DEFCON Prequal 2019

Vitor

#Android #DEX #nativelib #Javascript

THEORI

STEP 1: p1EncFn

- `ooo.vitor.MainActivity`
  - goal : `fc.mValid == true`

ooo.vitor.MainActivity

```
88              }
89          });
90      }
91
92      public void updateFlagWidget() {
93          runOnUiThread(new Runnable() {
94              public void run() {
95                  String str;
96                  int i;
97                  String str2 = "OOO";
98                  if (fc.mValid) {
99                      Log.e(str2, "Flag is valid!");
100                     i = -16737536;
101                     str = "Valid flag!";
102                 } else {
103                     Log.e(str2, "Flag is not valid dude");
104                     i = SupportMenu.CATEGORY_MASK;
105                     str = "Invalid flag";
106                 }
107                 MainActivity.this.mResultWidget.setText(str);
108                 MainActivity.this.mResultWidget.setTextColor(i);
109             }
110         });
111     }
```

STEP 1: p1EncFn

- len(OOO{SOMETHING}) == 45
- g0(<…SOMETHING…>)
- dp1()
- cf()

```
public class fc {
    private static final byte[] initVector = {19, 55, 19, 55, 19, 55, 19, 55, 19, 55, 19, 55, 19, 55, 19, 55};
    public static boolean mValid = false;
    public static String p1EncFn = "ckxalskuaewlkszdva";
    public static String p1Fn = "nsavlkureaasdqwecz";
    public static String p5EncFn = "cxnvhaekljlkjxxqkq";
    public static String rand2EncFn = "fwswzofqwkzhsgdxfr";
    public static String randEncFn = "zslzrfomygfttivyac";

    public static boolean cf(MainActivity mainActivity, String str) {
        try {
            cfa(mainActivity, p1EncFn);
            cfa(mainActivity, p5EncFn);
            cfa(mainActivity, randEncFn);
            cfa(mainActivity, rand2EncFn);
            if (str.startsWith("OOO{") && str.endsWith("}")) {
                if (str.length() == 45) {
                    if (!cf(mainActivity, dp1(mainActivity, new File(mainActivity.getFilesDir(), p1EncFn), g0(str.substring(4, 44))),
                        return false;
                    }
                    File file = new File(mainActivity.getFilesDir(), "bam.html");
                    WebView webView = mainActivity.mWebView;
                    StringBuilder sb = new StringBuilder();
                    sb.append("file:///");
                    sb.append(file.getAbsolutePath());
                    sb.append("?flag=");
                    sb.append(Uri.encode(str));
                    webView.loadUrl(sb.toString());
                    return mValid;
```

STEP 1: p1EncFn

- len(OOO{SOMETHING}) == 45
- g0(<…SOMETHING…>)
  - 키를 4byte씩 잘라서 XOR
  - bArr
- dp1()
- cf()

```java
public static byte[] g0(String str) {
    byte[] bArr = new byte[4];
    byte[] bytes = str.getBytes();
    for (int i = 0; i < 4; i++) {
        bArr[i] = 0;
    }
    for (int i2 = 0; i2 < 10; i2++) {
        for (int i3 = 0; i3 < 4; i3++) {
            bArr[i3] = (byte) (bArr[i3] ^ bytes[(i2 * 4) + i3]);
        }
    }
    return bArr;
}
```

STEP 1: p1EncFn

- len(OOO{SOMETHING}) == 45
- g0(<…SOMETHING…>)
  - bArr
- dp1()
  - MD5(bArr)해서 key로 사용
  - p1EncFn을 AES decrypt
  - p1Fn으로 리턴
- cf()

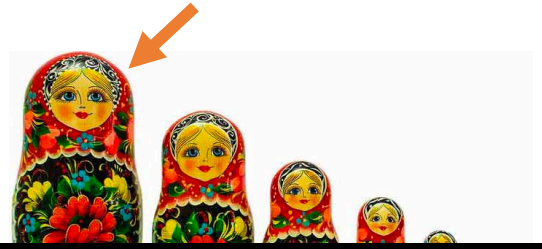
```
private static File dp1(Context context, File file, byte[] bArr) throws Exception {
    byte[] hash = hash(bArr);
    byte[] readAllBytes = Files.readAllBytes(file.toPath());
    try {
        IvParameterSpec ivParameterSpec = new IvParameterSpec(initVector);
        SecretKeySpec secretKeySpec = new SecretKeySpec(hash, "AES");
        Cipher instance = Cipher.getInstance("AES/CBC/PKCS5PADDING");
        instance.init(2, secretKeySpec, ivParameterSpec);
        byte[] doFinal = instance.doFinal(readAllBytes);
        File file2 = new File(context.getFilesDir(), p1Fn);
        FileOutputStream fileOutputStream = new FileOutputStream(file2);
        fileOutputStream.write(doFinal, 0, doFinal.length);
        fileOutputStream.flush();
        fileOutputStream.close();
        return file2;
    } catch (Exception unused) {
        return null;
    }
}
```

