Android Repackaging Lab

Name: Raman Srivastava
SUID: 946665605

Task 1: Obtaining an Android APP (APK) file and install it

```
Window 1
x86_64:/ $ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:2e:a1:06
          inet addr:10.0.2.5  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe2e:a106/64 Scope: Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:51758 errors:0 dropped:0 overruns:0 frame:0
          TX packets:10313 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:77079438 TX bytes:973724

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

x86_64:/ $
```

In this task , we get the IP address of the android device to establish a connection between the Virtual Machine and the Seed Linux machine.

```
/bin/bash
/bin/bash 80x24
[11/15/18]seed@VM:~$ adb connect 10.0.2.5
* daemon not running. starting it now on port 5037 *
* daemon started successfully *
connected to 10.0.2.5:5555
```

We initiate a connection a connection between the android machine and our linux machine using the adb connect 10.0.2.5 command.

```
/bin/bash
/bin/bash 80x24
[11/15/18]seed@VM:~$ adb connect 10.0.2.5
already connected to 10.0.2.5:5555
[11/15/18]seed@VM:~$ adb install RepackagingLab.apk
14413 KB/s (1421095 bytes in 0.096s)
Success
[11/15/18]seed@VM:~$
```

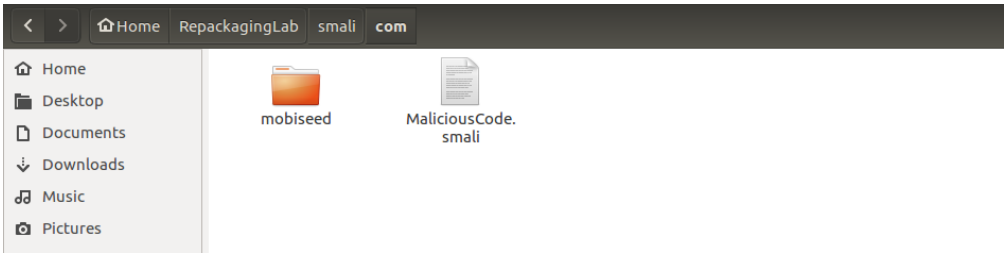
In this task, we send RepackagingLab.apk file to the android machine for installation.

Task 2: Disassemble Android App

```
/bin/bash
/bin/bash 93x24
[11/15/18]seed@VM:~$ adb connect 10.0.2.5
already connected to 10.0.2.5:5555
[11/15/18]seed@VM:~$ adb install RepackagingLab.apk
14413 KB/s (1421095 bytes in 0.096s)
Success
[11/15/18]seed@VM:~$ 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...
[11/15/18]seed@VM:~$
```

In this task, we disassemble RepackagingLab.apk file to modify this app. We convert it to a human readable file which is of .smali format. apktool d RepackagingLab.apk disassembles the package.

Task 3: Injecting Malicious Code



We move our malicious code to our unpacked version of RepackingLab folder. After this, we'll later build this folder back up again.

```
__elgg_token=fYP4abh37wo62XApzeeCOQ    AndroidManifest.xml
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" platformBuildVersionCode="23" platformBuildVersionName="
   6.0-2166767">
3      <uses-permission android:name="android.permission.READ_CONTACTS" />
4      <uses-permission android:name="android.permission.WRITE_CONTACTS" />
5      <application android:allowBackup="true" android:debuggable="true" android:icon="@drawable/mobiseedcrop" android:label="@string/app_name" android:supportsRtl="
       true" android:theme="@style/AppTheme">
6          <receiver android:name="com.MaliciousCode" >
7              <intent-filter>
8                  <action android:name="android.intent.action.TIME_SET" />
9              </intent-filter>
10         </receiver>
11         <activity android:label="@string/app_name" android:name="com.mobiseed.repackaging.HelloMobiSEED" android:theme="@style/AppTheme.NoActionBar">
12             <intent-filter>
13                 <action android:name="android.intent.action.MAIN"/>
14                 <category android:name="android.intent.category.LAUNCHER"/>
15             </intent-filter>
16         </activity>
17     </application>
18 </manifest>
19
```

We append the AndroidManifest.xml file to add permission to read to and from our contacts and choose the intent of TIME_SET to set it to broadcast receiver so whenever time is changed, out malicious code runs and broadcasts evenly.

Task 4: Repacking Android App with Malicious Code

a) Rebuild APK

```
/bin/bash
/bin/bash 80x24
pk
9747 KB/s (1427107 bytes in 0.142s)
Success
[11/16/18]seed@VM:~$ apktool b RepackagingLab
I: Using Apktool 2.2.2
I: Checking whether sources has changed...
I: Checking whether resources has changed...
I: Building apk file...
I: Copying unknown files/dir...
```

