EasyCM_WP

发表于 2021-01-15 分类于 Challenge， 2020， 安洵杯， Reverse
Challenge | 2020 | 安洵杯 | Reverse | EasyCM_WP

点击此处获得更好的阅读体验

WriteUp来源

https://xz.aliyun.com/t/8582

题目考点

解题思路

1.程序框架

通过TLS回调函数对程序关键加密函数进行SMC自解码，用户输入字符串，通过关键加密函数加密后与字符串比对。

2.关键加密函数

通过SMC自解密后可以查看

类似base64的重组位之后查表，同时另一个线程对表进简单的换表操作，线程同步进行。

（base表被简单加密隐藏）

3.反调试

静态反调试：几处花指令。

动态反调试：CheckRemoteDebuggerPresent

3.三个TLS回调函数

进入程序有三个TLS回调函数：

TlsCallBack_0

这两个函数都加了花指令。其中SMC自解码过程

TlsCallBack_1

TlsCallBack_2

4.两个子线程：

1. *子线程1，对 假flag 进行初次加密 得到比较字符串。*
2. *子线程2，先对程序中的一串数据进行加密后得到base表，再与主线程进行线程同步换表，且第一次换表在前。（这个线程加了花指令）*

其实base表解密之后就是标准的base64码表，不过下面要变换一下。

5.进入主函数

关键函数内部，因为添加花指令，循环调用关键加密函数 的部分缺失。

对SMC自解码代码解密 脚本

IDA中 IDC对SMC自解码代码解密 脚本

```
//IDA中 IDC对SMC自解码代码解密 脚本
auto address_s = 0x41E000;
auto address_e = 0x41F200;
auto i = 0;

for (; address_s+i < address_e; i++){
    if (i % 4 == 0){
        PatchByte(address_s+i, Byte(address_s+i) ^ 'D');
    }else if (i % 4 == 1){
        PatchByte(address_s+i, Byte(address_s+i) ^ '0');
    }else if (i % 4 == 2){
        PatchByte(address_s+i, Byte(address_s+i) ^ 'g');
    }else if (i % 4 == 3){
        PatchByte(address_s+i, Byte(address_s+i) ^ '3');
    }
}
Message("\ndone\n");
```

或者手动对PE文件中的节区（节区名：‘.cyzcc’）数据进行解密。（解密之后的代码也添加了花指令，需要去除一下）

代码解密之后

*这部分关键代码，就是对字节的**位**进行一个重组，之后查表，换一次表加密数据一次。*

6.脚本

编程语言：c，编译环境：vc++6.0

```
//table数组就是被加密隐藏的base表，得事先对一串数据解密后得到，我这里直接解密后贴过来了。
#include<stdio.h>
char table[150] = { 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P',
        'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', 'a', 'b', 'c', 'd', 'e', 'f',
        'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v',
        'w', 'x', 'y', 'z', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '+', '/' };
char RTS[] =  "D0g3{cyzcc_have_ten_girlfriends}";
char rra[] = { 0x23,0x7a,0x3d,0x60,0x34,0x7,0x11,0x36,0x2c,0x5,
               0xc,0x20,0xb,0x22,0x3f,0x6f,0x16,0x0,0x37,0xd,
               0x36,0xf,0x1e,0x20,0x37,0x14,0x2,0x9,0x2,0xf,
               0x1b,0x39, };
char str[100] = {0};

int main()
{
    unsigned char a = 0 ;
    unsigned char b = 0 ;
    unsigned char c = 0 ;
    unsigned char d = 0 ;
    unsigned int k = 0 ;
    unsigned int i = 0 ;

    for( i=0 ; RTS[i] ; i++)
        RTS[i] ^= rra[i];

    char* p = table;

    for(int j = 0; RTS[j] ; j += 4 )
    {
        //这里开始循环换表
        *(p+64) = *p;
        p++;

        for(i =0 ; i<64 ; i++)
            if( RTS[j] == *(p+i) )
            {
                a = i;
                break;
            }
        for(i =0 ; i<64 ; i++)
            if( RTS[j+1] == *(p+i) )
            {
                b = i;
                break;
            }
        for(i =0 ; i<64 ; i++)
            if( RTS[j+2] == *(p+i) )
            {
                c = i;
                break;
            }
        for(i =0 ; i<64 ; i++)
            if( RTS[j+3] == *(p+i) )
            {
                d = i;
                break;
            }
        k = (j/4)*3;
        str[k+0] = a<<2&0xC0 | c<<2&0x30 | d>>2&0xC | b&0x3;
        str[k+1] = b<<2&0xC0 | a<<2&0x30 | d>>0&0xC | c&0x3;
        str[k+2] = c<<2&0xC0 | b<<2&0x30 | d<<2&0xC | a&0x3;
    }
    printf("%s\n",str);
    return 0;
}
```

Flag

- ***本文作者：*** *CTFHub*
- ***本文链接：*** *https://writeup.ctfhub.com/Challenge/2020/安洵杯/Reverse/5JXMBvWKTW73BBqD54TvLj.html*
- ***版权声明：*** *本博客所有文章除特别声明外，均采用 BY-NC-SA 许可协议。转载请注明出处！*

 # Challenge # 2020 # Reverse # 安洵杯

IO_FILE
anxun3