XCTF_easyapk的WriteUp

原创

windy_ll 于 2020-03-03 17:26:41 发布 33 收藏

文章标签： 安卓 安全

本文链接：https://blog.csdn.net/qq_41374107/article/details/104636938

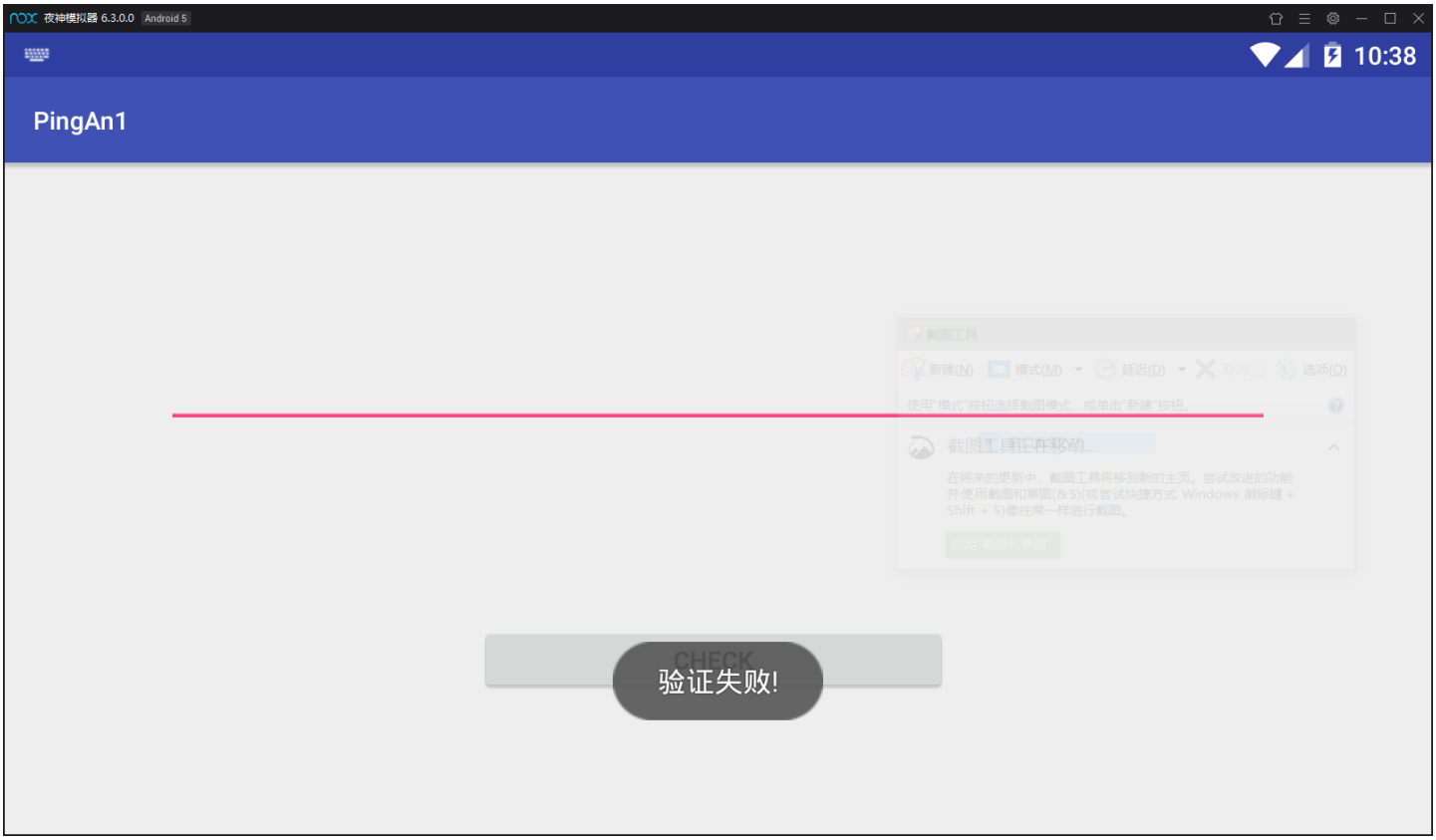
XCTF_easyapk的WriteUp

一、题目来源

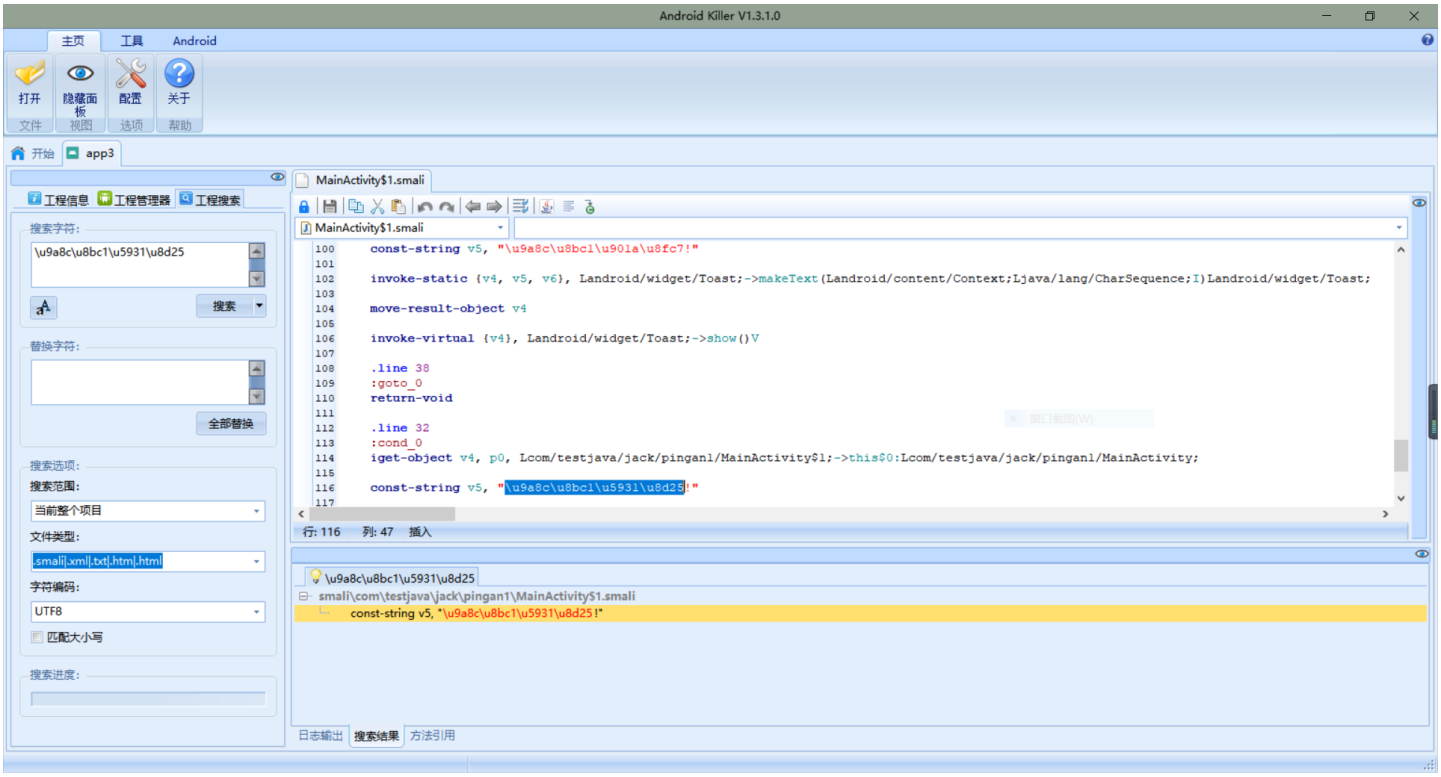
