 4 so we can install application and do attack



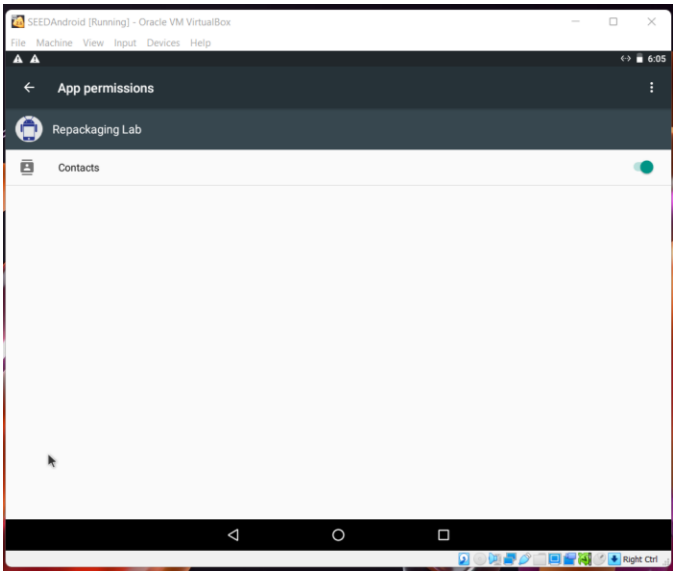
Proof that app install successfully



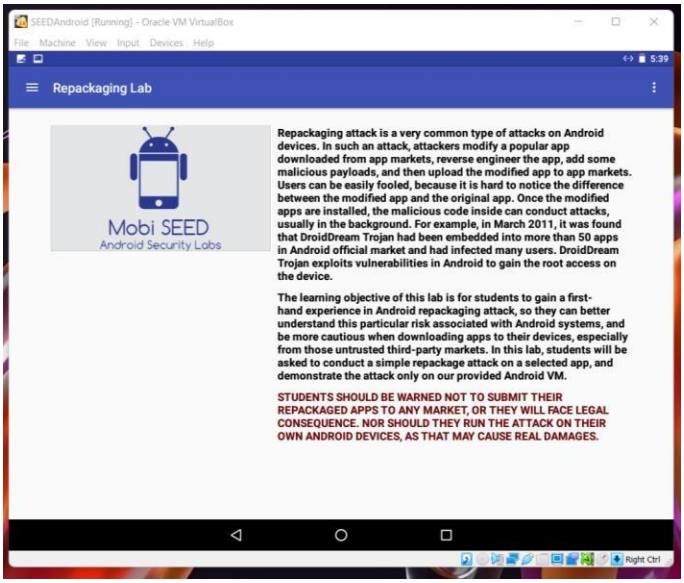
Step 5: We store some contacts for demo to check our code works properly or not.



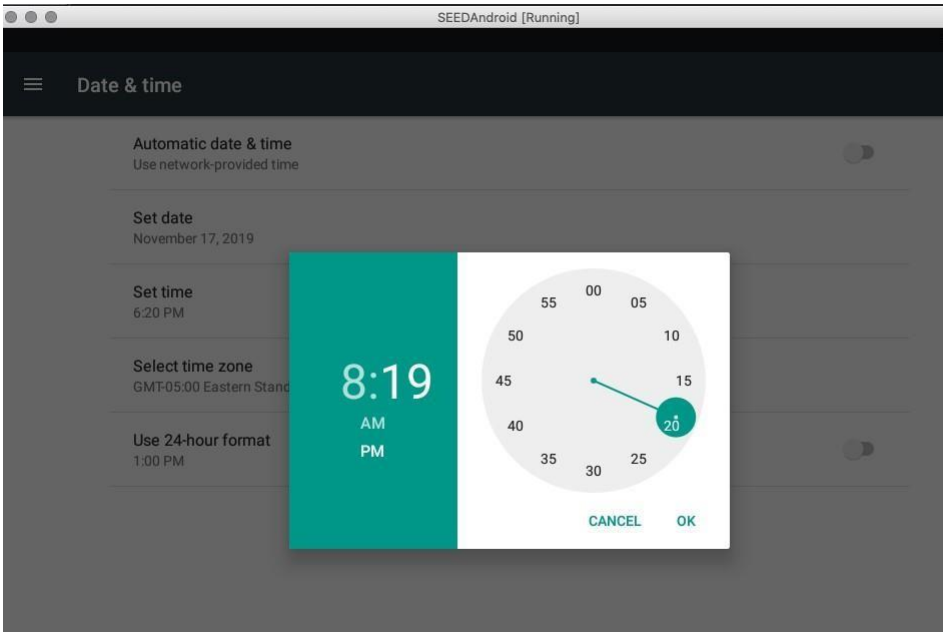
Step 6: Before running Repackaging app we will give it permission to read and write contacts because by default it has no permission.