STEP 1: p1EncFn

- len(OOO{SOMETHING}) == 45
- g0(<…SOMETHING…>)
- dp1()
- cf()
  - dp1()에서 복호화한 클래스 로드

```
private static boolean cf(Context context, File file, String str) {
    File file2 = new File(context.getFilesDir().getAbsolutePath());
    DexClassLoader dexClassLoader = new DexClassLoader(file.getAbsolutePath(), file2.getAbsolutePath(),
    boolean z = false;
    try {
        Class loadClass = dexClassLoader.loadClass("ooo.p1.P1");
        z = ((Boolean) loadClass.getDeclaredMethod("cf", new Class[]{Context.class, String.class}).invo
        return z;
    } catch (Exception unused) {
        return z;
    }
```

STEP 1: p1EncFn



- Goal
  - p1EncFn을 decrypt하는 키의 일부 찾기
    - `public static String p1EncFn = "ckxalskuaewlkszdva";`
  - PKZIP으로 시작할 것이라는 추측 가능
    - 해당 magic number로 시작하도록 뽑아주는 키만 수집
    - 그 중 제대로 복호화해주는 것 골라내기

```
(14:37)[~/ctf/defcon_prequal2019/vitor/work]$ python3 step1key.py
b'\x17\x01/\x03'
```

- '17 01 2F 03'

```python
from Crypto.Cipher import AES
from itertools import product
from multiprocessing import Pool
import hashlib

num_threads = 8
with open("../vitor_unzip/vitor/assets/ckxalskuaewlkszdva", "rb") as f:
    file_content = f.read()[:16]

magic = b"PK\x03\x04" #PKZIP magic
cnt = 0
iv = [19, 55, 19, 55, 19, 55, 19, 55]
magic_iv_xor = bytes([magic[i] ^ iv[i] for i in range(4)])

def f(num):
    key_list = []
    for c in product(range(128), repeat=3):
        r = 128 // num_threads
        for i in range(r * num, r * (num + 1)):
            key_str = bytes(c) + bytes([i])
            key = hashlib.md5(key_str).digest()
            aes = AES.new(key, AES.MODE_ECB)
            if aes.decrypt(file_content)[:4] == magic_iv_xor[:4]:
                print(key_str)
                key_list.append(key_str)
    return key_list

with Pool(processes=8) as p:
    key_list = p.map(f, range(8))
```

STEP 2: 또;;

- ooo.p1.P1
  - goal : cf의 return == true
    - g1()
  - 앗 데자뷰인가

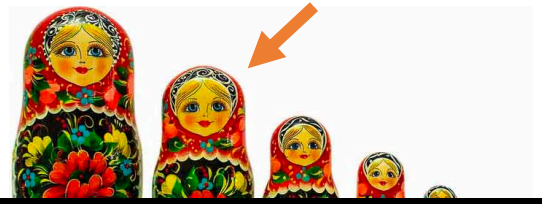
```java
public class P1 {
    public static String abc = "smnvlwkuelqkjsmxzz";
    public static String def = "mmdffuoscjdamcnssn";
    public static String ghi = "xtszswemcwohpluqmi";

    /* renamed from: ooo reason: collision with root package name */
    private static final byte[] f0ooo = {19, 55, 19, 55, 19, 55, 19, 55, 19, 55, 19, 55, 19, 55, 19, 55};

    private native String xxx(String str, String str2);

    public static boolean cf(Context ctx, String f) {
        try {
            byte[] K1 = g1(f.substring(4, 44));
            cfa(ctx, def);
            cfa(ctx, ghi);
            dp2(ctx, new File(ctx.getFilesDir(), def), K1);
            System.loadLibrary(abc);
            String tre = new P1().xxx(f, ctx.getFilesDir().getAbsolutePath());
            if (tre != null && new File(tre).isFile()) {
                return true;
            }
```

