安卓逆向学习 之 KGB Messenger的writeup（2)

原创

CowboyBeboppp 于 2021-05-20 20:26:44 发布 125 收藏 1

分类专栏： 安卓逆向 文章标签： 安全

本文链接：https://blog.csdn.net/weixin_38109420/article/details/116764373

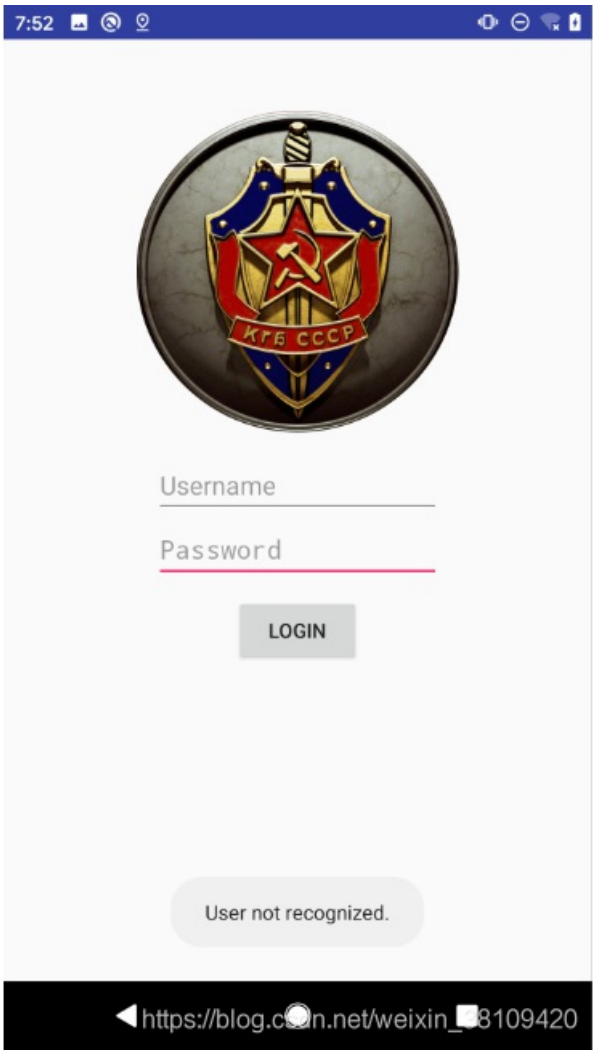
安卓逆向学习之KGB Messenger的writeup（2）

Login (Easy)

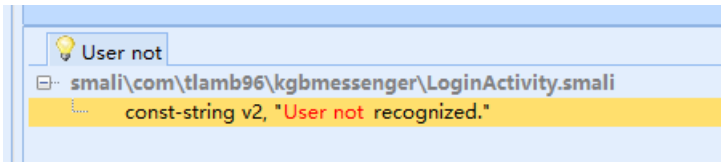
接上一题，去掉了烦人的弹窗后来到一个登录界面，现在要成功登录，并且登录密码全是小写。