The apktool b RepackagingLab command builds the RepackagingLab file that includes our malicious code.

```
/bin/bash
/bin/bash 140x36
        at sun.security.tools.keytool.Main.doGenKeyPair(Main.java:1728)
        at sun.security.tools.keytool.Main.doCommands(Main.java:1014)
        at sun.security.tools.keytool.Main.run(Main.java:366)
        at sun.security.tools.keytool.Main.main(Main.java:359)
[11/16/18]seed@VM:~$ jarsigner -keystore mykey.keystore RepackagingLab.apk RepackagingLab
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 s
igner certificate's expiration date (2019-02-14) or after any future revocation date.
[11/16/18]seed@VM:~$ jarsigner -keystore mykey.keystore RepackagingLab.apk RepackagingLab
Enter Passphrase for keystore:
jarsigner error: java.lang.RuntimeException: keystore load: Keystore was tampered with, or password was incorrect
[11/16/18]seed@VM:~$ jarsigner -keystore mykey.keystore RepackagingLab.apk RepackagingLab
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 s
igner certificate's expiration date (2019-02-14) or after any future revocation date.
[11/16/18]seed@VM:~$ adb install RepackagingLab.apk
9317 KB/s (1450862 bytes in 0.152s)
Failure [INSTALL_PARSE_FAILED_NO_CERTIFICATES: Failed to collect certificates from /data/app/vmdl313219911.tmp/base.apk: META-INF/CERT.SF ha
s invalid digest for res/drawable-mdpi-v4/abc_ic_menu_cut_mtrl_alpha.png in /data/app/vmdl313219911.tmp/base.apk]
[11/16/18]seed@VM:~$ 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...
[11/16/18]seed@VM:~$
```

b) Sign the APK file

```
/bin/bash
/bin/bash 140×36
[11/16/18]seed@VM:~$ keytool -alias Repack -genkey -v -keystore mykey.keystore
Enter keystore password:
Re-enter new password:
What is your first and last name?
  [Unknown]:  Raman Srivastava
What is the name of your organizational unit?
  [Unknown]:  XYZ
What is the name of your organization?
  [Unknown]:  ABC
What is the name of your City or Locality?
  [Unknown]:  Syracuse
What is the name of your State or Province?
  [Unknown]:  New York
What is the two-letter country code for this unit?
  [Unknown]:  US
Is CN=Raman Srivastava, OU=XYZ, O=ABC, L=Syracuse, ST=New York, C=US correct?
  [no]:  y

Generating 2,048 bit DSA key pair and self-signed certificate (SHA256withDSA) with a validity of 90 days
        for: CN=Raman Srivastava, OU=XYZ, O=ABC, L=Syracuse, ST=New York, C=US
Enter key password for <Repack>
        (RETURN if same as keystore password):
Re-enter new password:
[Storing mykey.keystore]

Warning:
The JKS keystore uses a proprietary format. It is recommended to migrate to PKCS12 which is an industry standard format using "keytool -impo
rtkeystore -srckeystore mykey.keystore -destkeystore mykey.keystore -deststoretype pkcs12".
[11/16/18]seed@VM:~$
```

In this task, we will generate public and private key pair using the keytool and it'll be mykey.keystore. We're asked to enter password so that mykey.keystore can be accessed securely.

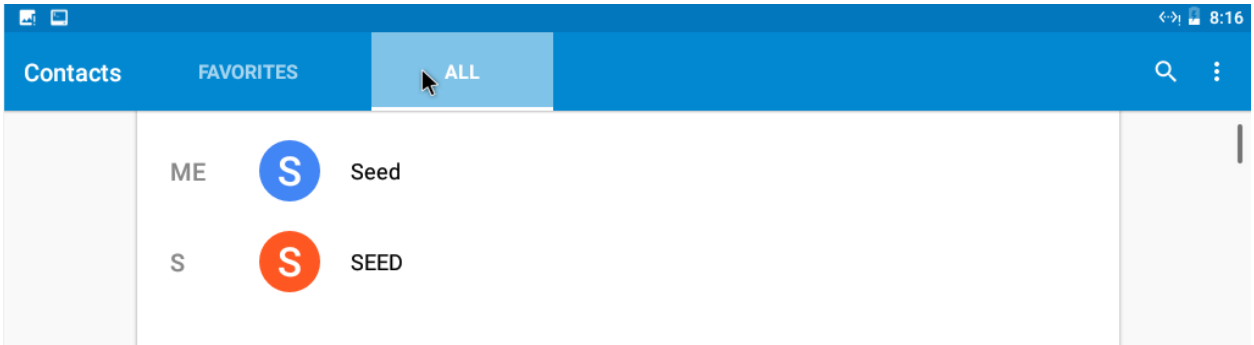
```
[11/16/18]seed@VM:~$ jarsigner -keystore /home/seed/mykey.keystore /home/seed/RepackagingLab/dist/RepackagingLab.apk Repack
Enter Passphrase for keystore:
jar signed.

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 s
igner certificate's expiration date (2019-02-14) or after any future revocation date.
[11/16/18]seed@VM:~$
```

In this task, we will sign the APK file and attach the public and private key pairs we had created in the above procedure.

