Writeup of Imageprc(reverse) in reversing.kr

原创

C0ss4ck 于 2018-02-19 18:27:52 发布 363 收藏

分类专栏： Reverse_of_CTF

Reverse_of_CTF 专栏收录该内容
7 篇文章 0 订阅
订阅专栏

做这道题……收获是……了解了几个WINAPI吧

0x00 Program Logic

首先运行程序，发现出现一个空白画板，用光标作图，再点击'Check'Button，弹窗Wrong
把程序扔进IDA，观察代码逻辑。首先看WinMain函数

```
int __stdcall WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nShowCmd)
{
  int v4; // ST14_4@1
  int v5; // eax@1
  HWND v6; // eax@1
  int result; // eax@3
  struct tagMSG Msg; // [sp+4h] [bp-44h]@1
  WNDCLASSA WndClass; // [sp+20h] [bp-28h]@1

  dword_4084D8 = (int)hInstance;
  WndClass.cbClsExtra = 0;
  WndClass.cbWndExtra = 0;
  WndClass.hbrBackground = (HBRUSH)GetStockObject(0);
  WndClass.hCursor = LoadCursorA(0, (LPCSTR)0x7F00);
  WndClass.hInstance = hInstance;
  WndClass.hIcon = LoadIconA(0, (LPCSTR)0x7F00);
  WndClass.lpfnWndProc = (WNDPROC)sub_401130;
  WndClass.lpszClassName = lpWindowName;
  WndClass.lpszMenuName = 0;
  WndClass.style = 3;
  RegisterClassA(&WndClass);
  v4 = GetSystemMetrics(1) / 2 - 75;
  v5 = GetSystemMetrics(0);
  v6 = CreateWindowExA(0, lpWindowName, lpWindowName, 0xCA0000u, v5 / 2 - 100, v4, 200, 150, 0, 0, hInstanc
  ShowWindow(v6, 5);
  if ( GetMessageA(&Msg, 0, 0, 0) )
  {
    do
    {
      TranslateMessage(&Msg);
      DispatchMessageA(&Msg);
    }
    while ( GetMessageA(&Msg, 0, 0, 0) );
    result = Msg.wParam;
  }
  else
  {
    result = Msg.wParam;
  }
  return result;
}
```

- 用GetStockObject选取NULL_BRUSH画刷样式填充主窗口背景色，
- 用LoadCursor选取IDC_ARROW(Standard arrow)作为光标样式，
- 用LoadIcon选取IDI_APPLICATION(Default application icon)作为所加载的图标资源，
- 随后lpfnWndProc来处理所接收到的键盘消息和鼠标消息(关键函数sub_401130)，
- lpszClassName描述窗口类名为lpWindowNmae，lpszMenuName设置菜单为NULL，style设置为3，
- 注册窗口；
- GetSystemMetrics(1)获取以像素为单位的计算机屏幕高度(SM_CYSCREEN)，GetSystemMetrics(0)获取以像素为单位的计算机屏幕宽度(SM_CXSCREEN)，其实也可以通过GetDeviceCaps获取计算机屏幕宽高，
- 用CreateWindowEx创建窗口，dwExStyle是WS_EX_LTRREADING(默认样式)，**dwStyle是0xCA0000(然而我并没有在MSDN上查到该数值对应的风格？)，高为150像素，宽为200像素，坐标......自己看吧**
- **ShowWindow**窗口展示；
- 随后**GetMessage**，再通过**TranslateMessage**和**DisPatchMessage**将用户的键盘消息和鼠标消息转发到过程函数**sub_401130**，在该函数内完成**judge**。

