MoeCTF 2023

Reverse入门指北

> moectf{F1rst_St3p_1s_D0ne}

base_64

base64换表

换的表为：ZYXWVUTSRQPONMLKJIHGFEDCBAzyxwvutsrqponmlkjihgfedcba0123456789+/

解密后

> moectf{pYc_And_Base64~}

Xor

异或

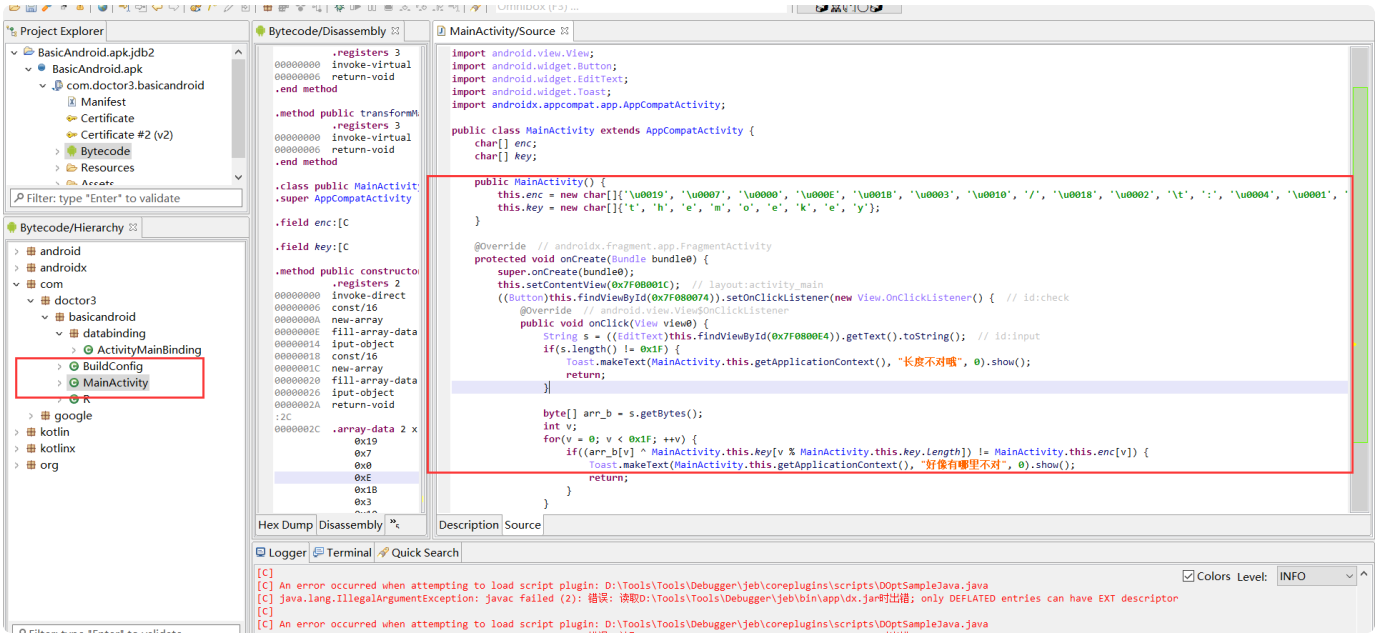
```
''.join([chr(get_wide_byte(0x14000C010+i)^0x39)for i in range(28)])
```

> moectf{You_kn0w_h0w_t0_X0R!}

Android

异或

直接JEB开逆，找MainActivity




```
c=['\u0019', '\u0007', '\u0000', '\u000E', '\u001B', '\u0003', '\u0010', '/', '\u0018', '\u0002', '\t', ':', '\u0004', '\u0001', ':', '*', '\u000B', '\u001D', '\u0006', '\u0007', '\f', '\t', '0', 'T', '\u0018', ':', '\u001C', '\u0015', '\u001B', '\u001C', '\u0010']
c=[ord(x) for x in c]
key=[ord(x) for x in 'themoekey']
print("".join([chr(c[i]^key[i%len(key)]) for i in range(len(c))]))
```

> moectf{Java_in_Android_1s_easy}

UPX!

UPX 异或

直接upx -d脱壳

主函数在sub_140079760()

```
__int64 sub_140079760()
{
  char *v0; // rdi
  __int64 i; // rcx
  unsigned __int64 v2; // rax
  char v4[32]; // [rsp+0h] [rbp-20h] BYREF
  char v5; // [rsp+20h] [rbp+0h] BYREF
  char v6[76]; // [rsp+28h] [rbp+8h] BYREF
  int j; // [rsp+74h] [rbp+54h]
  unsigned __int64 v8; // [rsp+148h] [rbp+128h]

  v0 = &v5;
  for ( i = 34i64; i; --i )
  {
    *(_DWORD *)v0 = -858993460;
    v0 += 4;
  }
  sub_140075557(&unk_1401A7008);
  sub_140073581("welcome to moectf");
  sub_140073581("I put a shell on my program to prevent you from reversing it, you will never be able to reverse it hhhh~~");
  sub_140073581("Now tell me your flag:");
  memset(v6, 0, 0x2Aui64);
  sub_1400727F8("%s", v6);
  for ( j = 0; ; ++j )
  {
    v8 = j;
    v2 = sub_140073829(v6);
    if ( v8 >= v2 )
      break;
    v6[j] ^= 0x67u;
    if ( byte_140196000[j] != v6[j] )
    {
      sub_140073973("try again~~");
      sub_1400723F7(0i64);
    }
  }
  sub_140073973("you are so clever!");
  sub_140074BCF(v4, &unk_140162070);
  return 0i64;
}
```

异或，直接一把梭

```
print(''.join([chr(get_wide_byte(0x140196000+i)^0x67) for i in range(41)]))
```

Equation

z3求解

```c
int __cdecl main(int argc, const char **argv, const char **envp)
{
  char v4[24]; // [rsp+20h] [rbp-60h] BYREF
  int v5; // [rsp+38h] [rbp-48h]
  __int16 v6; // [rsp+3Ch] [rbp-44h]
  char v7; // [rsp+3Eh] [rbp-42h]

  _main(argc, argv, envp);
  puts("I have trouble in solving equations");
  puts("Could you help me? qwq:");
  memset(v4, 0, sizeof(v4));
  v5 = 0;
  v6 = 0;
  v7 = 0;
  scanf("%31s", v4);
  if ( 334 * (char)v6
     + 100 * SHIBYTE(v5)
     + 369 * SBYTE2(v5)
     + 124 * SBYTE1(v5)
     + 278 * (char)v5
     + 158 * v4[23]
     + 162 * v4[22]
     + 145 * v4[19]
     + 27 * v4[17]
     + 91 * v4[15]
     + 195 * v4[14]
```

改v4的定义为char v4[31]就可以方便看了

```
IDA View-A        Pseudocode-A        Hex View-1
int __cdecl main(int argc, const char **argv, const char **envp)
{
  char v4[31]; // [rsp+20h] [rbp-60h] BYREF

  _main(argc, argv, envp);
  puts("I have trouble in solving equations");
  puts("Could you help me? qwq:");
  memset(v4, 0, sizeof(v4));
  scanf("%31s", v4);
  if ( 334 * v4[28]
     + 100 * v4[27]
     + 369 * v4[26]
     + 124 * v4[25]
     + 278 * v4[24]
     + 158 * v4[23]
     + 162 * v4[22]
     + 145 * v4[19]
     + 27 * v4[17]
     + 91 * v4[15]
     + 195 * v4[14]
     + 342 * v4[13]
     + 391 * v4[10]
     + 204 * v4[9]
     + 302 * v4[8]
     + 153 * v4[7]
     + 292 * v4[6]
     + 382 * v4[5]
     + 221 * v4[4]
     + 316 * v4[3]
     + 118 * v4[2]
     + 295 * v4[1]
     + 247 * v4[0]
     + 236 * v4[11]
```

