REVERSE
- easyasm
- `creakme`
- flag checker*
- 猫头鹰不是猫

CTYPTO
- `easy rsa`
- English Novel
- `Matryoshka`

MISC
- 好康的流量*
- 这个压缩包有点麻烦*

IOT
- 饭卡的`uno`

end

REVERSE

easyasm

分析汇编

```
seg003:000D                 cmp     si, 1Ch
seg003:0010                 jz      short loc_10134
seg003:0012                 xor     ax, ax
seg003:0014                 mov     al, [si]
seg003:0016                 shl     al, 1
seg003:0018                 shl     al, 1
seg003:001A                 shl     al, 1
seg003:001C                 shl     al, 1
seg003:001E                 push    ax
seg003:001F                 xor     ax, ax
seg003:0021                 mov     al, [si]
seg003:0023                 shr     al, 1
seg003:0025                 shr     al, 1
seg003:0027                 shr     al, 1
seg003:0029                 shr     al, 1
seg003:002B                 pop     bx
seg003:002C                 add     ax, bx
seg003:002E                 xor     ax, 17h
seg003:0031                 add     si, 1
seg003:0034                 cmp     al, es:[si]
seg003:0037                 jnz     short loc_100DD
seg003:0039                 mov     ax, 0B800h
seg003:003C                 mov     es, ax
```

脚本解密，直接暴力破解

```
a=[  0x91, 0x61, 0x01, 0xC1, 0x41, 0xA0, 0x60, 0x41, 0xD1, 0x21,
```

```
  0x14, 0xC1, 0x41, 0xE2, 0x50, 0xE1, 0xE2, 0x54, 0x20, 0xC1,
  0xE2, 0x60, 0x14, 0x30, 0xD1, 0x51, 0xC0, 0x17]

b='ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=_)
(*&^%$#@!+~|}{":'
p=''
print(len(a))
for j in range(len(a)):
    for i in b:
        i = ord(i)
        #print(i)
        t=((i%16)*16+int(i/16))^0x17
        if(t==a[j]):
            p+=chr(i)

print(p)
```

creakme

tea加密算法，但是还加了一个异或，还有把算法里面的数变了一下，去网上找了一份解密算法哈哈哈找到了c语言的算法，就在这个基础上修改了一下

```
#include<stdio.h>
void DecryptTEA(unsigned int *firstChunk, unsigned int *secondChunk, unsigned
int* key)
{
    unsigned int  sum = 0;
    unsigned int  y = *firstChunk;
    unsigned int  z = *secondChunk;
    unsigned int  delta = 0x12345678;

    sum = delta << 5; //32轮运算，所以是2的5次方；16轮运算，所以是2的4次方；8轮运算，所以
是2的3次方

    for (int i = 0; i < 32; i++) //32轮运算
    {
        z -= ((y << 4) + key[0]) ^ (y + sum) ^ ((y >> 5) + key[1]) ^ sum;
        y -= ((z << 4) + key[2]) ^ (z + sum) ^ ((z >> 5) + key[3]) ^ sum;
        sum -= delta;
    }

    *firstChunk = y;
    *secondChunk = z;
}
//buffer：输入的待解密数据buffer，在函数中直接对元数据buffer进行解密；size：buffer长度；
key是密钥；
void DecryptBuffer(char* buffer, int size, unsigned int* key)
{
    char *p = buffer;
    int leftSize = size;
    while (p < buffer + size &&
        leftSize >= sizeof(unsigned int) * 2)
    {
```

```
        DecryptTEA((unsigned int *)p, (unsigned int *)(p + sizeof(unsigned
int)), key);
        p += sizeof(unsigned int) * 2;

        leftSize -= sizeof(unsigned int) * 2;
    }
}

int main(){

    //int key[4] = {0x41424344,0x45464748,0x494a4b4c,0x4d4e4f50};
    unsigned int *key = (unsigned int *)"ABCDEFGHIJKLMNOPQRST";
    char pBuffer[32] = {
      0x88, 0x34, 0xD9, 0x48, 0x4C, 0x14, 0x0C, 0x03, 0xC2, 0x78,
      0xEB, 0x52, 0xED, 0xE5, 0x9C, 0xED,
      0xE6, 0xED, 0x1F, 0xAE, 0x6D, 0x12, 0x5A, 0xBA, 0xAA, 0x84,
      0x92, 0xCF, 0xE3, 0xF2, 0xE0, 0x65};
    //因为发现最后解密出来数组是24位的，就分开四组算了一下
    //有一点疑惑就是最后一组里面解密以后ascii码全是0，为什么还要设置最后这一组数据。
    int pSize = 32;
    DecryptBuffer(pBuffer, pSize, key);
    for(int i=0;i<32;i++){
        printf("%c",pBuffer[i]);
    }

}
```

