# ISCC2024 WriteUp

AkyOI 李承达 [3433778745@qq.com](mailto:3433778745@qq.com)

## mobile+ChallengeMobile

## 解题思路

Jeb打开分析，动态加载了Checker类并且调用了isflag的方法。

同时仔细观察解密ming文件的方法是调用了a函数，因此我们只要拿到a函数的返回值就能拿到Checker这个类的dex文件（其实是这个文件的每一个字节），之后转为dex文件在用jeb打开就能看到isflag的实现了。

其中拿到dex文件的frida的hook脚本如下：

// frida script to call mainactivity a to get dexfile

Java.perform(function() {

  // load main

    let MainActivity = Java.use("com.example.challengemobile.MainActivity");

    MainActivity["a"].implementation = function (bArr) {//call a

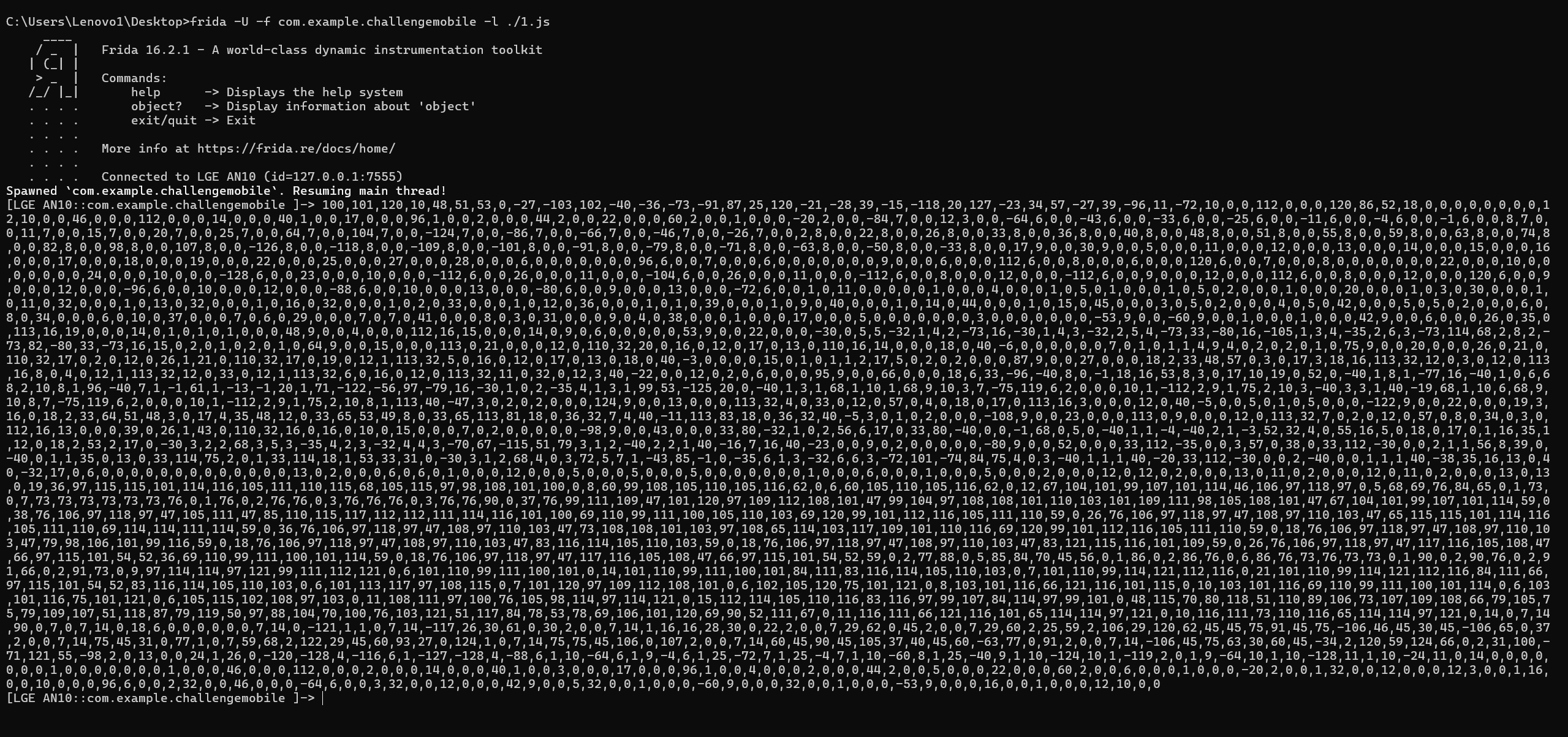
      let dexdexdexfffiiillleee = this['a'](bArr);

      console.log(dexdexdexfffiiillleee);

      return dexdexdexfffiiillleee;

