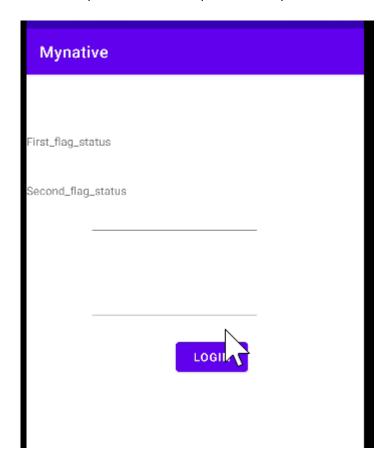
Solution for MemDump

Memdump CTF Solution (arm64-v8a)



Using jadx check the apk source code

By analyzing the source code we can observe that magic method is getting loaded from "InmemoryDexloader" it means the dex file is stored in application memory

```
Final Working.apk
                                                                                                 MainActivity
 Source code
                                                                                                                  WebView webView4 = this.webView;
    mandroid.support.p000v4
   □ androidx
                                                                                                                        Intrinsics.throwUninitializedPropertyAccessException("webView");
   □ com
                                                                                                                       webView = webView4

    example.mynative

        m databinding
                                                                                                                  webView.setVisibility(8);
         C0570R
                                                                                                     Memoryload((string) Androidmemory.element);
btn_click_me.setOnClickListener(new View.OnClickListener() {
// from class: com.example.mynative.MainActivity$$ExternalSyntheticLambda0
         MainActivity
           @ Companion
                                                                                                                       @Override // android.view.View.OnClickListener
           % Companion MainActivity$Compar
                                                                                                                       public final void onClick(View view) {
   MainActivity.onCreate$lambda$0(MainActivity.this, Androidmemory, view);
           % binding ActivityMainBinding
% btBuffer ByteBuffer
            €webView WebView
           /* JADX INFO: Access modifiers changed from: private */
public static final void onCreate$lambda$0(MainActivity this$0, Ref.ObjectRef Androidmemory, View it) {
   Intrinsics.checkNotNullParameter(this$0, "this$0");
   Intrinsics.checkNotNullParameter(Androidmemory, "$Androidmemory");
   if (this$0.buffer == null) {
        this$0.Memoryload((String) Androidmemory.element);
   }
}
            Nemoryload(String)
           Memoryload$lambda$1(Ref$ObjectRef, MainActivity)
           🛚 getBtBuffer() ByteBuffer
           mgetBuffer() byte[]
            mewcall(String) String
            @onCreate(Bundle) void
           📠 onCreate$lambda$0(MainActivity, Ref$ObjectRef, View
                                                                                                                 J
ByteBuffer wrap = ByteBuffer wrap(this$0.getBuffer());
Intrinsics.checkNotNullExpre sionValue(wrap, "wrap(buffer)");
           setBtBuffer(ByteBuffer) void
            % setBuffer(byte[]) void
                                                                                                                  this$0.setBtBuffer(wrap);
InMemoryDexClassLoader(this$0.getBtBuffer(), this$0.getClass().get
           % stringFromJNI() String
                                                                                                                 Class mt = lder.loadClass("com.example.ctf.Encrypt");
Method checkWethodInMemory = mt.getWethod("magic", String.class);
Object new1 = mt.newInstance();
ActivityMainBinding activityMainBinding = this$0.binding;
           % webFromJNI() String
                                                                                                 62
      m google
   min.arjsna.lib
   _ kotlin
                                                                                                                 ActivityMainBinding activityMainBinding2 = null;
if (activityMainBinding == null) {
    Intrinsics.throwUninitializedPropertyAccessException("binding");
   kotlinx.coroutines
   org
 ■ Resources
                                                                                                                       activityMainBinding = null;
 APK signature
                                                                                                                     ring params = activityMainBinding.editTextText.getText().toString();
                                                                                                                 ActivityMainBinding activityMainBinding3 = this$0.binding;
if (activityMainBinding3 == null) {
    Intrinsics.throwUninitializedPropertyAccessException("binding");
                                                                                                                       activityMainBinding3 = null;
```