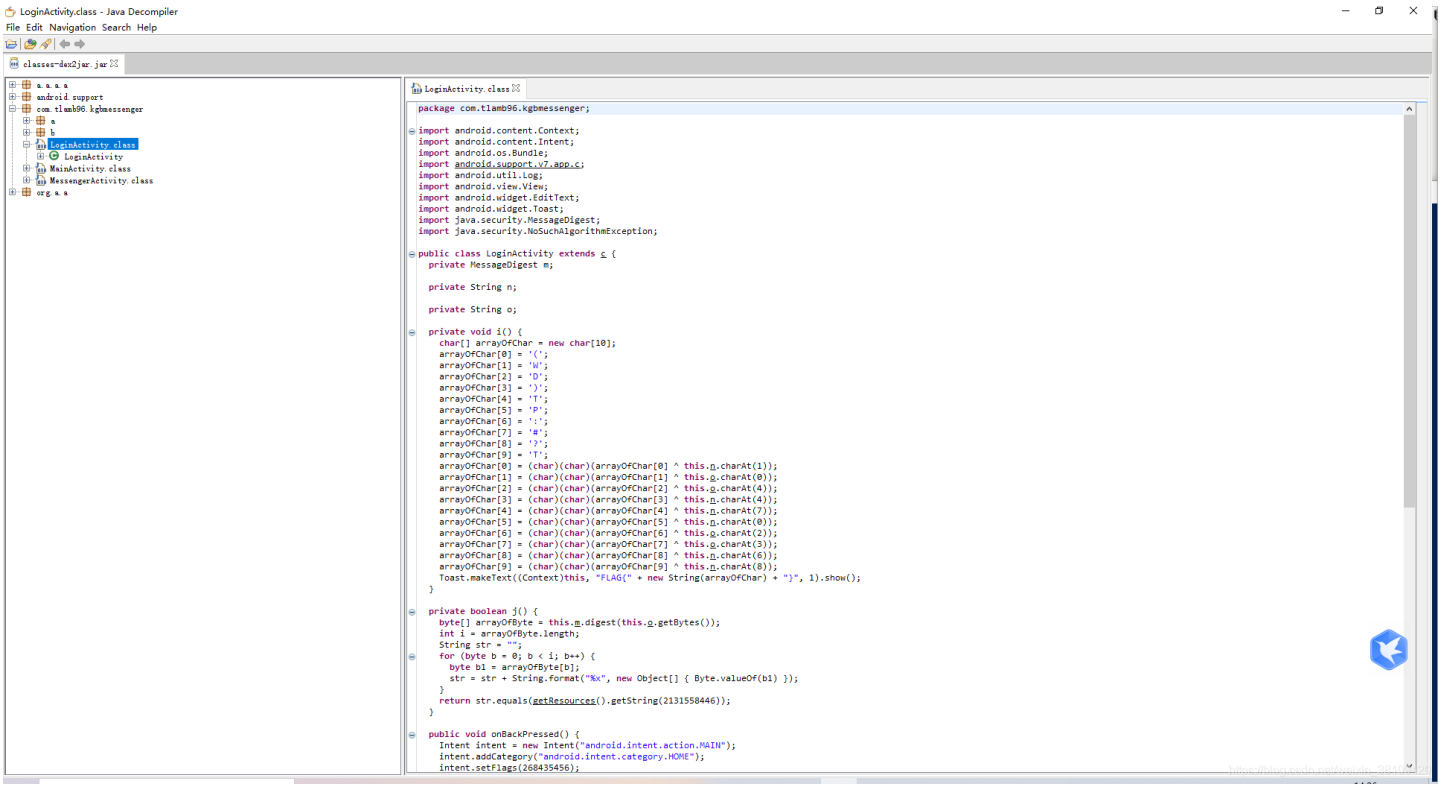
思路

首先随便输入账号密码点击login按钮会出现下面的字样：



在AndroidKiller中搜索字符串“User not recognized”,发现在在LoginActivity中，反编译用jd-gui打开后进行分析：


User not
smali\com\tlamb96\kgbmessenger\LoginActivity.smali
const-string v2, "User not recognized."


LoginActivity.class - Java Decompiler
File Edit Navigation Search Help
classes-dex2jar.jar
a.a.a.a
android.support
com.tlamb96.kgbmessenger
a
b
LoginActivity.class
LoginActivity
MainActivity.class
MessengerActivity.class
org.a.a
LoginActivity.class
package com.tlamb96.kgbmessenger;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.c;
import android.util.Log;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
public class LoginActivity extends c {
private MessageDigest m;
private String n;
private String o;
private void i() {
char[] arrayOfChar = new char[10];
arrayOfChar[0] = '(';
arrayOfChar[1] = 'W';
arrayOfChar[2] = 'D';
arrayOfChar[3] = ')';
arrayOfChar[4] = 'T';
arrayOfChar[5] = 'P';
arrayOfChar[6] = ':';
arrayOfChar[7] = '#';
arrayOfChar[8] = '?';
arrayOfChar[9] = 'T';
arrayOfChar[0] = (char)(char)(arrayOfChar[0] ^ this.n.charAt(1));
arrayOfChar[1] = (char)(char)(arrayOfChar[1] ^ this.o.charAt(0));
arrayOfChar[2] = (char)(char)(arrayOfChar[2] ^ this.o.charAt(4));
arrayOfChar[3] = (char)(char)(arrayOfChar[3] ^ this.n.charAt(4));
arrayOfChar[4] = (char)(char)(arrayOfChar[4] ^ this.n.charAt(7));
arrayOfChar[5] = (char)(char)(arrayOfChar[5] ^ this.n.charAt(0));
arrayOfChar[6] = (char)(char)(arrayOfChar[6] ^ this.o.charAt(2));
arrayOfChar[7] = (char)(char)(arrayOfChar[7] ^ this.o.charAt(3));
arrayOfChar[8] = (char)(char)(arrayOfChar[8] ^ this.n.charAt(6));
arrayOfChar[9] = (char)(char)(arrayOfChar[9] ^ this.n.charAt(8));
Toast.makeText((Context)this, "FLAG{" + new String(arrayOfChar) + "}", 1).show();
}
private boolean j() {
byte[] arrayOfByte = this.m.digest(this.o.getBytes());
int i = arrayOfByte.length;
String str = "";
for (byte b = 0; b < i; b++) {
byte b1 = arrayOfByte[b];
str = str + String.format("%x", new Object[] { Byte.valueOf(b1) });
}
return str.equals(getResources().getString(2131558446));
}
public void onBackPressed() {
Intent intent = new Intent("android.intent.action.MAIN");
intent.addCategory("android.intent.category.HOME");
intent.setFlags(268435456);

