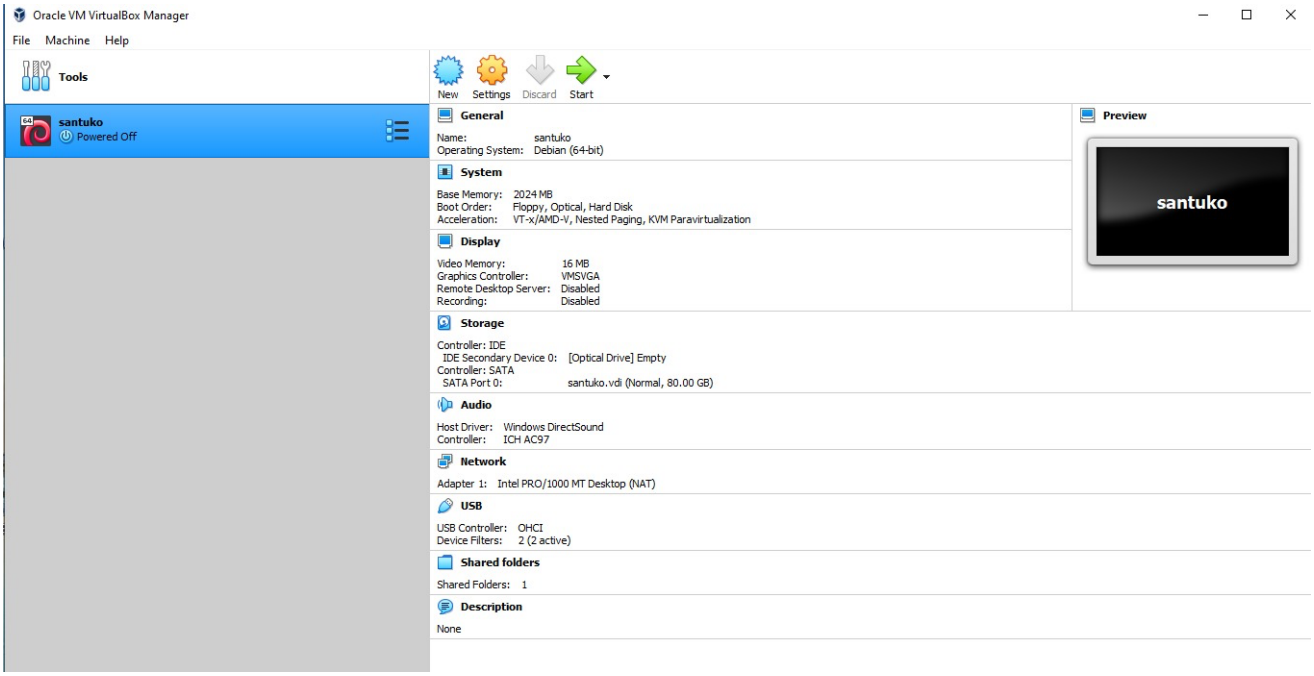
Malicious APK File Creation
No. 11

The installation and the creation of malicious android application are as follows:

Step-1: Installation of Santoku in the Virtual Machine.



Step-2:
Open the terminal and enter the command

pwd

and then enter the command

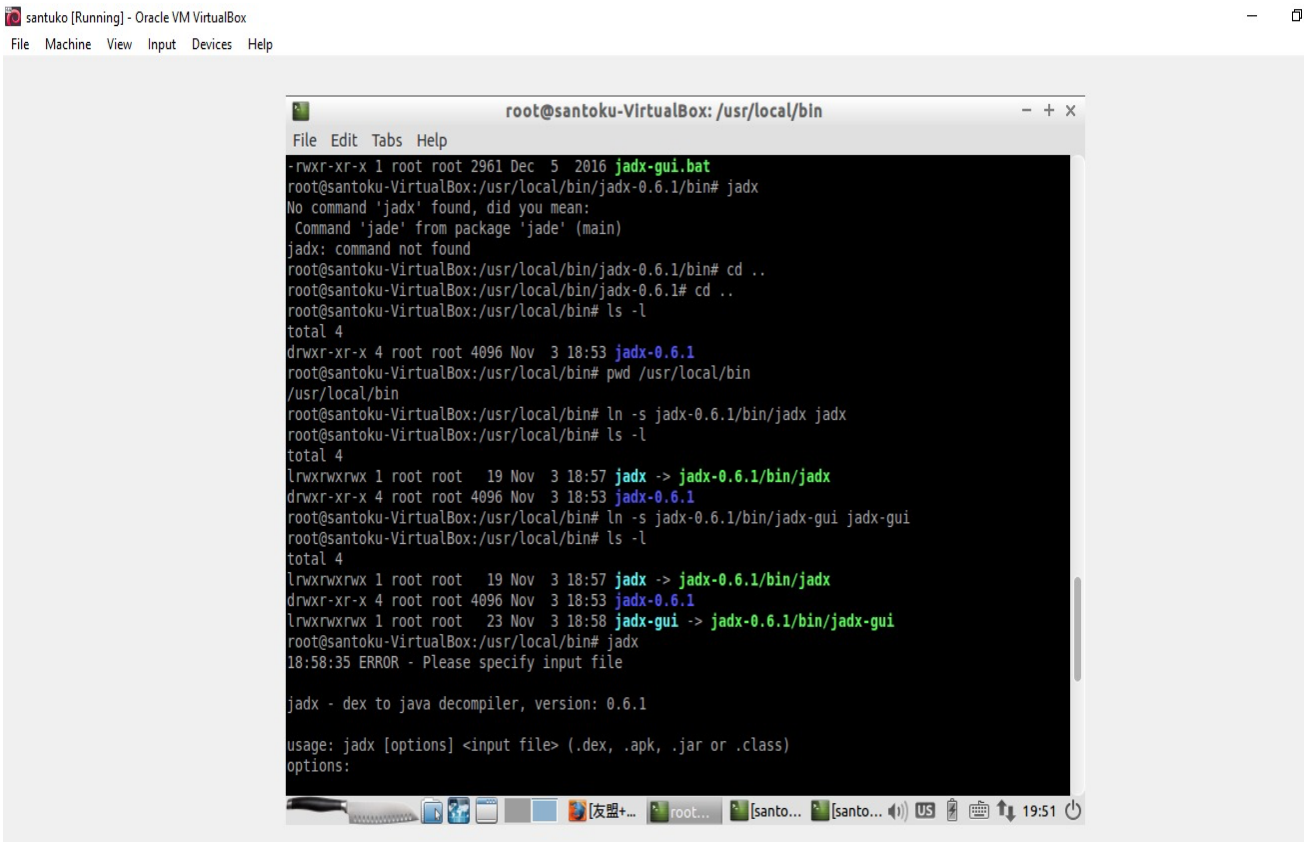
cd/usr/local/bin

This will create a folder in the above path.

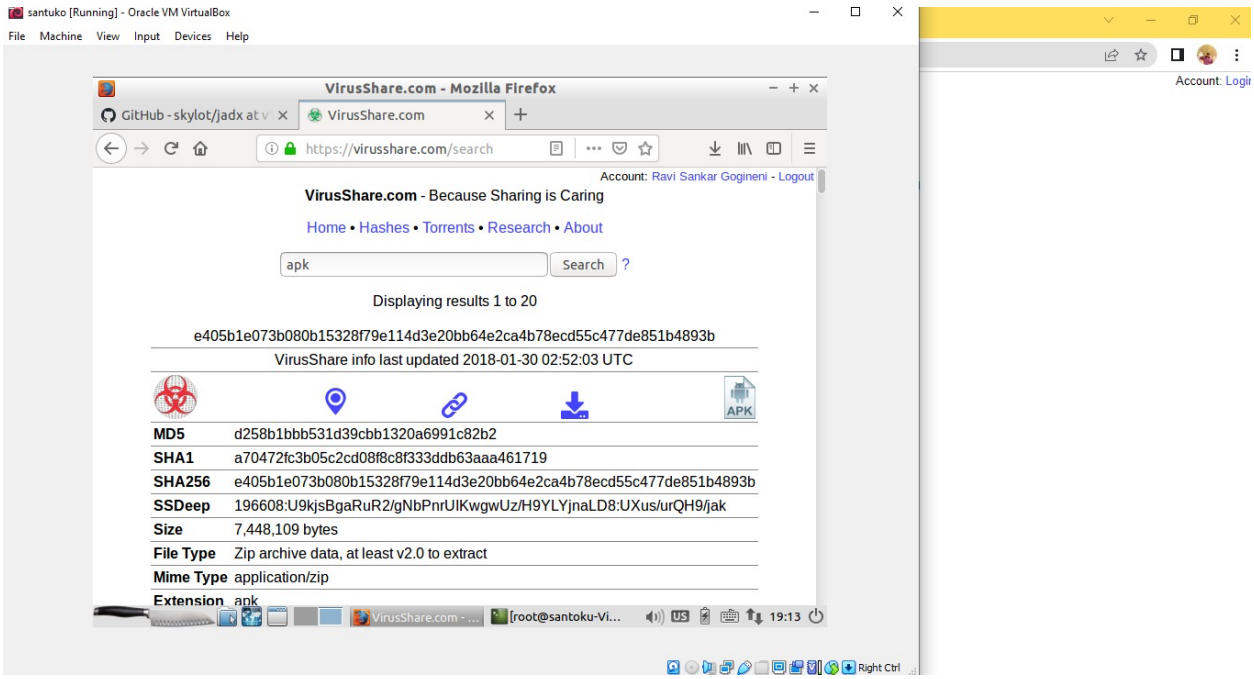
Step-3: Dex to Java Decompiler

We need to install jadx-gui. This is to decompile the application.
This operation will let us know about the application permissions, details of the blocks, information leakage etc. The operation result is as shown below.