    };

});



之后将这些字节写入文件：

arr = [100,101,120,10,48,51,53,0,-27,-103,102,-40,-36,-73,-91,87,25,120,-21,-28,39,-15,-118,20,127,-23,34,57,-27,39,-96,11,-72,10,0,0,112,0,0,0,120,86,52,18,0,0,0,0,0,0,0,0,12,10,0,0,46,0,0,0,112,0,0,0,14,0,0,0,40,1,0,0,17,0,0,0,96,1,0,0,2,0,0,0,44,2,0,0,22,0,0,0,60,2,0,0,1,0,0,0,-20,2,0,0,-84,7,0,0,12,3,0,0,-64,6,0,0,-43,6,0,0,-33,6,0,0,-25,6,0,0,-11,6,0,0,-4,6,0,0,-1,6,0,0,8,7,0,0,11,7,0,0,15,7,0,0,20,7,0,0,25,7,0,0,64,7,0,0,104,7,0,0,-124,7,0,0,-86,7,0,0,-66,7,0,0,-46,7,0,0,-26,7,0,0,2,8,0,0,22,8,0,0,26,8,0,0,33,8,0,0,36,8,0,0,40,8,0,0,48,8,0,0,51,8,0,0,55,8,0,0,59,8,0,0,63,8,0,0,74,8,0,0,82,8,0,0,98,8,0,0,107,8,0,0,-126,8,0,0,-118,8,0,0,-109,8,0,0,-101,8,0,0,-91,8,0,0,-79,8,0,0,-71,8,0,0,-63,8,0,0,-50,8,0,0,-33,8,0,0,17,9,0,0,30,9,0,0,5,0,0,0,11,0,0,0,12,0,0,0,13,0,0,0,14,0,0,0,15,0,0,0,16,0,0,0,17,0,0,0,18,0,0,0,19,0,0,0,22,0,0,0,25,0,0,0,27,0,0,0,28,0,0,0,6,0,0,0,0,0,0,0,96,6,0,0,7,0,0,0,6,0,0,0,0,0,0,0,9,0,0,0,6,0,0,0,112,6,0,0,8,0,0,0,6,0,0,0,120,6,0,0,7,0,0,0,8,0,0,0,0,0,0,0,22,0,0,0,10,0,0,0,0,0,0,0,24,0,0,0,10,0,0,0,-128,6,0,0,23,0,0,0,10,0,0,0,-112,6,0,0,26,0,0,0,11,0,0,0,-104,6,0,0,26,0,0,0,11,0,0,0,-112,6,0,0,8,0,0,0,12,0,0,0,-112,6,0,0,9,0,0,0,12,0,0,0,112,6,0,0,8,0,0,0,12,0,0,0,120,6,0,0,9,0,0,0,12,0,0,0,-96,6,0,0,10,0,0,0,12,0,0,0,-88,6,0,0,10,0,0,0,13,0,0,0,-80,6,0,0,9,0,0,0,13,0,0,0,-72,6,0,0,1,0,11,0,0,0,0,0,1,0,0,0,4,0,0,0,1,0,5,0,1,0,0,0,1,0,5,0,2,0,0,0,1,0,0,0,20,0,0,0,1,0,3,0,30,0,0,0,1,0,11,0,32,0,0,0,1,0,13,0,32,0,0,0,1,0,16,0,32,0,0,0,1,0,2,0,33,0,0,0,1,0,12,0,36,0,0,0,1,0,1,0,39,0,0,0,1,0,9,0,40,0,0,0,1,0,14,0,44,0,0,0,1,0,15,0,45,0,0,0,3,0,5,0,2,0,0,0,4,0,5,0,42,0,0,0,5,0,5,0,2,0,0,0,6,0,8,0,34,0,0,0,6,0,10,0,37,0,0,0,7,0,6,0,29,0,0,0,7,0,7,0,41,0,0,0,8,0,3,0,31,0,0,0,9,0,4,0,38,0,0,0,1,0,0,0,17,0,0,0,5,0,0,0,0,0,0,0,3,0,0,0,0,0,0,0,-53,9,0,0,-60,9,0,0,1,0,0,0,1,0,0,0,42,9,0,0,6,0,0,0,26,0,35,0,113,16,19,0,0,0,14,0,1,0,1,0,1,0,0,0,48,9,0,0,4,0,0,0,112,16,15,0,0,0,14,0,9,0,6,0,0,0,0,0,53,9,0,0,22,0,0,0,-30,0,5,5,-32,1,4,2,-73,16,-30,1,4,3,-32,2,5,4,-73,33,-80,16,-105,1,3,4,-35,2,6,3,-73,114,68,2,8,2,-73,82,-80,33,-73,16,15,0,2,0,1,0,2,0,1,0,64,9,0,0,15,0,0,0,113,0,21,0,0,0,12,0,110,32,20,0,16,0,12,0,17,0,13,0,110,16,14,0,0,0,18,0,40,-6,0,0,0,0,0,0,7,0,1,0,1,1,4,9,4,0,2,0,2,0,1,0,75,9,0,0,20,0,0,0,26,0,21,0,110,32,17,0,2,0,12,0,26,1,21,0,110,32,17,0,19,0,12,1,113,32,5,0,16,0,12,0,17,0,13,0,18,0,40,-3,0,0,0,0,15,0,1,0,1,1,2,17,5,0,2,0,2,0,