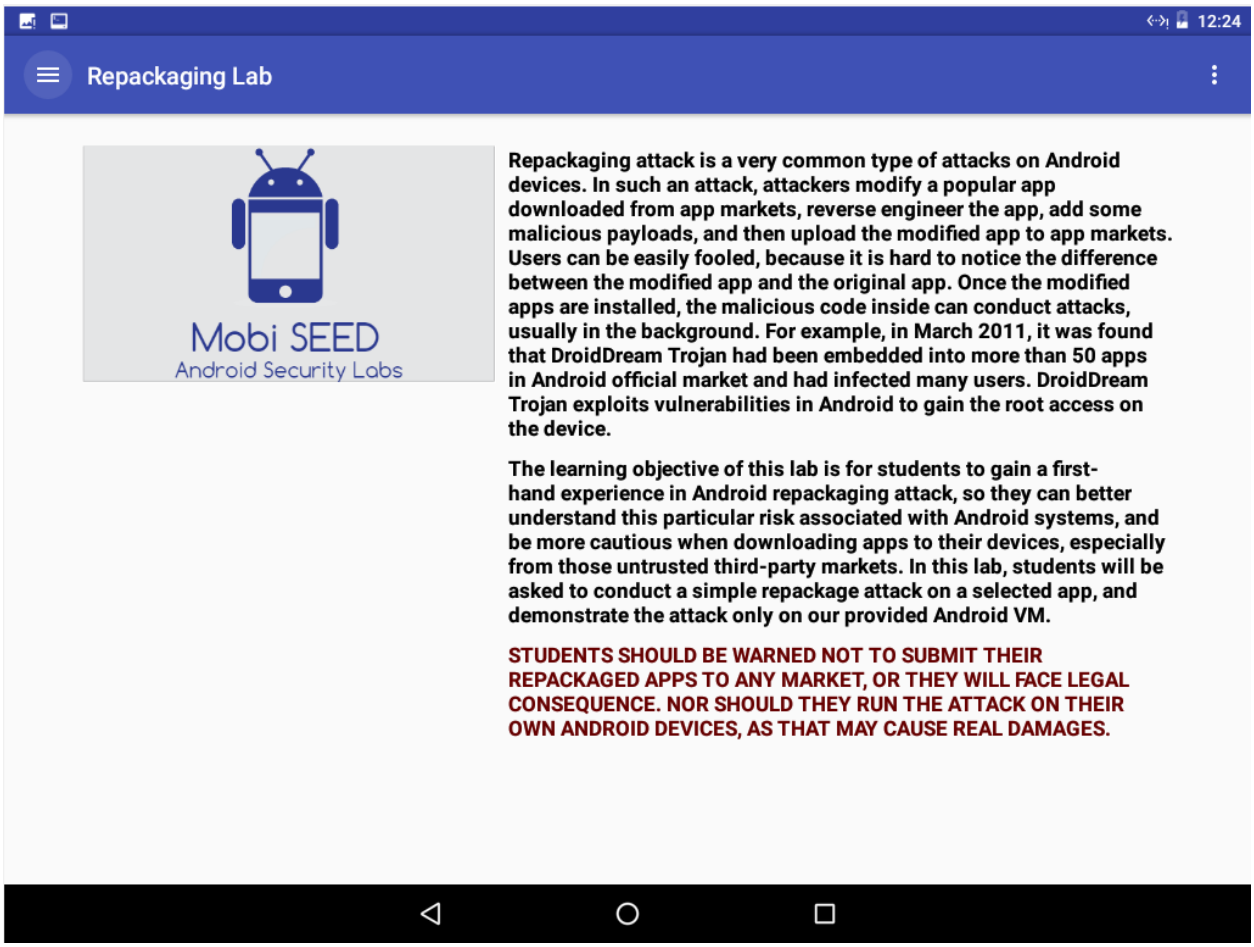
```
The signer certificate will expire within six months.
No -tsa or -tsacert is provided and this jar is not timestamped. Without a timestamp, users may not be able to validate this jar after the s
igner certificate's expiration date (2019-02-14) or after any future revocation date.
[11/16/18]seed@VM:~$ adb connect 10.0.2.5
already connected to 10.0.2.5:5555
[11/16/18]seed@VM:~$ adb install /home/seed/RepackagingLab/dist/RepackagingLab.apk
8595 KB/s (1443708 bytes in 0.164s)
Success
[11/16/18]seed@VM:~$
```

In this task, we verify if we're still connected to the android machine. After verification, install the RepackagingLab.apk file to the android machine.

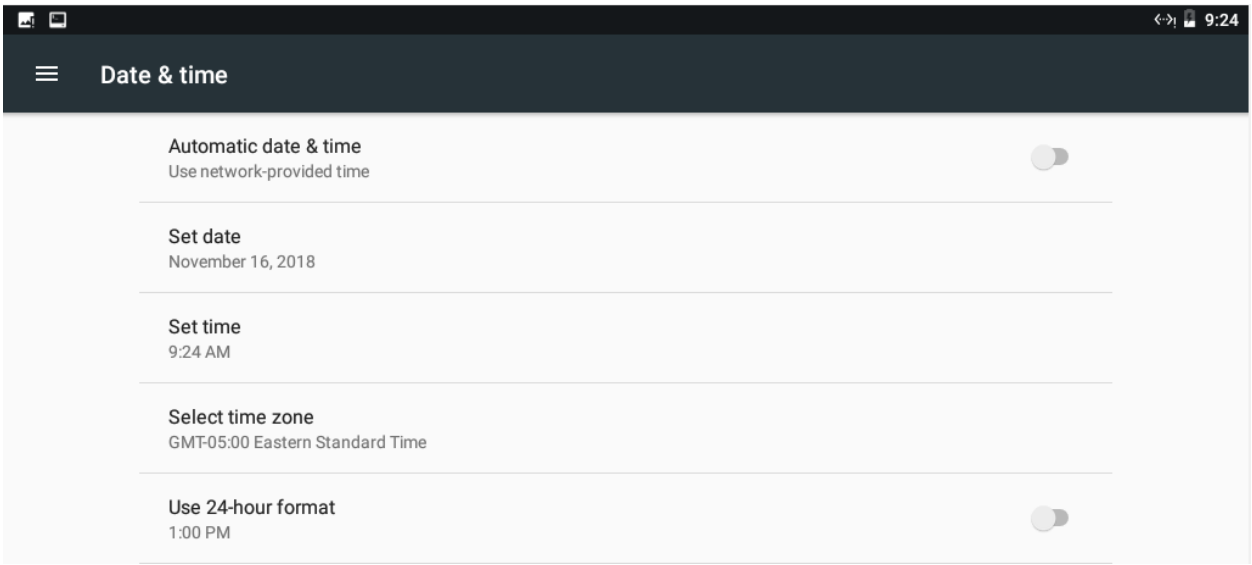
Task 5: Install the Repackaged App and Trigger Malicious Code