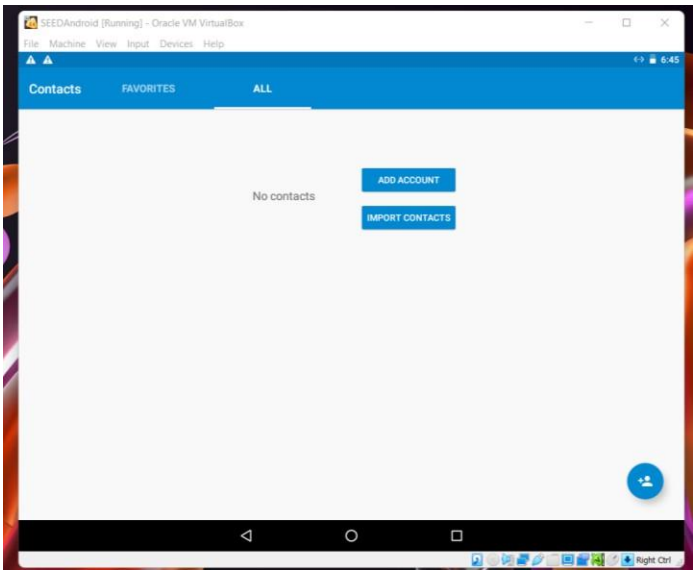
Step 7: Run the app once to execute malicious code



Step 8: Change time to activate malware



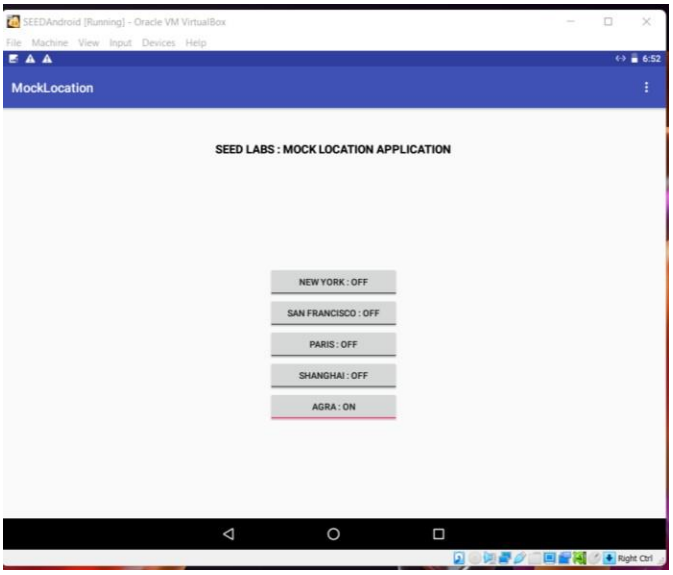
Step 9: Open contacts app and we can see all contacts are deleted and malware do its work well.



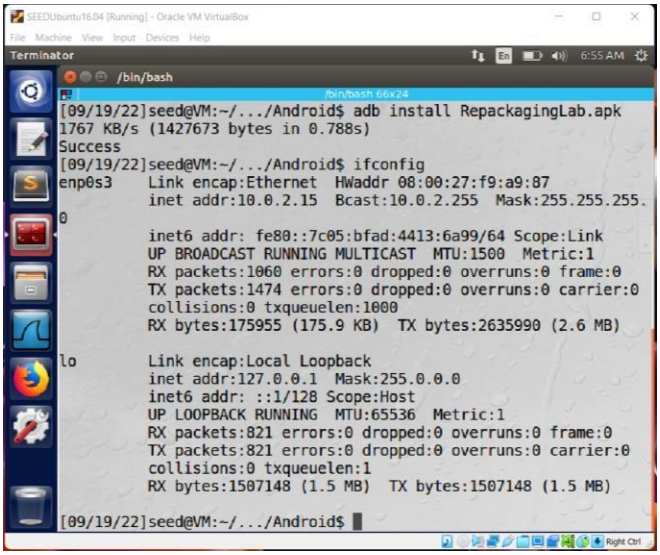
Point to be Noted: We first uninstall the app before installing otherwise signature mismatch error occur. App should run once to work fine.