flag checker*

反编译的时候遇到问题

> 使用`notepad++`文件浏览工具(任何能打开文件的工具都可以，例如记事本...)打开`dex`文件，**修改版本信息037为036或者035**，然后保存文件即可，再使用反编指令：`d2j-dex2jar.bat classes.dex` 反编即可成功。

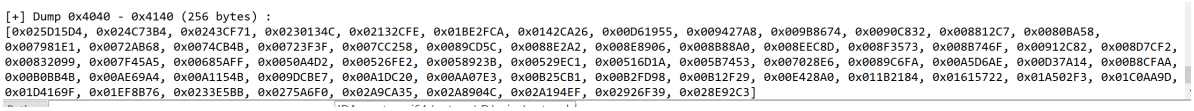
然后用`java decompiler`打开，找到主要函数

```
import javax.crypto.spec.SecretKeySpec;

public class MainActivity extends AppCompatActivity {
  public static byte[] encrypt(String paramString1, String paramString2) throws Exception {
    SecretKeySpec secretKeySpec = new SecretKeySpec(paramString2.getBytes(), 0, paramString2.length(), "RC4");
    Cipher cipher = Cipher.getInstance("RC4");
    cipher.init(1, secretKeySpec);
    return cipher.doFinal(paramString1.getBytes());
  }

  protected void onCreate(Bundle paramBundle) {
    super.onCreate(paramBundle);
    setContentView(2131296284);
    ((Button)findViewById(2131165218)).setOnClickListener(new View.OnClickListener() {
          public void onClick(View param1View) {
            String str = ((EditText)MainActivity.this.findViewById(2131165238)).getText().toString();
            byte[] arrayOfByte = new byte[0];
            try {
              byte[] arrayOfByte1 = MainActivity.encrypt(str, "carol");
              arrayOfByte = arrayOfByte1;
            } catch (Exception exception) {
              exception.printStackTrace();
            }
            if (Base64.encodeToString(arrayOfByte, 0).replace("\n", "").equals("mg6CITV6GEaFDTYnObFmENOAVjKcQmGncF90WhqvCFyhhsyqq1s=")) {
              Toast.makeText((Context)MainActivity.this, "Congratulations!!!", 1).show();
              return;
            }
            Toast.makeText((Context)MainActivity.this, "Fail,try again.", 1).show();
          }
        });
  }
}
```

有一串字符，是用`base64`加密过后去掉`\n`之后的字符串，感觉他的加密过程是先`rc4`，之后进行`base64`，再把换行符去掉，但`base64`是每76个字符才加一个换行符，这里就不知道怎么办了。

猫头鹰不是猫

`ida`打开，重点是`sub1537`和`sub12e6`这两个函数，`sub12e6`就是比较输入的字符串和`unk4040`处的数据，这里需要注意的是`unk4040`那里的数据本来的`byte`类型的，而比较的时候比较的是`dword`类型的数据，可以用`lazyida`提取一下数据