0,0,87,9,0,0,27,0,0,0,18,2,33,48,57,0,3,0,17,3,18,16,113,32,12,0,3,0,12,0,113,16,8,0,4,0,12,1,113,32,12,0,33,0,12,1,113,32,6,0,16,0,12,0,113,32,11,0,32,0,12,3,40,-22,0,0,12,0,2,0,6,0,0,0,95,9,0,0,66,0,0,0,18,6,33,-96,-40,8,0,-1,18,16,53,8,3,0,17,10,19,0,52,0,-40,1,8,1,-77,16,-40,1,0,6,68,2,10,8,1,96,-40,7,1,-1,61,1,-13,-1,20,1,71,-122,-56,97,-79,16,-30,1,0,2,-35,4,1,3,1,99,53,-125,20,0,-40,1,3,1,68,1,10,1,68,9,10,3,7,-75,119,6,2,0,0,0,10,1,-112,2,9,1,75,2,10,3,-40,3,3,1,40,-19,68,1,10,6,68,9,10,8,7,-75,119,6,2,0,0,0,10,1,-112,2,9,1,75,2,10,8,1,113,40,-47,3,0,2,0,2,0,0,0,124,9,0,0,13,0,0,0,113,32,4,0,33,0,12,0,57,0,4,0,18,0,17,0,113,16,3,0,0,0,12,0,40,-5,0,0,5,0,1,0,5,0,0,0,-122,9,0,0,22,0,0,0,19,3,16,0,18,2,33,64,51,48,3,0,17,4,35,48,12,0,33,65,53,49,8,0,33,65,113,81,18,0,36,32,7,4,40,-11,113,83,18,0,36,32,40,-5,3,0,1,0,2,0,0,0,-108,9,0,0,23,0,0,0,113,0,9,0,0,0,12,0,113,32,7,0,2,0,12,0,57,0,8,0,34,0,3,0,112,16,13,0,0,0,39,0,26,1,43,0,110,32,16,0,16,0,10,0,15,0,0,0,7,0,2,0,0,0,0,0,-98,9,0,0,43,0,0,0,33,80,-32,1,0,2,56,6,17,0,33,80,-40,0,0,-1,68,0,5,0,-40,1,1,-4,-40,2,1,-3,52,32,4,0,55,16,5,0,18,0,17,0,1,16,35,1,12,0,18,2,53,2,17,0,-30,3,2,2,68,3,5,3,-35,4,2,3,-32,4,4,3,-70,67,-115,51,79,3,1,2,-40,2,2,1,40,-16,7,16,40,-23,0,0,9,0,2,0,0,0,0,0,-80,9,0,0,52,0,0,0,33,112,-35,0,0,3,57,0,38,0,33,112,-30,0,0,2,1,1,56,8,39,0,-40,0,1,1,35,0,13,0,33,114,75,2,0,1,33,114,18,1,53,33,31,0,-30,3,1,2,68,4,0,3,72,5,7,1,-43,85,-1,0,-35,6,1,3,-32,6,6,3,-72,101,-74,84,75,4,0,3,-40,1,1,1,40,-20,33,112,-30,0,0,2,-40,0,0,1,1,1,40,-38,35,16,13,0,40,-32,17,0,6,0,0,0,0,0,0,0,0,0,0,0,0,0,13,0,2,0,0,0,6,0,6,0,1,0,0,0,12,0,0,0,5,0,0,0,5,0,0,0,5,0,0,0,0,0,0,0,1,0,0,0,6,0,0,0,1,0,0,0,5,0,0,0,2,0,0,0,12,0,12,0,2,0,0,0,13,0,11,0,2,0,0,0,12,0,11,0,2,0,0,0,13,0,13,0,19,36,97,115,115,101,114,116,105,111,110,115,68,105,115,97,98,108,101,100,0,8,60,99,108,105,110,105,116,62,0,6,60,105,110,105,116,62,0,12,67,104,101,99,107,101,114,46,106,97,118,97,0,5,68,69,76,84,65,0,1,73,0,7,73,73,73,73,73,73,76,0,1,76,0,2,76,76,0,3,76,76,76,0,3,76,76,90,0,37,76,99,111,109,47,101,120,97,109,112,108,101,47,99,104,97,108,108,101,110,103,101,109,111,98,105,108,101,47,67,104,101,99,107,101,114,59,0,38,76,106,97,118,97,47,105,111,47,85,110,115,117,112,112,111,114,116,101,100,69,110,99,111,100,105,110,103,69,120,99,101,112,116,105,111,110,59,0,26,76,106,97,118,97,47,108,97,110,103,47,65,115,115,101,114,116,105,111,110,69,114,114,111,114,59,0,36,76,106,97,