Task 6: Using Repackaging Attack to Track Victim's Location

Step 1: Due to hardware limitation of GPS in VM, we install mock location app in android



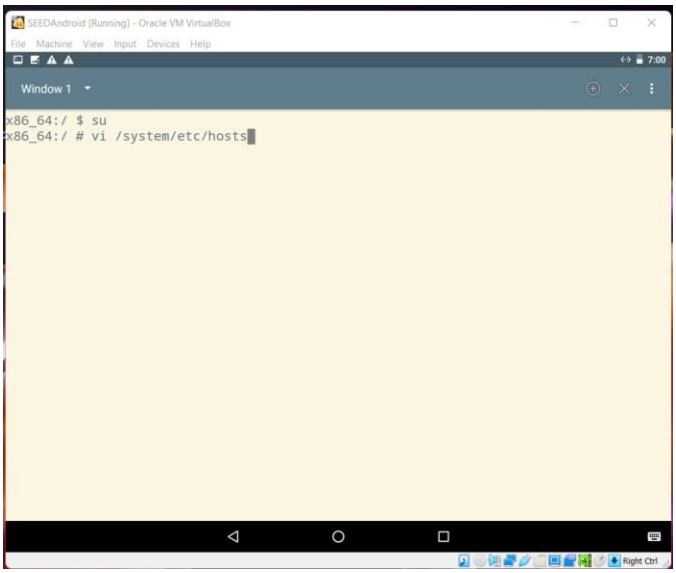
Step 2: Check Ip address of UBUNTU Machine which will we use later with command *ifconfig*, UBUNTU Machine ip: 10.0.2.15



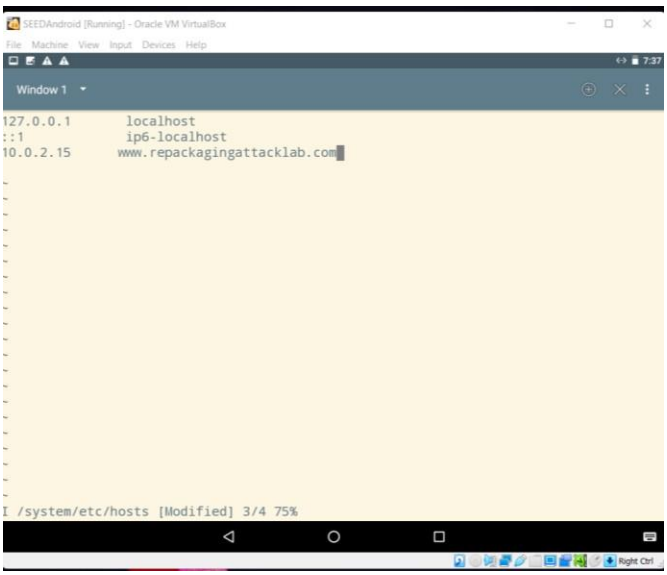
Step 3: In Android VM get super user access first and then open the host file to append,