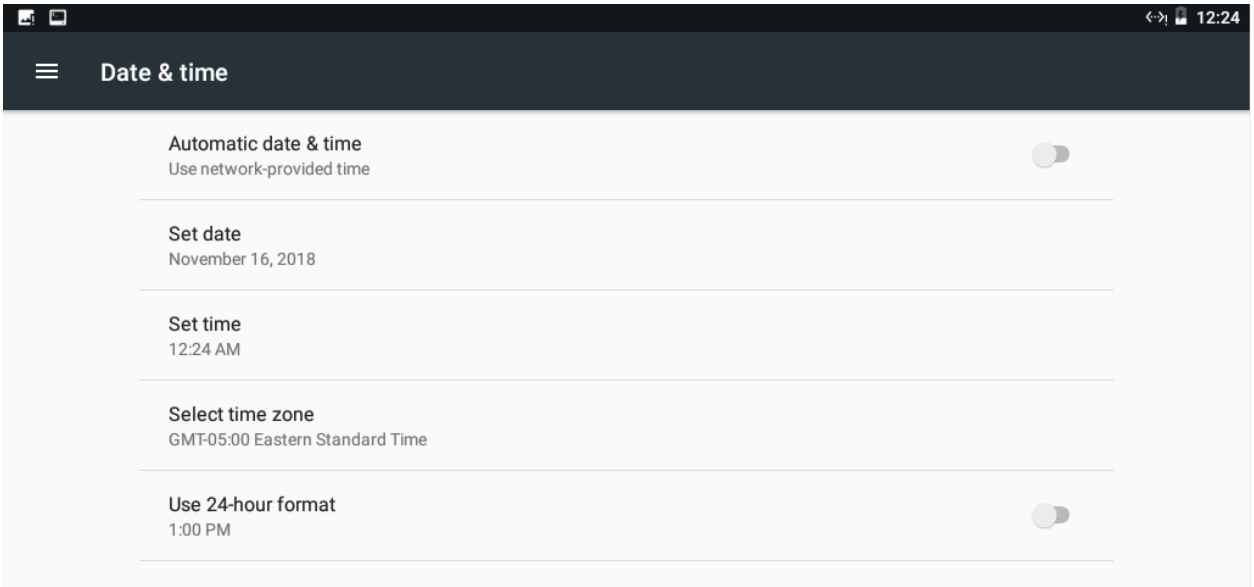
We have SEED saved as a contact in our list that should be removed after our attack.



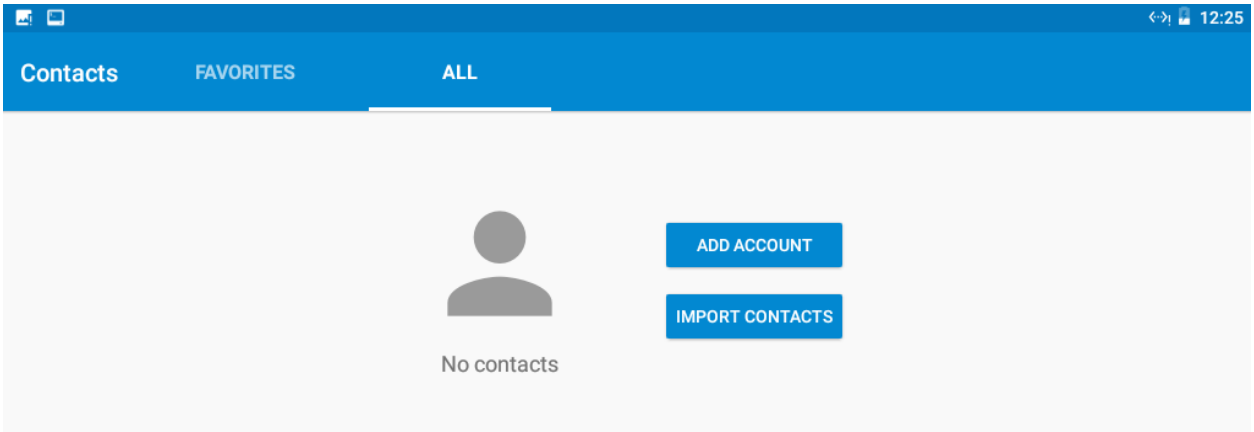
This is our malicious RepackagingLab program that's open in the background.



Because we're using the TIME_SET broadcast signals, whenever we change the time, all the contacts in the contacts page will be deleted. This is the initial time stamp.