来源：XCTF社区安卓题目easy_apk

二、破解思路

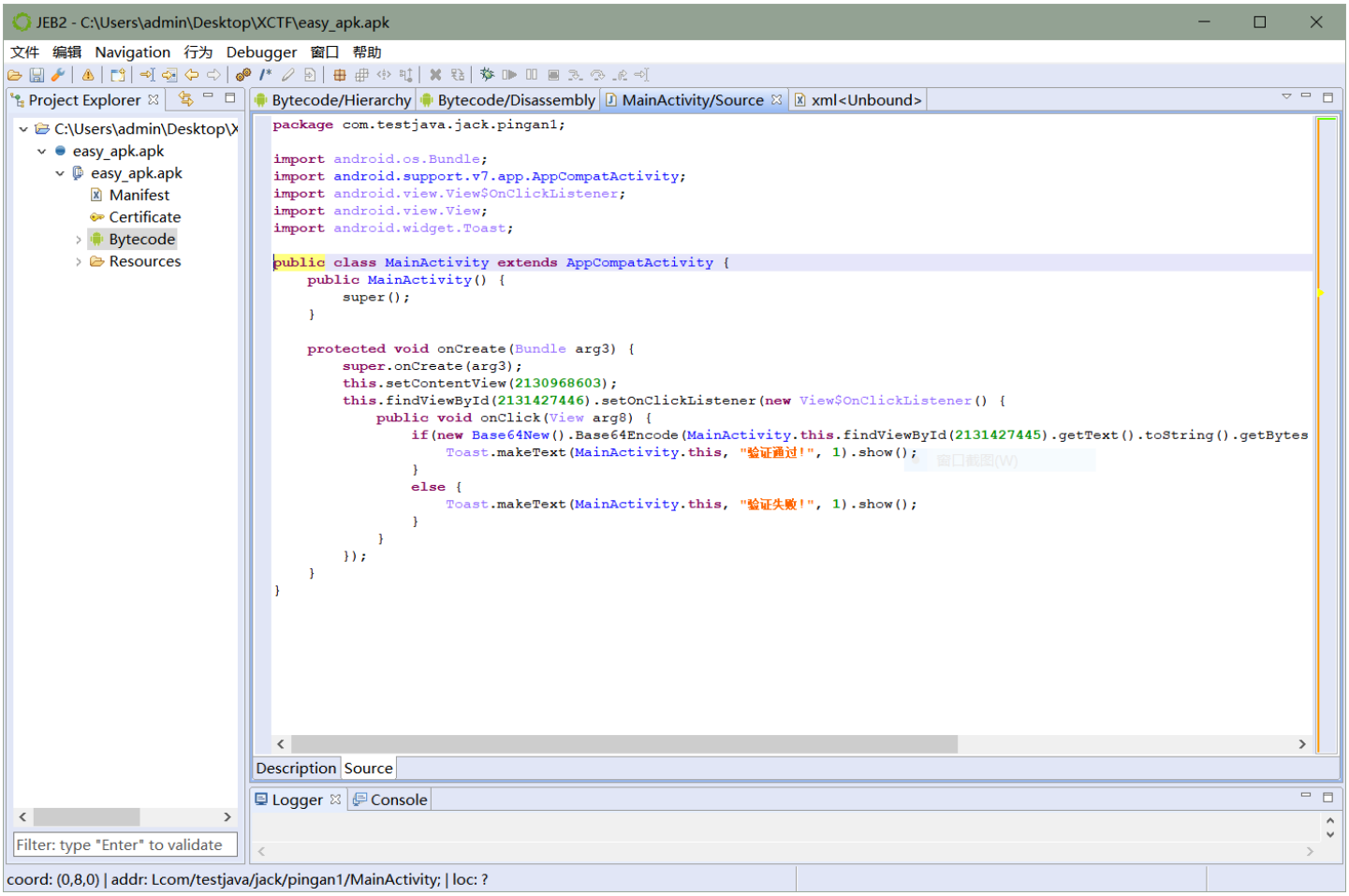
1、首先运行一下给的apk，发现就一个输入框和一个按钮，随便点击一下，发现弹出Toast`验证失败`。如下图所示：