接下来看看**sub_401130**的伪代码

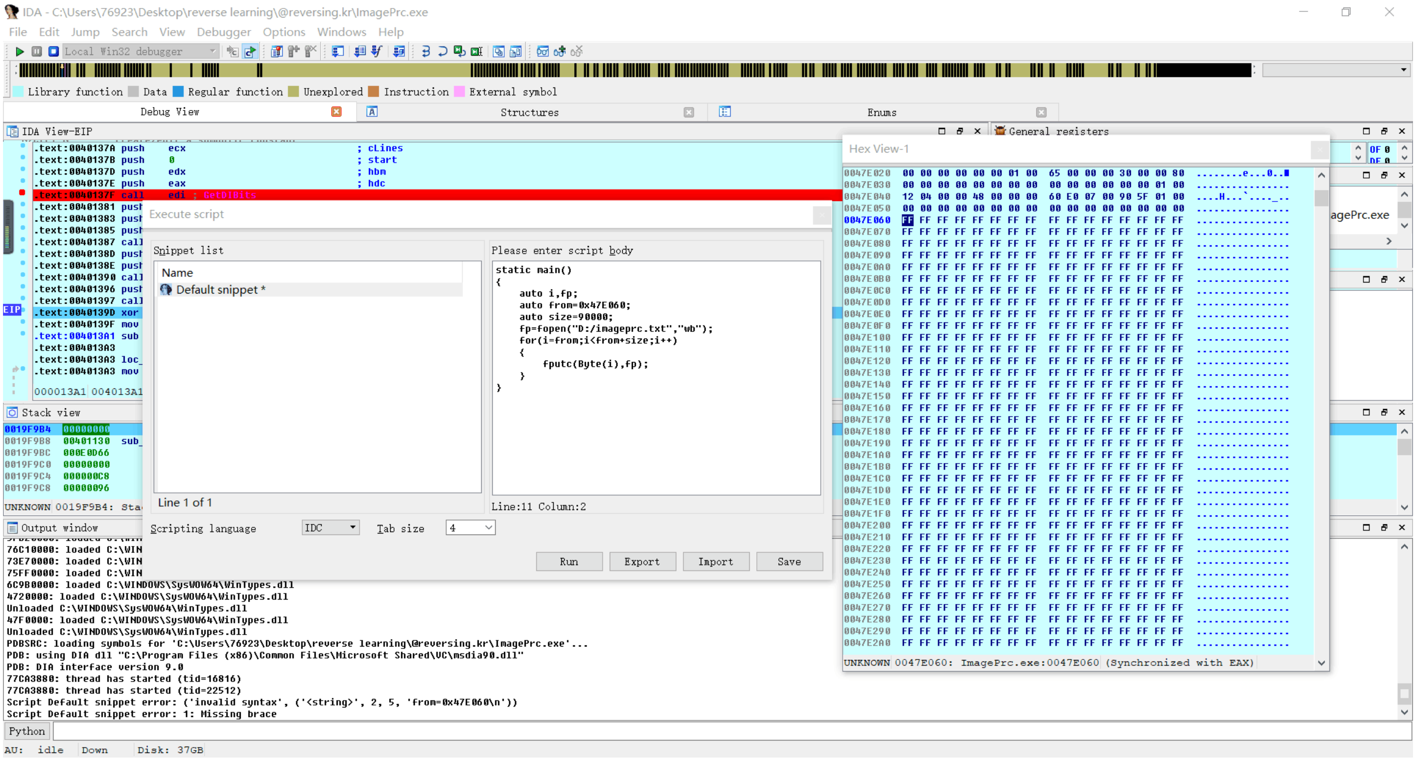
```
LRESULT __stdcall sub_401130(HWND hWnd, UINT Msg, WPARAM wParam, unsigned int lParam)
{
  HDC v4; // eax@6
  LRESULT result; // eax@6
  HGDIOBJ v6; // eax@7
  HDC v7; // esi@8
  void *v8; // esi@10
  HRSRC v9; // eax@10
  HGLOBAL v10; // eax@10
  _BYTE *v11; // eax@10
  signed int v12; // edi@10
  _BYTE *v13; // ecx@10
  int v14; // eax@10
  char pv; // [sp+8h] [bp-80h]@10
  LONG v16; // [sp+Ch] [bp-7Ch]@10
  UINT cLines; // [sp+10h] [bp-78h]@10
  struct tagBITMAPINFO bmi; // [sp+20h] [bp-68h]@6

  if ( Msg <= 0x111 )
  {
    if ( Msg != 273 )
    {
      if ( Msg == 1 )
      {
        v7 = GetDC(hWnd);
        hbm = CreateCompatibleBitmap(v7, 200, 150);
        hdc = CreateCompatibleDC(v7);
        h = SelectObject(hdc, hbm);
        Rectangle(hdc, -5, -5, 205, 205);
        ReleaseDC(hWnd, v7);
        ::wParam = (WPARAM)CreateFontA(12, 0, 0, 0, 400, 0, 0, 0, 0x81u, 0, 0, 0, 0x12u, pszFaceName);
        dword_4084E0 = (int)CreateWindowExA(
                              0,
                              ClassName,
                              WindowName,
                              0x50000000u,
                              60,
                              85,
                              80,
                              28,
                              hWnd,
                              (HMENU)0x64,
                              hInstance,
                              0);
        SendMessageA((HWND)dword_4084E0, 0x30u, ::wParam, 0);
        return 0;
      }
      if ( Msg == 2 )
      {
        v6 = SelectObject(hdc, h);
        DeleteObject(v6);
        DeleteDC(hdc);
        PostQuitMessage(0);
        return 0;
      }
      if ( Msg == 15 )
      {
