mobile2

libmobile04.so

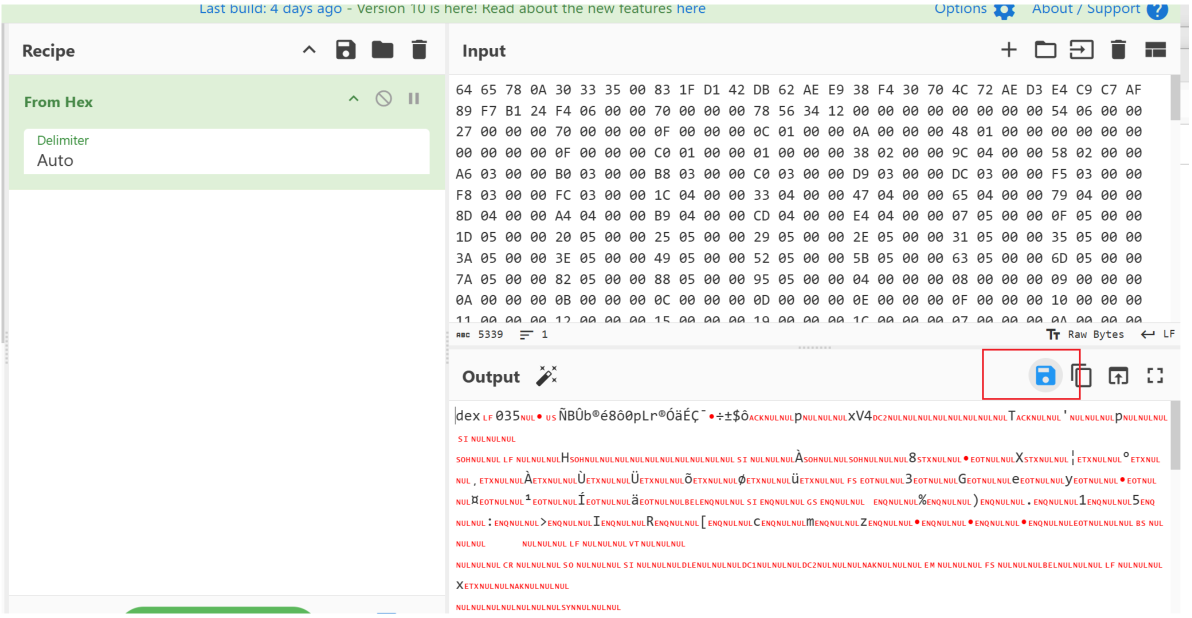
Java_com_example_mobile04_MainActivity_getEncryptedSegment里面最底下

```
__int64 __fastcall Java_com_example_mobile04_MainActivity_getEncryptedSegment(__inte
{
  int v3; // r15d
  int v4; // ebp
  unsigned int v5; // ebp
  __int64 v6; // r14

  v3 = 254 * a3;
  v4 = 1780;
  if ( a3 != 6 )
    v4 = v3 + 254;
  v5 = v4 - v3;
  v6 = (*(__int64 (__fastcall **)(__int64, _QWORD))(*(_QWORD *)a1 + 1408LL))(a1, v5)
  (*(void (__fastcall **)(__int64, __int64, _QWORD, _QWORD, char *))(*(_QWORD *)a1 +
    a1,
    v6,
    0LL,
    v5,
    &byte_1930[v3]);
  return v6;
}
```

```
000000000192E                   db    0
000000000192F                   db    0
0000000001930 byte_1930         db 64h                  ; DATA XREF: Java_
0000000001931                   db  65h ; e
0000000001932                   db  78h ; x
0000000001933                   db  0Ah
0000000001934                   db  30h ; 0
0000000001935                   db  33h ; 3
0000000001936                   db  35h ; 5
0000000001937                   db    0
0000000001938                   db  83h
0000000001939                   db  1Fh
000000000193A                   db 0D1h
000000000193B                   db  42h ; B
000000000193C                   db 0DBh
000000000193D                   db  62h ; b
```