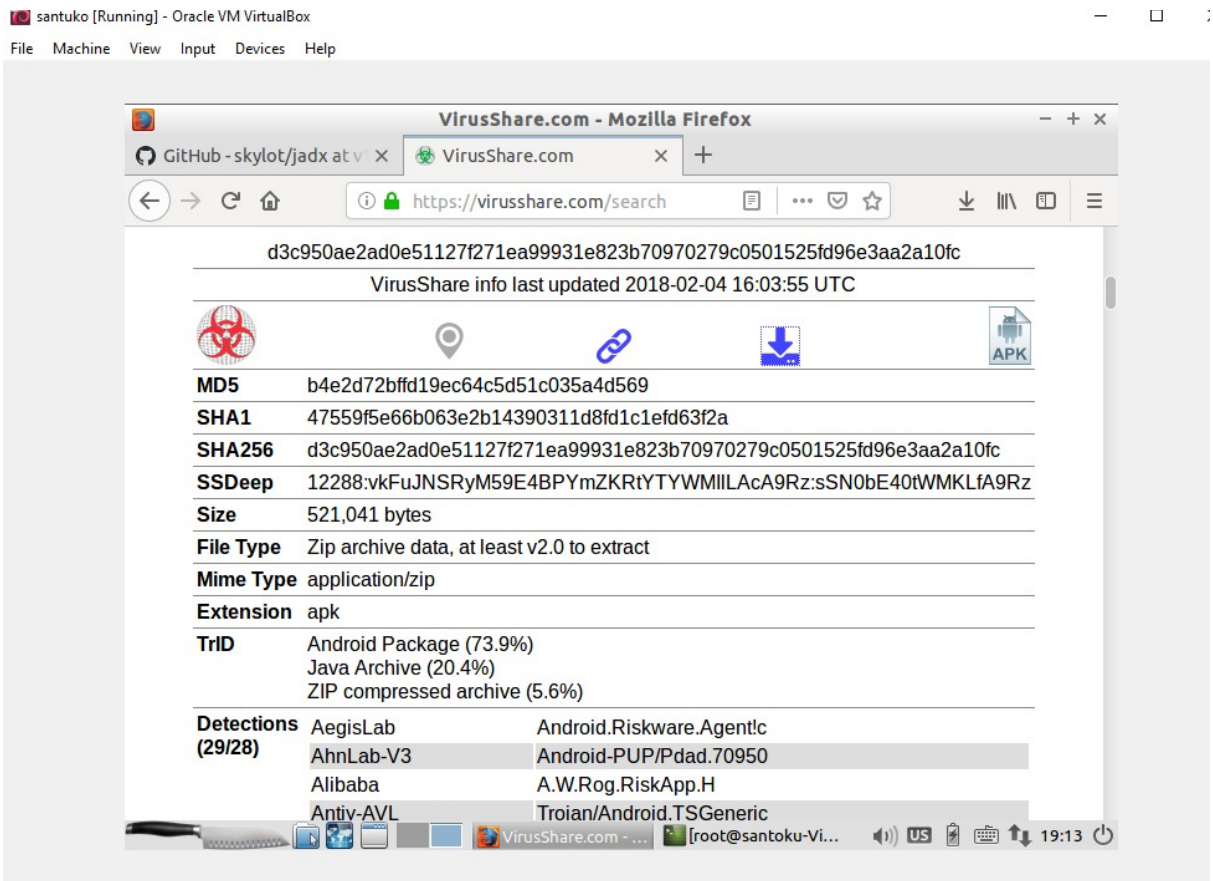
The link https://github.com/skylot/jadx/releases/download/v1.4.5/jadx-1.4.5.zip is used to wgethttps://github.com/skylot/jadx/releases/download/v1.4.5/jadx-1.4.5.zip.
This will lead us to get the application installation in terminal.



Step-4:
By using the https://virusshare.com/ we will download the malicious application.



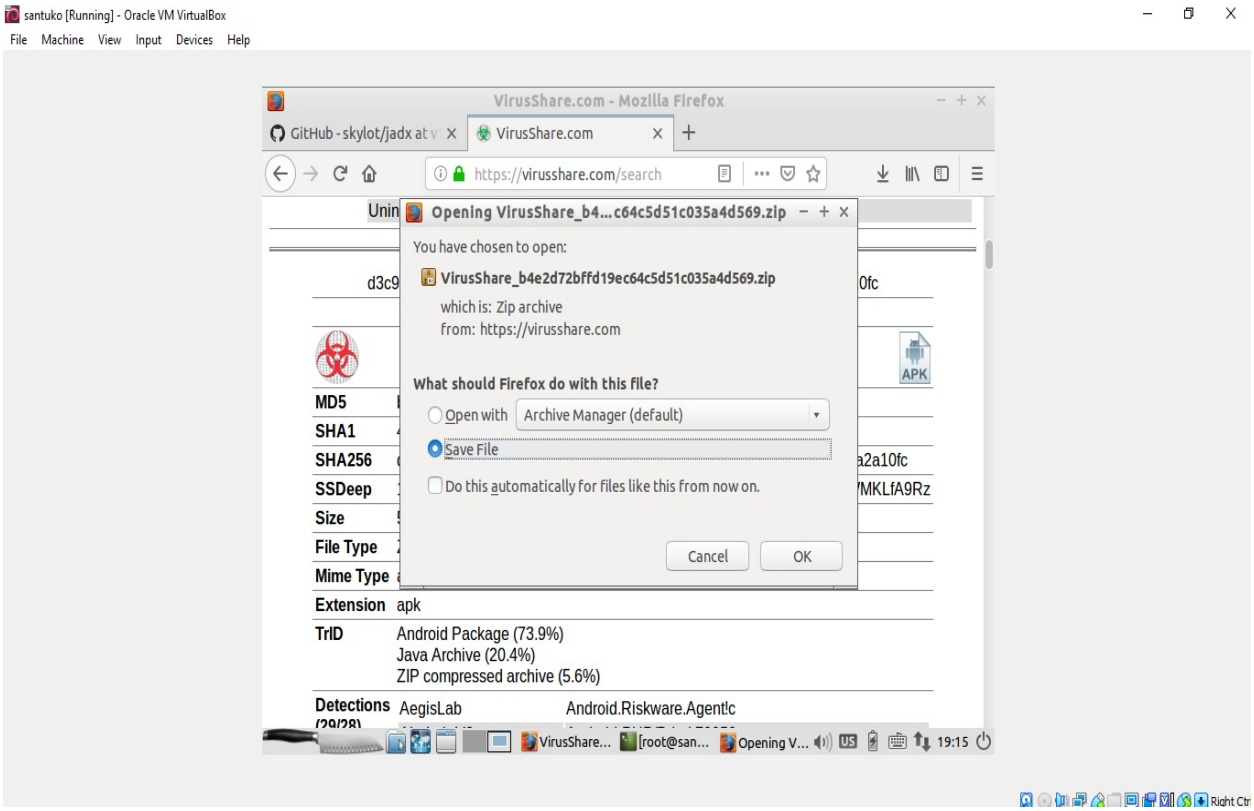
The resultant application details will be as follows:



The malicious part of the application is MD5

B4e2d72bffd19ec64c51c035a4d569

The application is saved as zip file as shown in below snapshot:



