Android逆向writeup,[原创]腾讯apk逆向系列WriteUp

转载

刘月岐 于 2021-05-31 10:52:57 发布 94 收藏
文章标签： Android逆向writeup

0x00 前言

正在学习安卓逆向的萌新SR绝赞刷题中，昨天做了三道有意思的题目，感觉很适合入门，于是写了个wp发了出来

题目已上传至文章附件，想摸的dalao们可以看看(

0x01 APP1

工具：JADX，安卓设备/虚拟机

安装并打开app，是一个简单的 输入-校验 式app

我们用JADX打开apk，看一下反编译代码

首先打开xml文件，这里写明了app启动时打开的Activity，确定是MainActivity后转到反编译代码处

```
        while (i < inputString.length() && i < versionCode.length()) {
            if (inputString.charAt(i) != (versionCode.charAt(i) ^ versionName)) {
                Toast.makeText(MainActivity.this, "再接再厉，加油~", 1).show();
                return;
            }
            i++;
        }
        if (inputString.length() == versionCode.length()) {
            Toast.makeText(MainActivity.this, "恭喜开启闯关之门！", 1).show();
            return;
        }
    } catch (PackageManager.NameNotFoundException e) {
    }
    Toast.makeText(MainActivity.this, "年轻人不要耍小聪明噢", 1).show();
}
```

如图，很容易确定代码逻辑——简单的异或操作

flag即为"versionName"的各个字符与"versionCode"进行异或操作的结果

```
package com.example.yaphetshan.tencentgreat;

public final class BuildConfig {
    public static final String APPLICATION_ID = "com.example.yaphetshan.tencentgreat";
    public static final String BUILD_TYPE = "debug";
    public static final boolean DEBUG = Boolean.parseBoolean("true");
    public static final String FLAVOR = "";
    public static final int VERSION_CODE = 15;
    public static final String VERSION_NAME = "X<cP[?PHNB<P?aj";
}
```

跟进代码，找到需要的两个常量，写解题脚本

得到flag：W3l_T0_GAM3_0ne

本题结束

0x02 APP2

工具：JADX，安卓设备/虚拟机，IDA，adb

如同分析APP1一般，我们安装并打开APP2，同时使用JADX进行反编译

APP2界面上要求填写账号密码，但无论什么账号密码都有效，影响不大

对反编译代码进行分析，发现无论我们怎么输入，都会打开SecondActivity界面

同时发现代码中有一段校验

```
public void onCreate(Bundle bundle) {
    super.onCreate(bundle);
    setContentView(R.layout.activity_main2);
    Intent intent = getIntent();
    String stringExtra = intent.getStringExtra("ili");
    String stringExtra2 = intent.getStringExtra("lil");
    if (Encryto.doRawData(this, stringExtra + stringExtra2).equals("VEIzd/V2UPYNdn/bxH3Xig==")) {
        intent.setAction("android.test.action.MoniterInstallService");
        intent.setClass(this, MoniterInstallService.class);
        intent.putExtra("company", "tencent");
        intent.putExtra("name", "hacker");
        intent.putExtra("age", 18);
        startActivity(intent);
        startService(intent);
    }
    SharedPreferences.Editor edit = getSharedPreferences("test", 0).edit();
    edit.putString("ilil", stringExtra);
    edit.putString("lili", stringExtra2);
    edit.commit();
}
```

我们跟进加密方法，发现是个native方法，即利用JNI，调用写好在so文件中的函数

使用IDA打开so文件

```
int __cdecl doRawData(int a1, int a2, int a3, int a4)
{
  char *v4; // esi
  int result; // eax
  char *v6; // esi
  size_t v7; // eax
  int v8; // [esp+0h] [ebp-2Ch]
  int (__cdecl *v9)(int, char *, size_t); // [esp+0h] [ebp-2Ch]
  char v10[20]; // [esp+4h] [ebp-28h] BYREF
  unsigned int v11; // [esp+18h] [ebp-14h]

  v11 = __readgsdword(0x14u);
  if ( checkSignature(a1, a2, a3) == 1 )
  {
    strcpy(v10, "thisisatestkey==");
    v4 = (char *)(*(int (__cdecl **)(int, int, _DWORD))(*(_DWORD *)a1 + 676))(a1, a4, 0);
    v8 = AES_128_ECB_PKCS5Padding_Encrypt(v4, (int)v10);
    (*(void (__cdecl **)(int, int, char *))(*(_DWORD *)a1 + 680))(a1, a4, v4);
    result = (*(int (__cdecl **)(int, int))(*(_DWORD *)a1 + 668))(a1, v8);
  }
  else
  {
    v6 = UNSIGNATURE[0];
    v9 = *(int (__cdecl **)(int, char *, size_t))(*(_DWORD *)a1 + 652);
    v7 = strlen(UNSIGNATURE[0]);
    result = v9(a1, v6, v7);
  }
  return result;
}
```