118,97,47,108,97,110,103,47,73,108,108,101,103,97,108,65,114,103,117,109,101,110,116,69,120,99,101,112,116,105,111,110,59,0,18,76,106,97,118,97,47,108,97,110,103,47,79,98,106,101,99,116,59,0,18,76,106,97,118,97,47,108,97,110,103,47,83,116,114,105,110,103,59,0,18,76,106,97,118,97,47,108,97,110,103,47,83,121,115,116,101,109,59,0,26,76,106,97,118,97,47,117,116,105,108,47,66,97,115,101,54,52,36,69,110,99,111,100,101,114,59,0,18,76,106,97,118,97,47,117,116,105,108,47,66,97,115,101,54,52,59,0,2,77,88,0,5,85,84,70,45,56,0,1,86,0,2,86,76,0,6,86,76,73,76,73,73,0,1,90,0,2,90,76,0,2,91,66,0,2,91,73,0,9,97,114,114,97,121,99,111,112,121,0,6,101,110,99,111,100,101,0,14,101,110,99,111,100,101,84,111,83,116,114,105,110,103,0,7,101,110,99,114,121,112,116,0,21,101,110,99,114,121,112,116,84,111,66,97,115,101,54,52,83,116,114,105,110,103,0,6,101,113,117,97,108,115,0,7,101,120,97,109,112,108,101,0,6,102,105,120,75,101,121,0,8,103,101,116,66,121,116,101,115,0,10,103,101,116,69,110,99,111,100,101,114,0,6,103,101,116,75,101,121,0,6,105,115,102,108,97,103,0,11,108,111,97,100,76,105,98,114,97,114,121,0,15,112,114,105,110,116,83,116,97,99,107,84,114,97,99,101,0,48,115,70,80,118,51,110,89,106,73,107,109,108,66,79,105,75,79,109,107,51,118,87,79,119,50,97,88,104,70,100,76,103,121,51,117,84,78,53,78,69,106,101,120,69,90,52,111,67,0,11,116,111,66,121,116,101,65,114,114,97,121,0,10,116,111,73,110,116,65,114,114,97,121,0,14,0,7,14,90,0,7,0,7,14,0,18,6,0,0,0,0,0,0,7,14,0,-121,1,1,0,7,14,-117,26,30,61,0,30,2,0,0,7,14,1,16,16,28,30,0,22,2,0,0,7,29,62,0,45,2,0,0,7,29,60,2,25,59,2,106,29,120,62,45,45,75,91,45,75,-106,46,45,30,45,-106,65,0,37,2,0,0,7,14,75,45,31,0,77,1,0,7,59,68,2,122,29,45,60,93,27,0,124,1,0,7,14,75,75,45,106,0,107,2,0,0,7,14,60,45,90,45,105,37,40,45,60,-63,77,0,91,2,0,0,7,14,-106,45,75,63,30,60,45,-34,2,120,59,124,66,0,2,31,100,-71,121,55,-98,2,0,13,0,0,24,1,26,0,-120,-128,4,-116,6,1,-127,-128,4,-88,6,1,10,-64,6,1,9,-4,6,1,25,-72,7,1,25,-4,7,1,10,-60,8,1,25,-40,9,1,10,-124,10,1,-119,2,0,1,9,-64,10,1,10,-128,11,1,10,-24,11,0,14,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,46,0,0,0,112,0,0,0,2,0,0,0,14,0,0,0,40,1,0,0,3,0,0,0,17,0,0,0,96,1,0,0,4,0,0,0,2,0,0,0,44,2,0,0,5,0,0,0,22,0,0,0,60,2,0,0,6,0,0,0,1,0,0,0,-20,2,0,0,1,32,0,0,12,0,0,0,12,3,0,0,1,16,0,0,10,0,0,0,96,6,0,0,2,32,0,0,46,0,0,0,-64,6,0,0,3,32,0,0,12,0,0,0,42,9,0,0,5,32,0,0,1,0,0,0,-60,9,0,0,0,32,0,0,1,0,0,0,-53,9,0,0,0,16,0,0,1,0,0,0,12,10,0,0]