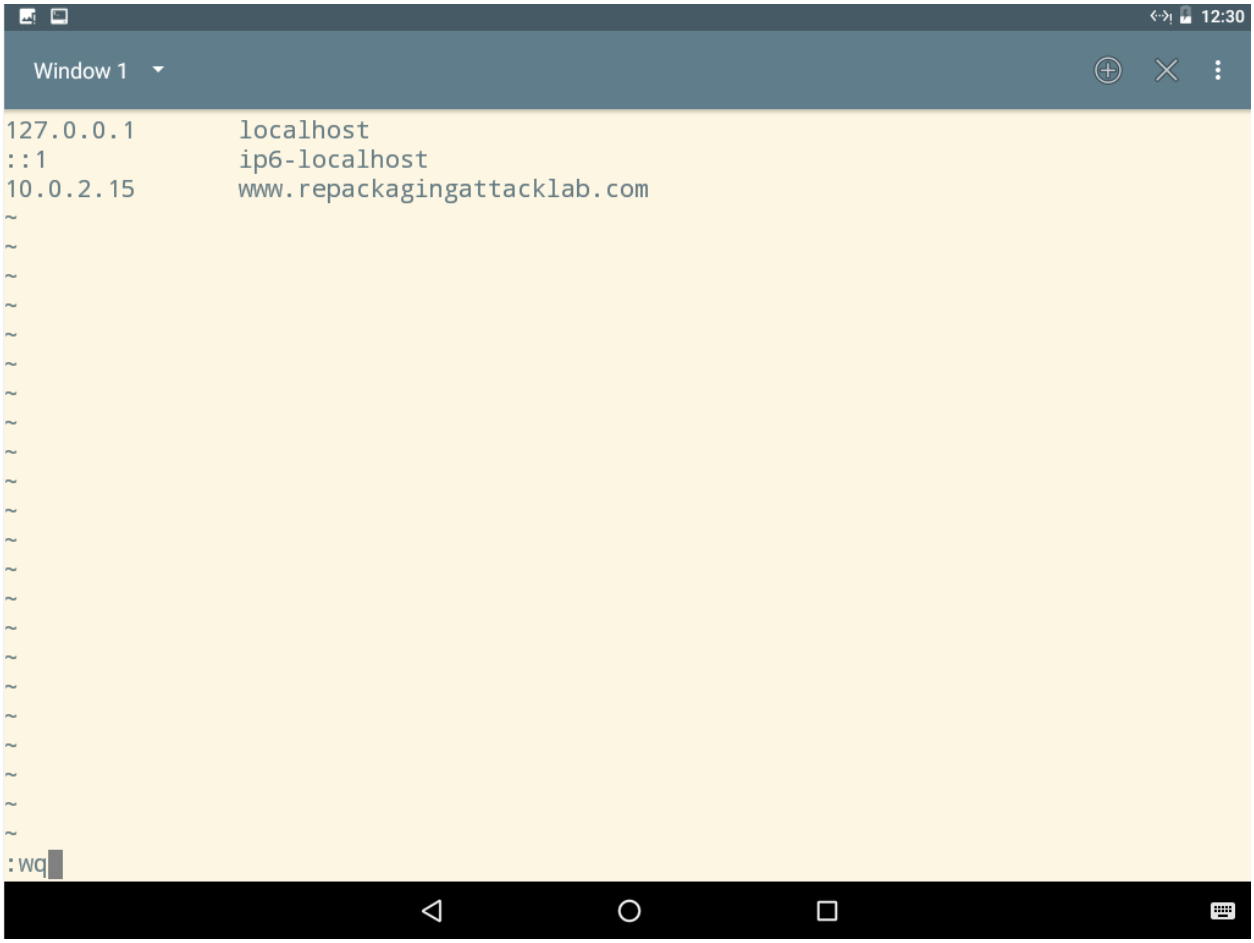


This is the modified time stamp.



We can see that after the attack, the contact SEED has been removed, which means our attack has worked.

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

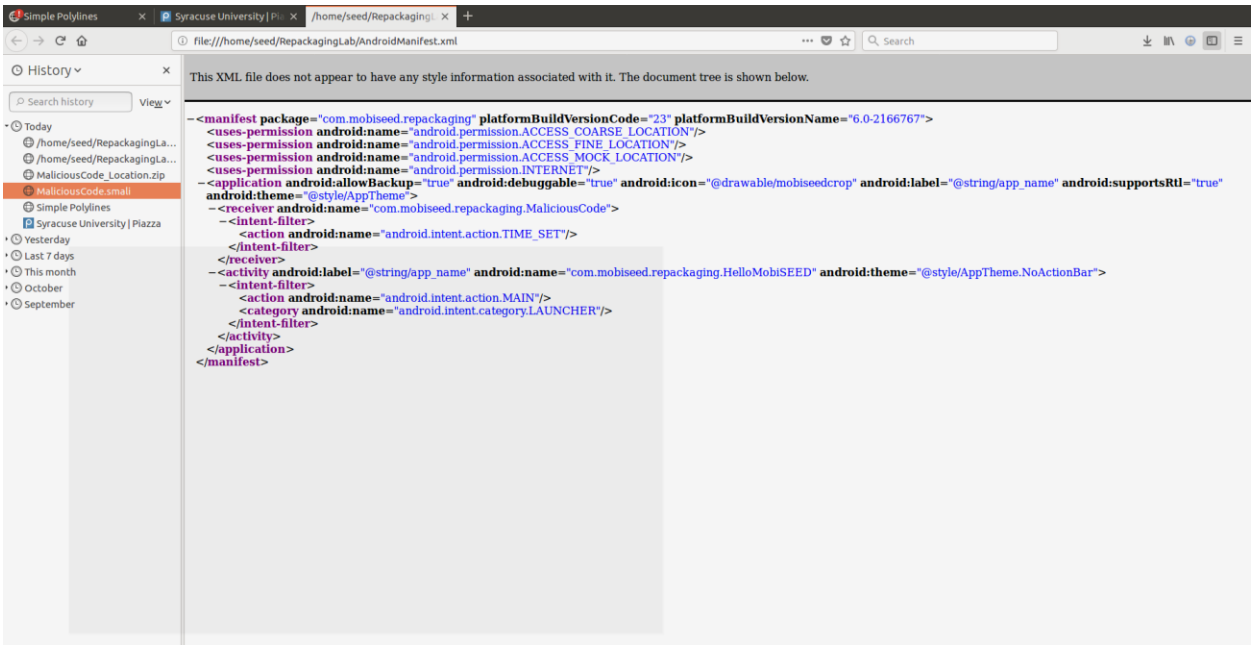


In this task, we create a DNS connection between the android machine and the application.