STEP 2: 또;;



- Goal
  - def decrypt하는 키의 일부 찾기
    - `public static String def = "mmdffuoscjdamcnssn";`
  - \x7fELF으로 시작할 것이라는 추측 가능
    - 해당 magic number로 시작하도록 뽑아주는 키만 수집
    - 그 중 제대로 복호화해주는 것 골라내기

```
(15:21)[~/ctf/defcon_prequal2019/vitor/work]$ python3 step2key.py
b"<'C`"
```

- '3C 27 43 60'

```python
from Crypto.Cipher import AES
from itertools import product
from multiprocessing import Pool
import hashlib

num_threads = 8
with open("../vitor_unzip/vitor/assets/mmdffuoscjdamcnssn", "rb") as f:
    file_content = f.read()[:16]

magic = b"\x7fELF" #ELF
cnt = 0
iv = [19, 55, 19, 55, 19, 55, 19, 55, 19, 55, 19, 55, 19, 55, 19, 55]
magic_iv_xor = bytes([magic[i] ^ iv[i] for i in range(4)])

def f(num):
    key_list = []
    for c in product(range(128), repeat=3):
        r = 128 // num_threads
        for i in range(r * num, r * (num + 1)):
            key_str = bytes(c) + bytes([i])
            key = hashlib.md5(key_str).digest()
            aes = AES.new(key, AES.MODE_ECB)
            if aes.decrypt(file_content)[:4] == magic_iv_xor[:4]:
                print(key_str)
                key_list.append(key_str)
    return key_list

with Pool(processes=8) as p:
    key_list = p.map(f, range(8))
```

STEP 3: libabc.so

- JNICALL Java_ooo_p1_P1_xxx()
  - g2()
  - dp3()
  - 쉘코드?!

```
xtszswem_enc = (unsigned __int8 *)mmap(0, (size_t)&stru_3E0.st_size, 7, 34, -1, 0);
if ( xtszswem_enc == (unsigned __int8 *)-1 )
  return 0;
v4 = strlen(path);
sfe = (char *)malloc(v4 + 19);
strcpy(sfe, path);
strcat(sfe, "/");
strcat(sfe, "xtszswemcwohpluqmi");
xtszswem = fopen(sfe, "rb");
if ( !xtszswem )
  return 0;
fread(xtszswem_enc, 0x190u, 1u, xtszswem);
fclose(xtszswem);
unlink(sfe);
K2_0 = g2((unsigned __int8 *)str);
dp3(xtszswem_enc, K2_0);
v5 = strlen(path);
qwe = (char *)malloc(v5 + 19);
strcpy(qwe, path);
strcat(qwe, "/");
strcat(qwe, "cxnvhaekljlkjxxqkq");
fffe = fopen(qwe, "rb");
if ( !fffe )
  return 0;
fseek(fffe, 0, 2);
req = ftell(fffe);
fseek(fffe, 0, 0);
mmm = (unsigned __int8 *)malloc(req);
fread(mmm, req, 1u, fffe);
fclose(fffe);
unlink(qwe);
__android_log_print(3, 168155, 168159);
if ( ((int (__cdecl *)(char *, unsigned __int8 *, size_t))xtszswem_enc)(str, mmm, req) !=
  return 0;
```

STEP 3: libabc.so

- JNICALL Java_ooo_p1_P1_xxx()
  - g2()
    - 키를 4바이트씩 잘라서 XOR
    - bArr
  - dp3()
    - 파일 xtszs… 를 4바이트씩 XOR
    - 결과는 쉘코드일것
      - 함수 프롤로그일까???

```
void __cdecl dp3(unsigned __int8 *p3enc, unsigned int bArr)
{
  int i; // [esp+8h] [ebp-4h]

  for ( i = 0; i < 100; ++i )
  {
    *(_DWORD *)&p3enc[4 * i] ^= bArr;
    bArr += 0x31333337;
  }
}
```

  - 쉘코드?!

