

BUU_re_[ACTF新生赛2020]usualCrypt

原创

A-Juan 于 2020-08-18 22:50:49 发布 206 收藏

分类专栏: [逆向](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: https://blog.csdn.net/weixin_46874126/article/details/108089679

版权



[逆向](#) 专栏收录该内容

8 篇文章 0 订阅

订阅专栏

查壳, 无壳, 用IDA32位打开, 找到主函数, 按F5查看其伪代码

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
    int v3; // esi
    int result; // eax
    int v5; // [esp+8h] [ebp-74h]
    int v6; // [esp+Ch] [ebp-70h]
    int v7; // [esp+10h] [ebp-6Ch]
    __int16 v8; // [esp+14h] [ebp-68h]
    char v9; // [esp+16h] [ebp-66h]
    char v10; // [esp+18h] [ebp-64h]

    sub_403CF8((int)&unk_40E140); // 输出
    scanf(aS, &v10); // 输入
    v5 = 0;
    v6 = 0;
    v7 = 0;
    v8 = 0;
    v9 = 0;
    sub_401080((int)&v10, strlen(&v10), (int)&v5); // 加密
    v3 = 0;
    while ( *((_BYTE *)&v5 + v3) == byte_40E0E4[v3] )
    {
        if ( ++v3 > strlen((const char *)&v5) )
            goto LABEL_6;
    }
    sub_403CF8((int)aError);
LABEL_6:
    if ( v3 - 1 == strlen(byte_40E0E4) )
        result = sub_403CF8((int)aAreYouHappyYes);
    else
        result = sub_403CF8((int)aAreYouHappyNo);
    return result;
}
```

分析, 猜测是输入, 加密, 然后比较, 如果比较结果相等, 则成功

byte_40E0E4为: MXHz3TlgnxLxJhFAdtZn2fFk3IYCrPC2I9

查看sub_401080

```
int __cdecl sub_401080(int a1, int a2, int a3)
{
    int v3; // edi
    int v4; // esi
    int v5; // edx
    int v6; // eax
    int v7; // ecx
    int v8; // esi
    int v9; // esi
    int v10; // esi
    int v11; // esi
    _BYTE *v12; // ecx
    int v13; // esi
    int v15; // [esp+18h] [ebp+8h]

    v3 = 0;
    v4 = 0;
    sub_401000();
    v5 = a2 % 3;
    v6 = a1;
    v7 = a2 - a2 % 3;
    v15 = a2 % 3;
    if ( v7 > 0 )
    {
        do
        {
            LOBYTE(v5) = *(_BYTE *)(a1 + v3);
            v3 += 3;
            v8 = v4 + 1;
            *(_BYTE *)(v8++ + a3 - 1) = byte_40E0A0[(v5 >> 2) & 0x3F];
            *(_BYTE *)(v8++ + a3 - 1) = byte_40E0A0[16 * *(_BYTE *)(a1 + v3 - 3) & 3
                + (((signed int)*(unsigned __int8 *) (a1 + v3 - 2) >> 4) & 0xF)];
            *(_BYTE *)(v8 + a3 - 1) = byte_40E0A0[4 * *(_BYTE *)(a1 + v3 - 2) & 0xF
                + (((signed int)*(unsigned __int8 *) (a1 + v3 - 1) >> 6) & 3)];
            v5 = *(_BYTE *)(a1 + v3 - 1) & 0x3F;
            v4 = v8 + 1;
            *(_BYTE *)(v4 + a3 - 1) = byte_40E0A0[v5];
        }
        while ( v3 < v7 );
        v5 = v15;
    }
    if ( v5 == 1 )
    {
        LOBYTE(v7) = *(_BYTE *)(v3 + a1);
        v9 = v4 + 1;
        *(_BYTE *)(v9 + a3 - 1) = byte_40E0A0[(v7 >> 2) & 0x3F];
        v10 = v9 + 1;
        *(_BYTE *)(v10 + a3 - 1) = byte_40E0A0[16 * *(_BYTE *)(v3 + a1) & 3];
        *(_BYTE *)(v10 + a3) = 61;
    LABEL_8:
        v13 = v10 + 1;
        *(_BYTE *)(v13 + a3) = 61;
        v4 = v13 + 1;
        goto LABEL_9;
    }
    if ( v5 == 2 )
    {
        v11 = v4 + 1;
```

```

v11 = v7 + 1;
*(_BYTE *)(v11 + a3 - 1) = byte_40E0A0[((signed int)*(unsigned __int8 *)(v3 + a1) >> 2) & 0x3F];
v12 = (_BYTE *)(v3 + a1 + 1);
LOBYTE(v6) = *v12;
v10 = v11 + 1;
*(_BYTE *)(v10 + a3 - 1) = byte_40E0A0[16 * (*(_BYTE *)(v3 + a1) & 3) + ((v6 >> 4) & 0xF)];
*(_BYTE *)(v10 + a3) = byte_40E0A0[4 * (*v12 & 0xF)];
goto LABEL_8;
}
}
LABEL_9:
*(_BYTE *)(v4 + a3) = 0;
return sub_401030(a3);
}

```

这个函数可以分为三个部分:开头sub_401000函数;中间base64加密;结尾sub_401030函数

sub_401000函数:

```

signed int sub_401000()
{
    signed int result; // eax
    char v1; // cl

    result = 6;
    do
    {
        v1 = byte_40E0AA[result];
        byte_40E0AA[result] = byte_40E0A0[result];
        byte_40E0A0[result++] = v1;
    }
    while ( result < 15 );
    return result;
}

```

sub_401000这个函数的作用是改变密钥对应表。调用后，部分字符的意义发生了变化。

sub_401030函数:

```

int __cdecl sub_401030(const char *a1)
{
    __int64 v1; // rax
    char v2; // al

    v1 = 0i64;
    if ( strlen(a1) != 0 )
    {
        do
        {
            v2 = a1[HIDWORD(v1)];
            if ( v2 < 97 || v2 > 122 )
            {
                if ( v2 < 65 || v2 > 90 )
                    goto LABEL_9;
                LOBYTE(v1) = v2 + 32;
            }
            else
            {
                LOBYTE(v1) = v2 - 32;
            }
            a1[HIDWORD(v1)] = v1;
LABEL_9:
            LODWORD(v1) = 0;
            ++HIDWORD(v1);
        }
        while ( HIDWORD(v1) < strlen(a1) );
    }
    return v1;
}

```

sub_401000函数功能为大小写字母转换

解密的思路是：反过来，先将byte_40E0E4大小写互换，构造base64变表，再利用变表将byte_40E0E4转换为正常的base64解密。

脚本：

```

import base64

secret = 'zMXHz3TIgnxLxJhFAdtZn2fFk3lYCrtPC219'.swapcase() #大小写转换
a = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'
dict = {}
offset = 10
flag = ''
for i in range(len(a)):
    dict[a[i]] = a[i]
for i in range(6,15): #构造修改后的表(字典)
    b = dict[a[i]]
    dict[a[i]] = dict[a[i+offset]]
    dict[a[i+offset]] = b
for i in range(len(secret)):
    flag += dict[secret[i]]
flag = base64.b64decode(flag)
print(flag)

#flag{bAse64_h2s_a_Surprise}

```