提出来赛博厨子解出来dex



jadx打开dex

```java
package com.example.mobile04;

import java.util.Arrays;
import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;

/* loaded from: C:\Users\p3cd0wn\Downloads\download (1).dex */
public class Sunday5 {
    public static native byte[] getKey();

    static {
        System.loadLibrary("Sunday");
    }

    public static byte[] encrypt(byte[] bArr) {
        try {
            byte[] key = getKey();
            if (key == null || key.length != 24) {
                throw new RuntimeException("Invalid key from native");
            }
            SecretKeySpec secretKeySpec = new SecretKeySpec(key, "DESede");
            Cipher cipher = Cipher.getInstance("DESede/ECB/PKCS5Padding");
            cipher.init(1, secretKeySpec);
            return cipher.doFinal(bArr);
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }

    public static boolean checkFlag(String str) {
        return Arrays.equals(encrypt(str.getBytes()), new byte[]{69, 49, 52, 68, 53, 53, 53, 66, 50, 65, 57, 55, 56, 50, 52, 53});
    }
}
```

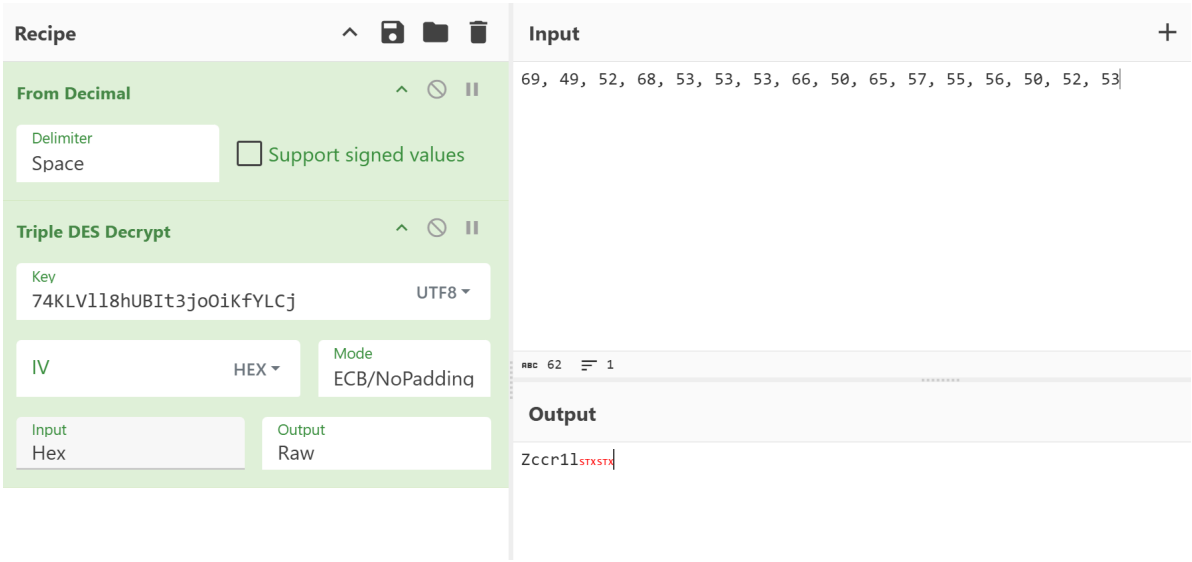
这里面是第一段的密文

libsunday.so找Java_com_example_mobile04_Sunday_getKey

```c
__int64 __fastcall Java_com_example_mobile04_Sunday_getKey(__int64 a1)
{
  __int64 v1; // rbx
  __int128 v3[2]; // [rsp+0h] [rbp-38h] BYREF
  unsigned __int64 v4; // [rsp+20h] [rbp-18h]

  v4 = __readfsqword(0x28u);
  qmemcpy(v3, "74KLVll8hUBIt3joOiKfYLCj", 24);
  v1 = (*(__int64 (__fastcall **)(__int64, __int64))(*(_QWORD *)a1 + 1408LL))(a1, 24LL);
  (*(void (__fastcall **)(__int64, __int64, _QWORD, __int64, __int128 *))(*(_QWORD *)a1 + 1664
    a1,
    v1,
    0LL,
    24LL,
    v3);
  return v1;
}
```