从关键字AES_128_ECB_PKCS5Padding中即可知道加密方式

使用在线工具解密字符串，即可获得flag：“aimagetencent”

不过，得到的flag是错误的，我们继续分析代码

观察左侧的几个类，我们发现了第三个Activity：FileDataActivity

进入后发现存在使用了相同方式加密的字符串

```java
public class FileDataActivity extends BaseActivity {

    /* renamed from: c */
    private TextView f13c;

    /* access modifiers changed from: protected */
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_main3);
        this.f13c = (TextView) findViewById(R.id.textView1);
        this.f13c.setText(Encryto.decode(this, "9YuQ2dk8CSaCe7DTAmaqAA=="));
    }
}
```

解密后获得真正的flag：Cas3_0f_A_CAK3

提交后，考虑到我们在app中从未进入过FileDataActivity，决定打开这个Act一探究竟

使用adb自带的am命令(adb配置与连接的方法不多赘述)

手机app便跳转到了这个Activity，如图所示

使用这个方法，同样可以获得flag

本题结束

0x03 APP3

工具：JADX，DB Browser for SQLite，android-backup-extractor

题目文件是ab文件，即adb的备份文件，我们使用android-backup-extractor工具将其解压

压缩包内能找到apk文件和两个数据库文件

使用JADX打开apk，发现反编译出来的代码有点复杂(

我们还是跟踪到MainActivity处进行分析

```java
/* renamed from: a */
private void m20a() {
    SQLiteDatabase.loadLibs(this);
    this.f33b = new DatabaseManager(this, "Demo.db", (SQLiteDatabase.CursorFactory) null, 1);
    ContentValues contentValues = new ContentValues();
    contentValues.put("name", "Stranger");
    contentValues.put("password", 123456);
    Cipher aVar = new Cipher();
    String a = aVar.mo6317a(contentValues.getAsString("name"), contentValues.getAsString("password"));
    this.f32a = this.f33b.getWritableDatabase(aVar.mo6316a(a + aVar.mo6318b(a, contentValues.getAsString("password"))).substring(0, 7));
    this.f32a.insert("TencentMicrMsg", (String) null, contentValues);
}
```

关键字demo.db告诉我们，要找的关键代码就在这里，简单阅读了下代码逻辑后得出以下结论：

程序对Strange和123456字符串进行了一些操作，作为数据库的密码

至于进行了什么操作，我们慢慢分析

首先，调用了Cipher类的"mo6317a"函数(此处由JADX的反混淆功能生成，不同设备可能有所不同)，该函数的作用是取两个字符串的前四个字符进行拼接

即"Stra1234"

然后以得到的字符串为参数，调用了Cipher类的"mo6318b"函数，跟踪过去发现，该函数进行了MD5和base16操作

最后，在加密结果的首部填上"Stra1234"，以此为参数调用”mo6316a“函数，该函数的作用是：在参数尾部接上”yaphetshan“，进行SHA-1和base16操作

