BUUCTF Reverse/[ACTF新生赛2020]usualCrypt

原创

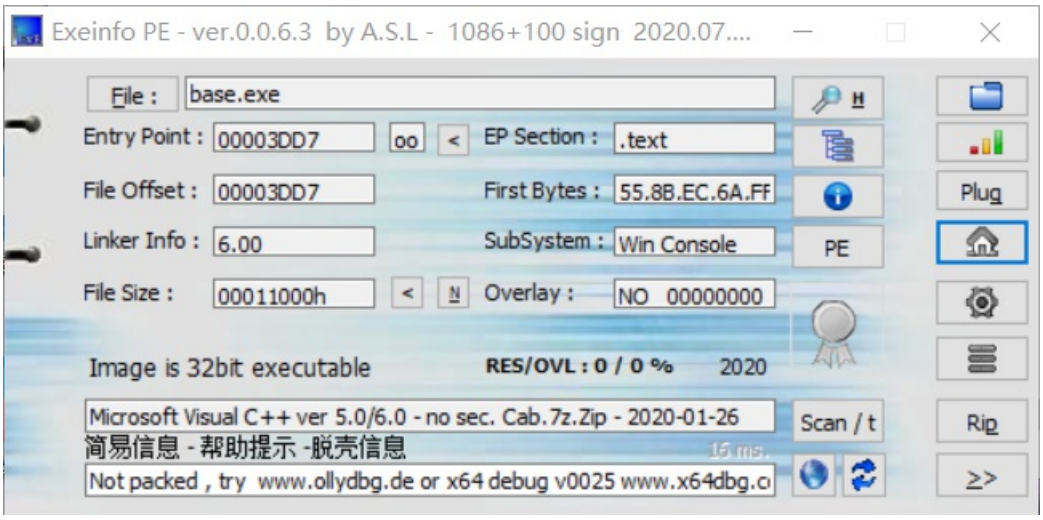
这就是强者的世界么 于 2021-07-22 21:48:28 发布 146 收藏

分类专栏： # BUUCTF Reverse

本文链接：https://blog.csdn.net/ookami6497/article/details/119002676

BUUCTF Reverse/[ACTF新生赛2020]usualCrypt

> 先查看文件信息：没有加壳且为32位程序



用IDA32位打开找到main函数查看伪代码

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  int v3; // esi
  int result; // eax
  int v5[3]; // [esp+8h] [ebp-74h] BYREF
  __int16 v6; // [esp+14h] [ebp-68h]
  char v7; // [esp+16h] [ebp-66h]
  char v8[100]; // [esp+18h] [ebp-64h] BYREF

  sub_403CF8((int)&unk_40E140);
  scanf("%s", v8);
  v5[0] = 0;
  v5[1] = 0;
  v5[2] = 0;
  v6 = 0;
  v7 = 0;
  sub_401080((int)v8, strlen(v8), (int)v5);
  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;
}
```

> 一点点分析代码

> 跟进查看发现这是个输出函数，又是一道字符串比较类型的题目

```
  int v3; // esi
  int result; // eax
  int v5[3]; // [esp+8h] [ebp-74h] BYREF
  __int16 v6; // [esp+14h] [ebp-68h]
  char v7; // [esp+16h] [ebp-66h]
  char v8[100]; // [esp+18h] [ebp-64h] BYREF

  sub_403CF8((int)&unk_40E140);
  scanf("%s", v8);
  v5[0] = 0;
  v5[1] = 0;
  v5[2] = 0;
  v6 = 0;
  v7 = 0;
  sub_401080((int)v8, strlen(v8), (int)v5);
  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;
}
```

```
.data:0040E141                  db 0B4h
.data:0040E142                  db 0C0h
.data:0040E143                  db 0B4h
.data:0040E144                  db 0C0h
.data:0040E145                  db 0B4h
.data:0040E146                  db 0A3h
.data:0040E147                  db 0A6h
.data:0040E148                  db  47h ; G
.data:0040E149                  db  69h ; i
.data:0040E14A                  db  76h ; v
.data:0040E14B                  db  65h ; e
.data:0040E14C                  db  20h
.data:0040E14D                  db  6Dh ; m
.data:0040E14E                  db  65h ; e
.data:0040E14F                  db  20h
.data:0040E150                  db  79h ; y
.data:0040E151                  db  6Fh ; o
.data:0040E152                  db  75h ; u
.data:0040E153                  db  72h ; r
.data:0040E154                  db  20h
.data:0040E155                  db  66h ; f
.data:0040E156                  db  6Ch ; l
.data:0040E157                  db  61h ; a
.data:0040E158                  db  67h ; g
.data:0040E159                  db  3Ah ; :
.data:0040E15A                  db    0
.data:0040E15B                  db    0
.data:0040E15C dword_40E15C     dd 0FFFFFFFFh           ; DATA XREF: sub_40149D+11↑r
.data:0040E15C                                          ; sub_40149D+1C↑w ...
.data:0040E160 ; public class std::ios_base /* mdisp:0 */
0000E143 0040E143: .data:0040E143 (Synchronized with Hex View-1)
```

输入的flag进行变换后与byte_40E0E4中的值进行比较