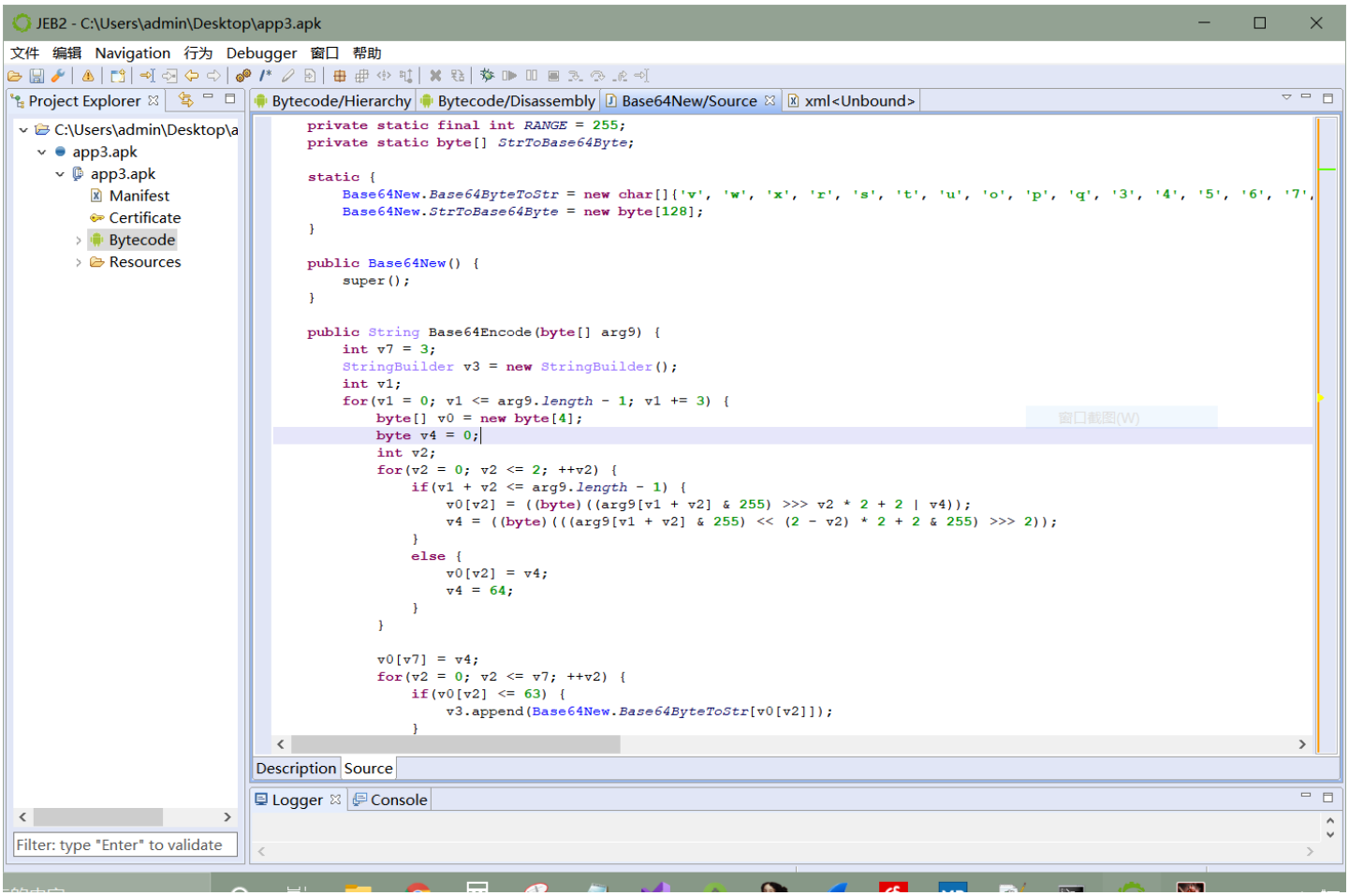


2、将该APK直接拖进AndroidKiller中反编译，搜索关键字验证失败，如下图所示：

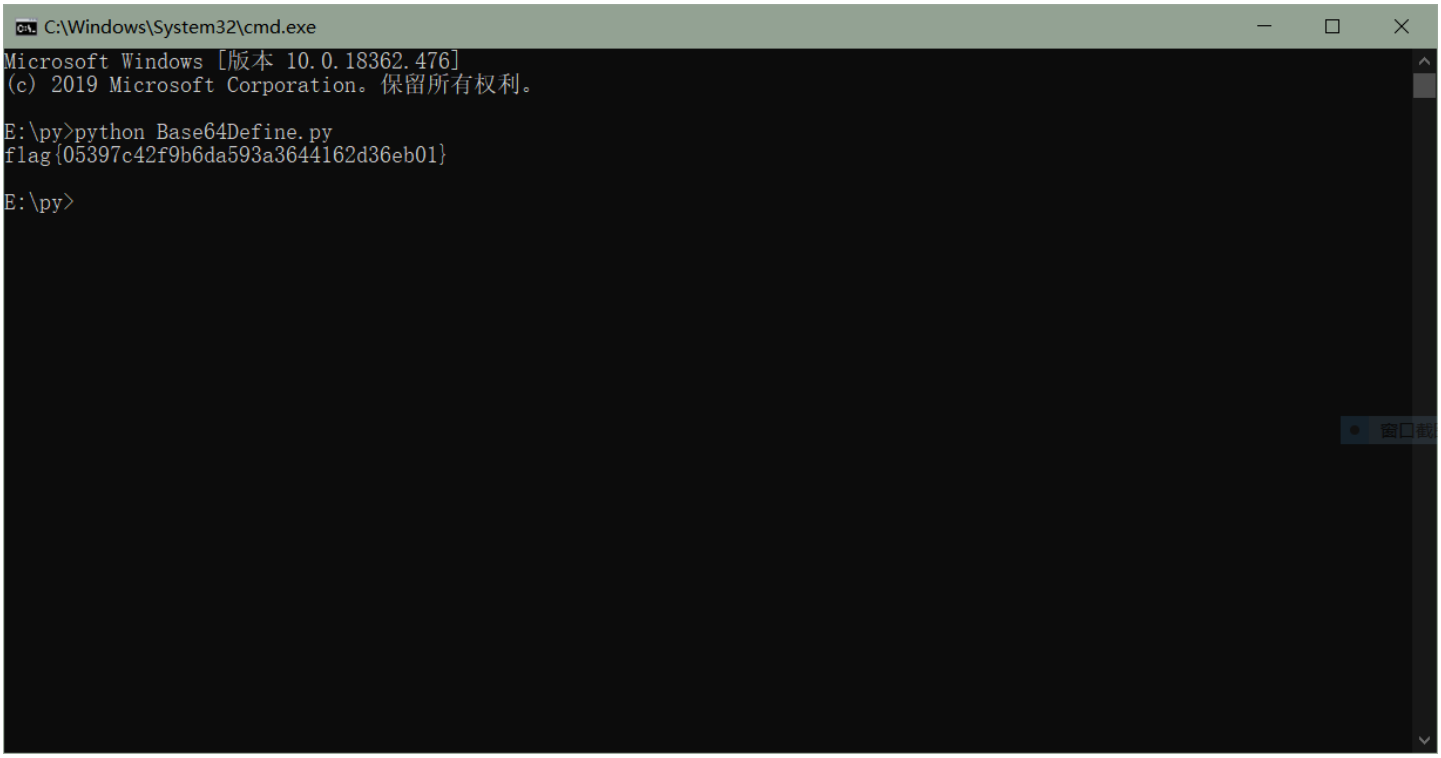


3、关键代码就在MainCtivity.java处，使用jeb直接查看Java代码（偷懒不看smali汇编了，看到脑壳痛），发现代码逻辑为将用户输入的字符串使用Base64New类中的Base64Encode()函数加密后和一个字符串相比较，若相等，则验证通过，否则验证失败，点击跳转到Base64New类中，发现Base64Encode()函数首先将用户输入得到的字符串转为字节数组作为参数，然后每3个字节为一组，扩展为4个字节，最后查表即可（感觉很眼熟的样子，emmmm就是base64编码，只是对照表跟base64对照表不同，可惜我在写完解密脚本之后才发现是这东西），那么解密过程也很简单了，直接对字符串4个字节一组分组，缩减为3个字节一组（语文水平感觉很差的样子写道这里。。。。），然后转为字符拼接成字符串即可。


JEB2 - C:\Users\admin\Desktop\XCTF\easy_apk.apk
文件 编辑 Navigation 行为 Debugger 窗口 帮助
Project Explorer
C:\Users\admin\Desktop\X
easy_apk.apk
easy_apk.apk
Manifest
Certificate
Bytecode
Resources
Bytecode/Hierarchy
Bytecode/Disassembly
MainActivity/Source
xml<Unbound>
package com.testjava.jack.pingan1;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View$OnClickListener;
import android.view.View;
import android.widget.Toast;
public class MainActivity extends AppCompatActivity {
public MainActivity() {
super();
}
protected void onCreate(Bundle arg3) {
super.onCreate(arg3);
this.setContentView(2130968603);
this.findViewById(2131427446).setOnClickListener(new View$OnClickListener() {
public void onClick(View arg8) {
if(new Base64New().Base64Encode(MainActivity.this.findViewById(2131427445).getText().toString().getBytes
Toast.makeText(MainActivity.this, "验证通过!", 1).show();
}
else {
Toast.makeText(MainActivity.this, "验证失败!", 1).show();
}
}
});
}
}
Description
Source
Logger
Console
Filter: type "Enter" to validate
coord: (0,8,0) | addr: Lcom/testjava/jack/pingan1/MainActivity; | loc: ?