Step 5:
By entering the command

ls -l

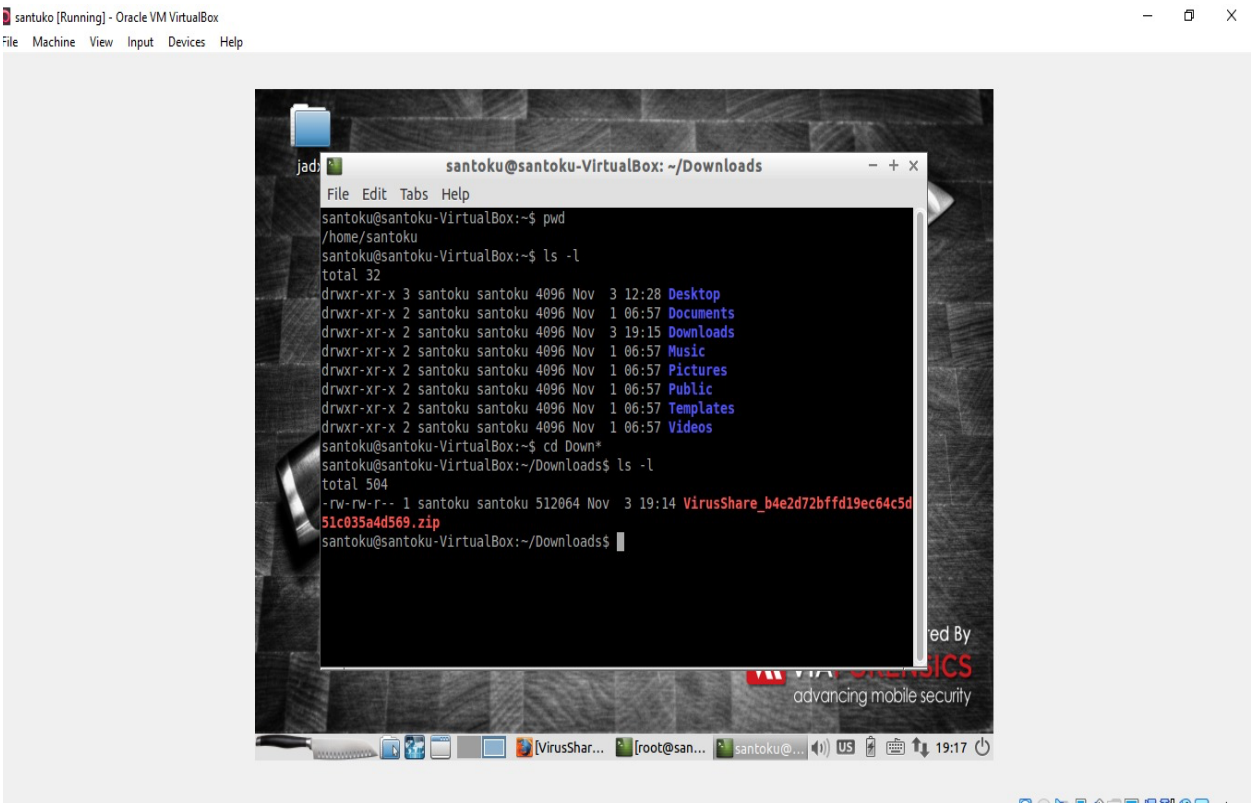
we will get the folders that are inside the application.
By using the command

cd Down*

we will get directed to the path by which the virus is downloaded. After that command, we enter

ls -l

we will get the list of the virus zip file. It appears as shown below:



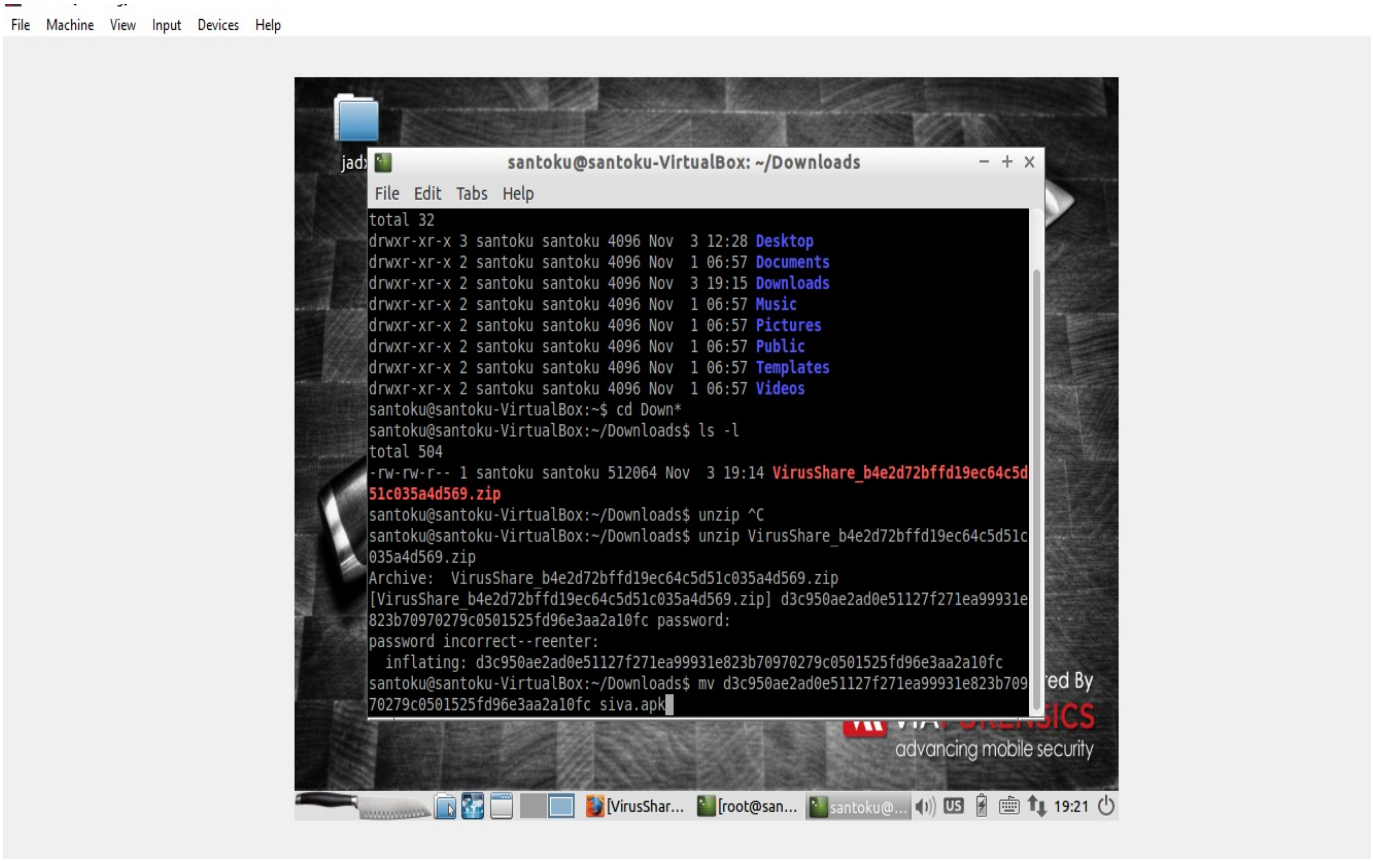
Step 6:

We unzip the folder by using the command

unzip *folder_name*

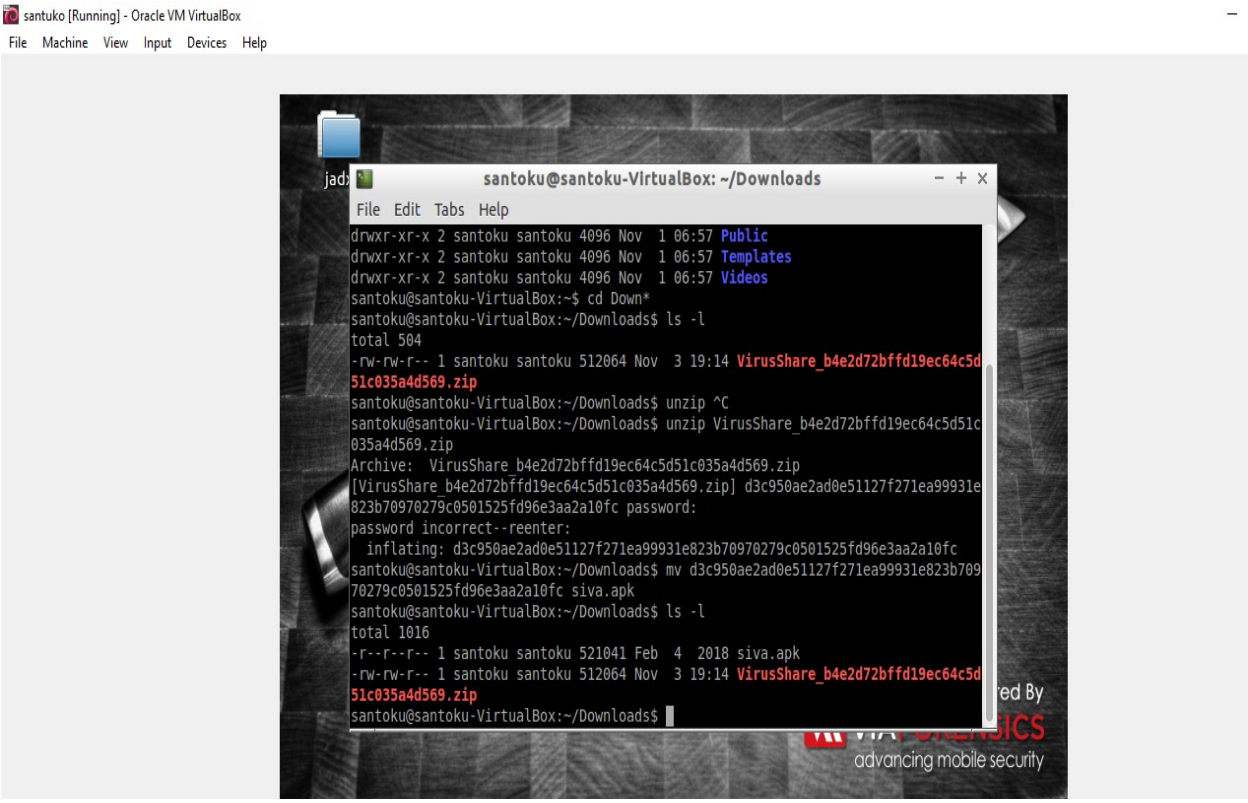
The password for the folder is infected.