剩下的就是解一个方程组了，但是这里有一点比较坑的就是，这个只有30个方程，但是因为flag格式限制死了，所以可以解。

```
from z3 import *
s=Solver()
v4 = [BitVec('u%d' % i, 8) for i in range(0, 31)]
s.add(v4[0] == ord('m'))
s.add(v4[1] == ord('o'))
s.add(334 * v4[28]+ 100 * v4[27]+ 369 * v4[26]+ 124 * v4[25]+ 278 * v4[24]
+ 158 * v4[23]+ 162 * v4[22]+ 145 * v4[19]+ 27 * v4[17]+ 91 * v4[15]+ 195
* v4[14]+ 342 * v4[13]+ 391 * v4[10]+ 204 * v4[9]+ 302 * v4[8]+ 153 * v4[7
]+ 292 * v4[6]+ 382 * v4[5]+ 221 * v4[4]+ 316 * v4[3]+ 118 * v4[2]+ 295 *
v4[1]+ 247 * v4[0]+ 236 * v4[11]+ 27 * v4[12]+ 361 * v4[16]+ 81 * v4[18]+
105 * v4[20]+ 65 * v4[21]+ 67 * v4[29]+ 41 * v4[30] == 596119)
s.add(371 * v4[29]+ 338 * v4[28]+ 269 * v4[27]+ 312 * v4[26]+ 67 * v4[25]+
 299 * v4[24]+ 235 * v4[23]+ 294 * v4[22]+ 303 * v4[21]+ 211 * v4[20]+ 122
 * v4[19]+ 333 * v4[18]+ 341 * v4[15]+ 111 * v4[14]+ 253 * v4[13]+ 68 * v4
[12]+ 347 * v4[11]+ 44 * v4[10]+ 262 * v4[9]+ 357 * v4[8]+ 323 * v4[5]+ 14
1 * v4[4]+ 329 * v4[3]+ 378 * v4[2]+ 316 * v4[1]+ 235 * v4[0]+ 59 * v4[6]+
 37 * v4[7]+ 264 * v4[16]+ 73 * v4[17]+ 126 * v4[30] == 634009)
s.add(337 * v4[29]+ 338 * v4[28]+ 118 * v4[27]+ 82 * v4[26]+ 239 * v4[21]+
 58 * v4[20]+ 304 * v4[19]+ 330 * v4[18]+ 377 * v4[17]+ 306 * v4[16]+ 221
* v4[13]+ 345 * v4[12]+ 124 * v4[11]+ 272 * v4[10]+ 270 * v4[9]+ 229 * v4[
8]+ 377 * v4[7]+ 373 * v4[6]+ 297 * v4[5]+ 112 * v4[4]+ 386 * v4[3]+ 90 *
v4[2]+ 361 * v4[1]+ 236 * v4[0]+ 386 * v4[14]+ 73 * v4[15]+ 315 * v4[22]+
33 * v4[23]+ 141 * v4[24]+ 129 * v4[25]+ 123 * v4[30] == 685705)
s.add(367 * v4[29]+ 55 * v4[28]+ 374 * v4[27]+ 150 * v4[24]+ 350 * v4[23]+
 141 * v4[22]+ 124 * v4[21]+ 366 * v4[20]+ 230 * v4[19]+ 307 * v4[18]+ 191
 * v4[17]+ 153 * v4[12]+ 383 * v4[11]+ 145 * v4[10]+ 109 * v4[9]+ 209 * v4
[8]+ 158 * v4[7]+ 221 * v4[6]+ 188 * v4[5]+ 22 * v4[4]+ 146 * v4[3]+ 306 *
 v4[2]+ 230 * v4[1]+ 13 * v4[0]+ 287 * v4[13]+ 257 * v4[14]+ 137 * v4[15]+
 7 * v4[16]+ 52 * v4[25]+ 31 * v4[26]+ 355 * v4[30] == 557696)
s.add(100 * v4[29]+ 191 * v4[28]+ 362 * v4[27]+ 55 * v4[26]+ 210 * v4[25]+
 359 * v4[24]+ 348 * v4[21]+ 83 * v4[20]+ 395 * v4[19]+ 350 * v4[16]+ 291
* v4[15]+ 220 * v4[12]+ 196 * v4[11]+ 399 * v4[8]+ 68 * v4[7]+ 84 * v4[6]+
 281 * v4[5]+ 334 * v4[4]+ 53 * v4[3]+ 399 * v4[2]+ 338 * v4[0]+ 18 * v4[1
]+ 148 * v4[9]+ 21 * v4[10]+ 174 * v4[13]+ 36 * v4[14]+ 2 * v4[17]+ 41 * v
4[18]+ 137 * v4[22]+ 24 * v4[23]+ 368 * v4[30] == 538535)
s.add(188 * v4[29]+ (v4[26] << 7)+ 93 * v4[25]+ 248 * v4[24]+ 83 * v4[23]+
 207 * v4[22]+ 217 * v4[19]+ 309 * v4[16]+ 16 * v4[15]+ 135 * v4[14]+ 251
* v4[13]+ 200 * v4[12]+ 49 * v4[11]+ 119 * v4[10]+ 356 * v4[9]+ 398 * v4[8
]+ 303 * v4[7]+ 224 * v4[6]+ 208 * v4[5]+ 244 * v4[4]+ 209 * v4[3]+ 189 *
v4[2]+ 302 * v4[1]+ 395 * v4[0]+ 314 * v4[17]+ 13 * v4[18]+ 310 * v4[20]+
21 * v4[21]+ 67 * v4[27]+ 127 * v4[28]+ 100 * v4[30] == 580384)
s.add(293 * v4[29]+ 343 * v4[28]+ 123 * v4[27]+ 387 * v4[26]+ 114 * v4[25]
+ 303 * v4[24]+ 248 * v4[23]+ 258 * v4[21]+ 218 * v4[20]+ 180 * v4[19]+ 19
6 * v4[18]+ 398 * v4[17]+ 398 * v4[14]+ 138 * v4[9]+ 292 * v4[8]+ 38 * v4[
7]+ 179 * v4[6]+ 190 * v4[5]+ 57 * v4[4]+ 358 * v4[3]+ 191 * v4[2]+ 215 *
v4[1]+ 88 * v4[0]+ 22 * v4[10]+ 72 * v4[11]+ 357 * v4[12]+ 9 * v4[13]+ 389
 * v4[15]+ 81 * v4[16]+ 85 * v4[30] == 529847)
```

```
s.add(311 * v4[29]+ 202 * v4[28]+ 234 * v4[27]+ 272 * v4[26]+ 55 * v4[25]+ 328 * v4[24]+ 246 * v4[23]+ 362 * v4[22]+ 86 * v4[21]+ 75 * v4[20]+ 142 * v4[17]+ 244 * v4[16]+ 216 * v4[15]+ 281 * v4[14]+ 398 * v4[13]+ 322 * v4[12]+ 251 * v4[11]+ 357 * v4[8]+ 76 * v4[7]+ 292 * v4[6]+ 389 * v4[5]+ 275 * v4[4]+ 312 * v4[3]+ 200 * v4[2]+ 110 * v4[1]+ 203 * v4[0]+ 99 * v4[9]+ 21 * v4[10]+ 269 * v4[18]+ 33 * v4[19]+ 356 * v4[30] == 631652)
s.add(261 * v4[29]+ 189 * v4[26]+ 55 * v4[25]+ 23 * v4[24]+ 202 * v4[23]+ 185 * v4[22]+ 182 * v4[21]+ 285 * v4[20]+ 217 * v4[17]+ 157 * v4[16]+ 232 * v4[15]+ 132 * v4[14]+ 169 * v4[13]+ 154 * v4[12]+ 121 * v4[11]+ 389 * v4[10]+ 376 * v4[9]+ 292 * v4[6]+ 225 * v4[5]+ 155 * v4[4]+ 234 * v4[3]+ 149 * v4[2]+ 241 * v4[1]+ 312 * v4[0]+ 368 * v4[7]+ 129 * v4[8]+ 226 * v4[18]+ 288 * v4[19]+ 201 * v4[27]+ 288 * v4[28]+ 69 * v4[30] == 614840)
s.add(60 * v4[29]+ 118 * v4[28]+ 153 * v4[27]+ 139 * v4[26]+ 23 * v4[25]+ 279 * v4[24]+ 396 * v4[23]+ 287 * v4[22]+ 237 * v4[19]+ 266 * v4[18]+ 149 * v4[17]+ 193 * v4[16]+ 395 * v4[15]+ 97 * v4[14]+ 16 * v4[13]+ 286 * v4[12]+ 105 * v4[11]+ 88 * v4[10]+ 282 * v4[9]+ 55 * v4[8]+ 134 * v4[7]+ 114 * v4[6]+ 101 * v4[5]+ 116 * v4[4]+ 271 * v4[3]+ 186 * v4[2]+ 263 * v4[1]+ 313 * v4[0]+ 149 * v4[20]+ 129 * v4[21]+ 145 * v4[30] == 510398)
s.add(385 * v4[29]+ 53 * v4[28]+ 112 * v4[27]+ 8 * v4[26]+ 232 * v4[25]+ 145 * v4[24]+ 313 * v4[23]+ 156 * v4[22]+ 321 * v4[21]+ 358 * v4[20]+ 46 * v4[19]+ 382 * v4[18]+ 144 * v4[16]+ 222 * v4[14]+ 329 * v4[13]+ 161 * v4[12]+ 335 * v4[11]+ 50 * v4[10]+ 373 * v4[9]+ 66 * v4[8]+ 44 * v4[7]+ 59 * v4[6]+ 292 * v4[5]+ 39 * v4[4]+ 53 * v4[3]+ 310 * v4[0]+ 154 * v4[1]+ 24 * v4[2]+ 396 * v4[15]+ 81 * v4[17]+ 355 * v4[30] == 558740)
s.add(249 * v4[29]+ 386 * v4[28]+ 313 * v4[27]+ 74 * v4[26]+ 22 * v4[25]+ 168 * v4[24]+ 305 * v4[21]+ 358 * v4[20]+ 191 * v4[19]+ 202 * v4[18]+ 14 * v4[15]+ 114 * v4[14]+ 224 * v4[13]+ 134 * v4[12]+ 274 * v4[11]+ 372 * v4[10]+ 159 * v4[9]+ 233 * v4[8]+ 70 * v4[7]+ 287 * v4[6]+ 297 * v4[5]+ 318 * v4[4]+ 177 * v4[3]+ 173 * v4[2]+ 270 * v4[1]+ 163 * v4[0]+ 77 * v4[16]+ 25 * v4[17]+ 387 * v4[22]+ 18 * v4[23]+ 345 * v4[30] == 592365)
s.add(392 * v4[29]+ 385 * v4[28]+ 302 * v4[27]+ 13 * v4[25]+ 27 * v4[24]+ 99 * v4[22]+ 343 * v4[19]+ 324 * v4[18]+ 223 * v4[17]+ 372 * v4[16]+ 261 * v4[15]+ 181 * v4[14]+ 203 * v4[13]+ 232 * v4[12]+ 305 * v4[11]+ 393 * v4[10]+ 325 * v4[9]+ 231 * v4[8]+ 92 * v4[7]+ 142 * v4[6]+ 22 * v4[5]+ 86 * v4[4]+ 264 * v4[3]+ 300 * v4[2]+ 387 * v4[1]+ 360 * v4[0]+ 225 * v4[20]+ 127 * v4[21]+ 2 * v4[23]+ 80 * v4[26]+ 268 * v4[30] == 619574)
s.add(270 * v4[28]+ 370 * v4[27]+ 235 * v4[26]+ 96 * v4[22]+ 85 * v4[20]+ 150 * v4[19]+ 140 * v4[18]+ 94 * v4[17]+ 295 * v4[16]+ 19 * v4[14]+ 176 * v4[12]+ 94 * v4[11]+ 258 * v4[10]+ 302 * v4[9]+ 171 * v4[8]+ 66 * v4[7]+ 278 * v4[6]+ 193 * v4[5]+ 251 * v4[4]+ 284 * v4[3]+ 218 * v4[2]+ (v4[1] << 6)+ 319 * v4[0]+ 125 * v4[13]+ 24 * v4[15]+ 267 * v4[21]+ 160 * v4[23]+ 111 * v4[24]+ 33 * v4[25]+ 174 * v4[29]+ 13 * v4[30] == 480557)
s.add(87 * v4[28]+ 260 * v4[27]+ 326 * v4[26]+ 210 * v4[25]+ 357 * v4[24]+ 170 * v4[23]+ 315 * v4[22]+ 376 * v4[21]+ 227 * v4[20]+ 43 * v4[19]+ 358 * v4[18]+ 364 * v4[17]+ 309 * v4[16]+ 282 * v4[15]+ 286 * v4[14]+ 365 * v4[13]+ 287 * v4[12]+ 377 * v4[11]+ 74 * v4[10]+ 225 * v4[9]+ 328 * v4[6]+ 223 * v4[5]+ 120 * v4[4]+ 102 * v4[3]+ 162 * v4[2]+ 123 * v4[1]+ 196 * v4[0]+ 29 * v4[7]+ 27 * v4[8]+ 352 * v4[30] == 666967)