su

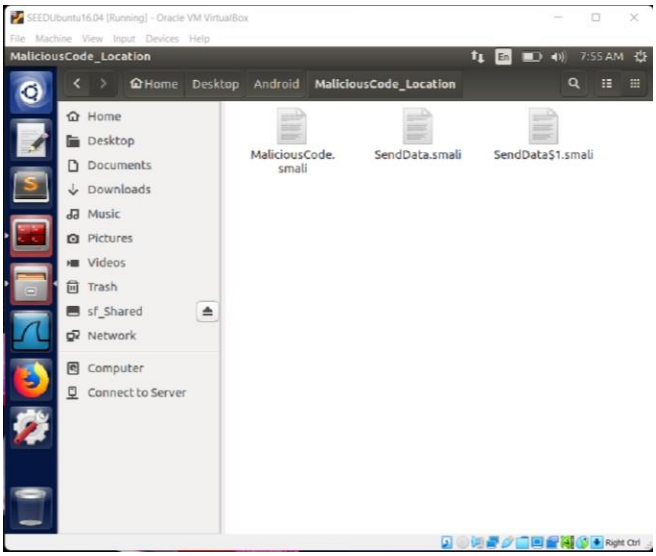
vi /system/etc/hosts



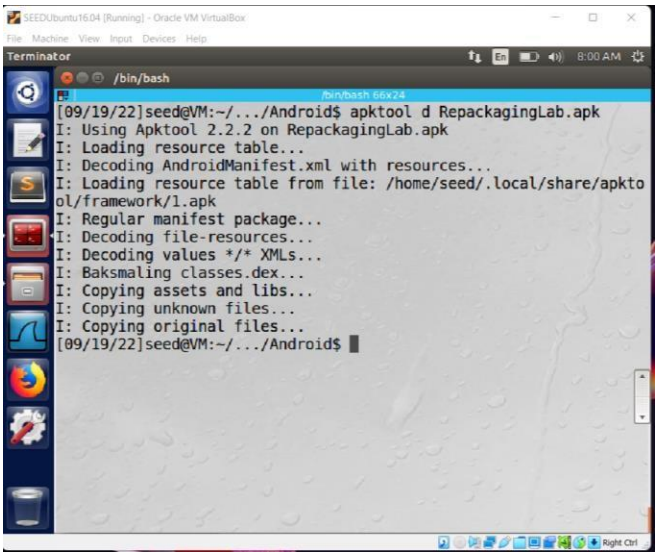
Step 4: Add UBUNTU ip address with www.repackagingattacklab.com so future updates we can see there



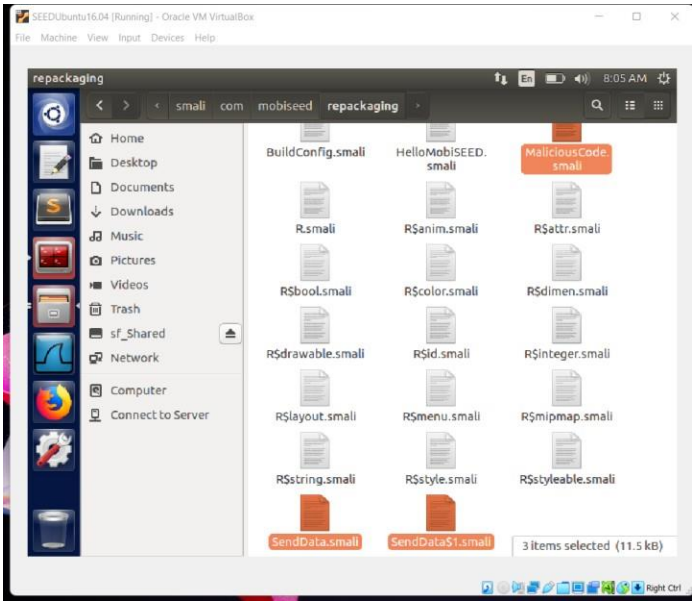
Step 5: Unzip Location malicious code files, MaliciousCode.smali, SendData$1.smali, and SendData.smali.