import struct

def write\_bytes\_to\_file(arr, file\_name):

    with open(file\_name, 'wb') as f:

        for num in arr:

            if -127 <= num <= 127:

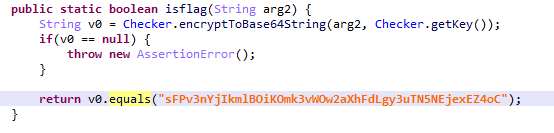
                f.write(struct.pack('b', num))

# arr = [1, -2, 3, -4, 5, -6, 7, -8, 9, -10, 11, -12, 13, -14, 15, -16, 17, -18, 19, -20]

file\_name = 'output.bin'

write\_bytes\_to\_file(arr, file\_name)

用jeb打开就是Checker类了，里面有密文。



但是getKey函数实现在了native层，，但是可以通过frida hook脚本来加载Checker这个类之后调用getKey函数直接得到返回值，也就是key，调用getKey函数并且输出返回值的脚本：

Java.perform(function () {

    var class0001 = Java.use("java.lang.Class");

    var classloader000 = Java.use("dalvik.system.DexClassLoader");

    classloader000.loadClass.overload("java.lang.String").implementation =

    function (name) {

        var retValue = this.loadClass(name, false);

        var hookname = "com.example.challengemobile.Checker";

        if (name == hookname) {

            var hook0001castsss=Java.cast(hookClass, class0001);

            var hookClass=retValue;

            var classmethod000=hook0001castsss.getMethod("getKey", []);

            var retValue=classmethod000.invoke(null, []);

            console.log("retValue:", retValue);

            return retValue;

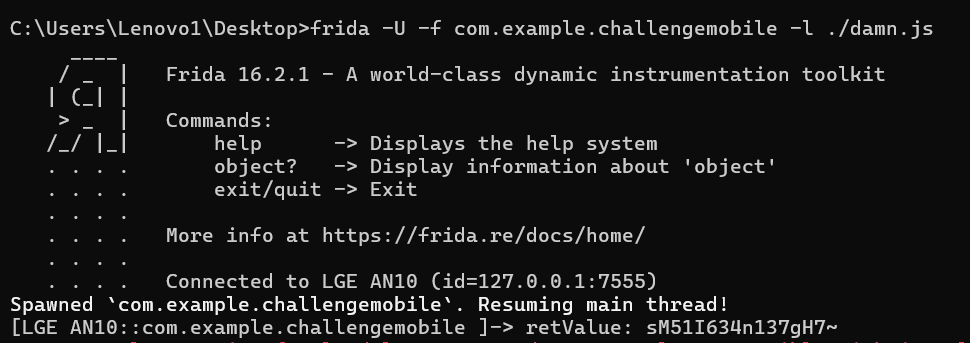
        }

        return retValue;

    };

    }

);



可见拿到了key，于是直接对密文进行base64解密之后再用key进行xxtea解密就行了

解密脚本：

import base64

import xxtea

str=b"sFPv3nYjIkmlBOiKOmk3vWOw2aXhFdLgy3uTN5NEjexEZ4oC"

a=base64.b64decode(str)

print(a)

print(len(a))

key=b"sM51I634n137gH7~"

print(xxtea.decrypt(a,key))