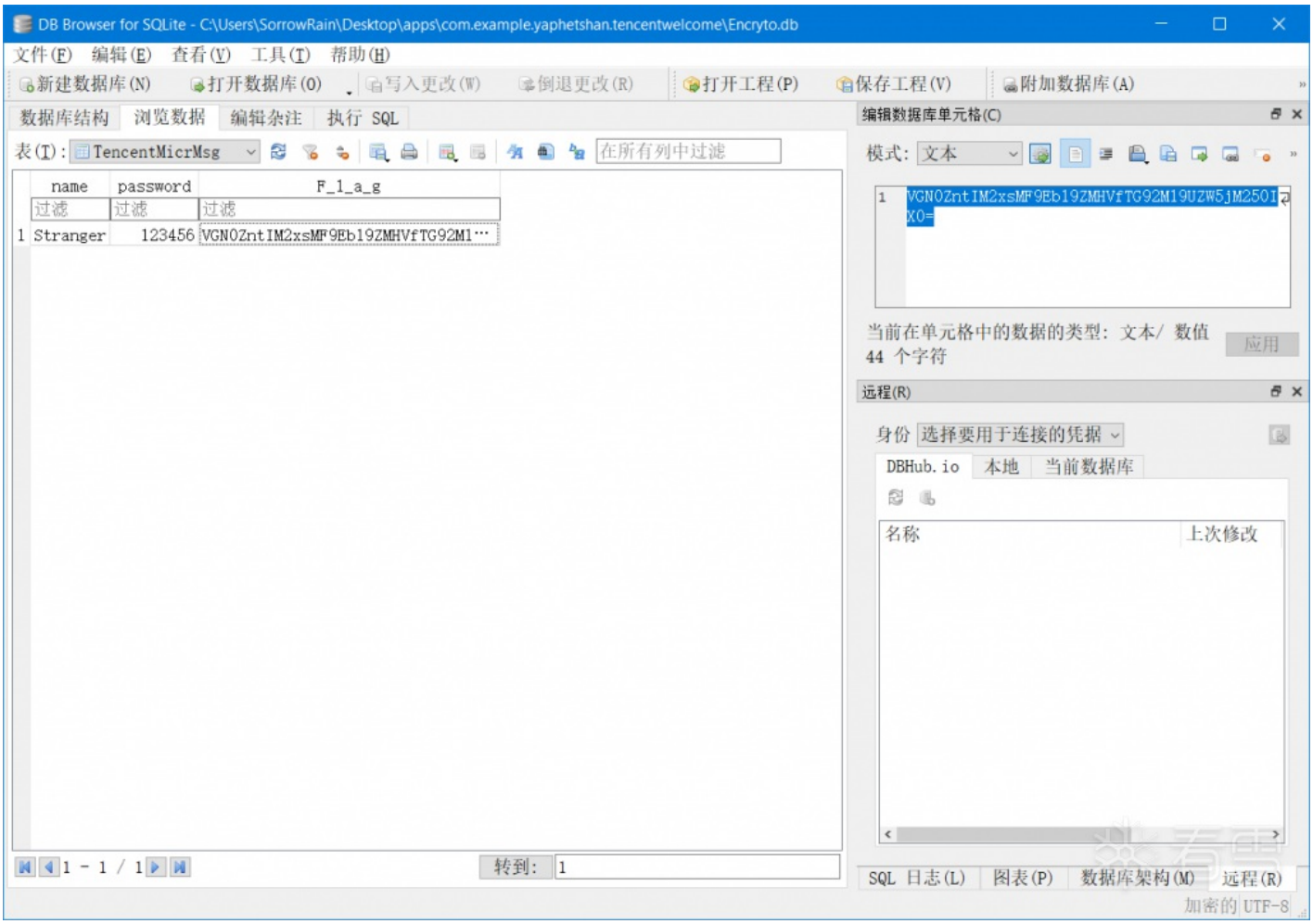
得到结果的前八位即为数据库密码：ae56f99

```
    /* renamed from: a */
    private String f36a = "yaphetshan";

    /* renamed from: a */
    public String mo6317a(String str, String str2) {
        String substring = str.substring(0, 4);
        return substring + str2.substring(0, 4);
    }

    /* renamed from: b */
    public String mo6318b(String str, String str2) {
        new SHA1Manager();
        return SHA1Manager.m24a(str);
    }

    /* renamed from: a */
    public String mo6316a(String str) {
        new SHA1Manager();
        return SHA1Manager.m25b(str + this.f36a);
    }
}
```

```java
public class SHA1Manager {
    /* renamed from: a */
    public static final String m24a(String str) {
        char[] cArr = {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b', 'c', 'd', 'e', 'f'};
        try {
            byte[] bytes = str.getBytes();
            MessageDigest instance = MessageDigest.getInstance("MD5");
            instance.update(bytes);
            char[] cArr2 = new char[(r4 * 2)];
            int i = 0;
            for (byte b : instance.digest()) {
                int i2 = i + 1;
                cArr2[i] = cArr[(b >>> 4) & 15];
                i = i2 + 1;
                cArr2[i2] = cArr[b & 15];
            }
            return new String(cArr2);
        } catch (Exception e) {
            return null;
        }
    }

    /* renamed from: b */
    public static final String m25b(String str) {
        char[] cArr = {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b', 'c', 'd', 'e', 'f'};
        try {
            byte[] bytes = str.getBytes();
            MessageDigest instance = MessageDigest.getInstance("SHA-1");
            instance.update(bytes);
            char[] cArr2 = new char[(r4 * 2)];
            int i = 0;
            for (byte b : instance.digest()) {
                int i2 = i + 1;
                cArr2[i] = cArr[(b >>> 4) & 15];
                i = i2 + 1;
                cArr2[i2] = cArr[b & 15];
            }
            return new String(cArr2);
        } catch (Exception e) {
            return null;
        }
    }
}
```


上方两个图片为Cipher类及其调用的加密类的代码，可以对照查看

使用DB Browserd打开db文件，输入密码，得到flag



将其进行base64解密，拿到最终的flag：Tctf{H3ll0_Do_Y0u_Lov3_Tenc3nt!}

本题结束

0x04 结语

若有疑问可以在评论区提出！感谢各位的阅读与支持啦！

最后于 2021-4-8 17:41

被郁雨编辑

，原因：

上传的附件：

app1.apk

(1.54MB，29次下载)

app2.apk

(593.97kb，18次下载)

app3.ab

(8.43MB，22次下载)