- 在loginactivaty中找到了验证用户名和密码的逻辑

```
public void onLogin(View paramView) {
  EditText editText2 = (EditText)findViewById(2131165247);
  EditText editText1 = (EditText)findViewById(2131165246);
  this.n = editText2.getText().toString();
  this.o = editText1.getText().toString();
  if (this.n != null && this.o != null && !this.n.isEmpty() && !this.o.isEmpty()) {
    if (!this.n.equals(getResources().getString(2131558450))) {
      Toast.makeText((Context)this, "User not recognized.", 0).show();
      editText2.setText("");
      editText1.setText("");
      return;
    }
    if (!i()) {
      Toast.makeText((Context)this, "Incorrect password.", 0).show();
      editText2.setText("");
      editText1.setText("");
      return;
    }
    i();
    startActivity(new Intent((Context)this, MessengerActivity.class));
  }
 }
}
```

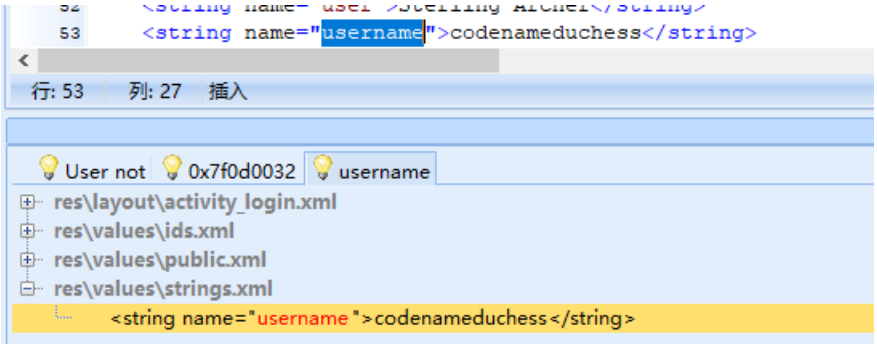

- 我们发现有两个判断，一个是对用户名的判断，如果判断错误会出现“User not recognized”，而下面的“Incorrect password”还没有出现，猜测应该是用户名正确，密码错误时的弹窗信息。所以寻找2131558450的字符串到底是啥。在androidkiller中已经将该字符串转换为了16进制，于是进行搜索找到其对应位置发现正是public.xml中name为“username”的一个字符串：

```
iget-object v2, p0, Lcom/tlamb96/kgbmessenger/LoginActivity;->n:Ljava/lang/String;

invoke-virtual {p0}, Lcom/tlamb96/kgbmessenger/LoginActivity;->getResources()Landroid/content/res/Resources;

move-result-object v3

const v4, 0x7f0d0032

invoke-virtual {v3, v4}, Landroid/content/res/Resources;->getString(I)Ljava/lang/String;

move-result-object v3

invoke-virtual {v2, v3}, Ljava/lang/String;->equals(Ljava/lang/Object;)Z

move-result v2

if-nez v2, :cond_2

const-string v2, "User not recognized."
```



```
<public type="string" name="username" id="0x7f0d0032" />
```