Dumping the dex file using frida

https://github.com/hluwa/frida-dexdump

```
https://github.com/hluwa/frida-dexdump
NFO:Agent:DexDumpAgent<Connection(pid=Session(pid=13476), connected:True), attached=True>: Attach.
NFO:frida-dexdump:[+] Searching...
NFO:frida-dexdump:[*] Successful found 19 dex, used 8 time.
NFO:frida-dexdump:[+] Starting dump to '/tmp/Mynative'...
NFO:frida-dexdump:[+] DexMd5=52c3b7b88941484c9f4e438a92f01b15, SavePath=/tmp/Mynative/classes.dex, DexSize=0x4ec
NFO:frida-dexdump:[+] DexMd5=52c3b7b88941484c9f4e438a92f01b15,
NFO:frida-dexdump:[+] DexMd5=b3824797f5e5137ebc067ab0be58bba1,
NFO:frida-dexdump:[+] DexMd5=2dfcd5726f3f24c47d0ae5fe1170-ref
                                                                                                    SavePath=/tmp/Mynative/classes02.dex, DexSize=0x289ef4
SavePath=/tmp/Mynative/classes03.dex, DexSize=0x91e4ec
NFO:frida-dexdump:[+] DexMd5=4d22e8f158c512d7d70335ebe6f71ee,
NFO:frida-dexdump:[+] DexMd5=5b75de85d1963c518cc39440e203e80a,
                                                                                                     SavePath=/tmp/Mynative/classes04.dex, DexSize=0x7bf98
                                                                                                     SavePath=/tmp/Mynative/classes05.dex, DexSize=0x78bac
NFO:frida-dexdump:[+] DexMd5=ecf7cddd075183ac84db1677966211d0,
                                                                                                     SavePath=/tmp/Mynative/classes06.dex, DexSize=0x1274a4
NFO:frida-dexdump:[+] DexMd5=e920130e06b5687afe980ddb8e3b4425,
                                                                                                     SavePath=/tmp/Mynative/classes07.dex,
                                                                                                                                                                  DexSize=0x1557b4
NFO:frida-dexdump:[+] DexMd5=ae71af4d4bbeb1159bf32cfc69ffc152,
NFO:frida-dexdump:[+] DexMd5=97a38419f73794ace5f7e37eafc7fcc4,
NFO:frida-dexdump:[+] DexMd5=a5624a6b12dd9e111ecbcf33d9e04fe7,
                                                                                                     SavePath=/tmp/Mynative/classes08.dex,
                                                                                                                                                                 DexSize=0x3259b4
                                                                                                    SavePath=/tmp/Mynative/classes09.dex, DexSize=0x4ba028
SavePath=/tmp/Mynative/classes10.dex, DexSize=0x7180c
SavePath=/tmp/Mynative/classes11.dex, DexSize=0x63abc
                                                                                                                                                                 DexSize=0x4ba028
NFO:frida-dexdump:[+] DexMd5=e1a66fc4ca8eddd80c1916e426dbb7d1,
NFO:frida-dexdump:[+] DexMd5=a9ff2dd9e894b96d2e51f9613f1b3497,
NFO:frida-dexdump:[+] DexMd5=baf6cce0de7c214e1add564ece343ebf,
                                   DexMd5=e1a66fc4ca8eddd80c1916e426dbb7d1,
                                                                                                    SavePath=/tmp/Mynative/classes12.dex, DexSize=0x224c
SavePath=/tmp/Mynative/classes13.dex, DexSize=0x33c
NFO:frida-dexdump:[+] DexMd5=c276
NFO:frida-dexdump:[*] All done...
                                   DexMd5=c276056fed4094a8148b17fa896e2e23,
                                                                                                    SavePath=/tmp/Mynative/classes14.dex,
                                                                                                                                                                  DexSize=0xa6c
  /tmp
```

Open the classes.dex file in jadx and observed that magic function is there and it perform Bytecode XOR of salt and AUTH_TOKEN to get flag

```
classes.dex
                                                                         @ Encrypt
 Source code
                                                                           package com.example.ctf;

    com.example.ctf

    e Encrypt
                                                                           import java.util.Arrays;

salt String
                                                                            /* Loaded from: D:\document\CTF _inti\classes.dex */
      MAUTH_TOKEN byte[]
      magic(String) String
      magic(String, byte[]) byte[]
                                                                                private static final byte[] AUTH_TOKEN = {42, 45, 51, 22, 11, 11, 27, 12, 15, 13, 0, 48
Resources
                                                                                public static byte[] magic(String passparam, byte[] crypt) {

□ Summary

                                                                                    byte[] bytes = passparam.getBytes();
byte[] crypt2 = new byte[bytes.length];
for (int i = 0; i < bytes.length; i++) {
    crypt2[i] = (byte) (bytes[i] ^ crypt[i % crypt.length]);</pre>
                                                                        14
                                                                        16
                                                                                    return crypt2;
                                                                                public static String magic(String passparam)
                                                                        21
                                                                                    if (!Arrays.equals(magic(passparam, salt.getBytes()), AUTH_TOKEN)) {
                                                                                        return "False";
                                                                        22
                                                                                    return "True";
```