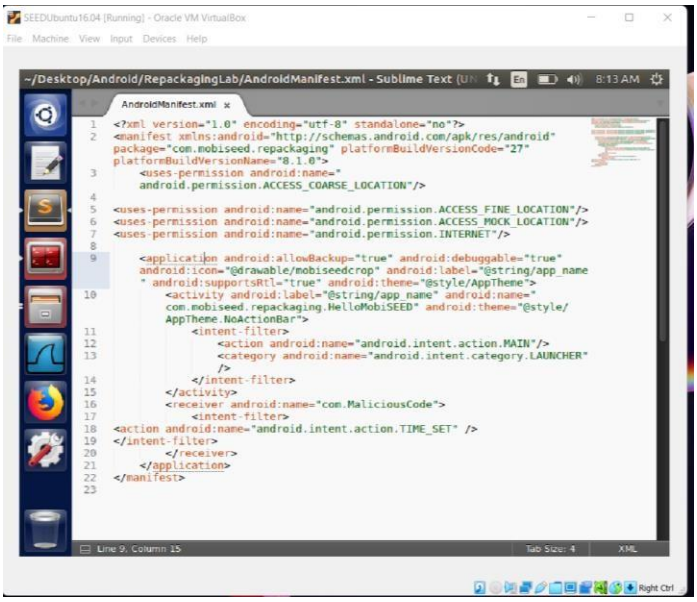
Step 6: Unpack Repackaging.apk file



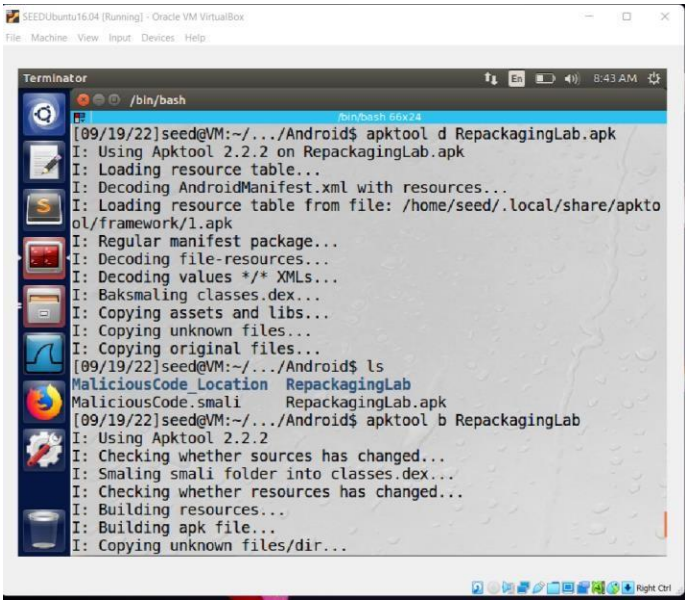
Step 7: Place three malicious files MaliciousCode.smali, SendData$1.smali, and SendData.smali which we got before by unzipping file



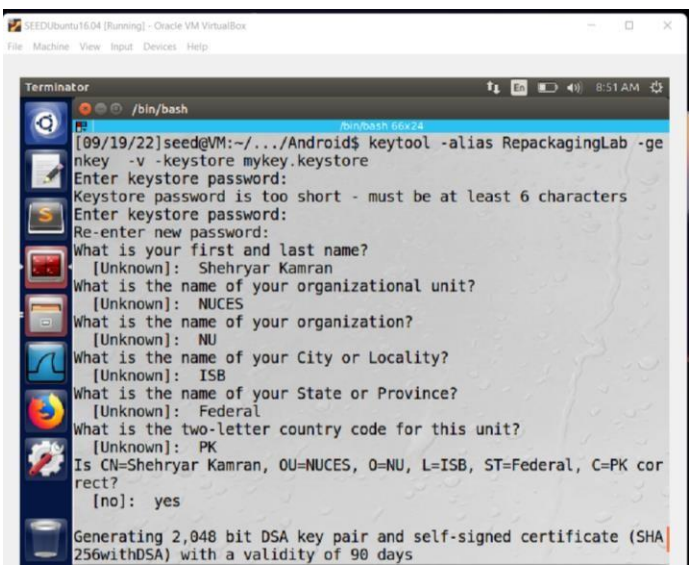
Step 8: Modify Manifest.xml file by adding permissions for internet and location access.

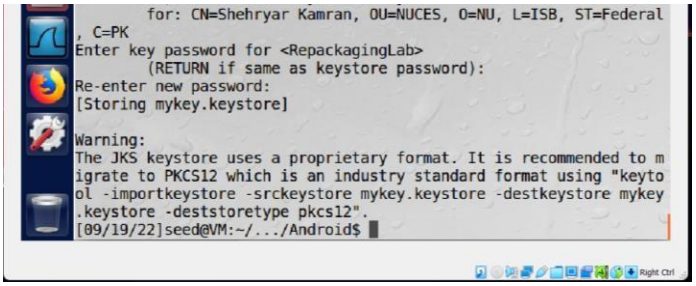


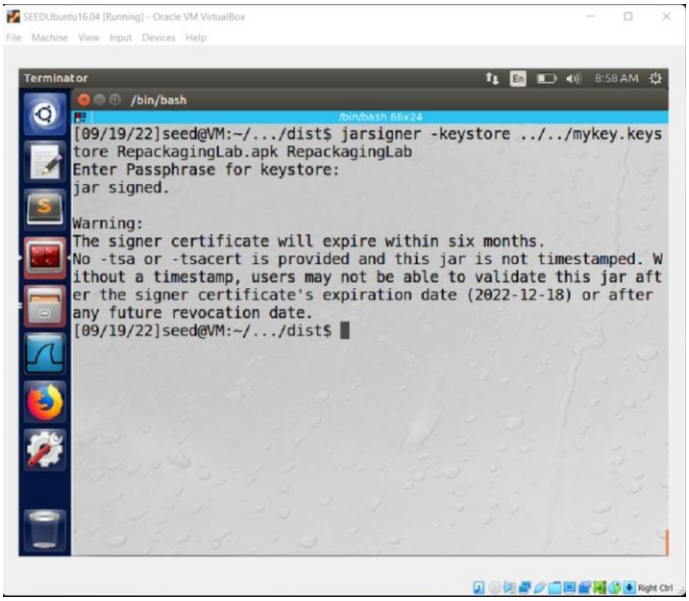
Step 9: Repack android apk file like we did before.



