某新生院赛CTF 移动题writeup

原创

xbalien 于 2014-05-22 11:41:33 发布 9635 收藏
分类专栏： 【CTF】 文章标签： CTF android逆向分析

题目提供了一个apk，常规考察内容一般为逆向、挖洞~

1.程序初步分析

获取一个apk后第一件事情肯定是运行，看看题目到底是要干嘛，是要逆向分析算法，还是利用一些Androd机制。截图如下：

程序只有一个按钮，运行之后出现了爱的提示，那意味着这道题很可能不是逆向算法找出flag，很有可能是让我们利用一些其他手段。那么也得要了解程序大致逻辑吧，没有源码只能逆向。

我的思路一般是先找入口点，包括main activity，broadcast receiver等等，目测只有一个main入口

main activity啥也没干，就把爱放了出来~但是这道题目算是给新生的入门题目，程序代码目录下都出现了很多爱，包括这么一个类com.rois.mobi.RightWay

也就是说，只要我能分析出这个方法具体干了什么flag就迎刃而解了。该方法是获取了自己报名的签名，去前面16字节尽心一个sm4算法，最后算出flag，一起输出~因为该方法跟本没法调用。所以这题本意就不是什么分析算法了，题目也放了一个爱的提示：师傅密码学不好。所以方法基本上应该锁定在如何调用这个方法。也就是出这题目的本意：动态加载。当然，后面自己做这道题目的时候发现了一种更为简单的党法，就是修改静态函数也可以做到~

2.解题分析

既然知道了做题思路，那么可以开始动手了。还是先上最简单的方法吧。

（1）把com.rois.mobi.RightWay.rightGetKey变成静态方法

于是使用VTS打开apk，找到类smail，原型签名为Lcom/rois/mobi/right/RightWay;->rightGetKey(Landroid/content/Context;)V，将其修改为静态类，成为静态类之后，就不存在p0表示this指针，因此，现在所有p0表示的是参数Landroid/content/Context;，原来表示该参数的p1就不存在了；之前java源码中，该方法中所有this.成员也要修改为static，在这个程序中是
改：.field private static mPm:Landroid/content/pm/PackageManager;，因此下面所有所有涉及mPm的使用都要由i系列的操作改为s系列，例如：iget-object修改为 sget-object（ 实例操作变为类操作）。稍微小调整一下就可以了，最后该方法代码为（可以原来程序smali对比）：

```
.method public static rightGetKey(Landroid/content/Context;)V
    .locals 14
    .parameter "context"

    .prologue
    const/16 v13, 0x10

    const/4 v5, 0x0

    .line 66
    const-string v10, "ROISMobi2"

    const-string v11, "go go check out"

    invoke-static {v10, v11}, Landroid/util/Log;->i(Ljava/lang/String;Ljava/lang/String;)I

    .line 67
    new-instance v0, Lcom/rois/mobi/uitls/Do;

    invoke-direct {v0}, Lcom/rois/mobi/uitls/Do;-><init>()V

    .line 69
    .local v0, cy:Lcom/rois/mobi/uitls/Do;
    const/4 v9, 0x0

    .line 70
    .local v9, str:Ljava/lang/String;
    new-array v4, v13, [B

    .line 71
    .local v4, out:[B
    new-array v1, v13, [B

    .line 72
    .local v1, in:[B
    invoke-virtual {p0}, Landroid/content/Context;->getPackageManager()Landroid/content/pm/PackageManager;

    move-result-object v10

    sput-object v10, Lcom/rois/mobi/right/RightWay;->mPm:Landroid/content/pm/PackageManager;

    .line 73
    const/4 v8, 0x0

    .line 75
    .local v8, signatures:[Landroid/content/pm/Signature;
    :try_start_0
    sget-object v10, Lcom/rois/mobi/right/RightWay;->mPm:Landroid/content/pm/PackageManager;

    const-string v11, "com.rois.mobi"

……
.end method
```

根据方法的原型签名：Lcom/rois/mobi/right/RightWay;->rightGetKey(Landroid/content/Context;)V知道需要要传递一个Context参数，在main activity中，p0正好合适。于是在main activity中加入：

invoke-static {p0}, Lcom/rois/mobi/right/RightWay;->rightGetKey(Landroid/content/Context;)V

这样程序就完成了修改，可以运行出flag了。

（2）动态加载

动态加载是搞android安全必须要了解的，利用类加载器，反射，可以实现对未加载类的加载，相关函数的调用，可以做很多是事情。了解了简单的使用方法后，就可以编程实现了。代码内容简单。

```
protected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.activity_first);
  sendButton = (Button)findViewById(R.id.sendSMS);
  sendButton.setOnClickListener(new OnClickListener() {
   @SuppressLint("NewApi")
   @Override
   public void onClick(View v) {
    DexClassLoader apk = new DexClassLoader(Environment.getExternalStorageDirectory().getAbsolutePath()+"/R
      getApplicationContext().getFilesDir().getAbsolutePath(),null,ClassLoader.getSystemClassLoader());
    try {
     Class<?> rw = apk.loadClass("com.rois.mobi.right.RightWay");
     Constructor<?> rwConstructor = rw.getConstructor();

     Method rightGetKeyMid = rw.getMethod("rightGetKey",android.content.Context.class);
     rightGetKeyMid.setAccessible(true);
     Object rwInstance = rwConstructor.newInstance();

     rightGetKeyMid.invoke(rwInstance,MainActivity.this);

    } catch (ClassNotFoundException e) {
     e.printStackTrace();
    } catch (InstantiationException e) {
     e.printStackTrace();
    } catch (IllegalAccessException e) {
     e.printStackTrace();
    } catch (NoSuchMethodException e) {
     e.printStackTrace();
    } catch (IllegalArgumentException e) {
     e.printStackTrace();
    } catch (InvocationTargetException e) {
     e.printStackTrace();
    }
   }
  });
 }
```

这个就是当时题目的本意。不过能解出来，什么方法都是好方法