`sub1537`那里的函数需要仔细分析一下，也是先把`unk4140`和`unk8140`处的数据提取出来，然后逆`sub1537`那里就好了

解密算法

```
for j in range(len(unk4140)):
    unk4140[j] =int(unk4140[j]/10)
    unk8140[j] =int(unk8140[j]/10)

mat4140=np.matrix(unk4140).reshape((64,64))
mat8140=np.matrix(unk8140).reshape((64,64))
mat4040=unk4040

print(mat4040)
B_mat=np.linalg.inv(mat8140)
#print(B_mat)
B=np.dot(mat4040,B_mat)
A_mat=np.linalg.inv(mat4140)
A=np.dot(B,A_mat)
a=A.A
print(a)

flag=''
for i in range(64):
    if(a[0][i]-int(a[0][i])>0.5):
        flag+=chr(int(a[0][i]+1))
    else:
        flag+=chr(int(a[0][i]))
print(flag)
```

CTYPTO

easy rsa

查看代码，直接破解

```
#e, p, q, pow(ord(c), e, p * q)
```

```
#38位
flag=''
a = [(12433, 149, 197, 104), (8147, 131, 167, 6633), (10687, 211, 197, 35594),
(19681, 131, 211, 15710), (33577, 251, 211, 38798), (30241, 157, 251, 35973),
(293, 211, 157, 31548), (26459, 179, 149, 4778), (27479, 149, 223, 32728),
(9029, 223, 137, 20696), (4649, 149, 151, 13418), (11783, 223, 251, 14239),
(13537, 179, 137, 11702), (3835, 167, 139, 20051), (30983, 149, 227, 23928),
(17581, 157, 131, 5855), (35381, 223, 179, 37774), (2357, 151, 223, 1849),
(22649, 211, 229, 7348), (1151, 179, 223, 17982), (8431, 251, 163, 30226),
(38501, 193, 211, 30559), (14549, 211, 151, 21143), (24781, 239, 241, 45604),
(8051, 179, 131, 7994), (863, 181, 131, 11493), (1117, 239, 157, 12579), (7561,
149, 199, 8960), (19813, 239, 229, 53463), (4943, 131, 157, 14606), (29077, 191,
181, 33446), (18583, 211, 163, 31800), (30643, 173, 191, 27293), (11617, 223,
251, 13448), (19051, 191, 151, 21676), (18367, 179, 157, 14139), (18861, 149,
191, 5139), (9581, 211, 193, 25595)]
for i in range(38):
    e = a[i][0]
    p = a[i][1]
    q = a[i][2]
    for j in range(33,126):
        if(pow(j, e, p * q)==a[i][3]):
            flag+=chr(j)

print(flag)
print(len(flag))
```

English Novel

打乱并且加密了，查看加密文件的的代码，发现加密算法只改变原文中的大写小写字母，先匹配一下每一个加密文件和原文件

```
'''
先全文大小比较，再找文中的空格看源文件和加密后的文件中前几个空格是否位置匹配
'''
import os
def findSubStrIndex(substr, str, time):
    times = str.count(substr)
    if (times == 0) or (times < time):
        pass
    else:
        i = 0
        index = -1
        while i < time:
            index = str.find(substr, index+1)
            i+=1
        return index

if __name__ == '__main__':
    for k in range(410):
        filePath1 = 'D:/mine/network engineering/ctf/challenges/English
Novel/original/part'+str(k)+'.txt'
```

```
        fileoperation1 = open(filePath1, 'rb')
        url1 = fileoperation1.read()
        lengh = len(url1)
        fileoperation1.close()

        for i in range(410):
            filePath2 ='D:/mine/network engineering/ctf/challenges/English
Novel/encrypt/part'+str(i)+'.txt'
            fileoperation2 = open(filePath2, 'rb')
            url2 = fileoperation2.read()
            fileoperation2.close()
            if(lengh == len(url2)):
                for j in range(200):
                    if(findSubStrIndex(' ',str(url2) , j) !=findSubStrIndex('
',str(url1) ,j)):
                        break
                if(j==199):
                    print(k,i)
                    oldname = 'D:/mine/network
engineering/ctf/challenges/English Novel/encrypt/part'+str(i)+' - 副本.txt'
                    newname = 'D:/mine/network
engineering/ctf/challenges/English Novel/encrypt/anew'+str(k)+'.txt'
                    os.rename(oldname,newname)
```