STEP 3: libabc.so

- JNICALL Java_ooo_p1_P1_xxx()
  - g2()
    - 키를 4바이트씩 잘라서 XOR
    - bArr
  - dp3()
    - 파일 xtszs… 를 4바이트씩 XOR
    - 결과는 쉘코드일것
      - ~~함수 프롤로그일까???~~ 땡
      - NOPsled가 있었네용
  - 쉘코드?!

```
void __cdecl dp3(unsigned __int8 *p3enc, unsigned int bArr)
{
  int i; // [esp+8h] [ebp-4h]

  for ( i = 0; i < 100; ++i )
  {
    *(_DWORD *)&p3enc[4 * i] ^= bArr;
    bArr += 0x31333337;
  }
}
```

```
__android_log_print(3, 0x290DB, 0x290DF);
.rodata:000290DF aJumpingToNopsl db 'Jumping to nopsled in 3, 2, 1, ...',0
```

STEP 3: libabc.so

- Goal
  - 쉘코드를 만들어주는 키의 일부 찾기
  - NOPsled로 시작할 것이라는 추측 가능
    - \x90\x90\x90\x90으로 시작하도록
      - ‘6E 63 27 4E’

```python
from itertools import product
from multiprocessing import Pool
import struct
def p32(x):
    return struct.pack("<L", x)

num_threads = 8
with open("../vitor_unzip/vitor/assets/xtszswemcwohpluqmi", "rb") as f:
    file_content = f.read()

#magic = b"\xb8\x37\x13\x03\x00"
#magic = b"\xff\xf5"
magic = b"\x90\x90\x90\x90" #NOPsled

ttlist = []
for j in range(100):
    ttlist.append(struct.unpack("<L", file_content[4*j : 4*j+4])[0])

def decrypt(ct, key):
    pt = b""
    for i in range(len(ct) // 4):
        pt += p32(ttlist[i] ^ key)
        key += 0x31333337
        key %= 0x100000000
    return pt

def f(num):
    key_list = []
    for i, j, k in product(range(128), repeat=3):
        r = 128 // num_threads
        for l in range(r * num, r * (num + 1)):
            key = (l << 24) ^ (k << 16) ^ (j << 8) ^ i
            if magic in decrypt(file_content, key):
                print(key)
                key_list.append(key)
    return key_list
```

Step 4: Shellcode

- 아직 끝나지 않았다
  - v5 = 2
  - v4 = 인자로 받아온 flag index
    - `flag[0x10:0x14]^flag[0x20:0x24]^flag[0x30:0x34]`
  - 쉘코드의 뒤쪽 (offset 0xC8부터) decrypt 한번 더
    - offset 0xC8부터 XOR
  - `flag[0x10:0x14] ^ flag[0x20:0x24] ^ flag[0x30:0x34] = 42 18 33 13`

```
// xtszswem_enc(flag, mmm, size)
int __cdecl sub_20(char *flag, char *mmm, int size)
{
  int v3; // ecx
  char *v4; // edi
  int v5; // edx
  __int16 *v6; // eax
  int v7; // edx
  __int16 *v8; // eax
  int v9; // edx

  v3 = 0;
  v4 = flag + 0x10;
  v5 = 2;
  do
  {
    v3 ^= *(_DWORD *)v4;
    v4 += 0x10;
    --v5;
  }                                     // fla
  while ( v5 );
  v6 = decryptme;
  v7 = 0x32;
  do
  {
    *(_DWORD *)v6 ^= v3;
    v6 += 2;
    --v7;                               // XOR
  }
  while ( v7 );
  v8 = &decryptme[50];
  v9 = 0x19;
  do
  {
    *(_DWORD *)v8 = *(_DWORD *)v8;
    v8 += 2;
    --v9;
  }
  while ( v9 );
  return (int)&decryptme[50];
}
```