Now it will display the virus inflating as shown in below image:



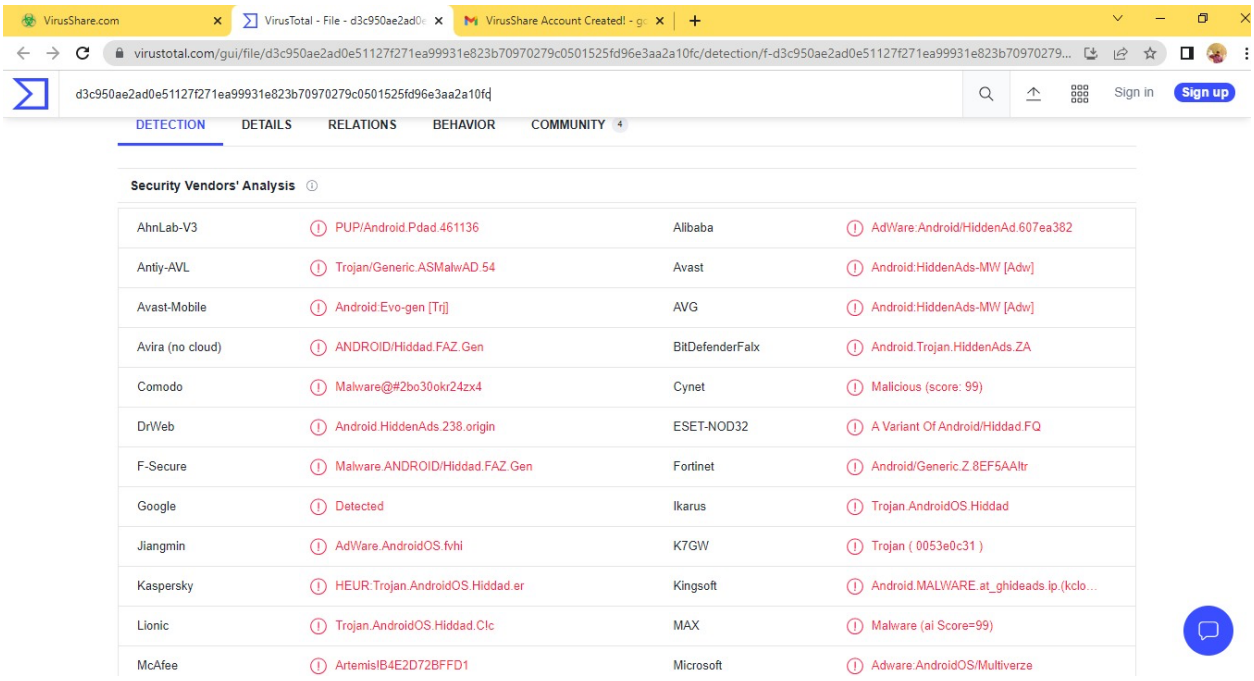
Step 7:

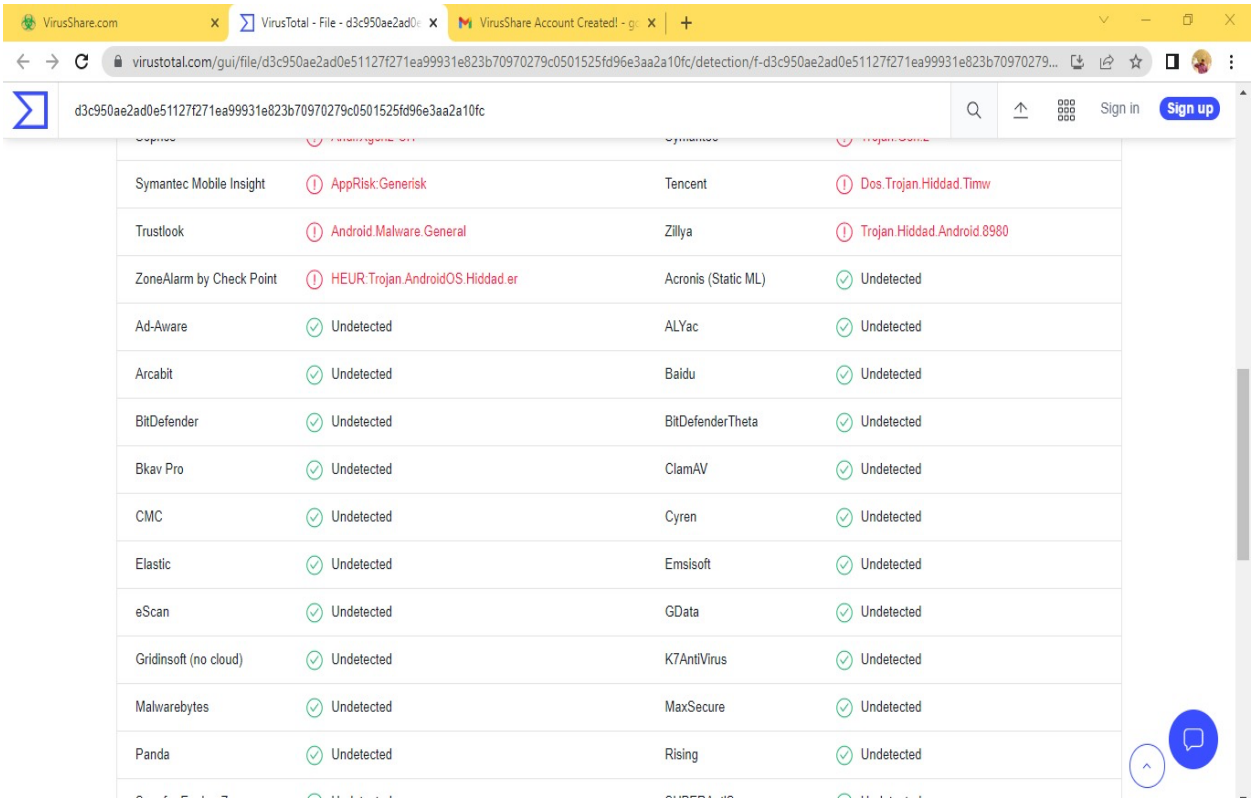
After unzipping the application, we will create a new folder named ravi and we move the malicious application to that folder. And we also change the malicious application name to siva.apk.
The snapshot of the created folder and the application are shown below:



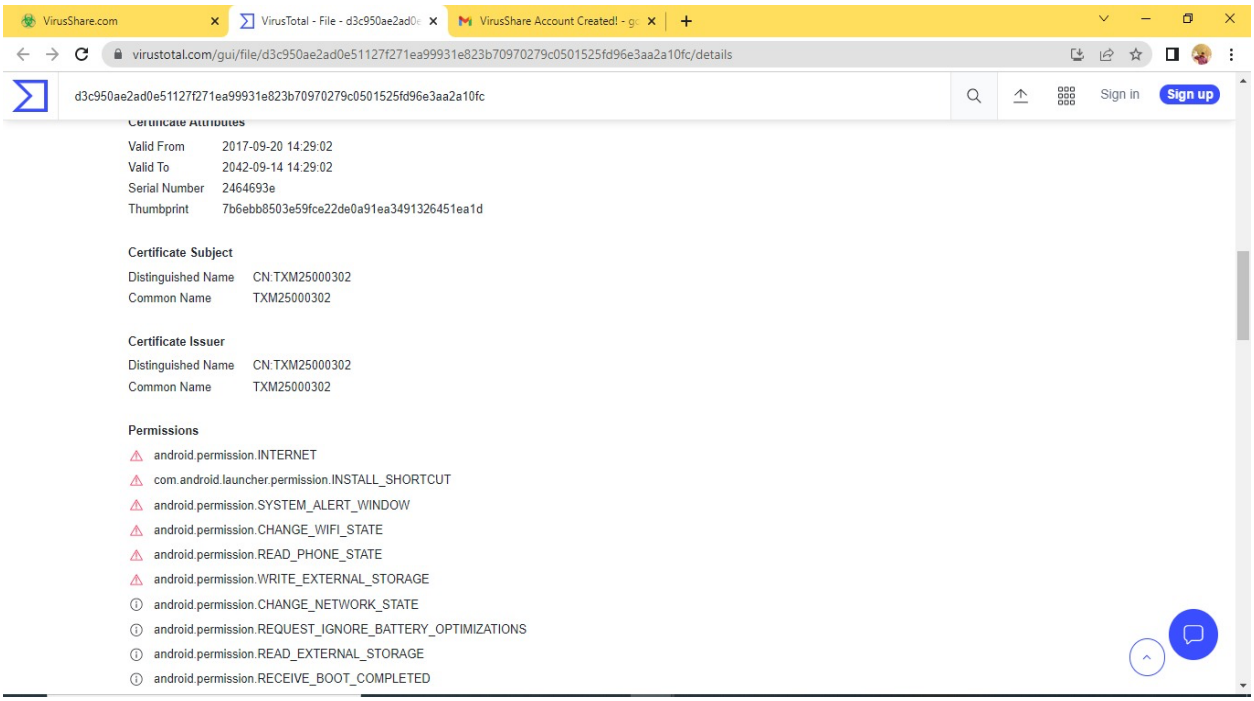
Step 8:
Using https://www.virustotal.com/gui/home/upload, we will get to know what trojans it has and what kind of permissions does it need and logos, activities, services and receivers.

Security analysis:

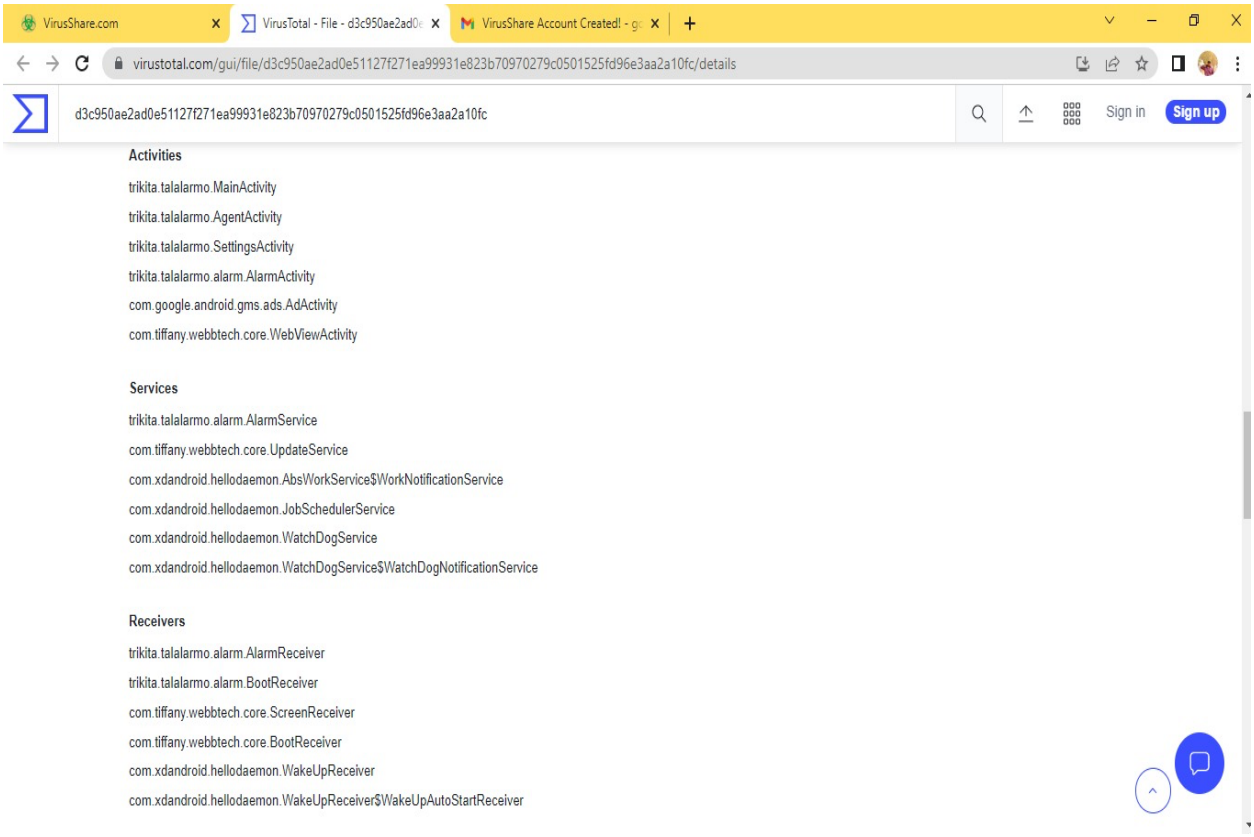




Application permissions:

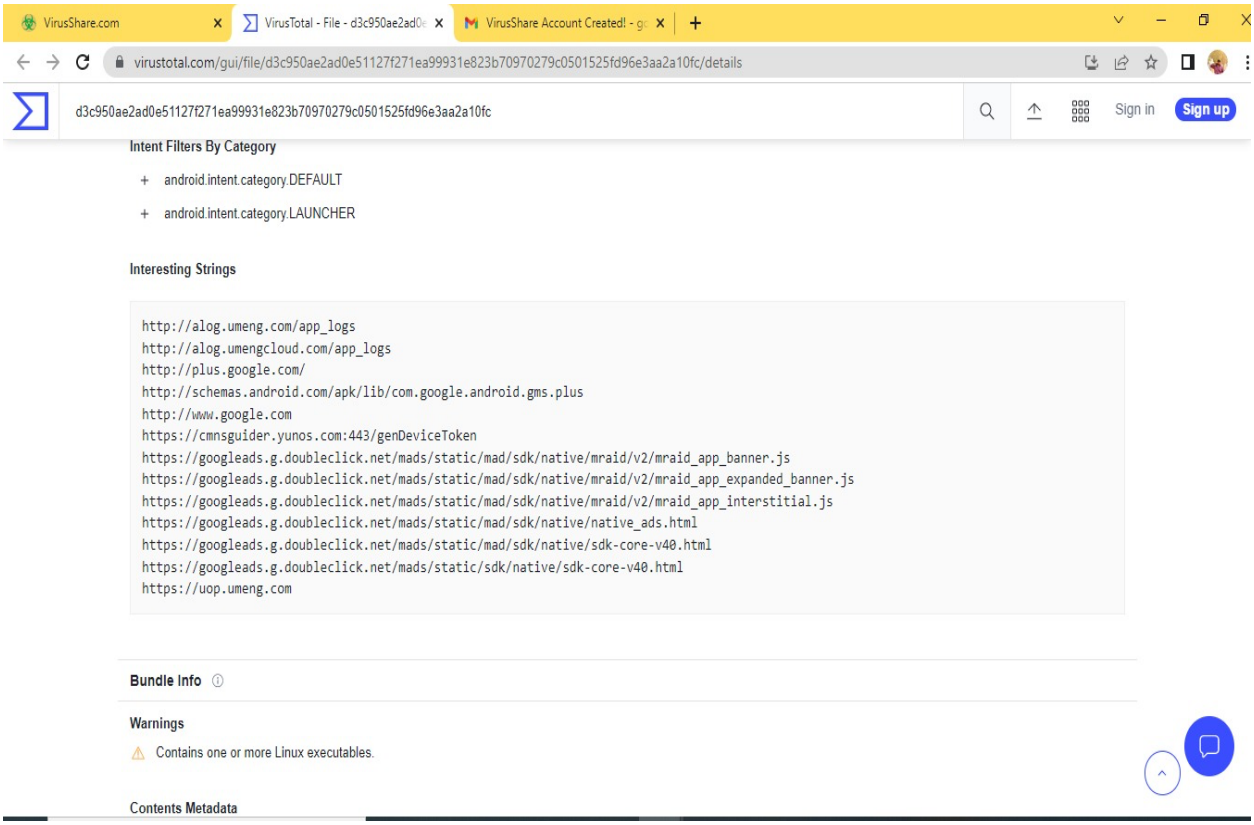


Main activity:



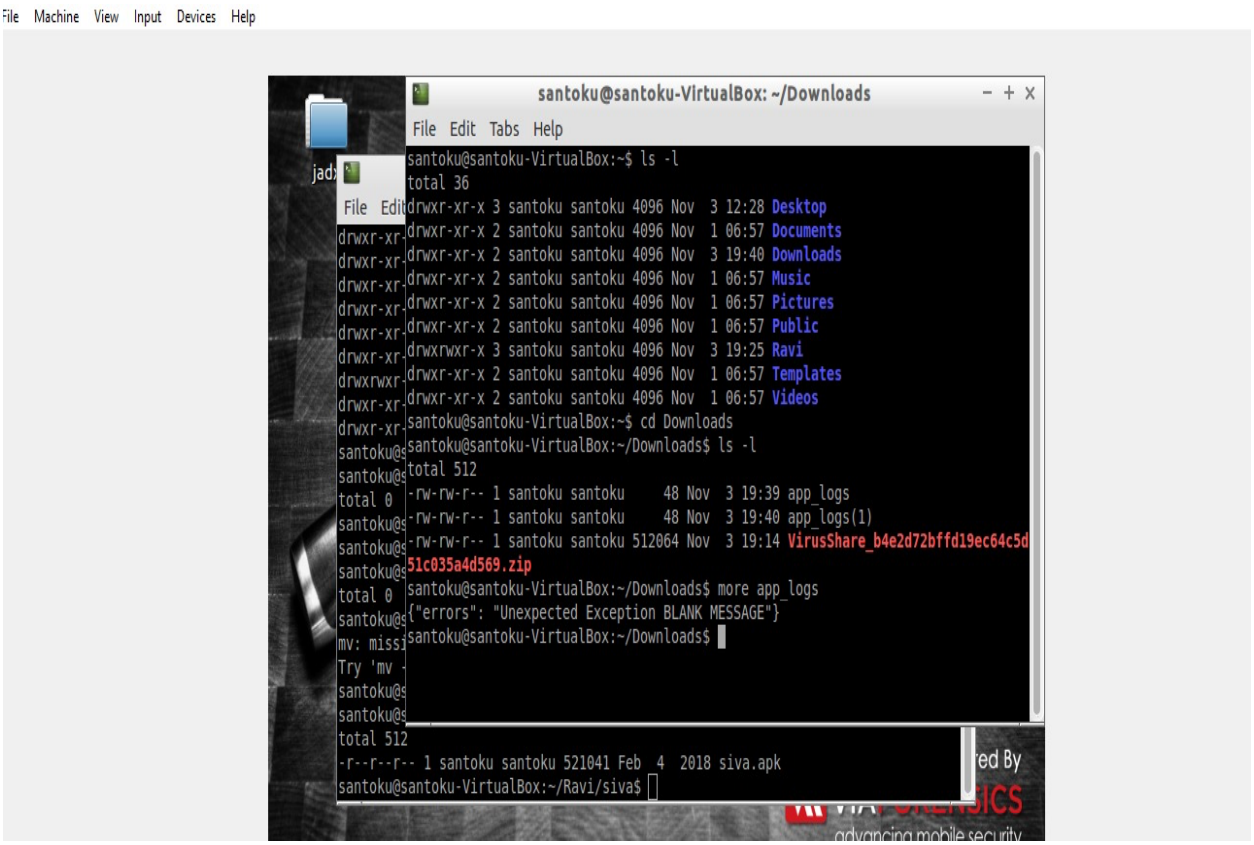
Step 9:

Strings are as shown below:



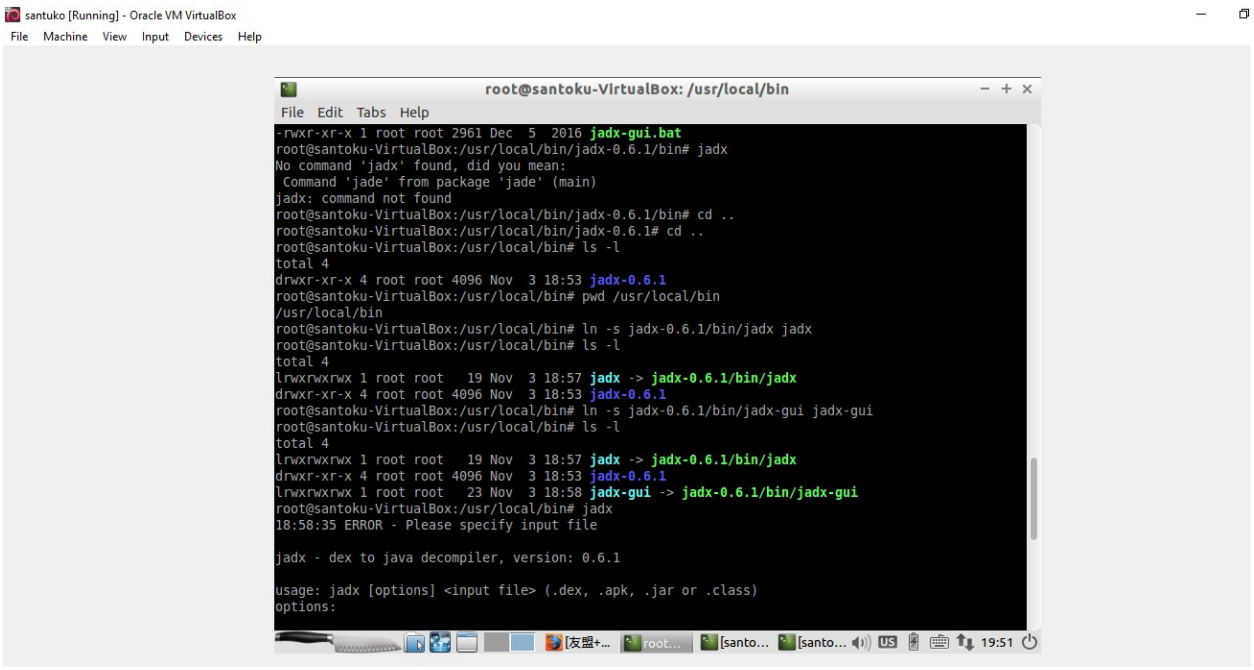
Step 10:

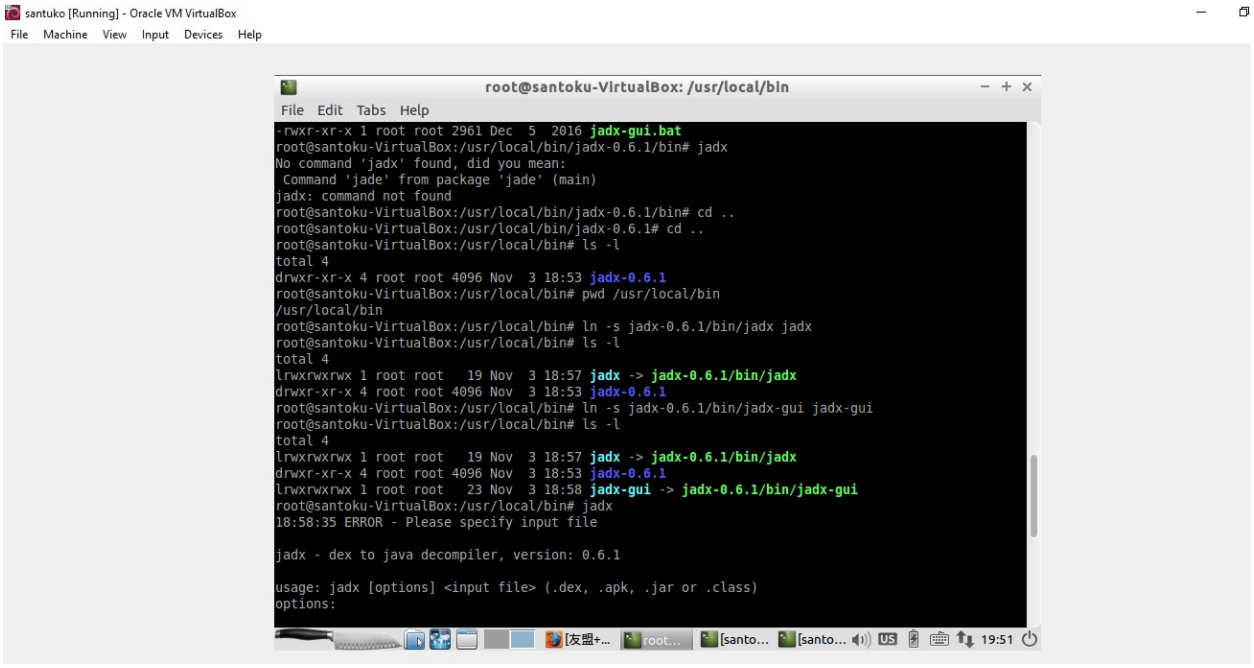
Installation of application logos:
We create the application logos by downloading from the links below:
http://alog.umeng.com/app_logs
http://alog.umengcloud.com/app_logs