```

```
s.add(61 * v4[29]+ 195 * v4[28]+ 125 * v4[27]+ (v4[26] << 6)+ 260 * v4[25]
+ 202 * v4[24]+ 116 * v4[23]+ 230 * v4[22]+ 326 * v4[21]+ 211 * v4[20]+ 37
1 * v4[19]+ 353 * v4[16]+ 124 * v4[13]+ 188 * v4[12]+ 163 * v4[11]+ 140 *
v4[10]+ 51 * v4[9]+ 262 * v4[8]+ 229 * v4[7]+ 100 * v4[6]+ 113 * v4[5]+ 15
8 * v4[4]+ 378 * v4[3]+ 365 * v4[2]+ 207 * v4[1]+ 277 * v4[0]+ 190 * v4[14
]+ 320 * v4[15]+ 347 * v4[17]+ 11 * v4[18]+ 137 * v4[30] == 590534)
s.add(39 * v4[28]+ 303 * v4[27]+ 360 * v4[26]+ 157 * v4[25]+ 324 * v4[24]+
 77 * v4[23]+ 308 * v4[22]+ 313 * v4[21]+ 87 * v4[20]+ 201 * v4[19]+ 50 *
v4[18]+ 60 * v4[17]+ 28 * v4[16]+ 193 * v4[15]+ 184 * v4[14]+ 205 * v4[13]
+ 140 * v4[12]+ 311 * v4[11]+ 304 * v4[10]+ 35 * v4[9]+ 356 * v4[8]+ 23 *
v4[5]+ 85 * v4[4]+ 156 * v4[3]+ 16 * v4[2]+ 26 * v4[1]+ 157 * v4[0]+ 150 *
 v4[6]+ 72 * v4[7]+ 58 * v4[29] == 429108)
s.add(157 * v4[29]+ 137 * v4[28]+ 71 * v4[27]+ 269 * v4[26]+ 161 * v4[25]+
 317 * v4[20]+ 296 * v4[19]+ 385 * v4[18]+ 165 * v4[13]+ 159 * v4[12]+ 132
 * v4[11]+ 296 * v4[10]+ 162 * v4[7]+ 254 * v4[4]+ 172 * v4[3]+ 132 * v4[0
]+ 369 * v4[1]+ 257 * v4[2]+ 134 * v4[5]+ 384 * v4[6]+ 53 * v4[8]+ 255 * v
4[9]+ 229 * v4[14]+ 129 * v4[15]+ 23 * v4[16]+ 41 * v4[17]+ 112 * v4[21]+
17 * v4[22]+ 222 * v4[23]+ 96 * v4[24]+ 126 * v4[30] == 563521)
s.add(207 * v4[29]+ 83 * v4[28]+ 111 * v4[27]+ 35 * v4[26]+ 67 * v4[25]+ 1
38 * v4[22]+ 223 * v4[21]+ 142 * v4[20]+ 154 * v4[19]+ 111 * v4[18]+ 341 *
 v4[17]+ 175 * v4[16]+ 259 * v4[15]+ 225 * v4[14]+ 26 * v4[11]+ 334 * v4[1
0]+ 250 * v4[7]+ 198 * v4[6]+ 279 * v4[5]+ 301 * v4[4]+ 193 * v4[3]+ 334 *
 v4[2]+ 134 * v4[0]+ 37 * v4[1]+ 183 * v4[8]+ 5 * v4[9]+ 270 * v4[12]+ 21
* v4[13]+ 275 * v4[23]+ 48 * v4[24]+ 163 * v4[30] == 493999)
s.add(393 * v4[29]+ 176 * v4[28]+ 105 * v4[27]+ 162 * v4[26]+ 148 * v4[25]
+ 281 * v4[24]+ 300 * v4[23]+ 342 * v4[18]+ 262 * v4[17]+ 152 * v4[12]+ 43
 * v4[11]+ 296 * v4[10]+ 273 * v4[9]+ 75 * v4[6]+ 18 * v4[4]+ 217 * v4[2]+
 132 * v4[1]+ 112 * v4[0]+ 210 * v4[3]+ 72 * v4[5]+ 113 * v4[7]+ 40 * v4[8
]+ 278 * v4[13]+ 24 * v4[14]+ 77 * v4[15]+ 11 * v4[16]+ 55 * v4[19]+ 255 *
 v4[20]+ 241 * v4[21]+ 13 * v4[22]+ 356 * v4[30] == 470065)
s.add(369 * v4[29]+ 231 * v4[28]+ 285 * v4[25]+ 290 * v4[24]+ 297 * v4[23]
+ 189 * v4[22]+ 390 * v4[21]+ 345 * v4[20]+ 153 * v4[19]+ 114 * v4[18]+ 25
1 * v4[17]+ 340 * v4[16]+ 44 * v4[15]+ 58 * v4[14]+ 335 * v4[13]+ 359 * v4
[12]+ 392 * v4[11]+ 181 * v4[8]+ 103 * v4[7]+ 229 * v4[6]+ 175 * v4[5]+ 20
8 * v4[4]+ 92 * v4[3]+ 397 * v4[2]+ 349 * v4[1]+ 356 * v4[0]+ (v4[9] << 6)
+ 5 * v4[10]+ 88 * v4[26]+ 40 * v4[27]+ 295 * v4[30] == 661276)
s.add(341 * v4[27]+ 40 * v4[25]+ 374 * v4[23]+ 201 * v4[22]+ 77 * v4[21]+
215 * v4[20]+ 283 * v4[19]+ 213 * v4[18]+ 392 * v4[17]+ 224 * v4[16]+ v4[1
5]+ 270 * v4[12]+ 28 * v4[11]+ 75 * v4[8]+ 386 * v4[7]+ 298 * v4[6]+ 170 *
 v4[5]+ 287 * v4[4]+ 247 * v4[3]+ 204 * v4[2]+ 103 * v4[1]+ 21 * v4[0]+ 84
 * v4[9]+ 27 * v4[10]+ 159 * v4[13]+ 192 * v4[14]+ 213 * v4[24]+ 129 * v4[
26]+ 67 * v4[28]+ 27 * v4[29]+ 361 * v4[30] == 555288)
s.add(106 * v4[29]+ 363 * v4[28]+ 210 * v4[27]+ 171 * v4[26]+ 289 * v4[25]
+ 240 * v4[24]+ 164 * v4[23]+ 342 * v4[22]+ 391 * v4[19]+ 304 * v4[18]+ 21
8 * v4[17]+ 32 * v4[16]+ 350 * v4[15]+ 339 * v4[12]+ 303 * v4[11]+ 222 * v
4[10]+ 298 * v4[9]+ 47 * v4[8]+ 48 * v4[6]+ 264 * v4[4]+ 113 * v4[3]+ 275
* v4[2]+ 345 * v4[1]+ 312 * v4[0]+ 171 * v4[5]+ 384 * v4[7]+ 175 * v4[13]+
 5 * v4[14]+ 113 * v4[20]+ 19 * v4[21]+ 263 * v4[30] == 637650)
```

```
s.add(278 * v4[29]+ 169 * v4[28]+ 62 * v4[27]+ 119 * v4[26]+ 385 * v4[25]+ 289 * v4[24]+ 344 * v4[23]+ 45 * v4[20]+ 308 * v4[19]+ 318 * v4[18]+ 270 * v4[17]+ v4[16]+ 323 * v4[15]+ 332 * v4[14]+ 287 * v4[11]+ 170 * v4[10]+ 163 * v4[9]+ 301 * v4[8]+ 303 * v4[7]+ 23 * v4[6]+ 327 * v4[5]+ 169 * v4[3]+ 28 * v4[0]+ 365 * v4[1]+ 15 * v4[2]+ 352 * v4[12]+ 72 * v4[13]+ 140 * v4[21]+ 65 * v4[22]+ 346 * v4[30] == 572609)
s.add(147 * v4[29]+ 88 * v4[28]+ 143 * v4[27]+ 237 * v4[26]+ 63 * v4[24]+ 281 * v4[22]+ 388 * v4[21]+ 142 * v4[20]+ 208 * v4[19]+ 60 * v4[18]+ 354 * v4[15]+ 88 * v4[14]+ 146 * v4[13]+ 290 * v4[12]+ 349 * v4[11]+ 43 * v4[10]+ 230 * v4[9]+ 267 * v4[6]+ 136 * v4[5]+ 383 * v4[4]+ 35 * v4[3]+ 226 * v4[2]+ 385 * v4[1]+ 238 * v4[0]+ 348 * v4[7]+ 20 * v4[8]+ 158 * v4[16]+ 21 * v4[17]+ 249 * v4[23]+ 9 * v4[25]+ 343 * v4[30] == 603481)
s.add(29 * v4[29]+ 323 * v4[26]+ 159 * v4[25]+ 118 * v4[20]+ 326 * v4[19]+ 211 * v4[18]+ 225 * v4[17]+ 355 * v4[16]+ 201 * v4[15]+ 149 * v4[14]+ 296 * v4[13]+ 184 * v4[12]+ 315 * v4[11]+ 364 * v4[10]+ 142 * v4[9]+ 75 * v4[8]+ 313 * v4[7]+ 142 * v4[6]+ 396 * v4[5]+ 348 * v4[4]+ 272 * v4[3]+ 26 * v4[2]+ 206 * v4[1]+ 173 * v4[0]+ 155 * v4[21]+ 144 * v4[22]+ 366 * v4[23]+ 257 * v4[24]+ 148 * v4[27]+ 24 * v4[28]+ 253 * v4[30] == 664504)
s.add(4 * v4[29]+ 305 * v4[28]+ 226 * v4[27]+ 212 * v4[26]+ 175 * v4[25]+ 93 * v4[24]+ 165 * v4[23]+ 341 * v4[20]+ 14 * v4[19]+ 394 * v4[18]+ (v4[17] << 8)+ 252 * v4[16]+ 336 * v4[15]+ 38 * v4[14]+ 82 * v4[13]+ 155 * v4[12]+ 215 * v4[11]+ 331 * v4[10]+ 230 * v4[9]+ 241 * v4[8]+ 225 * v4[7]+ 186 * v4[4]+ 90 * v4[3]+ 50 * v4[2]+ 62 * v4[1]+ 34 * v4[0]+ 237 * v4[5]+ 11 * v4[6]+ 336 * v4[21]+ 36 * v4[22]+ 29 * v4[30] == 473092)
s.add(353 * v4[29]+ 216 * v4[28]+ 252 * v4[27]+ 8 * v4[26]+ 62 * v4[25]+ 233 * v4[24]+ 254 * v4[23]+ 303 * v4[22]+ 234 * v4[21]+ 303 * v4[20]+ (v4[19] << 8)+ 148 * v4[18]+ 324 * v4[17]+ 317 * v4[16]+ 213 * v4[15]+ 309 * v4[14]+ 28 * v4[13]+ 280 * v4[11]+ 118 * v4[10]+ 58 * v4[9]+ 50 * v4[8]+ 155 * v4[7]+ 161 * v4[6]+ (v4[5] << 6)+ 303 * v4[4]+ 76 * v4[3]+ 43 * v4[2]+ 109 * v4[1]+ 102 * v4[0]+ 93 * v4[30] == 497492)
s.add(89 * v4[29]+ 148 * v4[28]+ 82 * v4[27]+ 53 * v4[26]+ 274 * v4[25]+ 220 * v4[24]+ 202 * v4[23]+ 123 * v4[22]+ 231 * v4[21]+ 169 * v4[20]+ 278 * v4[19]+ 259 * v4[18]+ 208 * v4[17]+ 219 * v4[16]+ 371 * v4[15]+ 181 * v4[12]+ 104 * v4[11]+ 392 * v4[10]+ 285 * v4[9]+ 113 * v4[8]+ 298 * v4[7]+ 389 * v4[6]+ 322 * v4[5]+ 338 * v4[4]+ 237 * v4[3]+ 234 * v4[0]+ 261 * v4[1]+ 10 * v4[2]+ 345 * v4[13]+ 3 * v4[14]+ 361 * v4[30] == 659149)
s.add(361 * v4[29]+ 359 * v4[28]+ 93 * v4[27]+ 315 * v4[26]+ 69 * v4[25]+ 137 * v4[24]+ 69 * v4[23]+ 58 * v4[22]+ 300 * v4[21]+ 371 * v4[20]+ 264 * v4[19]+ 317 * v4[18]+ 215 * v4[17]+ 155 * v4[16]+ 215 * v4[15]+ 330 * v4[14]+ 239 * v4[13]+ 212 * v4[12]+ 88 * v4[11]+ 82 * v4[10]+ 354 * v4[9]+ 85 * v4[8]+ 310 * v4[7]+ 84 * v4[6]+ 374 * v4[5]+ 380 * v4[4]+ 215 * v4[3]+ 351 * v4[2]+ 141 * v4[1]+ 115 * v4[0]+ 108 * v4[30] == 629123)
if s.check()==sat:
    m=s.model()
    flag=''.join([chr(m[v4[i]].as_long()) for i in range(31)])
    print(flag)
```