```
while ( *((_BYTE *)v5 + v3) == byte_40E0E4[v3] )
 {
   if ( ++v3 > strlen((const char *)v5) )
     goto LABEL_6;
 }
```

```
.data:0040E0E1                 align 4
.data:0040E0E4 byte_40E0E4     db 7Ah                  ; DATA XREF: _main+5C↑r
.data:0040E0E4                                         ; _main:loc_401238↑o
.data:0040E0E5 aMxhz3tignxlxjh db 'MXHz3TIgnxLxJhFAdtZn2fFk3lYCrtPC2l9',0
.data:0040E10C aAreYouHappyNo  db 'Are you happy?No!',0Ah,0
.data:0040E10C                                         ; DATA XREF: _main:loc_40125F↑o
```

这里对输入的flag进行了变换，跟进查看

```
sub_401080((int)v8, strlen(v8), (int)v5);
```

```
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)
                                           + (((int)*(unsigned __int8 *)(a1 + v3 - 2) >> 4) & 0xF)];
      *(_BYTE *)(++v8 + a3 - 1) = byte_40E0A0[4 * (*(_BYTE *)(a1 + v3 - 2) & 0xF)
                                           + (((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;
    *(_BYTE *)(v11 + a3 - 1) = byte_40E0A0[((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);
}
```

似曾相识的感觉，跟进byte_40E0A0

```
v5 == 1 )

3YTE(v7) = *(_BYTE *)(v3 + a1);
 = v4 + 1;
_BYTE *)(v9 + a3 - 1) = byte_40E0A0[(v7 >> 2) & 0x3F];
 = v9 + 1;
_BYTE *)(v10 + a3 - 1) = byte_40E0A0[16 * (*(_BYTE *)(v3 + a1) & 3)];
_BYTE *)(v10 + a3) = 61;
3:
 = v10 + 1;
_BYTE *)(v13 + a3) = 61;
 = v13 + 1;
to LABEL_9;

 v5 == 2 )

l = v4 + 1;
_BYTE *)(v11 + a3 - 1) = byte_40E0A0[((int)*(unsigned __int8 *)(v3 + a1) >> 2) & 0x3F];
2 = (_BYTE *)(v3 + a1 + 1);
3YTE(v6) = *v12;
 = v11 + 1;
_BYTE *)(v10 + a3 - 1) = byte_40E0A0[16 * (*(_BYTE *)(v3 + a1) & 3) + ((v6 >> 4) & 0xF)];
_BYTE *)(v10 + a3) = byte_40E0A0[4 * (*v12 & 0xF)];
to LABEL_8;

9:
/TE *)(v4 + a3) = 0;
```

看到这串字符，这不就是base64加密么。。。

```
.data:0040E094 dword_40E094    dd 0                    ; DATA XREF: _doexit:loc_40555C↑o
.data:0040E098                 align 10h
.data:0040E0A0 ; char byte_40E0A0[10]
.data:0040E0A0 byte_40E0A0     db 41h                  ; DATA XREF: sub_401000:loc_401005↑r
.data:0040E0A0                                         ; sub_401000+17↑w ...
.data:0040E0A1                 db  42h ; B
.data:0040E0A2                 db  43h ; C
.data:0040E0A3                 db  44h ; D
.data:0040E0A4                 db  45h ; E
.data:0040E0A5                 db  46h ; F
.data:0040E0A6                 db  47h ; G
.data:0040E0A7                 db  48h ; H
.data:0040E0A8                 db  49h ; I
.data:0040E0A9                 db  4Ah ; J
.data:0040E0AA byte_40E0AA     db 4Bh                  ; DATA XREF: sub_401000+B↑r
.data:0040E0AA                                         ; sub_401000+11↑w
.data:0040E0AB aLmnopqrstuvwxy db 'LMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/',0
.data:0040E0E1                 align 4
.data:0040E0E4 ; char byte_40E0E4[]
```

base64解码

Base编码系列：Base64 Base32 Base16

Base64编码是使用64个可打印ASCII字符（A-Z、a-z、0-9、+、/）将任意字节序列数据编码成ASCII字符串，另有“=”符号用作后缀用途。

Base64 索引表

数值	字符	数值	字符	数值	字符	数值	字符
0	A	16	Q	32	g	48	w
1	B	17	R	33	h	49	x
2	C	18	S	34	i	50	y
3	D	19	T	35	j	51	z
4	E	20	U	36	k	52	0
5	F	21	V	37	l	53	1
6	G	22	W	38	m	54	2
7	H	23	X	39	n	55	3
8	I	24	Y	40	o	56	4
9	J	25	Z	41	p	57	5
10	K	26	a	42	q	58	6
11	L	27	b	43	r	59	7
12	M	28	c	44	s	60	8
13	N	29	d	45	t	61	9
14	O	30	e	46	u	62	+
15	P	31	f	47	v	63	/

Base64将输入字符串按字节切分，取得每个字节对应的二进制值（若不足8比特则高位补0），然后将这些二进制数值串联起来，再按照6比特一组进行切分（因为2^6=64），最后一组若不足6比特则末尾补0。将每组二进制值转换成十进制，然后在上述表格中找到对应的符号并串联起来就是Base64编码结果。

但是除了base64加密，还有两个函数对字符串作了变换

这个函数对加密表进行变换，将 下标为 6 到14 的值 与 16 到24 的值进行调换