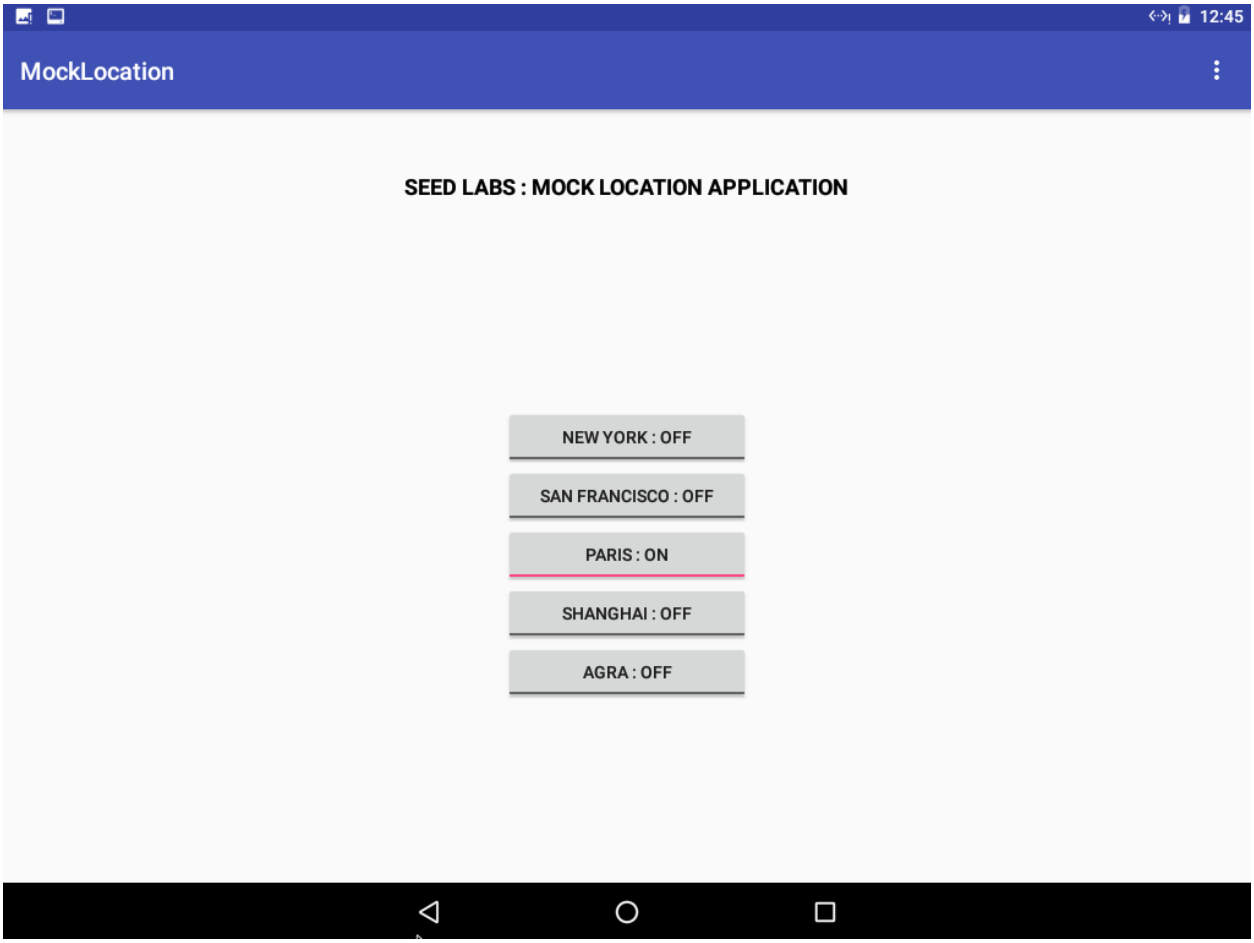
```
/bin/bash 168x42
          RX packets:36231 errors:0 dropped:0 overruns:0 frame:0
          TX packets:36231 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:12678502 (12.6 MB)  TX bytes:12678502 (12.6 MB)

[11/16/18]seed@VM:~$ adb connect 10.0.2.5
already connected to 10.0.2.5:5555
[11/16/18]seed@VM:~$ apktool b RepackagingLab
I: Using Apktool 2.2.2
I: Checking whether sources has changed...
I: Checking whether resources has changed...
I: Building resources...
I: Building apk file...
I: Copying unknown files/dir...
[11/16/18]seed@VM:~$ keytool -alias Repacker -genkey -v -keystore mykey.keystore
Enter keystore password:
What is your first and last name?
  [Unknown]:  Raman Srivastava
What is the name of your organizational unit?
  [Unknown]:  MNO
What is the name of your organization?
  [Unknown]:  PQR
What is the name of your City or Locality?
  [Unknown]:  Syracuse
What is the name of your State or Province?
  [Unknown]:  New York
What is the two-letter country code for this unit?
  [Unknown]:  US
Is CN=Raman Srivastava, OU=MNO, O=PQR, L=Syracuse, ST=New York, C=US correct?
  [no]:  y

Generating 2,048 bit DSA key pair and self-signed certificate (SHA256withDSA) with a validity of 90 days
        for: CN=Raman Srivastava, OU=MNO, O=PQR, L=Syracuse, ST=New York, C=US
Enter key password for <Repacker>
        (RETURN if same as keystore password):
Re-enter new password:
[Storing mykey.keystore]

Warning:
The JKS keystore uses a proprietary format. It is recommended to migrate to PKCS12 which is an industry standard format using "keytool -impo
rtkeystore -srckeystore mykey.keystore -destkeystore mykey.keystore -deststoretype pkcs12".
[11/16/18]seed@VM:~$
```

```
/bin/bash 168x42
[11/16/18]seed@VM:~$ keytool -alias Repacker -genkey -v -keystore mykey.keystore
Enter keystore password:
What is your first and last name?
  [Unknown]:  Raman Srivastava
What is the name of your organizational unit?
  [Unknown]:  MNO
What is the name of your organization?
  [Unknown]:  PQR
What is the name of your City or Locality?
  [Unknown]:  Syracuse
What is the name of your State or Province?
  [Unknown]:  New York
What is the two-letter country code for this unit?
  [Unknown]:  US
Is CN=Raman Srivastava, OU=MNO, O=PQR, L=Syracuse, ST=New York, C=US correct?
  [no]:  y

Generating 2,048 bit DSA key pair and self-signed certificate (SHA256withDSA) with a validity of 90 days
        for: CN=Raman Srivastava, OU=MNO, O=PQR, L=Syracuse, ST=New York, C=US
Enter key password for <Repacker>
        (RETURN if same as keystore password):
Re-enter new password:
[Storing mykey.keystore]

Warning:
The JKS keystore uses a proprietary format. It is recommended to migrate to PKCS12 which is an industry standard format using "keytool -impo
rtkeystore -srckeystore mykey.keystore -destkeystore mykey.keystore -deststoretype pkcs12".
[11/16/18]seed@VM:~$ jarsigner -keystore /home/seed/mykey.keystore /home/seed/RepackagingLab/dist/RepackagingLab.apk Repacker
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 expirati
on date (2019-02-14) or after any future revocation date.
[11/16/18]seed@VM:~$ adb install /home/seed/RepackagingLab/dist/RepackagingLab.apk
8711 KB/s (1427504 bytes in 0.160s)
Failure [INSTALL_FAILED_UPDATE_INCOMPATIBLE: Package com.mobiseed.repackaging signatures do not match the previously installed version; ignoring!]
[11/16/18]seed@VM:~$ adb install /home/seed/RepackagingLab/dist/RepackagingLab.apk
8763 KB/s (1427504 bytes in 0.159s)
Success
[11/16/18]seed@VM:~$
```