        v4 = BeginPaint(hWnd, (LPPAINTSTRUCT)bmi.bmiColors);
        BitBlt(v4, 0, 0, 200, 150, hdc, 0, 0, 0xCC0020u);
        EndPaint(hWnd, (const PAINTSTRUCT *)bmi.bmiColors);
        return 0;
      }
      return DefWindowProcA(hWnd, Msg, wParam, lParam);
    }
    if ( wParam == 100 )
    {
      GetObjectA(hbm, 24, &pv);
      memset(&bmi, 0, 0x28u);
      bmi.bmiHeader.biHeight = cLines;
      bmi.bmiHeader.biWidth = v16;
      bmi.bmiHeader.biSize = 40;
      bmi.bmiHeader.biPlanes = 1;
      bmi.bmiHeader.biBitCount = 24;
      bmi.bmiHeader.biCompression = 0;
      GetDIBits(hdc, (HBITMAP)hbm, 0, cLines, 0, &bmi, 0);
      v8 = (void *)sub_40150B(bmi.bmiHeader.biSizeImage);
      GetDIBits(hdc, (HBITMAP)hbm, 0, cLines, v8, &bmi, 0);
      v9 = FindResourceA(0, (LPCSTR)0x65, (LPCSTR)0x18);
      v10 = LoadResource(0, v9);
      v11 = LockResource(v10);
      v12 = 0;
      v13 = v8;
      v14 = v11 - (_BYTE *)v8;
      while ( *v13 == v13[v14] )
      {
        ++v12;
        ++v13;
        if ( v12 >= 90000 )
        {
          sub_401500(v8);
          return 0;
        }
      }
      MessageBoxA(hWnd, Text, Caption, 0x30u);
      sub_401500(v8);
      return 0;
    }
    return 0;
  }
  if ( Msg == 512 )
  {
    if ( dword_47D7F8 )
    {
      MoveToEx(hdc, x, y, 0);
      LineTo(hdc, (unsigned __int16)lParam, lParam >> 16);
      x = (unsigned __int16)lParam;
      y = lParam >> 16;
      InvalidateRect(hWnd, 0, 0);
    }
    return 0;
  }
  if ( Msg == 513 )
  {
    dword_47D7F8 = 1;
    y = lParam >> 16;
    x = (unsigned __int16)lParam;
    result = 0;
  }
  }
  else
  {
    if ( Msg != 514 )
      return DefWindowProcA(hWnd, Msg, wParam, lParam);
    dword_47D7F8 = 0;
    result = 0;
  }
  return result;
}
```

