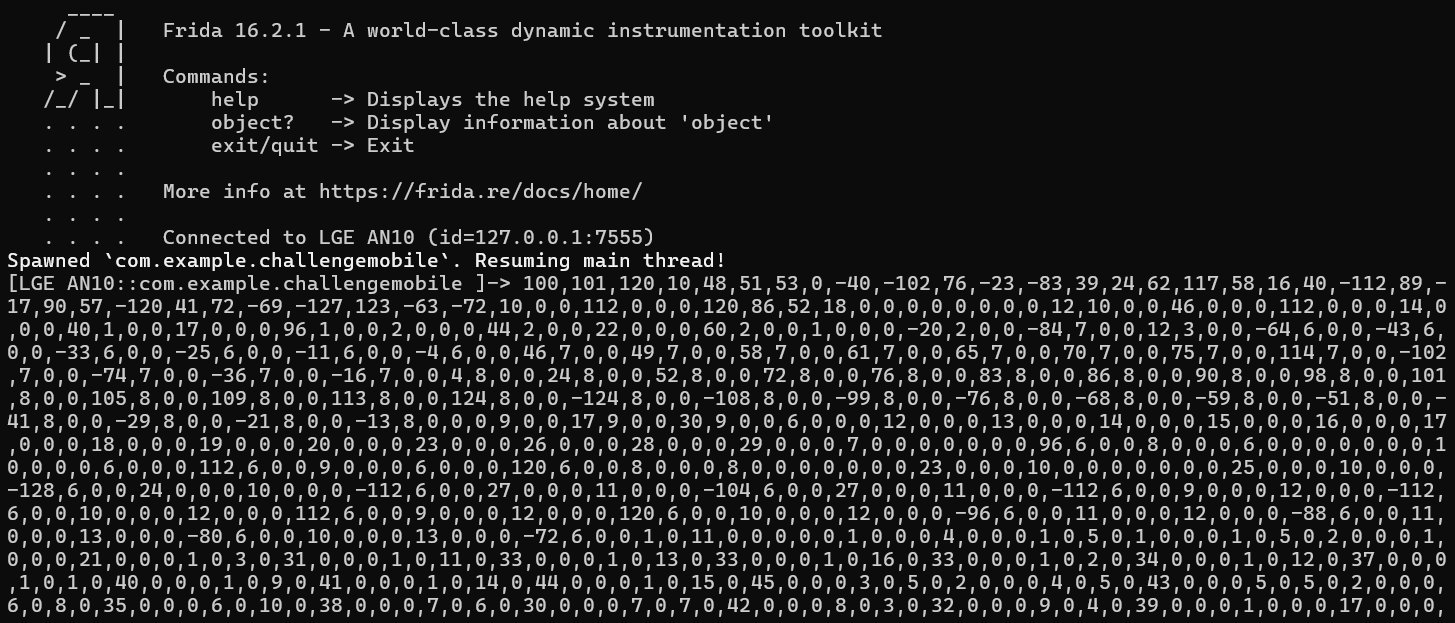
Nebu1ea 韩磊 [1504895347@qq.com](mailto:1504895347@qq.com)

Mobile: ChallengeMobile

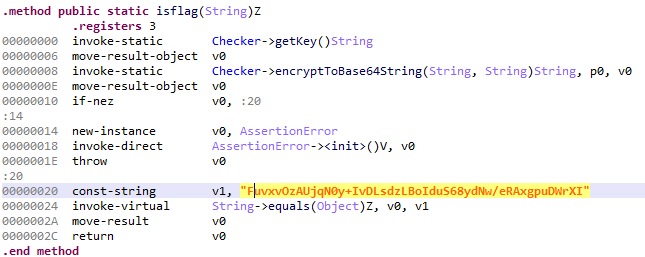
解题思路：

用Jeb打开分析，动态加载了Checker类，并调用了其中的isflag方法。在解密ming文件的过程中，注意到调用了a函数。因此，我们的目标是获取a函数的返回值，这样就能获取到Checker类的DEX文件，即该文件的每个字节。之后，将这些字节重新组合成DEX文件，并使用Jeb打开，即可查看isflag方法的实现。拿到frida的脚本为脚本1，跑完的效果如下：

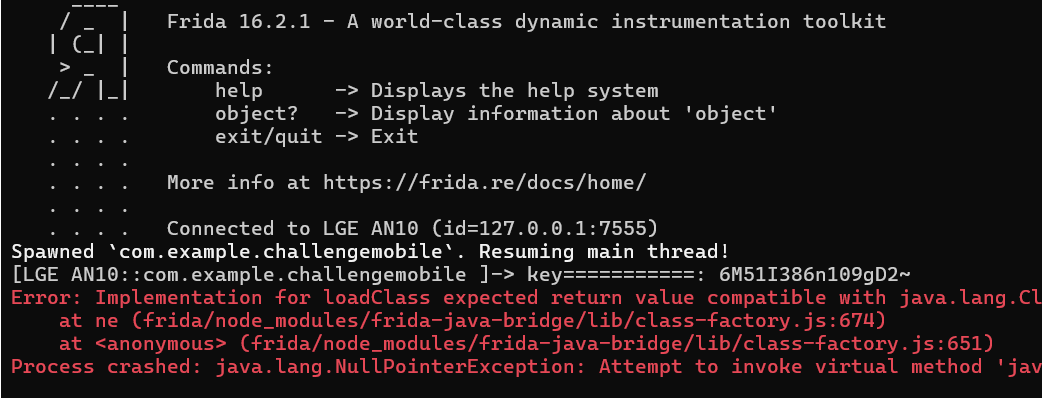


接着把这些字节写到文件里面去，用脚本2

再看checker的密文：



此时getKey函数虽然实现在了native层，但可以通过frida hook脚本加载后得到返回值key。调用的为脚本三，跑完效果：



Key拿到了，直接解密就行了，用脚本四。

Exp：

脚本一：

Java.perform(function() {

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

    TargetActivity["customMethodName"].implementation = function (byteArray) {

        let dexFile = this['originalMethodName'](byteArray);

        console.log(dexFile);

        return dexFile;

    };

});

脚本二：

import struct

input\_file\_path = "111.txt"

with open(input\_file\_path, "r") as input\_file:

    content = input\_file.read()

numbers = [int(num) for num in content.split(",")]

output\_file\_path = "1.nebu1ea"

with open(output\_file\_path, "wb") as output\_file:

    for num in numbers:

        output\_file.write(struct.pack('b',num))

脚本三;

Java.perform(function () {

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

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

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

    function (className) {

        var originalClass = this.loadClass(className, false);

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

        if (className == targetClassName) {

            var targetClass = Java.cast(originalClass, JavaClass);

            var getKeyMethod = targetClass.getMethod("getKey", []);

            var keyValue = getKeyMethod.invoke(null, []);

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

            return keyValue;

        }

        return originalClass;

    };

});

脚本四;

import base64

import xxtea

str=input("密文")

abab=base64.b64decode(str)

key=input("密钥")

print(xxtea.decrypt(abab,key))