> moectf{y0u_s0lv3d_Equati0ns!!!}

RRRRRc4

魔改RC4

分析main得到下面的代码。

```
IDA View-A    Pseudocode-B    Pseudocode-A    字串
  char v7[44]; // [rsp+278h] [rbp+248h] BYREF
  int v8; // [rsp+2A4h] [rbp+274h]
  int j; // [rsp+2C4h] [rbp+294h]

  v0 = &v4;
  for ( i = 172i64; i; --i )
  {
    *v0 = -858993460;
    v0 += 4;
  }
  sub_14007555C(&unk_1401A7007);
  memset(v5, 0, sizeof(v5));
  memset(v6, 0, sizeof(v6));
  strcpy(v7, "moectf2023");
  v8 = 0;
  printf("welcome to moectf!!!");
  printf("This is a very common algorithm ");
  printf("show your flag:");
  sub_1400727F8("%s", byte_140197260);
  if ( sub_140073829(byte_140197260) == 37 )
  {
    Rc4(v5, v6, byte_140197260, 38, v7, 10);
    for ( j = 0; j < 0x26; ++j )
    {
      if ( byte_140196000[j] == byte_140197260[j] )
        ++v8;
    }
  }
  if ( v8 == 37 )
    sub_140073973("right!flag is your input!");
  else
    sub_140073973("try again~");
  sub_140074BCF(v3, &unk_140162100);
  return 0i64;
}
```

在跟进RC4得到

```
IDA View-A   Pseudocode-B   Pseudocode-A   字串   Hex View-1   Enu
__int64 result; // rax
int i; // [rsp+24h] [rbp+4h]
int j; // [rsp+24h] [rbp+4h]
int v9; // [rsp+24h] [rbp+4h]
int v10; // [rsp+44h] [rbp+24h]
int v11; // [rsp+44h] [rbp+24h]
char v12; // [rsp+64h] [rbp+44h]
char v13; // [rsp+64h] [rbp+44h]
int v14; // [rsp+A4h] [rbp+84h]

result = GetCurrId(&unk_1401A7007);
v10 = 0;
v14 = 0;
for ( i = 0; i < 256; ++i )
{
  *(a1 + i) = i;
  *(a2 + i) = *(a5 + i % a6);
  result = (i + 1);
}
for ( j = 0; j < 256; ++j )
{
  v10 = (*(a2 + j) + *(a1 + j) + v10) % 256;
  v12 = *(a1 + v10);
  *(a1 + v10) = *(a1 + j);
  *(a1 + j) = v12;
  result = (j + 1);
}
v9 = 0;
v11 = 0;
while ( a4 )
{
  v9 = (v9 + 1) % 256;
  v11 = (*(a1 + v9) + v11) % 256;
  v13 = *(a1 + v11);
  *(a1 + v11) = *(a1 + v9);
  *(a1 + v9) = v13;
  *(a3 + v14++) ^= *(a1 + (*(a1 + v11) + *(a1 + v9)) % 256);
  result = --a4;
}
return result;
}
00007ADD sub_1400795E0:25 (1400796DD)
```

这是一个魔改的RC4，不过S盒被魔改成，*(a2 + i) = *(a5 + i % a6);

因为RC4是对称加密，所以我们直接patch这个函数就可以了。

```
.text:0000000140079B34 E8 F0 9C FF FF          call    sub_140073829
.text:0000000140079B34
.text:0000000140079B39 48 83 F8 25             cmp     rax, 25h ; '%'
.text:0000000140079B3D 0F 85 90 00 00 00       jnz     loc_140079BD3
.text:0000000140079B3D
.text:0000000140079B43 C7 44 24 28 0A 00 00 00 mov     [rsp+470h+var_448], 0Ah
.text:0000000140079B4B 48 8D 85 48 02 00 00    lea     rax, [rbp+440h+var_1F8]
.text:0000000140079B52 48 89 44 24 20          mov     [rsp+470h+var_450], rax
.text:0000000140079B57 41 B9 26 00 00 00       mov     r9d, 26h ; '&'
.text:0000000140079B5D 4C 8D 05 FC D6 11 00    lea     r8, byte_140197260
.text:0000000140079B64 48 8D 95 30 01 00 00    lea     rdx, [rbp+440h+var_310]
.text:0000000140079B6B 48 8D 4D 10             lea     rcx, [rbp+440h+var_430]
.text:0000000140079B6F E8 DE B4 FF FF          call    Rc4
.text:0000000140079B6F
```