这是第一部分的key，然后厨子解



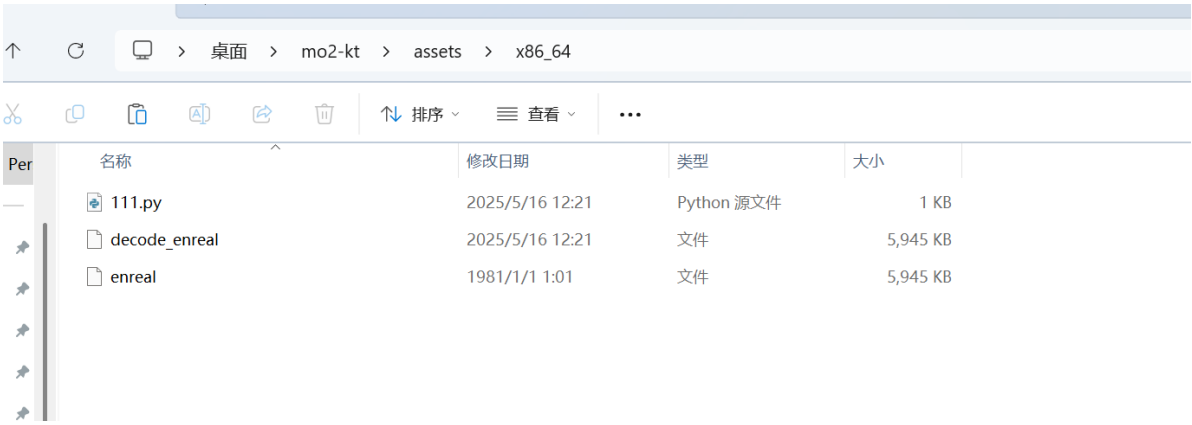
libMonday.so找到Java_com_example_mobile04_a_checkFlag2里面的位移异或这一段

```
        if ( v8 <= 0 )
        {
LABEL_19:
          v33 = (*(*a1 + 1408LL))(a1, v8);
          (*(*a1 + 1664LL))(a1, v33, 0LL, v8, v9);
          (*(*a1 + 1536LL))(a1, a4, v9, 0LL);
          LOBYTE(v5) = (*(*a1 + 1368LL))(a1, v33) > 0;
          return v5;
        }
        if ( v8 < 8 )
        {
          for ( i = 0LL; i != v8; ++i )
LABEL_18:
            *(v9 + i) = ((((*(v9 + i) >> 6) | (4 * *(v9 + i))) ^ 0x44) >> 3) | (32
                                                                              * ((((*(v9 + i) >> 6) | (4 * *(v9 + i))) ^ 0x44));
          goto LABEL_19;
        }
        if ( v8 >= 0x20 )
        {
          i = v8 & 0x7FFFFFE0;
          v11 = 0LL;
```

注意到xor了0x44

```python
with open("enreal", "rb") as f:
    data = list(f.read())
for i in range(len(data)):
    data[i] = (data[i] << 2) | (data[i] >> 6)
    data[i] &= 0xff
    data[i] ^= 0x44 #这里进行修改
    data[i] = (data[i] >> 3) | (data[i] << 5)
    data[i] &= 0xff
with open("decode_enreal", "wb") as f:
    f.write(bytes(data))
```