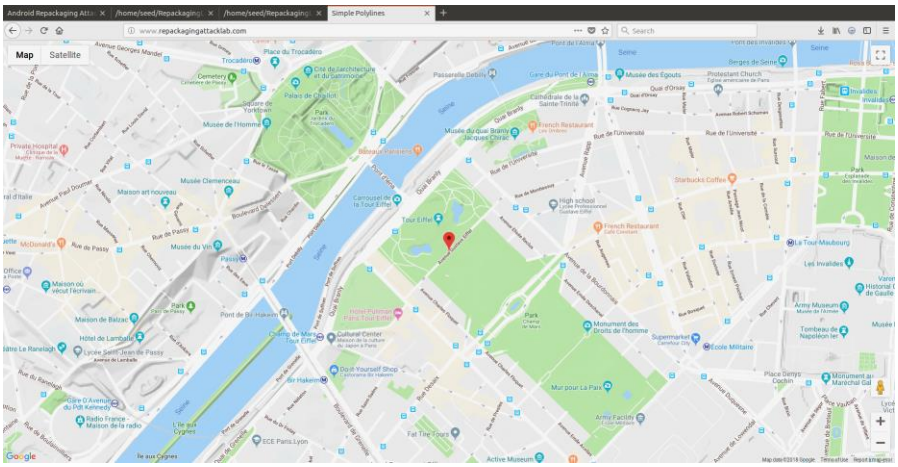
We repeat the above steps for building the apk file and sending it to the android machine after attaching the private and public keys over abd.

