

buuctf刷题记录10 [ACTF新生赛2020]usualCrypt

原创

ytj00 于 2020-07-30 17:19:01 发布 284 收藏

分类专栏: [ctf 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/ytj00/article/details/107694610>

版权



ctf 同时被 2 个专栏收录

28 篇文章 0 订阅

订阅专栏



逆向

27 篇文章 0 订阅

订阅专栏

ida打开, 进入main函数

```
未知 外部符号
IDA View-A 伪代码 Stack of _main 十六进制视图-1 结构体 枚举
段 ^
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     int v3; // esi
4     int result; // eax
5     int v5; // [esp+8h] [ebp-74h]
6     int v6; // [esp+Ch] [ebp-70h]
7     int v7; // [esp+10h] [ebp-6Ch]
8     __int16 v8; // [esp+14h] [ebp-68h]
9     char v9; // [esp+16h] [ebp-66h]
10    char v10; // [esp+18h] [ebp-64h]
11
12    sub_403CF8((int)&unk_40E140);
13    scanf(aS, &v10);
14    v5 = 0;
15    v6 = 0;
16    v7 = 0;
17    v8 = 0;
18    v9 = 0;
19    sub_401080((int)&v10, strlen(&v10), (int)&v5);
20    v3 = 0;
21    while ( *((_BYTE *)&v5 + v3) == byte_40E0E4[v3] )
22    {
23        if ( ++v3 > strlen((const char *)&v5) )
24            goto LABEL_6;
25    }
26    sub_403CF8((int)aError);
27 LABEL_6:
28    if ( v3 - 1 == strlen(byte_40E0E4) )
29        result = sub_403CF8((int)aAreYouHappyYes);
30    else
31        result = sub_403CF8((int)aAreYouHappyNo);
32    return result;
33 }
```

<https://blog.csdn.net/ytj00>

有一个关键函数, 和一个关键比较

先进入关键函数里面

```

7  sub_401000: // [ebp+0] [ebp+0]
8
9  v3 = 0;
10 v4 = 0;
11 sub_401000();
12 v5 = a2 % 3;
13 v6 = a1;
14 v7 = a2 - a2 % 3;
15 v15 = a2 % 3;
16 if ( v7 > 0 )
17 {
18     do
19     {
20         LOBYTE(v5) = *(_BYTE *)(a1 + v3);
21         v3 += 3;
22         v8 = v4 + 1;
23         *(_BYTE *)(v8++ + a3 - 1) = byte_40E0A0[(v5 >> 2) & 0x3F];
24         *(_BYTE *)(v8++ + a3 - 1) = byte_40E0A0[16 * (*(_BYTE *)(a1 + v3 - 3) & 3)
25             + (((signed int)*(unsigned __int8 *)(a1 + v3 - 2) >> 4) & 0xF)];
26         *(_BYTE *)(v8 + a3 - 1) = byte_40E0A0[4 * (*(_BYTE *)(a1 + v3 - 2) & 0xF)
27             + (((signed int)*(unsigned __int8 *)(a1 + v3 - 1) >> 6) & 3)];
28         v5 = *(_BYTE *)(a1 + v3 - 1) & 0x3F;
29         v4 = v8 + 1;
30         *(_BYTE *)(v4 + a3 - 1) = byte_40E0A0[v5];
31     }
32     while ( v3 < v7 );
33     v5 = v15;
34 }
35 if ( v5 == 1 )
36 {
37     LOBYTE(v7) = *(_BYTE *)(v3 + a1);
38     v9 = v4 + 1;
39     *(_BYTE *)(v9 + a3 - 1) = byte_40E0A0[(v7 >> 2) & 0x3F];
40     v10 = v9 + 1;
41     *(_BYTE *)(v10 + a3 - 1) = byte_40E0A0[16 * (*(_BYTE *)(v3 + a1) & 3)];

```

<https://blog.csdn.net/yij00>

```

43     v5 = v15;
44 }
45 if ( v5 == 1 )
46 {
47     LOBYTE(v7) = *(_BYTE *)(v3 + a1);
48     v9 = v4 + 1;
49     *(_BYTE *)(v9 + a3 - 1) = byte_40E0A0[(v7 >> 2) & 0x3F];
50     v10 = v9 + 1;
51     *(_BYTE *)(v10 + a3 - 1) = byte_40E0A0[16 * (*(_BYTE *)(v3 + a1) & 3)];
52     *(_BYTE *)(v10 + a3) = 61;
53 LABEL_8:
54     v13 = v10 + 1;
55     *(_BYTE *)(v13 + a3) = 61;
56     v4 = v13 + 1;
57     goto LABEL_9;
58 }
59 if ( v5 == 2 )
60 {
61     v11 = v4 + 1;
62     *(_BYTE *)(v11 + a3 - 1) = byte_40E0A0[(((signed int)*(unsigned __int8 *)(v3 + a1) >> 2) & 0x3F)];
63     v12 = *(_BYTE *)(v3 + a1 + 1);
64     LOBYTE(v6) = *v12;
65     v10 = v11 + 1;
66     *(_BYTE *)(v10 + a3 - 1) = byte_40E0A0[16 * (*(_BYTE *)(v3 + a1) & 3) + ((v6 >> 4) & 0xF)];
67     *(_BYTE *)(v10 + a3) = byte_40E0A0[4 * (*v12 & 0xF)];
68     goto LABEL_8;
69 }
70 LABEL_9:
71     *(_BYTE *)(v4 + a3) = 0;
72     return sub_401030((const char *)a3);
73 }