- GetDC指定hWnd为句柄，以后便可在GDI函数中用该句柄上下文环境中绘图，随后CreateCompatibleBitmap创建宽200高150的位图，CreateCompatibleDC创建上下文环境，然后那么一大通API就是用来让你在这个位图里绘图的OTZ(API太多了我已经不想解释了);
- 一通API之后就开始Judge，首先是FindResource，然后是LoadResource，最后开始把Resource里的内容和画上去的内容逐像素点比较(拆成RGB是90000次Judge);

以上......就是程序逻辑......

0x01 Dump

开始动态调试......在LoadResource后把90000个数据用IDC给Dump下来，脚本及内存截图如下



```
.text:0040137A push    ecx                         ; cLines
.text:0040137B push    0                           ; start
.text:0040137D push    edx                         ; hbm
.text:0040137E push    eax                         ; hdc
.text:0040137F call    edi ; GetDIBits
.text:00401381 push
.text:00401383 push
.text:00401385 push
.text:00401387 call
.text:0040138D push
.text:0040138E push
.text:00401390 call
.text:00401396 push
.text:00401397 call
.text:0040139D xor
.text:0040139F mov
.text:004013A1 sub
.text:004013A3
.text:004013A3 loc_
.text:004013A3 mov
000013A1 004013A1
```

```
static main()
{
    auto i,fp;
    auto from=0x47E060;
    auto size=90000;
    fp=fopen("D:/imageprc.txt","wb");
    for(i=from;i<from+size;i++)
    {
        fputc(Byte(i),fp);
    }
}
```

```
0019F9B4  00000000
0019F9B8  00401130  sub_
0019F9BC  000E0D66
0019F9C0  00000000
0019F9C4  000000C8
0019F9C8  00000096
UNKNOWN 0019F9B4: Sta
```

```
0047E020  00 00 00 00 00 00 01 00  65 00 00 00 30 00 00 80  ........e...0...
0047E030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 01 00  ................
0047E040  12 04 00 00 48 00 00 00  60 E0 07 00 90 5F 01 00  ....H...`...._..
0047E050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0047E060  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E070  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E080  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E090  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E0A0  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E0B0  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E0C0  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E0D0  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E0E0  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E0F0  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E100  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E110  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E120  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E130  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E140  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E150  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E160  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E170  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E180  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E190  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E1A0  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E1B0  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E1C0  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E1D0  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E1E0  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E1F0  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E200  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E210  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E220  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E230  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E240  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E250  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E260  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E270  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E280  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E290  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
0047E2A0  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ................
UNKNOWN 0047E060: ImagePrc.exe:0047E060 (Synchronized with EAX)
```

```
76C10000: loaded C:\WIN
73E70000: loaded C:\WIN
75FF0000: loaded C:\WIN
6C9B0000: loaded C:\WINDOWS\SysWOW64\WinTypes.dll
4720000: loaded C:\WINDOWS\SysWOW64\WinTypes.dll
Unloaded C:\WINDOWS\SysWOW64\WinTypes.dll
47F0000: loaded C:\WINDOWS\SysWOW64\WinTypes.dll
Unloaded C:\WINDOWS\SysWOW64\WinTypes.dll
PDBSRC: loading symbols for 'C:\Users\76923\Desktop\reverse learning\@reversing.kr\ImagePrc.exe'...
PDB: using DIA dll "C:\Program Files (x86)\Common Files\Microsoft Shared\VC\msdia90.dll"
PDB: DIA interface version 9.0
77CA3880: thread has started (tid=16816)
77CA3880: thread has started (tid=22512)
Script Default snippet error: ('invalid syntax', ('<string>', 2, 5, 'from=0x47E060\n'))
Script Default snippet error: 1: Missing brace
```

Run之后90000个bytes就被写到了文件里；

0x02 Restore bmp

不会PIL的哭了......
写了好久......还去查了官方文档......

```
from PIL import Image

width=200
height=150

f=open('imageprc.txt','rb')
data=f.read()
p=Image.frombytes('RGB',(width,height),data)
p=p.transpose(Image.FLIP_TOP_BOTTOM)
p.show()
```

那个FLIP_TOP_BOTTOM真是……后来在其他大师傅的脚本上才看到……加了上去……完善了一下

于是得到一张图……写着GOT(可是按道理不用TOP_BOTTOM应该能得到正确的GOT图啊？这里是一个疑问……也许和Resource加载的顺序有关？)

得到flag，即GOT