




# flagstore.apk移动逆向writeup

原创

b0ring  于 2017-10-10 22:10:35 发布  826  收藏

分类专栏: [CTF\\_RE](#) 文章标签: [移动 java](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: <https://blog.csdn.net/s1054436218/article/details/78198257>

版权



[CTF\\_RE 专栏收录该内容](#)

4 篇文章 3 订阅

订阅专栏

世安杯遇到的一道逆向题, 感觉还挺有意思的, 解决问题中遇到了很多问题 (其实是装工具时碰到的问题), 贴出来跟大家分享一下。

题目可以在这个地址下载:

链接: <http://pan.baidu.com/s/1qYv7ALA> 密码: rg7d

首先用jadx-gui查看一下java代码, MainActivity中代码如下:

```
package com.flagstore.ctf.flagstore;

import android.app.Activity;
import android.content.IntentFilter;
import android.os.Bundle;
import android.widget.TextView;
import com.flagstore.ctf.flagstore.Manifest.permission;

public class MainActivity extends Activity {
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        TextView tv = new TextView(getApplicationContext());
        tv.setText("To-do: UI pending");
        setContentView(tv);
        IntentFilter filter = new IntentFilter();
        filter.addAction("com.flagstore.ctf.INCOMING_INTENT");
        registerReceiver(new Send_to_Activity(), filter, permission._MSG, null);
    }
}
```

它一开始就是把程序的界面弄成一个文本: "To-do: UI pending", 然后设置了一个Broadcast监听, 关于Broadcast机制, 可以看这个文章:

<http://www.cnblogs.com/playing/archive/2011/03/23/1992030.html>

然后我们再看Send\_to\_Activity这个类:

```
package com.flagstore.ctf.flagstore;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.util.Log;
import android.widget.Toast;

public class Send_to_Activity extends BroadcastReceiver {
    public void onReceive(Context context, Intent intent) {
        if (intent.getStringExtra("msg").equalsIgnoreCase("OpenSesame")) {
            Log.d("Here", "Intent");
            context.startActivity(new Intent(context, CTFReceiver.class));
            return;
        }
        Toast.makeText(context, "Ah, ah, ah, you didn't say the magic word!", 1).show();
    }
}
```

代码的意思很简单，就是如果受到msg为OpenSesame的话，就会激活CTFReceiver这个类：

```

package com.flagstore.ctf.flagstore;

import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;

public class CTFReceiver extends AppCompatActivity {

    class C01581 implements OnClickListener {
        C01581() {
        }

        public void onClick(View v) {
            Intent intent = new Intent();
            intent.setAction("com.flagstore.ctf.OUTGOING_INTENT");
            String a = CTFReceiver.this.getResources().getString(C0159R.string.str3) + "fpcMpwfFurWGlWu`uDl
            String b = Utilities.doBoth(CTFReceiver.this.getResources().getString(C0159R.string.passphrase)
            String name = getClass().getName().split("\\.")[4];
            intent.putExtra("msg", CTFReceiver.this.getPhrase(a, b, Utilities.doBoth(name.substring(0, name
            CTFReceiver.this.sendBroadcast(intent);
        }
    }

    public native String getFlag(String str, String str2, String str3);

    public native String getPhrase(String str, String str2, String str3);

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        new TextView(this).setText("Clever Person!");
        Button button = new Button(this);
        button.setText("Broadcast");
        setContentView((View) button);
        button.setOnClickListener(new C01581());
    }

    static {
        System.loadLibrary("native-lib");
    }
}

```

逻辑大概就是一旦这个被激活，点击一下按钮，他就会发一个广播，于是我们的一个解决思路就是：打开APP后发送一个广播为“OpenSesame”，然后再监听接收。

然后用到一些工具：

[drozer](#)

[Genymotion](#)

两个工具安装过程中都遇到了一些问题，Genymotion在网上有很多安装教程，说两个值得一提的问题吧：

1、因为Genymotion需要安装Virtualbox作为依赖，但是Virtualbox可能会提示你缺少headers，但是你apt-get不到，网上的教程中没有几个提到的，直接搜这个问题也有很多让你apt-get的，但这其实是内核问题，你需要升级一下内核，再安装一下headers，具体操作如下：

```
apt-cache search linux-image|grep kali
```