其实好像不需要把所有的`part`都匹配上，匹配一部分就能完成后面解密的过程

看加密算法知道，现在需要把`key`找出来，暴力破解

```
if __name__ == '__main__':
    choose =
'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=_)(*&^%$#@!+~|}
{":'
    key = ''
    for j in range(44):
        for k in range(0,410):
            inkey = []
            filePath1 = 'D:/mine/network engineering/ctf/challenges/English
Novel/original/part'+str(k)+'.txt'
            fileoperation1 = open(filePath1, 'rb')
            url1 = fileoperation1.read()
            fileoperation1.close()

            filePath2 ='D:/mine/network engineering/ctf/challenges/English
Novel/encrypt/anew'+str(k)+'.txt'
            fileoperation2 = open(filePath2, 'rb')
            url2 = fileoperation2.read()
            fileoperation2.close()
            #print(chr(url2[0]),chr(url1[0]))
            if(chr(url2[j]).isupper() or chr(url2[j]).islower()):

                if chr(url2[j]).isupper():
                    for i in choose:
                        if(url2[j]==(url1[j] - ord('A') + ord(i)) % 26 +
ord('A')):
                            #print(j)
```

```
                            inkey.append(i)

                elif chr(url2[j]).islower():
                    for i in choose:
                        if(url2[j]==(url1[j] - ord('a') + ord(i)) % 26 +
ord('a')):
                            inkey.append(i)
            if(len(inkey)!=0):
                print(inkey)
```

得到key之后再解密`flag.enc`中的数据，还是暴力破解哈哈哈，因为他加密算法里面有一个取余数，不太好逆，就直接暴力破解了

也是因为他取了余数，所以直接暴力破解最后破解出来的都是大写字母，但是根据加密算法看，原始数据是小写加密之后应该还是小写，这里卡了好久，后面才想到他取了余数，可以在解密的基础上+26，或者+26+26

```
if __name__ == '__main__':

    choose =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=_)(*&^%$#@!+~|}
{:'"
    data = "klsyf{W0_j0v_ca0z_'Ks0ao-bln1qstxp_juqfqy'?}"
    key = "QSFZOLGXWYOOVFAGQZHNSXROWZNAJQMCNJMWOKSBBQQG"
    result = ''
    for i in range(44):
        if data[i].isupper():
            for j in choose:
                #print((ord(j)-ord("A")+ord(key[i]))%26 + ord('A'))
                if ( ord(data[i]) == (ord(j)-ord("A")+ord(key[i]))%26 +
ord('A')):
                    #print("!!!")
                    result+=j
                    break
        elif data[i].islower():
            for j in choose:
                if ( ord(data[i]) == (ord(j)-ord("a")+ord(key[i]))%26+ord('a')):
                    #print("!!!")
                    tt=chr(ord(j)+26)
                    if(tt>='a' and tt<='z'):
                        result+=tt
                        print(tt)
                    else:
                        result+=chr(ord(j)+26+26)
                    break
        else:
                result+=data[i]
    print(result)
```