- 在string.xml中找到该字符串得到用户名为 “codenameduchess”：

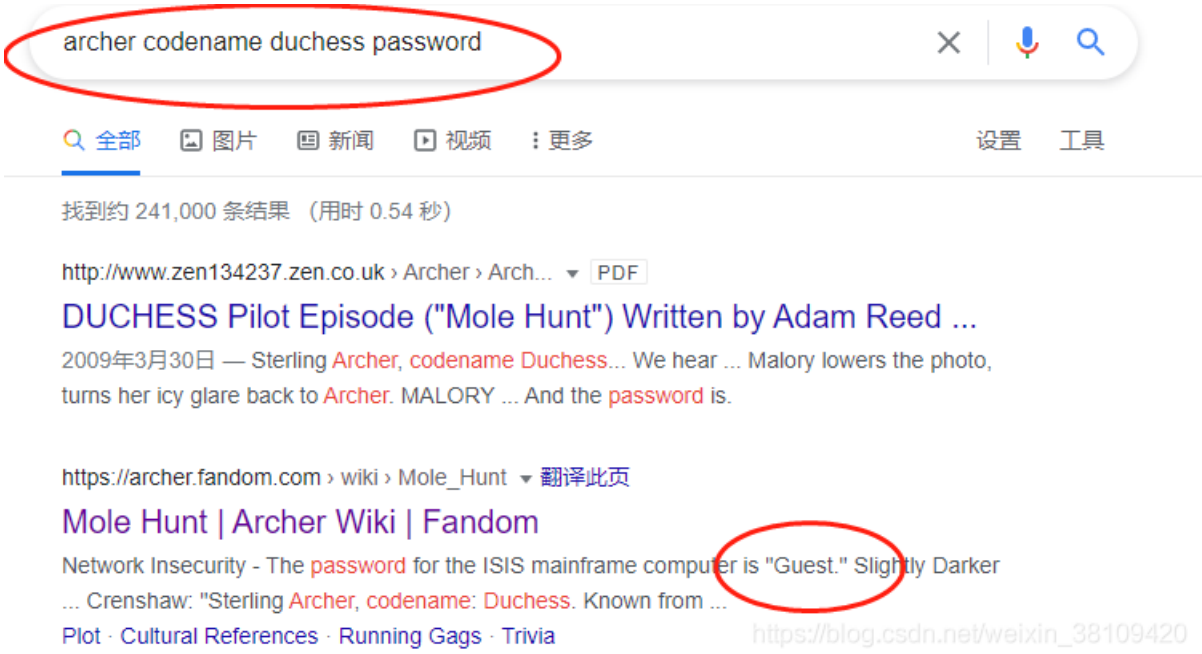


- 输入用户名"codenameduchess"后随便输入一个密码发现正是出现了"Incorret password",所以j()函数就是对密码的判断，查看j()函数发现是对输入的密码进行了md5哈希然后与一个固定值作比较：

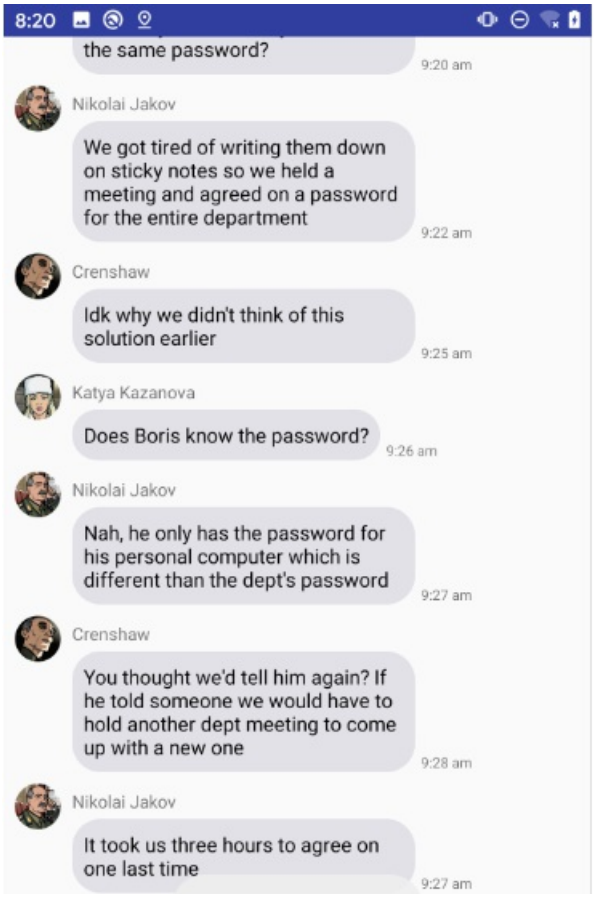
```
private boolean j() {
  byte[] arrayOfByte = this.m.digest(this.o.getBytes());
  int i = arrayOfByte.length;
  String str = "";
  for (byte b = 0; b < i; b++) {
```

```
            byte b1 = arrayOfByte[b];
            str = str + String.format("%x", new Object[] { Byte.valueOf(b1) });
          }
          return str.equals(getResources().getString(2131558446));
        }
```

- 由于哈希函数的不可逆性，这里好像没法获得原值（？或许可以用对照表什么的）
- workthrough里的解决方法是：
  - 用户名codenameduchess其实是一个动漫里Archer中特工的代号，代号为duchess (XD
  - 然后google一下就可以发现密码了



- 又因为题目说密码全是小写，就输入guest，然后就成功登录获得flag啦,flag也是googlepro,说明并不是想让我们通过hash值反推密码：




FLAG{G00G13_PR0}
Enter message
SEND
https://blog.csdn.net/weixin_38109420