将这一行patch成密文所在的地方

```
.text:0000000140079B5D 4C 8D 05 9C C4 11 00          lea     r8, byte_140196000          ; Keypatch modified this from:
```

然后输入37位的伪码动调执行到这里call的下面

```
.text:00007FF6713A9B21 lea     rcx, aS                          ; "%s"
.text:00007FF6713A9B28 call    sub_7FF6713A27F8
.text:00007FF6713A9B28
.text:00007FF6713A9B2D lea     rcx, byte_7FF6714C7260
.text:00007FF6713A9B34 call    sub_7FF6713A3829
.text:00007FF6713A9B34
.text:00007FF6713A9B39 cmp     rax, 25h ; '%'
.text:00007FF6713A9B3D jnz     loc_7FF6713A9BD3
.text:00007FF6713A9B3D
.text:00007FF6713A9B43 mov     [rsp+470h+var_448], 0Ah
.text:00007FF6713A9B4B lea     rax, [rbp+440h+var_1F8]
.text:00007FF6713A9B52 mov     [rsp+470h+var_450], rax
.text:00007FF6713A9B57 mov     r9d, 26h ; '&'
.text:00007FF6713A9B5D lea     r8, byte_7FF6714C6000            ; Keypatch modified this from:
.text:00007FF6713A9B5D                                          ;   lea r8, byte_140197260
.text:00007FF6713A9B5D                                          ; Keypatch modified this from:
.text:00007FF6713A9B5D                                          ;   lea r8, byte_7FF6714C7260
.text:00007FF6713A9B5D                                          ; Keypatch modified this from:
.text:00007FF6713A9B5D                                          ;   lea r8, byte_7FF6714C7260
.text:00007FF6713A9B64 lea     rdx, [rbp+440h+var_310]
.text:00007FF6713A9B6B lea     rcx, [rbp+440h+var_430]
.text:00007FF6713A9B6F call    Rc4
.text:00007FF6713A9B6F
RIP .text:00007FF6713A9B74 mov     [rbp+440h+var_1AC], 0
.text:00007FF6713A9B7E jmp     short loc_7FF6713A9B8E
.text:00007FF6713A9B7E
```

此时的rdx就是flag的指针

```
General registers                                   deREferencing - Registers
 RAX 0000000000000000
 RBX 0000000000000000
*RCX 0000000000000025 /* b'%' */
*RDX 00007FF6714C6000 (.data ! byte_7FF6714C6000) -> ("moectf{y0u_r3a11y_understand_rc4!!!!}")
 RDI 0000000CEFEFF923 -> CCCCCCCCCCCCCCCC
 RSI 00007FF67149214B (.rdata) -> 6C65770000000000 /* b'wel' */
*R8  0000000CEFEFF6E0 -> 1C7372B00F229546
 R9  0000000000000026 /* b'&' */
 R10 00007FFBE6B2D0D4 (ntdll.dll)
 R11 8101010101010100
 R12 0000000000000000
 R13 0000000000000000
 R14 0000000000000000
 R15 0000000000000000
 RBP 0000000CEFEFF6D0 -> CCCCCCCCCCCCCCCC
 RSP 0000000CEFEFF6A0 -> 0000000CEFEFF6E0 -> 1C7372B00F229546
*RIP 00007FF6713A9B74 (.text ! sub_7FF6713A9A70+104) -> mov [rbp+440h+var_1AC], 0
 EFL 0000000000000246
 GLE 0000000000000000 (ERROR_SUCCESS)

STACK | HEAP | CODE | DATA | RODATA | RWX | VALUE

ID
VIP
VIF
AC
VM
RF
NT
IOPL
OF
DF
IF
TF
SF
ZF
AF
PF
CF
```

> moectf{y0u_r3a11y_understand_rc4!!!!}

ezAndroid

Native层逆向 Maze

直接JEB跟到MainActivity

```
    public native int check(String arg1) {
    }

    @Override  // androidx.fragment.app.FragmentActivity
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        ActivityMainBinding activityMainBinding0 = ActivityMainBinding.inflate(this.getLayoutInflater());
        this.binding = activityMainBinding0;
        this.setContentView(activityMainBinding0.getRoot());
        ((Button)this.findViewById(0x7F080062)).setOnClickListener(new View.OnClickListener() {  // id:button
            @Override  // android.view.View$OnClickListener
            public void onClick(View view) {
                String s = ((EditText)this.findViewById(0x7F0800DF)).getText().toString();  // id:input
                if(s.length() != 23) {
                    Toast.makeText(MainActivity.this.getApplicationContext(), "长度不对哦", 0).show();
                    return;
                }

                if(MainActivity.this.check(s) == 1) {
                    Toast.makeText(MainActivity.this.getApplicationContext(), "OK!RIGHT,flag is moectf{" + s + "}", 0).show();
                    return;
                }

                Toast.makeText(MainActivity.this.getApplicationContext(), "Try to reverse the native lib!", 0).show();
            }
        });
    }
}
```

分析主函数，发现关键是check()这个方法

然后check的定义在上面是 public native check()

所以应该是native层逆向

将apk里面的lib目录中，随便找一个so文件（最好用x86架构的）解压下来，扔进IDA，

跟进JNI__load函数

```
IDA View-A   Pseudocode-C   Pseudocode-B   Pseudocode-A   Hex View-1   Enums
int __cdecl JNI_OnLoad(int a1)
{
  int v3; // [esp+28h] [ebp-B0h]
  int v4; // [esp+40h] [ebp-98h] BYREF
  char dest[136]; // [esp+44h] [ebp-94h] BYREF
  unsigned int v6; // [esp+CCh] [ebp-Ch]

  v6 = __readgsdword(0x14u);
  memcpy(
    dest,
    "******************@**************.************...****#..*****.********.*****.****.....*****.****.*********......****"
    "*******************",
    sizeof(dest));
  v3 = __strlen_chk(dest, 136);
  __memcpy_chk(asc_3934, dest, v3, 136);
  v4 = 0;
  if ( sub_1560(a1, &v4, 65540) )
    return -1;
  *(&off_2910 + 1075) = (Elf32_Dyn *)sub_15B0(v4, "com/doctor3/ezandroid/MainActivity");
  if ( !*(&off_2910 + 1075) )
    return -1;
  if ( (int)sub_15F0(v4, dword_39DC, off_2824, 1) >= 0 )
    return 65540;
  return -1;
```

可以看到memcpy了一个字符串

然后sub_1cc0()这个函数

IDA View-A | Pseudocode-C | Pseudocode-B | Pseudo

```
_BOOL4 __cdecl sub_CC0(_BYTE *a1)
{
  _BYTE *v1; // eax
  bool v3; // [esp+Bh] [ebp-Dh]
  char *v4; // [esp+10h] [ebp-8h]
  int v5; // [esp+14h] [ebp-4h]

  v4 = &asc_3934[18];
  while ( 2 )
  {
    v3 = 0;
    if ( *a1 )
      v3 = *v4 != 42;
    if ( !v3 )
      return *v4 == 35;
    v1 = a1++;
    switch ( *v1 )
    {
      case 'a':
        --v4;
        continue;
      case 'd':
        ++v4;
        continue;
      case 's':
        v4 += 15;
        continue;
      case 'w':
        v4 -= 15;
        continue;
      default:
        v5 = 0;
        break;
    }
    break;
  }
  return v5;
}
```

很明显是个每行15个字符的maze，wasd控制

直接提取出刚才的字符串手搓

```
***************
***@***********
***.***********
*...****#..****
*.********.****
*.****.....****
*.****.********
*......********
***************
```

> moectf{ssaassssdddddwwddddwwaa}

SMC

SMC 线性变换

如题，这是一道SMC

分析主函数

```
IDA View-A    Pseudocode-A    Hex View-1
int __cdecl main_0(int argc, const char **argv, const char **envp)
{
  char v4; // [esp+0h] [ebp-78h]
  char v5; // [esp+0h] [ebp-78h]
  char v6[104]; // [esp+Ch] [ebp-6Ch] BYREF

  printf(aPlzInputYourFl, v4);
  scanf(aS, v6);
  sub_4011E0();
  if ( sub_401050(v6) )
    printf(aGood, v5);
  else
    printf(aTryAgainPlease, v5);
  return 0;
}
```