Matryoshka

先是盲文，解密出来是这样的

```
....-/-..../--..--/-..../-..../--..--/....-/..---/--..--/--.../...../--..-
-/-..../-..../--..--/....-/...../--..--/....-/-..../--..--/-...././--..-
-/-..../-../--..--/....-/-.-./--..--/--.../...--/--..--/...--/-..../--..-
-/..../..../--..--/...--/...--/--..--/--.../...--/--..--/-..../----./--..-
-/...../----./--..--/--.../....-/--..--/....-/-.-./--..--/...--/-..../--..-
-/...../---../--..--/...--/..---/--..--/--.../-----/--..--/...--/..../--..-
-/-..../----./--..--/..../-/./--..--/...--/-----/--..--/-..../...--/--..-
-/-..../..../--..--/...../...--/--..--/-..../-.-./--..--/--.../----./--..-
-/-..../-.../--..--/-..../-../--..--/...--/----./--..--/--.../..---/--..-
-/...../.----/--..--/..../-/./--..--/...--/----./--..--/-..../..-./--..-
-/..../-../--..--/...../...--/--..--/...--/.----/--..--/-..../.-/--..-
-/-..../-.../--..--/--.../...--/--..--/...--/----./--..--/--.../..---/--..-
-/..../-.../--..--/...--/..---/--..--/...../..---/--..--/...--/-..../--..-
-/-..../-.../--..--/..../-.-./--..--/...--/---../--..--/-..../---../--..-
-/-..../..-./--..--/--.../..---/--..--/...--/-----/--..--/...--/-..
```

摩斯密码，解密出来有点奇怪，还包含一些特殊字符，而且提示说摩斯密码是第三层，想着可以把他逆序一下，再摩斯解密，出来的字符用`ascii`码就很好解了，接下来是`vigenere`，解密出来这样

```
YzBibXZnaHl6X3swUmF6X2d4eGOwdGhrem9fMG9iMG1fdm9rY2N6dF8hcnO
```

想着可以用`base64`试一下，

```
cObmvghyz_{ORaz_gxxmOthkzo_OobOm_vokcczt_!r}
```

题目有提示，先用`caesar`解出来

```
hOgralmde_{OWfe_lccrOympet_OtgOr_atphhey_!w}
```

看到隔着一位有`hgame{`，就把他再解一下就好了

```
a='hOgralmde_{OWfe_lccrOympet_OtgOr_atphhey_!w}'
flag=''
for i in range(len(a)):
    if(i%2==0):
        flag+=a[i]
for i in range(len(a)):
    if((i+1)%2==0):
        flag+=a[i]
print(flag)
```

MISC

好康的流量*

用`wiresharp`分析，根据题目提示应该是可以提取出一张图片的，直接搜索，还用`base64`加密过的，追踪`tcp`流之后把数据拷贝出来，解密到文件里面

```
import base64

filePath1 = 'C:/Users/admin/Desktop/1.txt'
fileoperation = open(filePath1, 'rb')
url = fileoperation.read()
fileoperation.close()

str_url = base64.b64decode(url)

filePath2 = 'C:/Users/admin/Desktop/2.txt'
writeFileOperation = open(filePath2, 'wb')
writeString = str_url
writeFileOperation.write(writeString)
writeFileOperation.close()
```

可以看到一张图片

放到`stegsolve`里面，换通道，看到一个条形码，解出来是flag的前半部分

`hgame{ez_1mg_`

这个压缩包有点麻烦*

第一个压缩包根据提示六位数字爆破

第二个压缩包根据文件里面的字符串进行字典攻击破解

第三个压缩包因为提示里面有`just store it`，明文压缩`txt`文件的时候压缩方式是存储，一段时间之后手动终止得到密钥

然后就得到一张图片

之后就没有找到flag...

IOT

饭卡的uno

额，这个题去网上搜了一下，.hex文件有的可以用`ida`打开,反编译他的程序，但打开后没有反编译出来，全是数据段，一开始想尝试把他变成代码，然后就在程序后半段发现了这个

```
05BD                    align 2
05BE                    db  68h ; h
05BF aGameF1rst5tep0    db 'game{F1rst_5tep_0F_IOT}',0
05D7                    db  0Dh
05D8                    db  0Ah
```

end