Normal2 WP

加密分析

静态分析可以看到`sub_140001000`内部有用到一个大数组，开头是

`63h, 7Ch, 77h, 7Bh, 0F2h, 6B...`，一共五百多个byte，但开头是AES加密的Sbox，可以在线查一下是完全对得上的，IDA里面创建数组，大小设置为256，后面还有两百多个数，容易发现是AES的逆S盒。

既然基本确定了是AES加密，那么函数里面过于复杂的部分不用分析了，直接运行看效果，我们输入一个32位长度的flag（根据密文猜测的长度），main函数运行完`sub_140012A0`时，栈上的flag前面一半已经加密了

自行测试可以发现就是AES的ECB模式，key的话可以在`sub_140001000`找到，

就在arr里面：

继续调试，这个部分

作用纯属就是把内存中16进制的数相当于做了`b16encode`处理，结果放到了

```
if ( lstrcmpA(&String1,
"934d8706bed74cd6eea683c7be86b2eb32616562363039383965386433333531") )
```

String1里面。

到此为止，程序的加密过程清晰了：

取前面16字节用KEY进行AES加密，后16字节直接转化成16进制。

解题脚本

windows下需要安装pycryptodome模块

```
from base64 import b16decode, b64encode
from Crypto.Cipher import AES

ciphertext = '934d8706bed74cd6eea683c7be86b2eb3261656236303938396538643333353 1'
flag2 = b16decode(ciphertext[32:].upper())

key = [0x1B, 0x2E, 0x35, 0x46, 0x58, 0x6E, 0x72, 0x86,
       0x9B, 0xA7, 0xB5, 0xC8, 0xD9, 0xEF, 0xff, 0x0C]

aes = AES.new(bytes(key), AES.MODE_ECB)
to_decrypt = b16decode(ciphertext[:32].upper())

flag1 = aes.decrypt(to_decrypt)

flag = flag1 + flag2
print(flag.decode())
```

得到flag：

`hxb2018{853ecfe52aeb60989e8d3351}`