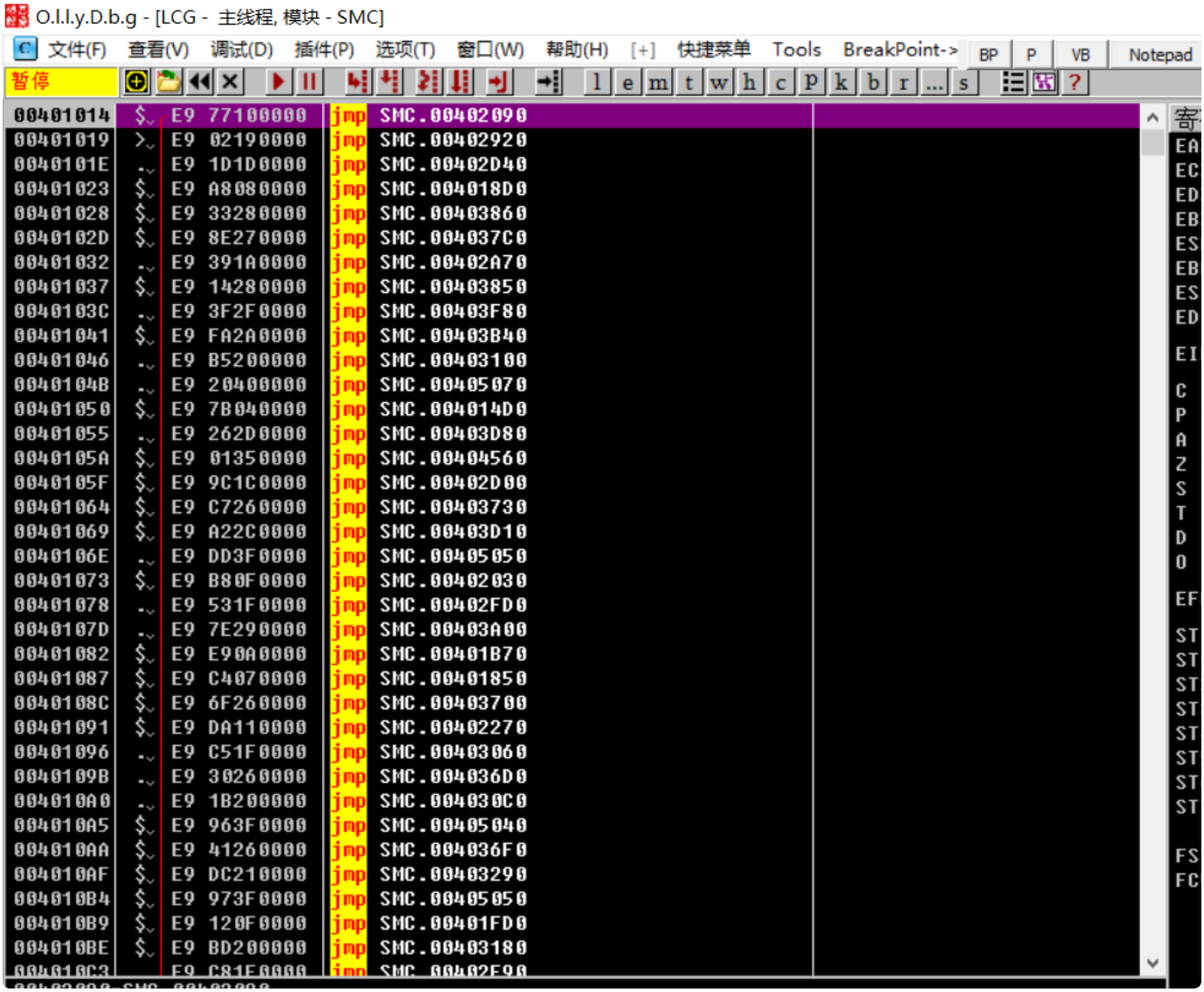
跟进sub_4011E0()

```
IDA View-A    Pseudocode-A    Hex View-1    Enums
int sub_401550()
{
  int result; // eax
  int i; // [esp+4h] [ebp-1Ch]
  DWORD flOldProtect[2]; // [esp+18h] [ebp-8h] BYREF

  flOldProtect[1] = -858993460;
  flOldProtect[0] = malloc(8u);
  result = VirtualProtect(&loc_4014D0 - &loc_4014D0 % 0x1000, 0x1000u, 0x80u, flOldProtect);
  for ( i = 0; i < 122; ++i )
  {
    *(&loc_4014D0 + i) ^= 0x66u;
    result = i + 1;
  }
  return result;
}
```

如图，VirtualProtect是SMC的典型特征，用于修改内存属性，下面是异或0x66的解密

上OD动调



OEP为00401014

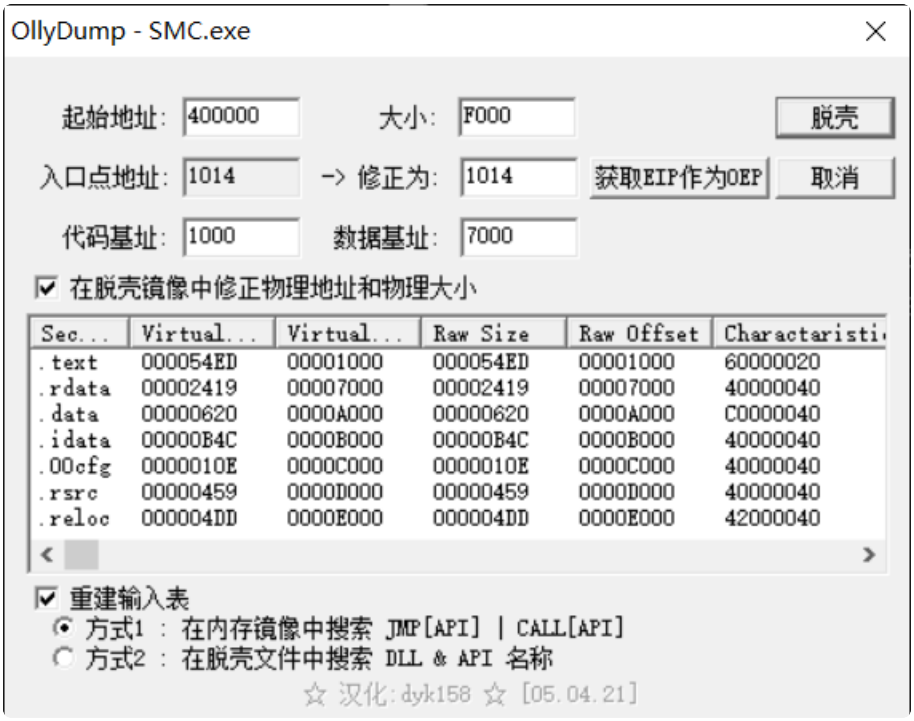
载入之后直接改eip飞到00401550

```
0040154D    CC          int3
0040154E    CC          int3
0040154F    CC          int3
00401550  > 55          push ebp
00401551  . 8BEC        mov ebp,esp
00401553  . 83EC 1C     sub esp,0x1C
00401556  . 56          push esi                                SMC.<ModuleEntryPoint>
00401557  . B8 CCCCCCCC mov eax,0xCCCCCCCC
0040155C  . 8945 E4     mov [local.7],eax
0040155F  . 8945 E8     mov [local.6],eax
00401562  . 8945 EC     mov [local.5],eax
00401565  . 8945 F0     mov [local.4],eax
00401568  . 8945 F4     mov [local.3],eax
0040156B  . 8945 F8     mov [local.2],eax
0040156E  . 8945 FC     mov [local.1],eax
00401571  . 8BF4        mov esi,esp
00401573  . 6A 08       push 0x8                                size = 0x8
00401575  . FF15 50B1400 call dword ptr ds:[<&ucrtbased.malloc>] malloc
0040157B  . 83C4 04     add esp,0x4
0040157E  . 3BF4        cmp esi,esp
00401580  . E8 DEFBFFFF call SMC.00401163
00401585  . 8945 F8     mov [local.2],eax
00401588  . C745 F0 D014 mov [local.4],SMC.004014D0
0040158F  . C745 EC 7A00 mov [local.5],0x7A
00401596  . 8B45 F0     mov eax,[local.4]
00401599  . 33D2        xor edx,edx                             SMC.<ModuleEntryPoint>
0040159B  . B9 00100000 mov ecx,0x1000
004015A0  . F7F1        div ecx                                 SMC.<ModuleEntryPoint>
004015A2  . 8B45 F0     mov eax,[local.4]
004015A5  . 2BC2        sub eax,edx                             SMC.<ModuleEntryPoint>
004015A7  . 8945 E8     mov [local.6],eax
```

然后再ctrl+f9执行到段尾retn

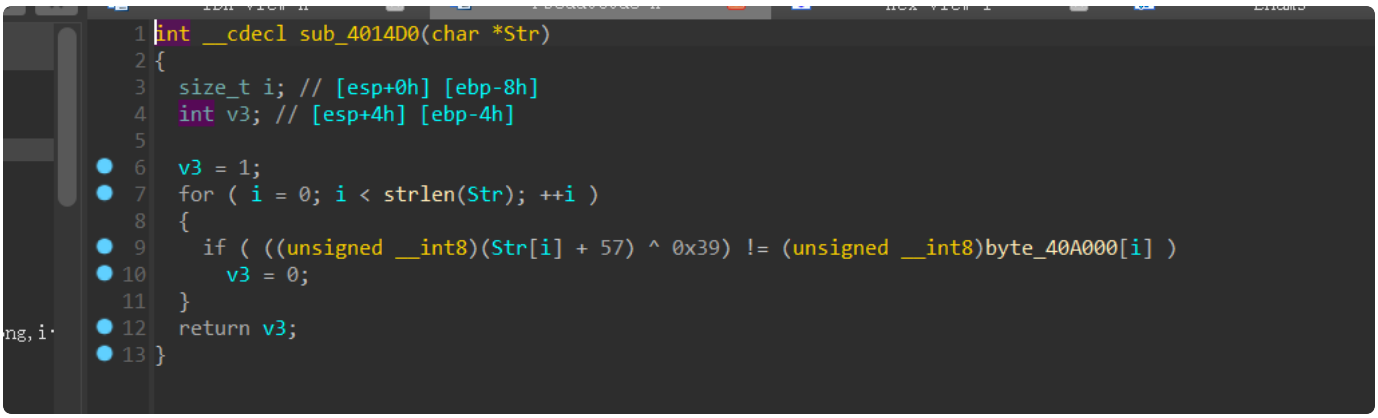
```
004015FB  > 52          push edx                                SMC.00401549
004015FC  . 8BCD        mov ecx,ebp
004015FE  . 50          push eax
004015FF  . 8D15 1C16400 lea edx,dword ptr ds:[0x40161C]
00401605  . E8 18FBFFFF call SMC.00401122
0040160A  . 58          pop eax                                 kernel32.761C00C9
0040160B  . 5A          pop edx                                 kernel32.761C00C9
0040160C  . 5E          pop esi                                 kernel32.761C00C9
0040160D  . 83C4 1C     add esp,0x1C
00401610  . 3BEC        cmp ebp,esp
00401612  . E8 4CFBFFFF call SMC.00401163
00401617  . 8BE5        mov esp,ebp
00401619  . 5D          pop ebp                                 kernel32.761C00C9
0040161A  . C3          retn
0040161B    90          nop
0040161C    01          db 01
0040161D    00          db 00
0040161E    00          db 00
0040161F    00          db 00
00401620  . 24164000    dd SMC.00401624
00401624    F8          db F8
00401625    FF          db FF
00401626    FF          db FF
00401627    FF          db FF
00401628    04          db 04
00401629    00          db 00
```