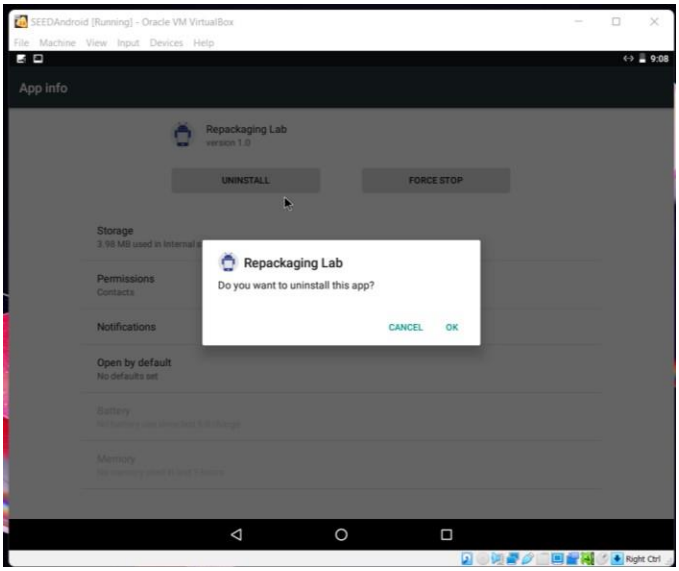
Step 10: Sign again our apk with public and private keys





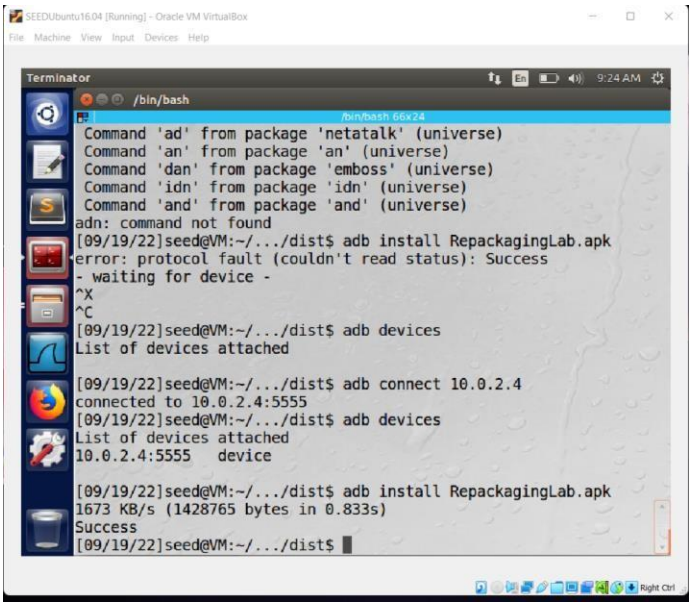


Step 11: Uninstall old android app from Android VM to install new one.

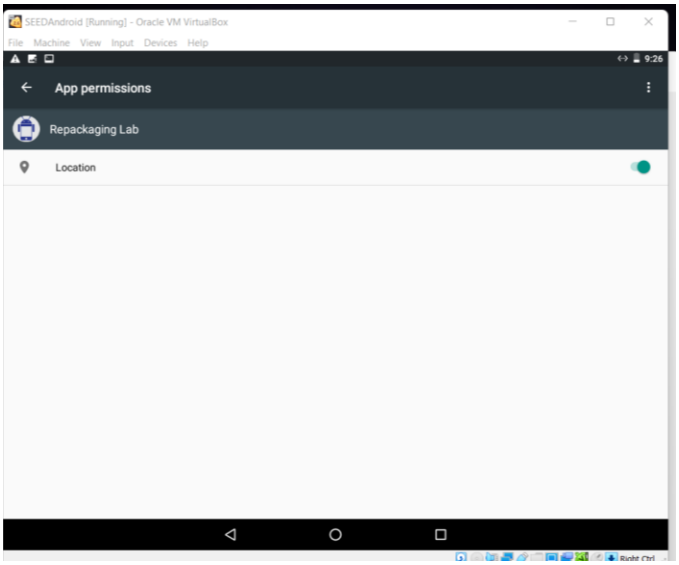


Step 12: Reinstall RepackagingLab.apk successfully with command:

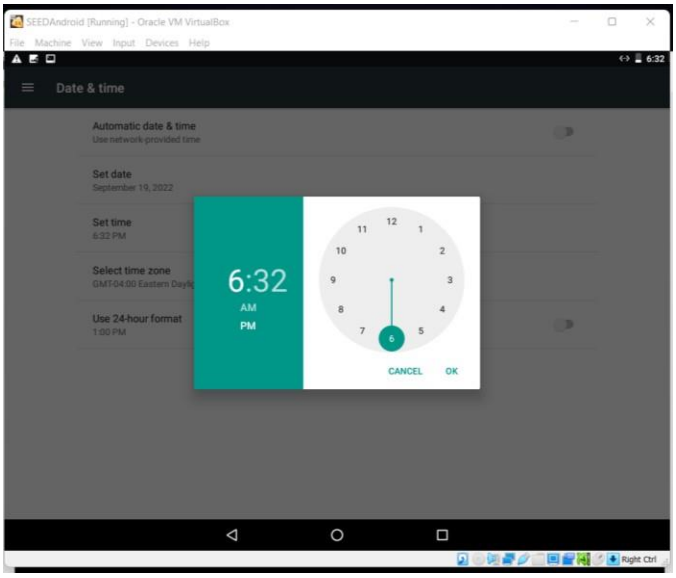
adb install RepackagingLab.apk



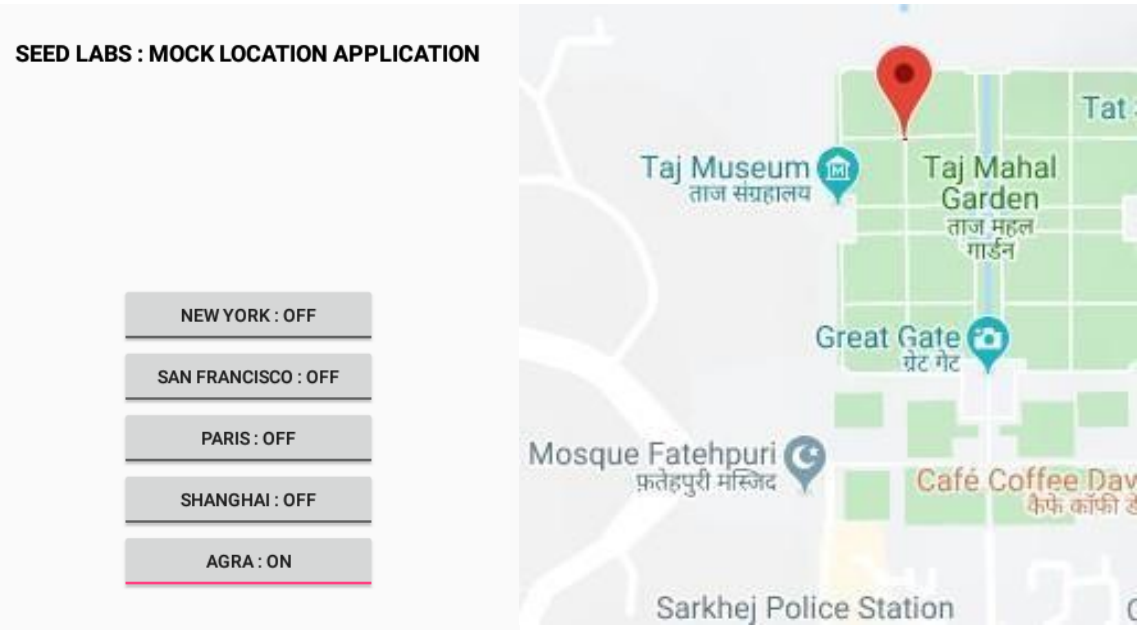
Step 13: Give Location permission before running the app from app settings



Step 14: Change time to activate malware



Step 15: Switch on Agra on Android VM Mock Location app and on Ubuntu VM we get same location by checking it on website: www.repackagingattacklab.com



Point to be Noted: Uninstall and reinstall app, change time is must to trigger malware. App should run once to work fine

This is the end of our Android Repackaging Lab Report.