Step 4: Shellcode

- ROP Chain

```
 c8:   fa                      cli
 c9:   18 33                   sbb    BYTE PTR [ebx],dh
 cb:   13 42 db                adc    eax,DWORD PTR [edx-0x25]
 ce:   b0 d3                   mov    al,0xd3
 d0:   46                      inc    esi
 d1:   db f0                   fcomi  st,st(0)
 d3:   9a b2 db b8 15 81 29    call   0x2981:0x15b8dbb2
 da:   3d d0 c1 de 37          cmp    eax,0x37dec1d0
 df:   d0 c1                   rol    cl,1
 e1:   f3 37                   repz aaa
 e3:   d0 3c 19                sar    BYTE PTR [ecx+ebx*1],1
 e6:   f0 4b                   lock dec ebx
 e8:   81 9b df 0b 81 a0 3a    sbb    DWORD PTR [ebx-0x5f7ef421],0x1842133a
 ef:   13 42 18
 f2:   f0 ab                   lock stos DWORD PTR es:[edi],eax
 f4:   75 0b                   jne    0x101
 f6:   30 13                   xor    BYTE PTR [ebx],dl
 f8:   81 91 eb d0 cb d3 f0    adc    DWORD PTR [ecx-0x2c342f15],0xdb8a9af0
 ff:   9a 8a db
102:   f2 d2 4a db             repnz ror BYTE PTR [edx-0x25],cl
106:   02 da                   add    bl,dl
108:   81 2b 7c 07 81 2b       sub    DWORD PTR [ebx],0x2b81077c
10e:   7c 3b                   jl     0x14b
110:   81 88 a3 83 d2 88 a3    or     DWORD PTR [eax-0x772d7c5d],0x88d283a3
```

```
 c8:   b8 00 00 00 00          mov    eax,0x0
 cd:   c3                      ret
 ce:   83 c0 04                add    eax,0x4
 d1:   c3                      ret
 d2:   c3                      ret
 d3:   89 f0                   mov    eax,esi
 d5:   c3                      ret
 d6:   8b 06                   mov    eax,DWORD PTR [esi]
 d8:   c3                      ret
 d9:   31 0e                   xor    DWORD PTR [esi],ecx
 db:   c3                      ret
 dc:   83 c6 04                add    esi,0x4
 df:   c3                      ret
 e0:   83 eb 04                sub    ebx,0x4
 e3:   c3                      ret
 e4:   7e 01                   jle    0xe7
 e6:   c3                      ret
 e7:   58                      pop    eax
 e8:   c3                      ret
 e9:   83 ec 18                sub    esp,0x18
 ec:   c3                      ret
 ed:   b8 09 00 00 00          mov    eax,0x9
 f2:   c3                      ret
 f3:   b8 37 13 03 00          mov    eax,0x31337
 f8:   c3                      ret
 f9:   89 d8                   mov    eax,ebx
```

Step 4: Shellcode

- ROP Chain
  - flag[0x14:0x18] ^ flag[0x28:0x2c]
    - 얘가 rol 하면서 4byte씩 xor함
- 뭘 만들어 주려고?

```
xor ecx, ecx
xor ecx, DWORD PTR [edi+0x14]
xor ecx, DWORD PTR [edi+0x28]
xor DWORD PTR [esi], ecx
add esi,0x4
rol ecx,0x8
sub ebx,0x4
          … <loop> …
mov eax,0x31337
ret
```

Step 4: Shellcode

- ROP Chain
  - flag[0x14:0x18] ^ flag[0x28:0x2c]
    - 얘가 rol 하면서 4byte씩 xor함
- 뭘 만들어 주려고? HTML
  - 이 밑에 `memcmp(mmm, "<html>", 6u)`
- cxnvh… 는 <html>로 시작
  - 56 2D 52 5D

```
xtszswem_enc = (unsigned __int8 *)mmap(0, (size_t)&stru_3E0.st_size, 7, 34, -1, 0);
if ( xtszswem_enc == (unsigned __int8 *)-1 )
  return 0;
v4 = strlen(path);
sfe = (char *)malloc(v4 + 19);
strcpy(sfe, path);
strcat(sfe, "/");
strcat(sfe, "xtszswemcwohpluqmi");
xtszswem = fopen(sfe, "rb");
if ( !xtszswem )
  return 0;
fread(xtszswem_enc, 0x190u, 1u, xtszswem);
fclose(xtszswem);
unlink(sfe);
K2_0 = g2((unsigned __int8 *)str);
dp3(xtszswem_enc, K2_0);
v5 = strlen(path);
qwe = (char *)malloc(v5 + 19);
strcpy(qwe, path);
strcat(qwe, "/");
strcat(qwe, "cxnvhaekljlkjxxqkq");
fffe = fopen(qwe, "rb");
if ( !fffe )
  return 0;
fseek(fffe, 0, 2);
req = ftell(fffe);
fseek(fffe, 0, 0);
mmm = (unsigned __int8 *)malloc(req);
fread(mmm, req, 1u, fffe);
fclose(fffe);
unlink(qwe);
__android_log_print(3, 168155, 168159);
if ( ((int (__cdecl *)(char *, unsigned __int8 *, size_t))xtszswem_enc)(str, mmm, req) !=
  return 0;
```