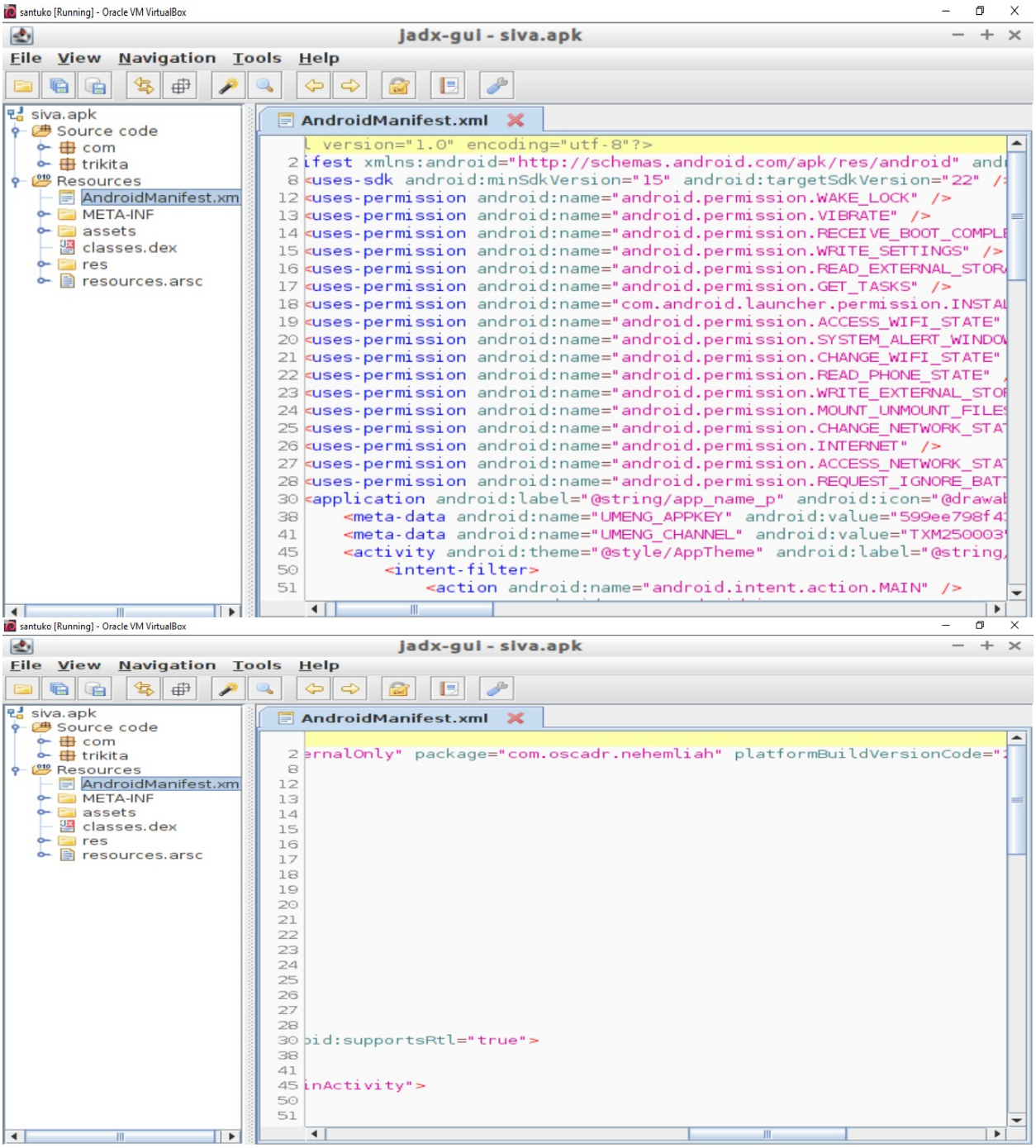


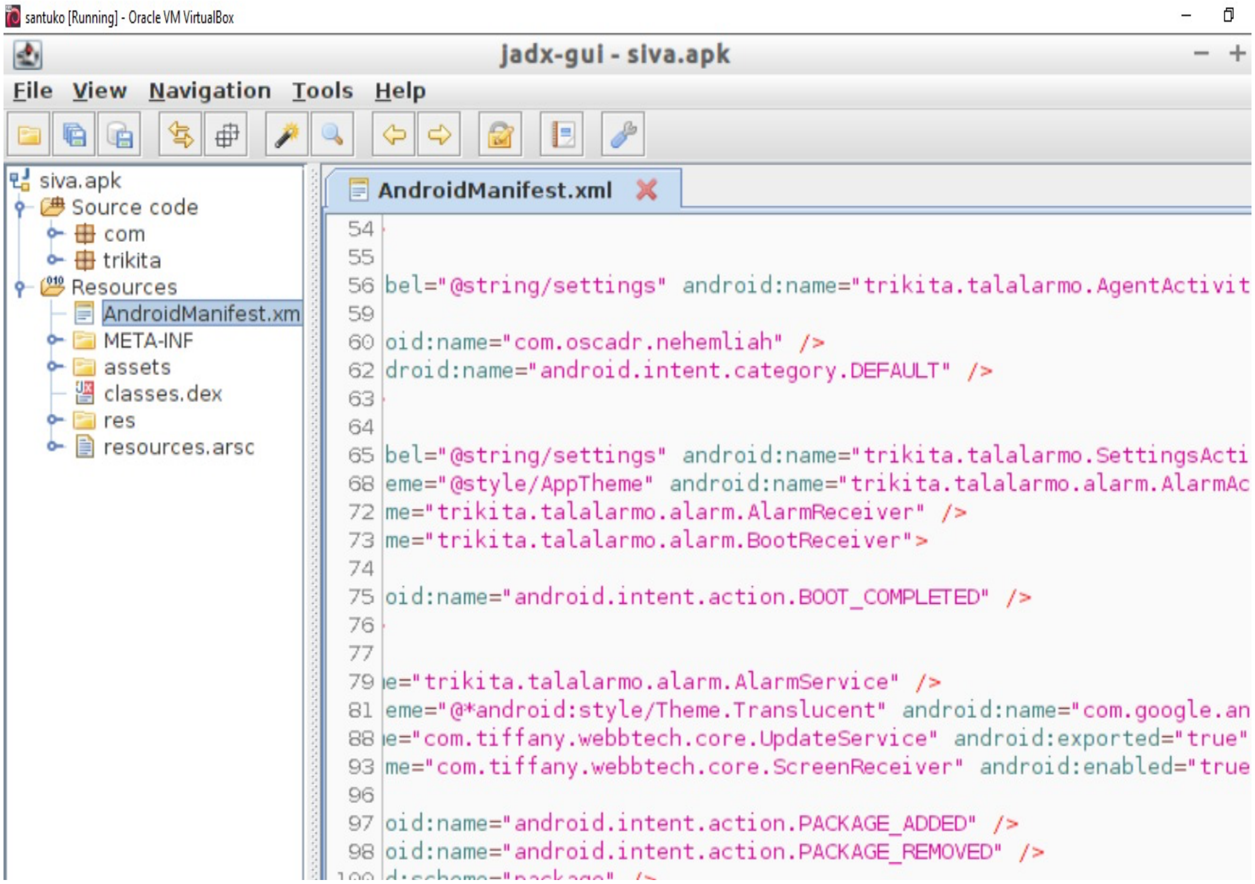
Step 11:

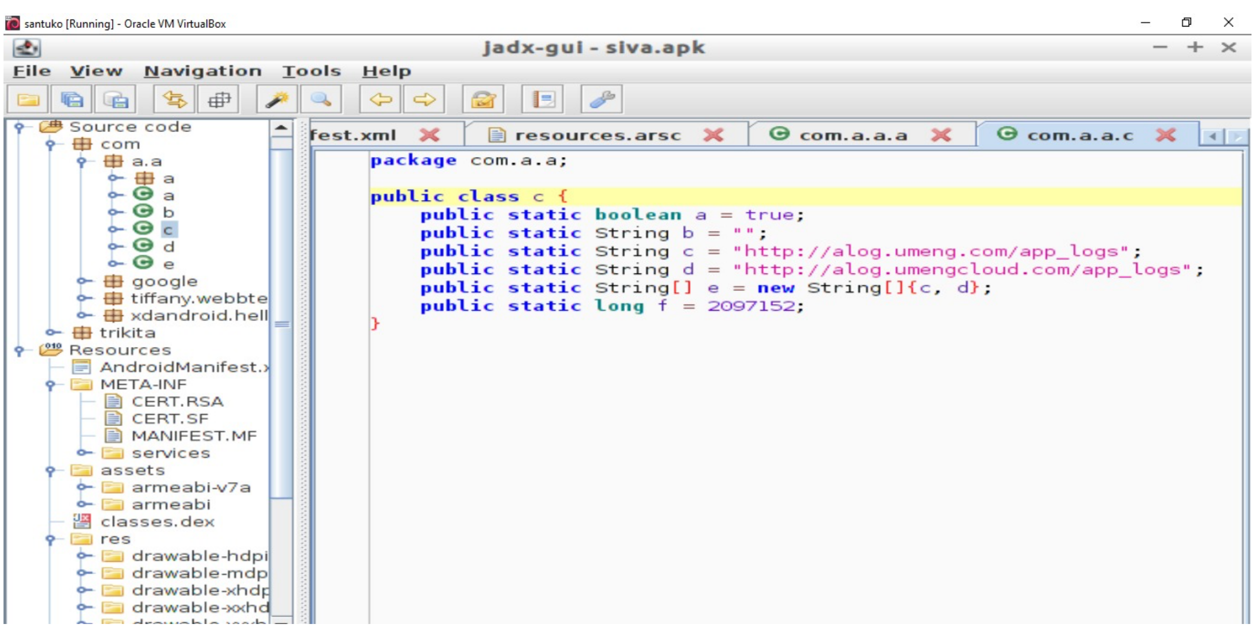
By using the command jadx-gui, we get the siva.apk application. In that application we see the source code and resources, androidmanifest.xml, clsses.dex, resources.arsc.





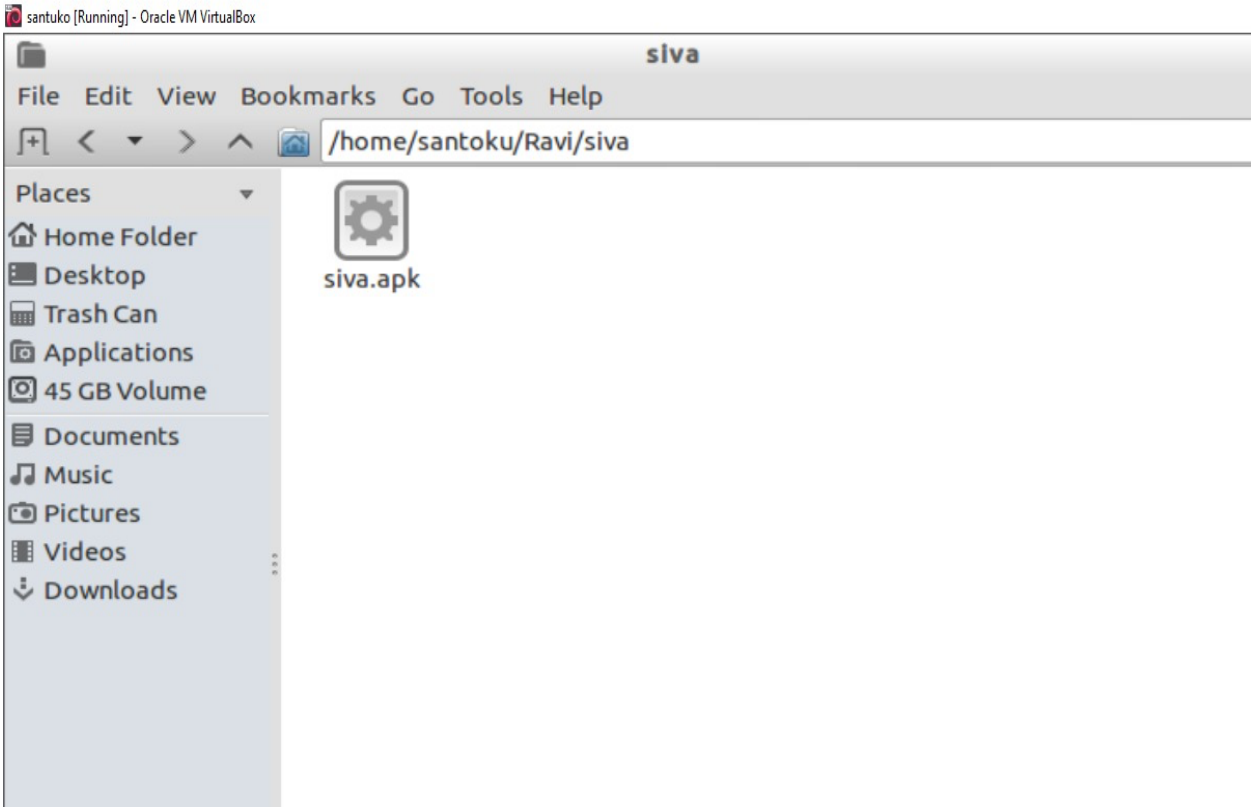

santuko [Running] - Oracle VM VirtualBox
jadx-gui - siva.apk
File View Navigation Tools Help
siva.apk
Source code
com
trikita
Resources
AndroidManifest.xm
META-INF
assets
classes.dex
res
resources.arsc
AndroidManifest.xml
l version="1.0" encoding="utf-8"?>
ifest xmlns:android="http://schemas.android.com/apk/res/android" andr
<uses-sdk android:minSdkVersion="15" android:targetSdkVersion="22" />
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="android.permission.VIBRATE" />
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLE
<uses-permission android:name="android.permission.WRITE_SETTINGS" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STOR
<uses-permission android:name="android.permission.GET_TASKS" />
<uses-permission android:name="com.android.launcher.permission.INSTAL
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDO
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE"
<uses-permission android:name="android.permission.READ_PHONE_STATE"
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STOR
<uses-permission android:name="android.permission.MOUNT_UNMOUNT_FILES
<uses-permission android:name="android.permission.CHANGE_NETWORK_STA
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STA
<uses-permission android:name="android.permission.REQUEST_IGNORE_BAT
<application android:label="@string/app_name_p" android:icon="@drawab
<meta-data android:name="UMENG_APPKEY" android:value="599ee798f4
<meta-data android:name="UMENG_CHANNEL" android:value="TXM250003
<activity android:theme="@style/AppTheme" android:label="@string/
<intent-filter>
<action android:name="android.intent.action.MAIN" />
santuko [Running] - Oracle VM VirtualBox
jadx-gui - siva.apk
File View Navigation Tools Help
siva.apk
Source code
com
trikita
Resources
AndroidManifest.xm
META-INF
assets
classes.dex
res
resources.arsc
AndroidManifest.xml
ernalOnly" package="com.oscadr.nehemliah" platformBuildVersionCode="
oid:supportsRtl="true">
inActivity">


santuko [Running] - Oracle VM VirtualBox
jadx-gui - siva.apk
File View Navigation Tools Help
siva.apk
Source code
com
trikita
Resources
AndroidManifest.xm
META-INF
assets
classes.dex
res
resources.arsc
AndroidManifest.xml
bel="@string/settings" android:name="trikita.talalarmo.AgentActivit
oid:name="com.oscadr.nehemliah" />
droid:name="android.intent.category.DEFAULT" />
bel="@string/settings" android:name="trikita.talalarmo.SettingsActi
eme="@style/AppTheme" android:name="trikita.talalarmo.alarm.AlarmAc
me="trikita.talalarmo.alarm.AlarmReceiver" />
me="trikita.talalarmo.alarm.BootReceiver">
oid:name="android.intent.action.BOOT_COMPLETED" />
e="trikita.talalarmo.alarm.AlarmService" />
eme="@*android:style/Theme.Translucent" android:name="com.google.an
e="com.tiffany.webbtech.core.UpdateService" android:exported="true"
me="com.tiffany.webbtech.core.ScreenReceiver" android:enabled="true
oid:name="android.intent.action.PACKAGE_ADDED" />
oid:name="android.intent.action.PACKAGE_REMOVED" />


santuko [Running] - Oracle VM VirtualBox
jadx-gui - siva.apk
File View Navigation Tools Help
Source code
com
a.a
a
a
b
c
d
e
google
tiffany.webbte
xdandroid.hell
trikita
Resources
AndroidManifest.x
META-INF
CERT.RSA
CERT.SF
MANIFEST.MF
services
assets
armeabi-v7a
armeabi
classes.dex
res
drawable-hdpi
drawable-mdp
drawable-xhdp
drawable-xxhd
fest.xml
resources.arsc
com.a.a.a
com.a.a.c
package com.a.a;
public class c {
public static boolean a = true;
public static String b = "";
public static String c = "http://alog.umeng.com/app_logs";
public static String d = "http://alog.umengcloud.com/app_logs";
public static String[] e = new String[]{c, d};
public static long f = 2097152;
}

Step 12:

The created malicious application is:



Step 13:

The secret code is **Texas tech**