之后再把eip改回00401014，然后右键用Olldump脱壳



再次分析dump出来的文件

此时再跟进sub_401050()



```
int __cdecl sub_4014D0(char *Str)
{
  size_t i; // [esp+0h] [ebp-8h]
  int v3; // [esp+4h] [ebp-4h]

  v3 = 1;
  for ( i = 0; i < strlen(Str); ++i )
  {
    if ( ((unsigned __int8)(Str[i] + 57) ^ 0x39) != (unsigned __int8)byte_40A000[i] )
      v3 = 0;
  }
  return v3;
}
```

已经解密完毕，此时可以发现是一个线性变换

一把梭

```
print(''.join([chr((get_wide_byte(0x40a000+i)^0x39)-57) for i in range(25)]))
```

> moectf{Self_Mod1f1cation}

RUST

RUST 异或

RUST这个逆起来头大，但是可以重命名函数变量来减轻阅读量

```
__int64 v49; // [rsp+2E0h] [rbp-8h]

NewAlloc(&v28);
NewArgumnets(&v29, __PAIR128__(1LL, &stru_57D80), &stru_45210);
print(v19);
NewArgumnets(&v30, __PAIR128__(1LL, &stru_57D90), &stru_45210);
print(v20);
*buf.vec.buf.alloc.gap0 = std::io::stdio::stdin::h8b590be40a6f0948();
readLine(&self, &buf);
*&v0.gap0[8] = &stru_57DA0;
*v0.gap0 = &self;
core::result::Result$LT$T$C$E$GT$::unwrap::hc3fcaeef2c7da20e(v0);
v1 = _$LT$alloc..string..String$u20$as$u20$core..ops..deref..Deref$GT$::deref::hf8c72e0a8093fc4f(&v28, &stru_57DA0);
v3 = core::str::_$LT$impl$u20$str$GT$::trim_end::he08a22e006303d78(v1, __PAIR128__(v2, v2));
*&v0.gap0[8] = v4;
*v0.gap0 = v3;
v26 = v3;
v27 = v4;
v46 = v3;
v47 = v4;
if ( core::str::_$LT$impl$u20$str$GT$::len::h88055d0d4e46e37e(v0) != 30 )
{
  NewArgumnets(&buf.vec.buf.cap, __PAIR128__(1LL, &stru_57DB8), &stru_45210);
  print(v21);
  std::process::exit::h0481127236e019a8(1);
}
v25 = alloc::alloc::exchange_malloc::hde43adfcaffa380d(0x1EuLL, 1uLL);
*v25 = -27;
v25[1] = -25;
v25[2] = -19;
v25[3] = -21;
v25[4] = -4;
v25[5] = -18;
v25[6] = -13;
v25[7] = -38;
v25[8] = -3;
v25[9] = -5;
v25[10] = -4;
v25[11] = -41;
v25[12] = -6;
v25[13] = -19;
```

以及　HIBYTE(v21.pieces.length) = BitXor(*v21.fmt.gap0, 0x88u);

可以猜测，这个题目是异或

```python
v25=[0]*30
v25[0]= 0xE5;
v25[1] = 0xE7;
v25[2] = 0xED;
v25[3] = 0xEB;
v25[4] = 0xFC;
v25[5] = 0xEE;
v25[6] = 0xF3;
v25[7] = 0xDA;
v25[8] = 0xFD;
v25[9] = 0xFB;
v25[10] = 0xFC;
v25[11] = 0xD7;
v25[12] = 0xFA;
v25[13] = 0xED;
v25[14] = 0xFE;
v25[15] = 0xD7;
v25[16] = 0xFF;
v25[17] = 0xE1;
v25[18] = 0xE4;
v25[19] = 0xE4;
v25[20] = 0xD7;
v25[21] = 0xEA;
v25[22] = 0xED;
v25[23] = 0xD7;
v25[24] = 0xE9;
v25[25] = 0xFF;
v25[26] = 0xEE;
v25[27] = 0xFD;
v25[28] = 0xB9;
v25[29] = 0xF5;
print(''.join([chr(x^0x88) for x in v25]))
```

> moectf{Rust_rev_will_be_awfu1}

GUI

GUI 线性变换

```
符号 Lumina 函数
IDA Vie… | Pseudocod… | Pseudocod… | Pseudocod… | Pseudocod… | Stack of sub_45B…
1  int __stdcall WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nShowCmd)
2  {
3    struct tagMSG Msg; // [esp+D0h] [ebp-54h] BYREF
4    WNDCLASSW WndClass; // [esp+F4h] [ebp-30h] BYREF
5
6    __CheckForDebuggerJustMyCode(&unk_528026);
7    WndClass.style = 0;
8    WndClass.cbClsExtra = 0;
9    WndClass.cbWndExtra = 0;
10   memset(&WndClass.hIcon, 0, 16);
11   WndClass.lpfnWndProc = sub_450CDF;
12   WndClass.hInstance = hInstance;
13   WndClass.lpszClassName = L"FlagCheckerWindowClass";
14   RegisterClassW(&WndClass);
15   hWnd = CreateWindowExW(
16            0,
17            L"FlagCheckerWindowClass",
18            L"Windows Flag Checker",
19            0xCF0000u,
20            100,
21            100,
22            300,
23            200,
24            0,
25            0,
26            hInstance,
27            0);
28   ShowWindow(hWnd, nShowCmd);
29   UpdateWindow(hWnd);
     while ( GetMessageW(&Msg, 0, 0, 0) )
0000E230 _WinMain@16_0:3 (45DE30)
```

分析这个窗口绑定的函数

```
外部符号 Lumina 函数
IDA Vie… | Pseudocod… | Pseudocod… | Pseudocod… | Pseudocod… | Stack of sub_45B…
19      }
20      else
21      {
22        if ( Msg != 273 )
23          return DefWindowProcW(hWndParent, Msg, wParam, lParam);
24        if ( (unsigned __int16)wParam == 1 )
25        {
26          DlgItem = GetDlgItem(hWndParent, 2);
27          GetWindowTextW(DlgItem, String, 1024);
28          sub_450C94(String);
29          v13 = 0;
30          sub_450A0A(v7, v8);
31          LOBYTE(v13) = 1;
32          sub_450C94(a91);
33          if ( (unsigned __int8)sub_4531AB(v7, v6) )
34            MessageBoxW(hWndParent, Text, L"hint", 0);
35          else
36            MessageBoxW(hWndParent, L"Sorry, flag error.", L"hint", 0);
37          sub_4529B8(v6);
38          LOBYTE(v13) = 0;
39          sub_4529B8(v7);
40          v13 = -1;
41          sub_4529B8(v8);
42        }
43      }
44    }
45    else
46    {
47      switch ( Msg )
        {
0000C6A1 sub_45BF90:19 (45C2A1)
```

首先是GetWindowTextW这个api可以获取输入在编辑框的内容

再猜测Line33是一个判断，跟进 sub_450A0A

是一个线性变换

```
2 {
3   int v3; // [esp+F8h] [ebp-54h]
4   _WORD *v4; // [esp+104h] [ebp-48h]
5   char v5[32]; // [esp+11Ch] [ebp-30h] BYREF
6   int v6; // [esp+148h] [ebp-4h]
7
8   __CheckForDebuggerJustMyCode(&unk_528026);
9   sub_4519E6(v5);
10  v6 = 0;
11  v4 = (_WORD *)sub_450956(a2);
12  v3 = sub_45017C(a2);
13  while ( v4 != (_WORD *)v3 )
14    sub_4516B7((*v4++ - 5) ^ 0x51);
15  sub_4510C7(v5);
16  v6 = -1;
17  sub_4529B8(v5);
18  return a1;
19 }
```

观察到这里有WORD*和上面的GetWindowTextW，可以猜测是Unicode存储的数据

上面的Line32是取密文，密文来自于0x4fe210

所以直接一把梭

```
print(''.join([chr((get_wide_word(0x4fe210+i*2)^0x51)+5) for i in range(30)]))
```

Junk_code

jmp db型花指令 线性变换

```
IDA View-A    Pseudocode-B    Pseudocode-A
int __cdecl main_0(int argc, const char **argv, const char **envp)
{
  int v4; // eax
  char Str[18]; // [esp+E8h] [ebp-30h] BYREF
  _BYTE v6[26]; // [esp+FAh] [ebp-1Eh] BYREF

  __CheckForDebuggerJustMyCode(&unk_543007);
  j__puts("welcome to moectf\nyour flag:");
  v6[18] = 0;
  sub_4591AE("%36s", Str);
  if ( j__strlen(Str) >> 1 == 18 )
  {
    sub_45A9A0(Str, 18);
    if ( v4 && sub_459EBF(v6, 18) )
      j__puts("congratulations!!!");
    else
      j__puts("WORNG!");
    return 0;
  }
  else
  {
    j__puts("WORNG!");
    return 0;
  }
}
```