Step 5: Web ;-(

- ???

```
<html>
<head>
<script type='text/javascript'>

function findGetParameter(parameterName) {
    var result = null,
        tmp = [];
    location.search
        .substr(1)
        .split("&")
        .forEach(function (item) {
          tmp = item.split("=");
          if (tmp[0] === parameterName) result = decodeURIComponent(tmp[1]);
        });
    return result;
}

var _0x2d96=['c2V0RmxhZ0FzVmFsaWQ=','ZmxhZw==','c3Vic3Ry','ID0+IHBIZF8xd19lNHJMMTNyOyl9','SlNJbnRlcmZhY2U='];(function(_0xff65eb,_0x3
{while(--_0x4ed2aa){_0xff65eb['push'](_0xff65eb['shift']());}};_0x36edd3(++_0x34f845);}(_0x2d96,0x18c));var _0x5983=function(_0x4f716
_0x467db9=_0x2d96[_0x4f7167];if(_0x5983['AteqUi']===undefined){(function(){var _0x1e3a59=function(){var
_0xb32e9b;try{_0xb32e9b=Function('return\x20(function()\x20'+'{}.constructor(\x22return\x20this\x22)(\x20)'+');')();}catch(_0x28a8a9)
_0x79e00a=_0x1e3a59();var _0x1081d6='ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=';_0x79e00a['atob']||(_0x79e00a
_0x2e5671=String(_0x5c0d69)['replace'](/=+$/,'');for(var _0x5b28bf=0x0,_0x540b09,_0x32967a,_0x5bb6b3=0x0,_0x1b1ba3='';_0x32967a=_0x2e
(_0x540b09=_0x5b28bf%0x4?_0x540b09*0x40+_0x32967a:_0x32967a,_0x5b28bf++%0x4)?_0x1b1ba3+=String['fromCharCode'](0xff&_0x540b09>>(-0x2*
{_0x32967a=_0x1081d6['indexOf'](_0x32967a);}return _0x1b1ba3;});}());_0x5983['AAZHzw']=function(_0x5f0357){var _0x404b9e=atob(_0x5f03
_0x488e18=0x0,_0x25edcb=_0x404b9e['length'];_0x488e18<_0x25edcb;_0x488e18++){_0x4001de+='%'+('00'+_0x404b9e['charCodeAt'](_0x488e18)[
decodeURIComponent(_0x4001de);};_0x5983['ttOyRt']={};_0x5983['AteqUi']=!![];}var _0x9b6309=_0x5983['ttOyRt'][_0x4f7167];if(_0x9b6309=
(_0x467db9);_0x5983['ttOyRt'][_0x4f7167]=_0x467db9;}else{_0x467db9=_0x9b6309;}return _0x467db9;};theflag=findGetParameter(_0x5983('0x
(0x18)==_0x5983('0x2')){window[_0x5983('0x3')][_0x5983('0x4')]();}

</script>
</head>

<body>

Argh, forgot to add a "web" tag to this chall? ;-)
  - @reyammer

</body>
</html
```

Step 5: Web ;-(

- ???
- 합리적 의심 타임
  - `var _0x2d96 = ['c2V0RmxhZ0FzVmFsaWQ=','ZmxhZw==','c3Vic3Ry','ID0+IHBIZF8xd19lNHJMMTNyOyl9','SlNJbnRlcmZhY2U='];`
  - `var _0x1081d6 = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=';`
- 속는 셈 치고 BASE64 decode
  - ['setFlagAsValid', 'flag', 'substr', ' => pHd_1w_e4rL13r;)}', 'JSInterface']
  - 키의 뒤쪽 21글자를 주었다!

Step 6: Z3

- 조건에 맞게 Z3 돌리기

```
step1key[j] = step1key[j] ^ flag[4 + 4*i + j]
step2key[j] = step2key[j] ^ flag[4 + 4*(i+1) + j]
step3key[j] = step3key[j] ^ flag[4 + 4*(i-1) + j]
step4key[j] = step4key[j] ^ flag[4*i + j]
step5key[j] = step5key[j] ^ flag[4*i + j]
```

- 이 때 각 step별로 찾았던 키들을 잘 넣어야 SAT

```
'OOO{pox&mpuzz,U_solve_it => pHd_1w_e4rL13r;)}'
```