```

00001105 sub_401080:38 (401105)

<https://blog.csdn.net/yij00>

最开始有一个sub_401000()函数，进入后

```
signed int sub_401000()
```

```
signed int result; // eax  
char v1; // cl
```

```
result = 6;  
do  
{  
    v1 = unk_40E0AA[result];  
    unk_40E0AA[result] = byte_40E0A0[result];  
    byte_40E0A0[result++] = v1;  
}  
while ( result < 15 );  
return result;
```

<https://blog.csdn.net/yij00>

```
.data:0040E0A0 aAbcdefghij      db 'ABCDEFGHJIJ'          ; DATA XREF: sub_401000:loc_401005↑r  
.data:0040E0A0                                     ; sub_401000+17↑w ...  
.data:0040E0AA ; char aKlmnopqrstuvw[]  
.data:0040E0AA aKlmnopqrstuvw db 'KLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'  
.data:0040E0AA                                     ; DATA XREF: sub_401000+B↑r  
.data:0040E0AA                                     ; sub_401000+11↑w
```

感觉有点像base64加密，然后这个函数是变换base64的码表，写脚本将变换后的码表求出来

```
#include <stdio.h>  
int main()  
{  
  
char b[]="ABCDEFGHJIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/" ;  
char c;  
int i=6;  
printf("sss\n");  
do  
{  
    c=b[i+10];  
    b[i+10]=b[i];  
    b[i++]=c;  
}  
while(i<15);  
  
printf("%s",b);  
  
}
```

得到码表: ABCDEFQRSTUVWXYPGHIJKLMNOPZabcdefghijklmnopqrstuvwxyz0123456789+/
注意: 这里有个P, 是原来base64码表里没有的, 是原来base64码表里第17个字符, 也就是原来base64码表里第17个字符, 也就是原来base64码表里第17个字符

然后注意到返回时的函数sub_401030()

```

段 ^ 1 int __cdecl sub_401030(const char *a1)
      2 {
      3     __int64 v1; // rax
      4     char v2; // al
      5
      6     v1 = 0i64;
      7     if ( strlen(a1) != 0 )
      8     {
      9         do
     10         {
     11             v2 = a1[HIDWORD(v1)];
     12             if ( v2 < 'a' || v2 > 'z' )
     13             {
     14                 if ( v2 < 'A' || v2 > 'Z' )
     15                     goto LABEL_9;
     16                 LOBYTE(v1) = v2 + 32;
     17             }
     18             else
     19             {
     20                 LOBYTE(v1) = v2 - 32;
     21             }
     22             a1[HIDWORD(v1)] = v1;
     23 LABEL_9:
     24             LODWORD(v1) = 0;
     25             ++HIDWORD(v1);
     26         }
     27         while ( HIDWORD(v1) < strlen(a1) );
     28     }
     29     return v1;
     30 }

```

<https://blog.csdn.net/ytyj00>

很明显是一个大小写互换的函数

所以我们可以通过main函数里的判断条件来逆推flag

```

v0 = 0,
v7 = 0;
v8 = 0;
v9 = 0;
sub_401080((int)&v10, strlen(&v10), (int)&v5);
v3 = 0;
while ( *((_BYTE *)&v5 + v3) == byte_40E0E4[v3] )
{
    if ( ++v3 > strlen((const char *)&v5) )
        goto LABEL_6;
}
sub_403CF8((int)aError);
ABEL_6:
if ( v3 - 1 == strlen(byte_40E0E4) )
    result = sub_403CF8((int)aError);

```

<https://blog.csdn.net/ytyj00>

```

.data:0040E0E4 byte_40E0E4 db 'z' ; DATA XREF: _main+5C↑
.data:0040E0E4 ; _main:loc_401238↑
.data:0040E0E5 aMxhz3tignxlxjh db 'MXHz3TIgnxLxJhFAdtZn2fFk3lYCrPC219',0
.data:0040E109 align 4

```

比较数据为: zMXHz3TlgnLxJhFAdtZn2fFk3lYCrPC2l9

先转换为大写: ZmxhZ3tiGNXlXjHfaDTzN2FfK3LycRTpc2L9

然后根据之前的码表解密

```
===== RESTART: D:\常用脚本py\密码学+进制\base64解密.py =====  
=====  
  
*****  
*      (1)encode      (2)decode      *  
*****  
  
Please select the operation you want to perform:  
2  
Please enter a string that needs to be decrypted:  
ZmxhZ3tiGNXlXjHfaDTzN2FfK3LycRTpc2L9  
Decrypted String:  
flag{bAse64_h2s_a_Surprise}  
>>>
```

<https://blog.csdn.net/ytyj00>

flag为: flag{bAse64_h2s_a_Surprise}