然后去/assets/x86_64解密里面的enreal



ida打开decode的enreal，real_check得到第二部分的key，密文是最底下的那串十六进制

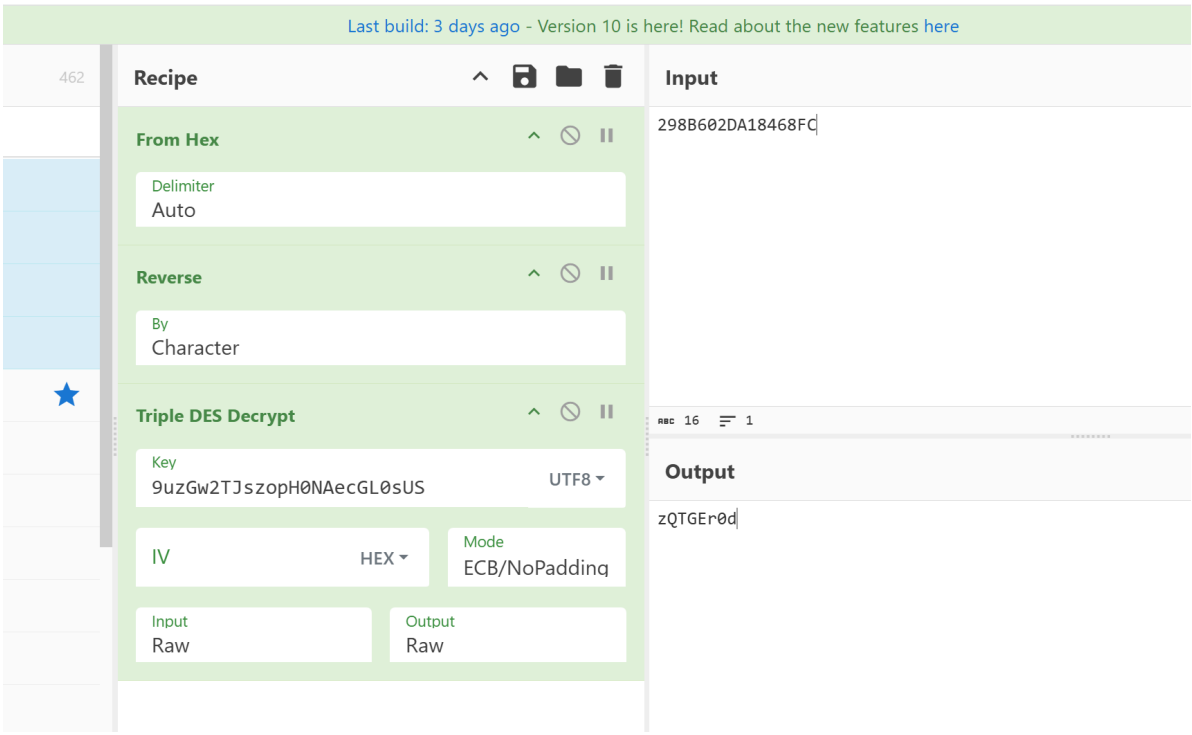
```
bool __fastcall real_check(__int64 a1, __int64 a2, __int64 a3)
{
  __int64 v4; // rbx
  unsigned int v5; // ebp
  __int64 v6; // r14
  __int64 v7; // rax
  char v9[4]; // [rsp+4h] [rbp-54h] BYREF
  __int64 v10; // [rsp+8h] [rbp-50h] BYREF
  __int128 v11[2]; // [rsp+10h] [rbp-48h] BYREF
  unsigned __int64 v12; // [rsp+30h] [rbp-28h]

  v12 = __readfsqword(0x28u);
  v4 = (*(__int64 (__fastcall **)(__int64, __int64, _QWORD))(*(_QWORD *)a1 + 1472LL))(a1, a3, 0LL);
  v5 = (*(__int64 (__fastcall **)(__int64, __int64))(*(_QWORD *)a1 + 1368LL))(a1, a3);
  qmemcpy(v11, "9uzGw2TJszopH0NAecGL0sUS", 24);
  v6 = EVP_CIPHER_CTX_new();
  v7 = EVP_des_ede3_ecb();
  EVP_EncryptInit_ex(v6, v7, 0LL, v11, 0LL);
  EVP_CIPHER_CTX_set_padding(v6, 0LL);
  EVP_EncryptUpdate(v6, &v10, v9, v4, v5);
  EVP_CIPHER_CTX_free(v6);
  return v10 == 0x298B602DA18468FCLL;
}
```

密文是

```
bool __fastcall real_check(__int64 a1, __int64 a2, __int64 a3)
{
  __int64 v4; // rbx
  unsigned int v5; // ebp
  __int64 v6; // r14
  __int64 v7; // rax
  char v9[4]; // [rsp+4h] [rbp-54h] BYREF
  __int64 v10; // [rsp+8h] [rbp-50h] BYREF
  __int128 v11[2]; // [rsp+10h] [rbp-48h] BYREF
  unsigned __int64 v12; // [rsp+30h] [rbp-28h]

  v12 = __readfsqword(0x28u);
  v4 = (*(__int64 (__fastcall **)(__int64, __int64, _QWORD))(*(_QWORD *)a1 + 1472LL
  v5 = (*(__int64 (__fastcall **)(__int64, __int64))(*(_QWORD *)a1 + 1368LL))(a1, a
  qmemcpy(v11, "9uzGw2TJszopH0NAecGL0sUS", 24);
  v6 = EVP_CIPHER_CTX_new();
  v7 = EVP_des_ede3_ecb();
  EVP_EncryptInit_ex(v6, v7, 0LL, v11, 0LL);
  EVP_CIPHER_CTX_set_padding(v6, 0LL);
  EVP_EncryptUpdate(v6, &v10, v9, v4, v5);
  EVP_CIPHER_CTX_free(v6);
  return v10 == 0x298B602DA18468FCLL;
}
```



两部分拼接包ISCC即可