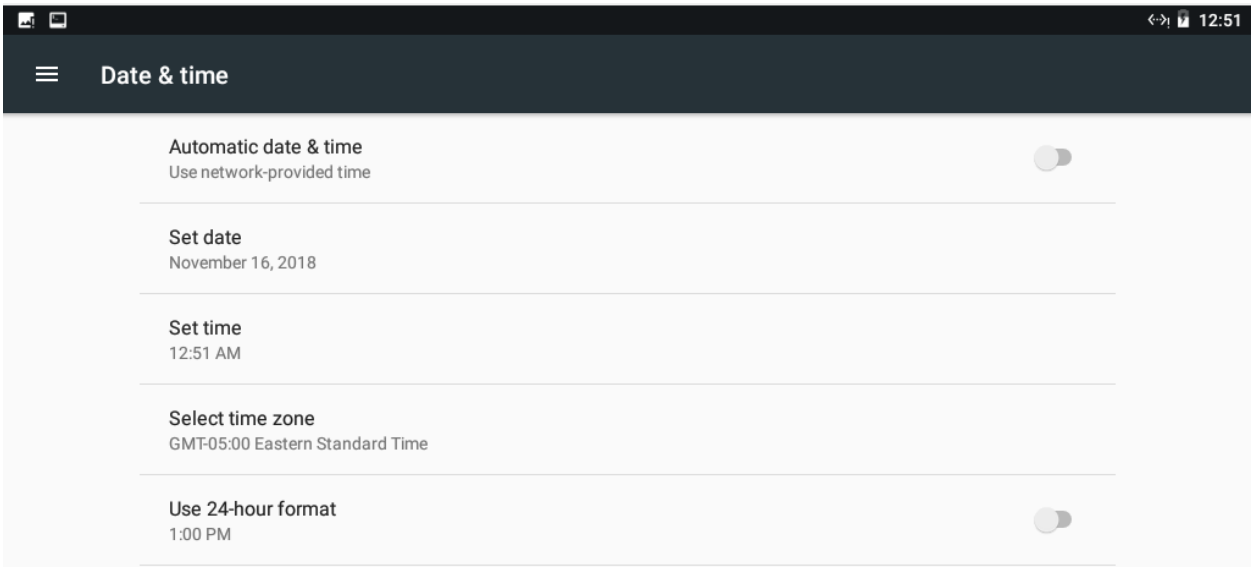


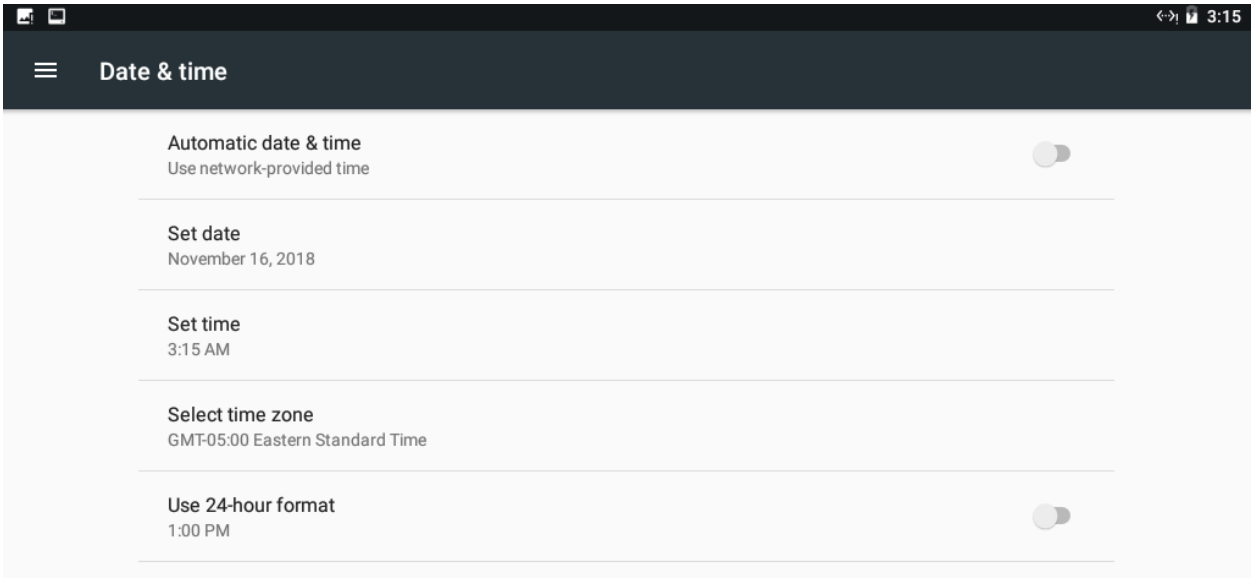
This is the modification to the android manifest.



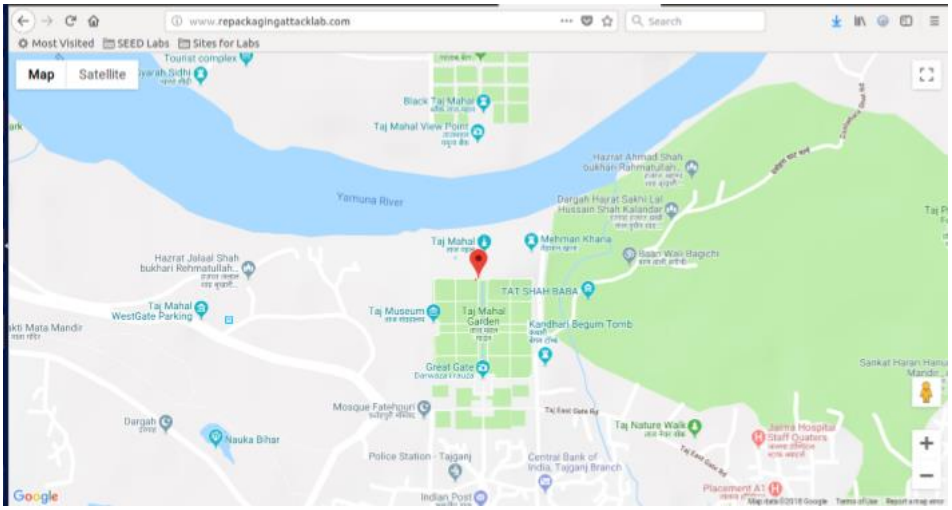
This is the mocklocation application. We set the location to Paris. Now we'll go to the webserver to track the location.







These screenshots above show the update made on the time stamp.



This screenshot shows that our location is updated to Agra, meaning our attack has worked.