By writing the below java code we can obtain the flag

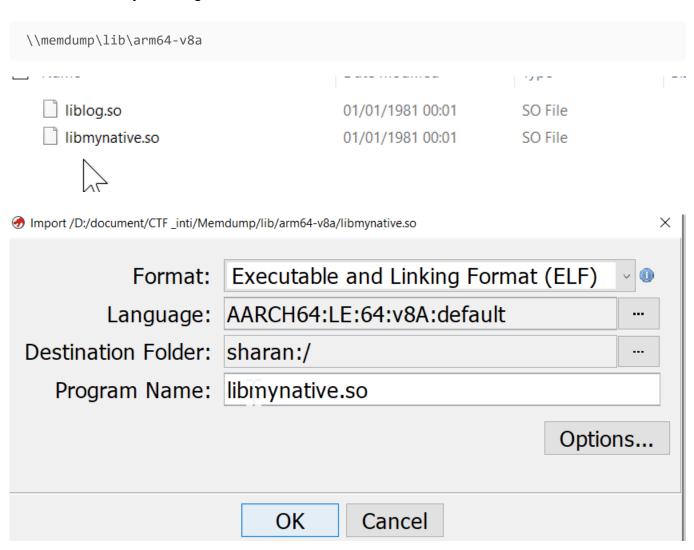
First part of flag

Flag{Intigritispecial

Second part of flag

By again observing with second editext we can observe the newcall is getting called which native function defined at the start

Native Code analysis using Ghidra

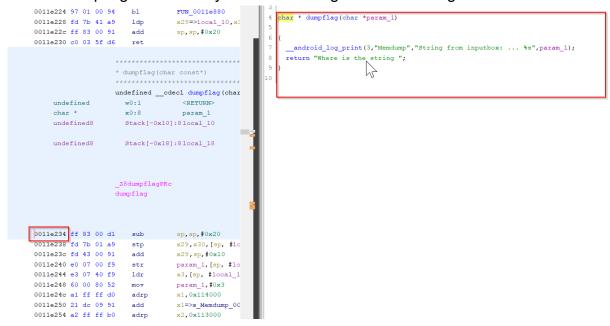


dumpflag function is getting called in native which return the varible which is get compared with

if return value matches "RheO5PB6mfL5N3YBH45e5XuCEaWpvWUFESqTYnZk" flag will be leaked in logcat android_log_print()

```
undefined8
 Java_com_example_mynative_MainActivity_newcall
           (_JNIEnv *param_1,undefined8 param_2,_jstring *param_3)
5
  {
   int iVar1;
  long lVar2;
  int iVar3;
  char *pcVar4;
  int local 2d4;
  undefined8 local 298;
  byte abStack576 [256];
  byte abStack320 [256];
  undefined8 uStack64;
  undefined8 uStack56;
  long local_28;
  1Var2 = cRead 8(tpidr el0);
)
  local 28 = *(long *)(lVar2 + 0x28);
   pcVar4 = (char *) JNIEnv::GetStringUTFChars(param 1, param 3, (uchar *) 0x0);
  pcVar4 = (char *)dumpflag(pcVar4);
   if (pcVar4 == "RheO5PB6mfL5N3YBH45e5XuCEaWpvWUFESqTYnZk") {
    uStack56 = 0x3f2a432354240505;
    uStack64 = 0x422b274624060720;
    __memcpy_chk(abStack320,"RheO5PB6mfL5N3YBH45e5XuCEaWpvWUFESqTYnZk",(long)iVar3,0x100);
     __memset_chk(abStack576,0,0x10,0x100);
    for (local_2d4 = 0; local_2d4 < 0x10; local_2d4 = local_2d4 + 1) {
      iVar1 = 0;
      if (iVar3 != 0) {
       iVar1 = local_2d4 / iVar3;
      abStack576[local_2d4] =
           *(byte *)((long)&uStack64 + (long)local 2d4) ^ abStack320[local 2d4 - iVar1 * iVar3];
     __android_log_print(3,"Memdump","Success Second part: ... %s",abStack576);
    local_298 = _JNIEnv::NewStringUTF(param_1,"True");
3
   }
)
   else {
     __android_log_print(3,"Memdump","Wrong value: ... %s",pcVar4);
    local_298 = _JNIEnv::NewStringUTF(param_1, "False");
3
   1Var2 = cRead_8(tpidr_el0);
   if (*(long *)(lVar2 + 0x28) == local_28) {
    return local_298;
                   /* WARNING: Subroutine does not return */
   __stack_chk_fail();
```