```
15
16  v3 = 0;
17  v4 = 0;
18  sub_401000();
19  v5 = a2 % 3;
20  v6 = a1;
21  v7 = a2 - a2 % 3;
22  v15 = a2 % 3;
23  if ( v7 > 0 )
24  {
```

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

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

> 这个函数对加密后的base64编码进行大小写转换

```
65    goto LABEL_8;
66  }
67LABEL_9:
68  *(_BYTE *)(v4 + a3) = 0;
69  return sub_401030(a3);
70}
```

```
0000116A sub_401080:60 (40116A)
```

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

  v1 = 0i64;
  if ( strlen(a1) )
  {
    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;
}
```

那么根据这个解题步骤就是先将 zMXHz3TlgnxLxJhFAdtZn2fFk3lYCrtPC2l9 中的大写字母转小写，小写字母转大写，然后将调换的字母调换回来，再进行base64解密

输出被调换前的表,发现这只交换了大写字母的部分

```
char b[] = "ABCDEFGHIJ"
           "KLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/";
    for(i = 6; i < 15; i++ )   //互换表
   {
       char t = b[i];
       b[i] = b[i + 10];
       b[i + 10] = t;
   }
```

```
C:\Users\86183\Desktop\oj\1EXAMPLE\bin\Debug\1EXAMPLE.exe
ABCDEFQRSTUVWXYPGHIJKLMNOZabcdefghijklmnopqrstuvwxyz0123456789+/

Process returned 0 (0x0)   execution time : 0.641 s
```

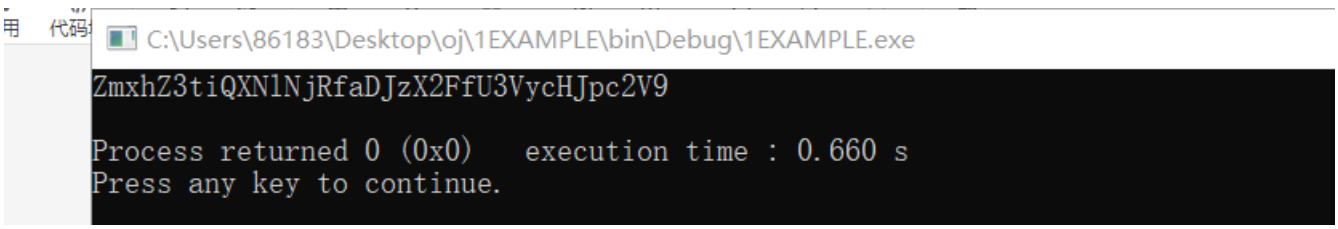
写出脚本

```c
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
int main()
{
    int i,j;
   char a[] = "zMXHz3TIgnxLxJhFAdtZn2fFk3lYCrtPC2l9";
  char flag[30];
   char b[] = "ABCDEFGHIJ"
              "KLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/";
   for(i = 0 ; i < strlen(a) ; i++)    //大小写互换
   {
       if(a[i] >= 'a' && a[i] <= 'z')
       {
           a[i] -= 32;
       }
       else if(a[i] >= 'A' && a[i] <= 'Z')
       {
           a[i] += 32;
       }

   }
   for(i = 0 ; i < strlen(a) ;i++)
   {
       if(a[i] >= 'A' && a[i] <= 'Z')   //只交换了大写字符
       {
           for(j = 6; j < 25 ; j++)
           {
               if(b[j] == a[i])
               {
                   if(j >= 6 && j < 15)
                        a[i] = b[j + 10];
                   else if(j >= 16 && j < 25)
                        a[i] = b[j - 10];
                    break;
               }
           }
       }
   }
  printf("%s\n",a);
   return 0;

}
```

运行得到结果 ZmxhZ3tiQXNlNjRfaDJzX2FfU3VycHJpc2V9



然后找个网站解密

flag: flag{bAse64_h2s_a_Surprise}