然后就获得了一个内核列表，我这边结果如下：

```
linux-headers-4.13.0-kali1-amd64 - Header files for Linux 4.13.0-kali1-amd64
linux-image-4.13.0-kali1-amd64 - Linux 4.13 for 64-bit PCs
linux-image-4.13.0-kali1-amd64-dbg - Debug symbols for linux-image-4.13.0-kali1-amd64
linux-headers-4.13.0-kali1-686 - Header files for Linux 4.13.0-kali1-686
linux-headers-4.13.0-kali1-686-pae - Header files for Linux 4.13.0-kali1-686-pae
linux-image-4.13.0-kali1-686 - Linux 4.13 for older PCs
linux-image-4.13.0-kali1-686-dbg - Debug symbols for linux-image-4.13.0-kali1-686
linux-image-4.13.0-kali1-686-pae - Linux 4.13 for modern PCs
linux-image-4.13.0-kali1-686-pae-dbg - Debug symbols for linux-image-4.13.0-kali1-686-pae
linux-image-4.6.0-kali1-amd64 - Linux 4.6 for 64-bit PCs
linux-headers-4.12.0-kali2-amd64 - Header files for Linux 4.12.0-kali2-amd64
linux-image-4.12.0-kali2-amd64 - Linux 4.12 for 64-bit PCs
```

然后apt-get下一样版本号和32位或64位的headers和image，比如：

```
apt-get install linux-headers-4.13.0-kali1-amd64
apt-get install linux-image-4.13.0-kali1-amd64
```

然后重启一下，就可以了。

2、第二个问题解决比较简单，它说你的CPU不支持虚拟化，这个问题是在虚拟机中运行时遇到的问题，在设置—处理器的虚拟化模式里选择虚拟化 Intel VT-x/EPT 或 AMD-V/RVI，就解决了。

drozer遇到的问题其实挺脑残的，按照readme中的方法安装完依赖包并且build以后，它很多文件夹都是隐藏的，于是我当时挺懵逼的，以为没生成，其实生成的就是你python setup.py xxx,安装文件就在xxx目录里，然后安装就行了。

随便下载一个安卓手机的虚拟机，然后运行，用adb查看并注入devices：

```
root@kali: adb devices
List of devices attached
192.168.57.101:5555 device
```

工具安装就可以秒杀这道题了，首先build一下drozer的agent，就是放在安卓虚拟机中监听的软件：

```
/opt/genymobile/genymotion# drozer agent build
I: Using Apktool 2.2.4 on standard-agent.apk
.....
Done: /tmp/xxxxxx/agent.apk
```

然后直接adb install一波，题目和agent.apk:

```
root@kali:adb install /tmp/xxxxxx/agent.apk
Success
root@kali:adb install flagstore.apk
Success
```

然后在手机中打开agent，然后连接一下:

```
root@kali:adb forward tcp:31415 tcp:31415
root@kali:drozer console connect
Selecting 505c86a4845f7221 (Genymotion Samsung Galaxy S8 - 7.0.0 - API 24 - 1440x2960 7.0)

..                ...
..o..             .r..
..a.. . . . . . . . .nd
  ro..idsnemesisan..pr
  .otectorandroidsneme.
  .,sisandprotectorandroids+.
  ..nemesisanprotectorandroidsn:.
  .emesisandprotectorandroidsnemes..
  ..isandp,..,rotectorandro,..,idsnem.
  .isisandp..rotectorandroid..snemisis.
  ,andprotectorandroidsnemisisandprotec.
  .torandroidsnemesisandprotectorandroid.
  .snemisisandprotectorandroidsnemesisan:
  .dprotectorandroidsnemesisandprotector.

drozer Console (v2.4.3)
dz> run app.broadcast.sniff --action "com.flagstore.ctf.OUTGOING_INTENT"
```

然后在新窗口中发送广播:

```
root@kali: adb shell
vbox86p:/ # su
vbox86p:/ # -a "com.flagstore.ctf.INCOMING_INTENT" --es msg "OpenSesame"
```

然后旧窗口就收到了flag:

```
Action: com.flagstore.ctf.OUTGOING_INTENT  
Raw: Intent { act=com.flagstore.ctf.OUTGOING_INTENT flg=0x10 (has extras) }  
Extra: msg=CongratsGoodWorkYouFoundItIHopeYouUsedADBFlag:TheseIntentsAreFunAndEasyToUse (java.lang.Stri
```