JEB2 - C:\Users\admin\Desktop\app3.apk
文件 编辑 Navigation 行为 Debugger 窗口 帮助
Project Explorer
C:\Users\admin\Desktop\a
app3.apk
app3.apk
Manifest
Certificate
Bytecode
Resources
Bytecode/Hierarchy
Bytecode/Disassembly
Base64New/Source
xml<Unbound>
private static final int RANGE = 255;
private static byte[] StrToBase64Byte;
static {
Base64New.Base64ByteToStr = new char[]{'v', 'w', 'x', 'r', 's', 't', 'u', 'o', 'p', 'q', '3', '4', '5', '6', '7',
Base64New.StrToBase64Byte = new byte[128];
}
public Base64New() {
super();
}
public String Base64Encode(byte[] arg9) {
int v7 = 3;
StringBuilder v3 = new StringBuilder();
int v1;
for(v1 = 0; v1 <= arg9.length - 1; v1 += 3) {
byte[] v0 = new byte[4];
byte v4 = 0;
int v2;
for(v2 = 0; v2 <= 2; ++v2) {
if(v1 + v2 <= arg9.length - 1) {
v0[v2] = ((byte)((arg9[v1 + v2] & 255) >>> v2 * 2 + 2 | v4));
v4 = ((byte)(((arg9[v1 + v2] & 255) << (2 - v2) * 2 + 2 & 255) >>> 2));
}
else {
v0[v2] = v4;
v4 = 64;
}
}
v0[v7] = v4;
for(v2 = 0; v2 <= v7; ++v2) {
if(v0[v2] <= 63) {
v3.append(Base64New.Base64ByteToStr[v0[v2]]);
}
Description
Source
Logger
Console
Filter: type "Enter" to validate

4、将解码过程写出python脚本跑出flag，填入验证通过！！！



三、总结

总结：很基础的一道题，唯一的难点就在必须写base64解码脚本了吧，emmmm要是像我一样木有认出来是base64，那一开始就有点难受了！！！下面附上Base64编码解码原理以及相关脚本！！！

Base64编码原理：

1、将字符串转为字节数组，然后每3个字节一组，一个24个比特，不足3个字节直接补0

2、在每一组3个字节24bit中，以6个bit构成一个字节（高两位补0），形成4个字节为一组

3、根据编码后的字节查找对照表，拼接成字符串，自此,Base64编码完成！！！

Base64解码原理：

1、将编码后的字符串查找对照表后的字节以4个字节为一组，出现`=`直接去掉即可

2、将这4个字节每个字节的高两位去掉，有32bit变为24bit，将这24bit以8个bit构成三个字节

3、将第二步得到的字节数组转为字符串即可！！！

附上python脚本(ps:由于markdown问题，所以有需要的同学更改一下某些地方的缩进)：

```
def Base64Decode(str_list):
list_base = []
a = str_list[0] << 2
c = str_list[1] & 15
b = str_list[1] >> 4
a = a | b
list_base.append(a)
c = c << 4
a = str_list[2] & 3
b = str_list[2] >> 2
c = c | b
list_base.append(c)
a = a << 6
a = a | str_list[3]
list_base.append(a)
return list_base

CodingTable = 'vwxrstuopq34567ABCDEFGHIJyz012PQRSTKLMNOZabcdUVWXYefghijklmn89+/'
Ciphertext = '5rFf7E2K6rqN7Hpiyush7E6S5fJg6rsi5NBf6NGT5rs='

i = 0
flag = 'flag{'

while i <= (len(Ciphertext) - 1):
list1 = []
n = 0
for k in range(4):
 if Ciphertext[i + k] == '=':
  list1.append(0)
  n = n + 1
 else:
  list1.append(CodingTable.index(Ciphertext[i + k]))
ba = Base64Decode(list1)
for j in range(3 - n):
 ch = chr(ba[j])
 flag = flag + str(ch)
i = i + 4

flag = flag + '}'
print(flag)
```