For the dumpflag function only return the string "Where is string "



Note the memory address of dumpflag function "0x0011e234"

Now to return the string "RheO5PB6mfL5N3YBH45e5XuCEaWpvWUFESqTYnZk" using frida we need find the address of the above string which is "0x00113ed5"

```
"libunwind: malformed DW CFA offset extended D...
                                                                                                  byte abStack320 [256];
         75 6e 77
                                                                                                  undefined8 uStack64;
         69 6e 64 ...
                                                                                                  undefined8 uStack56;
 00113ed2 78
                                    78h
                                                                                                  long local 28;
 00113ed3 37
                                    37h
                                                                                                  lVar2 = cRead 8(tpidr el0);
                                                                                                  local 28 = *(long *)(lVar2 + 0x28);
                                                                                 Java_c
                     s RheO5PB6mfL5N3YBH45e5XuCEaWpvWUF 00113ed5 XREF[5]:
                                                                                                  pcVar4 = (char *)_JNIEnv::GetStringUTFChars(param_1,param_3, (uchar *)0x0
                                                                                 Java_c
                                                                                                  pcVar4 = (char *)dumpflag(pcVar4);
                                                                                 Java_c
                                                                                                  if (pcVar4 == "RheO5PB6nfL5N3YBH45e5XuCEaWpvWUFESqTYnZk") {
                                                                                 Java_c
                                                                                                   uStack56 = 0x3f2a432354240505;
                                                                                                    uStack64 = 0x422b274624060720;
00113ed5 52 68 65
                                    "RheO5PB6mfL5N3YBH45e5XuCEaWpvWUFESgTYnZk"
                                                                                                    iVar3 = __strlen_chk("RheO5PB6mfL5N3YBH45e5XuCEaWpvWUFESqTYnZk",0xfff;
          4f 35 50
                                                                                                    __memcpy_chk(abStack320,"RheO5PB6mfL5N3YBH45e5XuCEaWpvWUFESqTYnZk",(1c
                                                                                                      memset chk(abStack576,0,0x10,0x100);
                                                                                                    for (local 2d4 = 0; local 2d4 < 0x10; local 2d4 = local 2d4 + 1) {
                                                                    XREF[1]:
                                                                                                      iVar1 = 0;
 00113efe 3e 20 74
                                                                                                      if (iVar3 != 0) {
          79 70 65
                                                                                                         iver1 = loce1 2d4 / iver3.
```

We have dumpflag =0x0011e234 string =0x00113ed5

using below frida script we can return the value of dumpflag to pointer 0x00113ed5

```
const ghidraImageBase = 0x00100000; // example value get the real value in Ghidra
from Window -> Memory map -> Set Image Base
const moduleName = "libmynative.so";
const moduleBaseAddress = Module.findBaseAddress(moduleName);
const functionRealAddress = moduleBaseAddress.add(0x0011e234 - ghidraImageBase);
const functionRealAddress2 = moduleBaseAddress.add(0x00113ed5 - ghidraImageBase);
Interceptor.attach(functionRealAddress, {
    onEnter: function(args) {
```

```
return 0;
},
onLeave: function(retval) {

console.log(Memory.readCString(functionRealAddress2))
retval.replace(functionRealAddress2)
console.log("Flag Found")

},
});
```

```
dock video/avc
const moduleName = **Himpmativa in: ;
const moduleName = **Himpmativa in: ;
const moduleBaseAddress = Module AindRaseAddress(moduleName);
const functionRealAddress = moduleBaseAddress.add(0x0811234 - ghidraIm
ageBase);
const functionRealAddress2 = moduleBaseAddress.add(0x080113ed5 - ghidraIm
ageBase);
Interceptor.attach(functionRealAddress, {
    onEnter: function(args) {
        return 0;
        onLeave: function(retval) {

        console.log(Memory.readCString(functionRealAddress2))
        return.replace(functionRealAddress2)

        console.log(Flag Found*)

        },
    };
};
{}
GM1913::com.example.mynative ]-> RheOSPB6mfLSN3YBH45e5XuCEaWpvWUFESqTYnZk
Flag Found

} console.log(**Pag Found**)

### Provided Research of the Consoleration of the Consolera
```

Second part of flag

rockswithchamps}

Full flag

Flag{Intigritispecialrockswithchamps}