分析主函数可知，flag为18位，

sub_45A9A0(Str, 18)和sub_459EBF(v6, 18)里面分析不出来，考虑是有花指令。

第一个花指令，典型的jmp db型，直接全部Nop掉

```
.text:0046060E 85 C0                    test    eax, eax
.text:00460610 74 01                    jz      short loc_460613
.text:00460610
.text:00460610                          ; ---------------------------
.text:00460612 E8                       db 0E8h
```

第二个花指令，双jmp db型，直接nop掉

```
.text:0046076B 74 03                    jz      short loc_460770
.text:0046076B
.text:0046076D 75 01                    jnz     short loc_460770
.text:0046076D
.text:0046076D                          ; ---------------------------
---------------------------------------------
.text:0046076F E8                       db 0E8h
.text:00460770                          ; ---------------------------
---------------------------------------------
```

再分析这两个函数

一个是，对输入的前18位，进行了一个-5再比较的操作

```
for ( j = 0; j < len; ++j )
    *(v7 + j) -= 5;
  for ( k = 0; k < len; ++k )
  {
    if ( aHjOavtPzmHQ[k] != *(v7 + k) )
      return 0;
  }
```

这一块是对后面的18位进行了一个异或操作

```
IDA View-A
BOOL __cdecl sub_460750(char *Str2, signed int MaxCount)
{
  signed int v3; // [esp+D4h] [ebp-8h]

  while ( v3 < MaxCount )
    Str2[v3++] ^= 0x66u;
  return j__strncmp(Str1, Str2, MaxCount) == 0;
}
```

然后IDApython一把梭

```
print(''.join([chr(get_wide_byte(0x53F000+i)+5) for i in range(18) ]+[chr(g
et_wide_byte(0x516E50+i)^0x66) for i in range(18)]))
```

> moectf{y0u_rem0v3d_th3_junk_c0d3!!!}

unwind

SEH异常处理 TEA

此程序为标准的TEA

程序的控制流是这样的

首先在0x41590f有一个非法的内存访问

```
.text:0041590F C7 05 00 00 00 00 00 00 00 00 mov     large dword ptr ds:0, 0
```

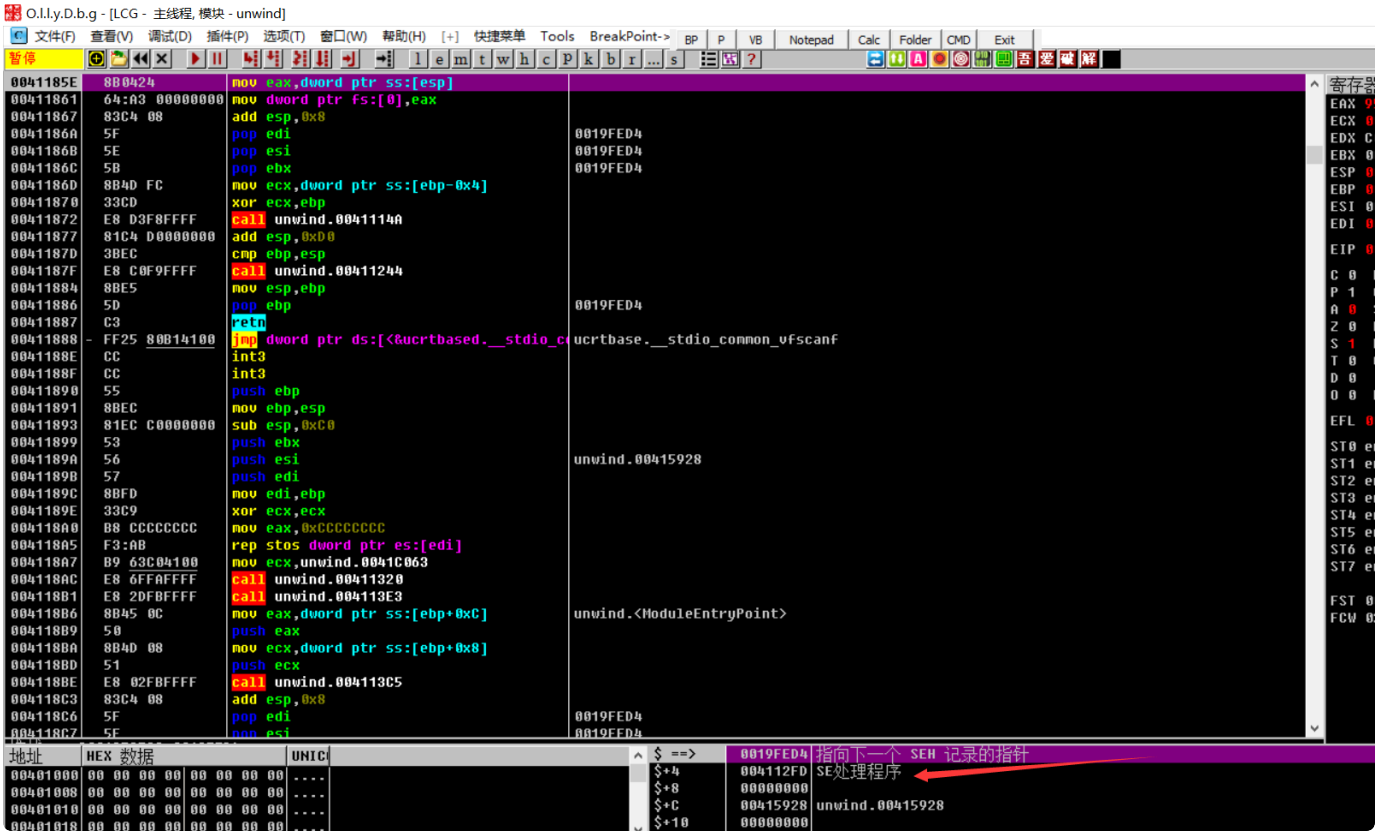
然后异常处理是对前面的4组进行一个TEA加密

然后是跟进去这个call

```
.text:0041597A                          ;   __try { // __except at loc_415995
.text:0041597A C7 45 FC 01 00 00 00     mov     [ebp+ms_exc.registration.TryLevel], 1
.text:00415981 E8 79 B7 FF FF           call    sub_4110FF
.text:00415981                          ;   } // starts at 41597A
```

```
.text:00411856 64 89 25 00 00 00 00     mov     large fs:0, esp
.text:0041185D CC                       int     3
                                        ; Trap to Debugger
.text:0041185E 8B 04 24                 mov     eax, [esp+0E4h+var_E4]
```

可以发现有个int3中断，所以这边是第二个异常处理的地方



OD跟到这边，可以发现这个SEH链的异常处理程序

IDA里面跟一下这个地址

来到了sub_411B50()这个函数

接着往下跟会发现这个函数被执行了两遍，这个call ecx

```
77D28A6A   8BFF            mov edi,edi
77D28A6C   55              push ebp
77D28A6D   8BEC            mov ebp,esp
77D28A6F   FF75 0C         push dword ptr ss:[ebp+0xC]
77D28A72   52              push edx                                  ntdll.77D28AB0
77D28A73   64:FF35 000000  push dword ptr fs:[0]
77D28A7A   64:8925 000000  mov dword ptr fs:[0],esp
77D28A81   FF75 14         push dword ptr ss:[ebp+0x14]
77D28A84   FF75 10         push dword ptr ss:[ebp+0x10]
77D28A87   FF75 0C         push dword ptr ss:[ebp+0xC]
77D28A8A   FF75 08         push dword ptr ss:[ebp+0x8]
77D28A8D   8B4D 18         mov ecx,dword ptr ss:[ebp+0x18]           unwind.004112FD
77D28A90   FFD1            call ecx                                  unwind.004112FD
77D28A92   64:8B25 000000  mov esp,dword ptr fs:[0]
77D28A99   64:8F05 000000  pop dword ptr fs:[0]                      ntdll.77D28A92
77D28AA0   8BE5            mov esp,ebp
77D28AA2   5D              pop ebp                                   ntdll.77D28A92
77D28AA3   C2 1400         retn 0x14
```

汇总起来

```
TEA(&byte_41A578, aDx3906);
TEA(&unk_41A580, aDoctor3);
TEA(&unk_41A588, aFux1aoyun);
TEA(&unk_41A590, aR3verier);
for i in range(2):
    TEA(&unk_41A598, aDx3906);
    TEA(&unk_41A5A0, aDoctor3);
    TEA(&unk_41A5A8, aFux1aoyun);
    TEA(&unk_41A5B0, aR3verier);

```

exp:

```python
from ctypes import *
from libnum import *
def tea_decrypt(v,k):
    v0=c_uint32(v[0])
    v1=c_uint32(v[1])
    delta=-0x61C88647
    sum1=c_int32(delta*32)
    for i in range(32):
        v1.value-=((v0.value<<4)+k[2])^(v0.value+sum1.value)^((v0.value>>5)+k[3])
        v0.value-=((v1.value<<4)+k[0])^(v1.value+sum1.value)^((v1.value>>5)+k[1])
        sum1.value-=delta
    return v0.value,v1.value
v=[get_wide_dword(0x41A000+i*4) for i in range(16)]#密文的地址
k=[get_wide_dword(0x41A044+i*4) for i in range(16)]#密钥的地址
flag=''
get=lambda s:n2s(s[0]).decode()[::-1]+n2s(s[1]).decode()[::-1]
for i in range(4):
    s=tea_decrypt(v[i*2:i*2+2],k[i*4:i*4+4])
    flag+=get(s)
for i in range(4):
    s=tea_decrypt(v[8+i*2:8+i*2+2],k[i*4:i*4+4])
    s=tea_decrypt(s,k[i*4:i*4+4])
    flag+=get(s)
print(flag)
```

> moectf{WoOo00Oow_S0_interesting_y0U_C4n_C41l_M3tW1c3_BY_Unw1Nd~}