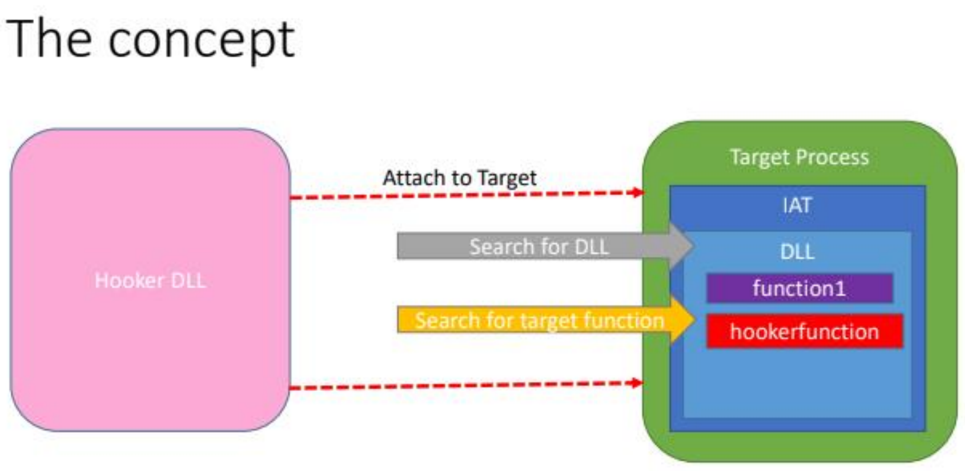
# Hooking the IAT

In this section, we will learn about Hooking the IAT and replacing any function with your function.



On the left, we have the hooker dll, and on the right, we have the Target process.

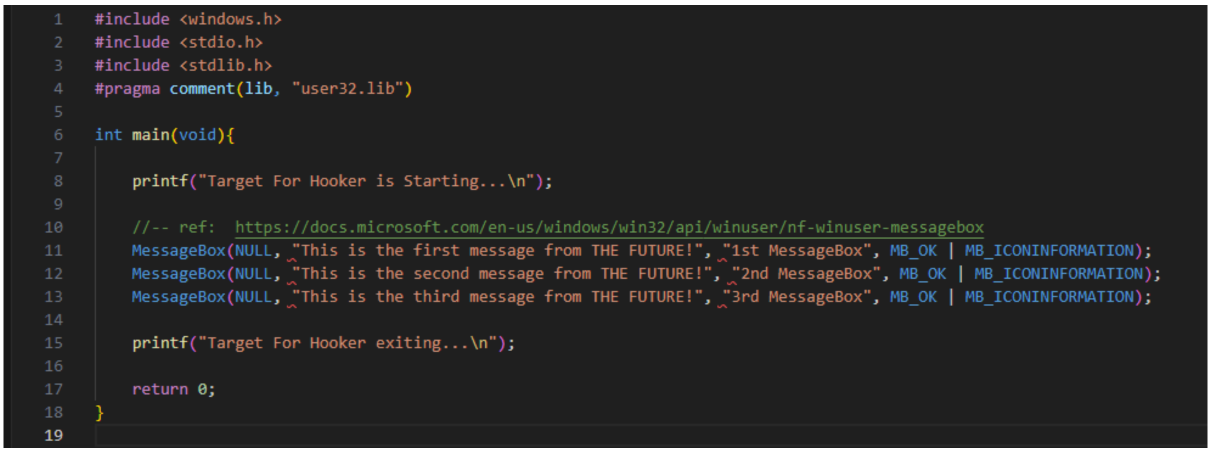
So, the hooker dll will look for a process to attach, and once it is attached, it will execute some hooking code.

Here the target process has an IAT(Import Address Table) and has all the PE executable files. Inside the IAT, there will be a list of DLLs, which are loaded by the target process.

The hooker dll will search for a specific dll, which it is interested in to find a specific function, and then it looks for a specific function, then it will replace a function with its function(Here it has replaced the function2 to hooker-function). So, this is the concept of IAT hooking.

So, now let’s go through the IAT hooking code:

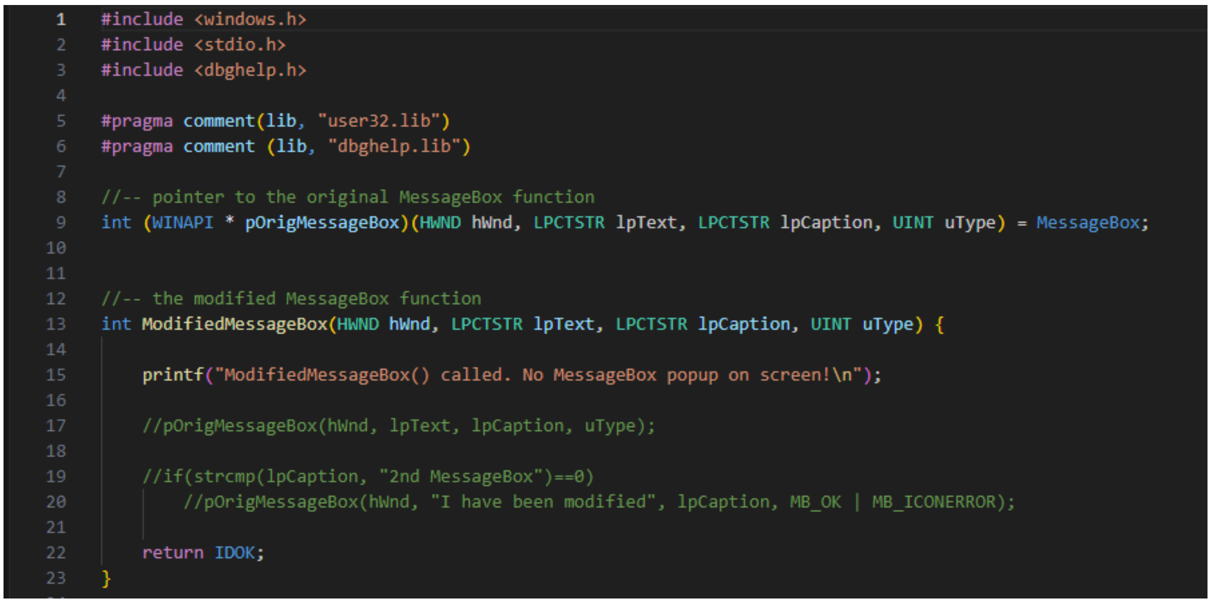
The target process looks like this:



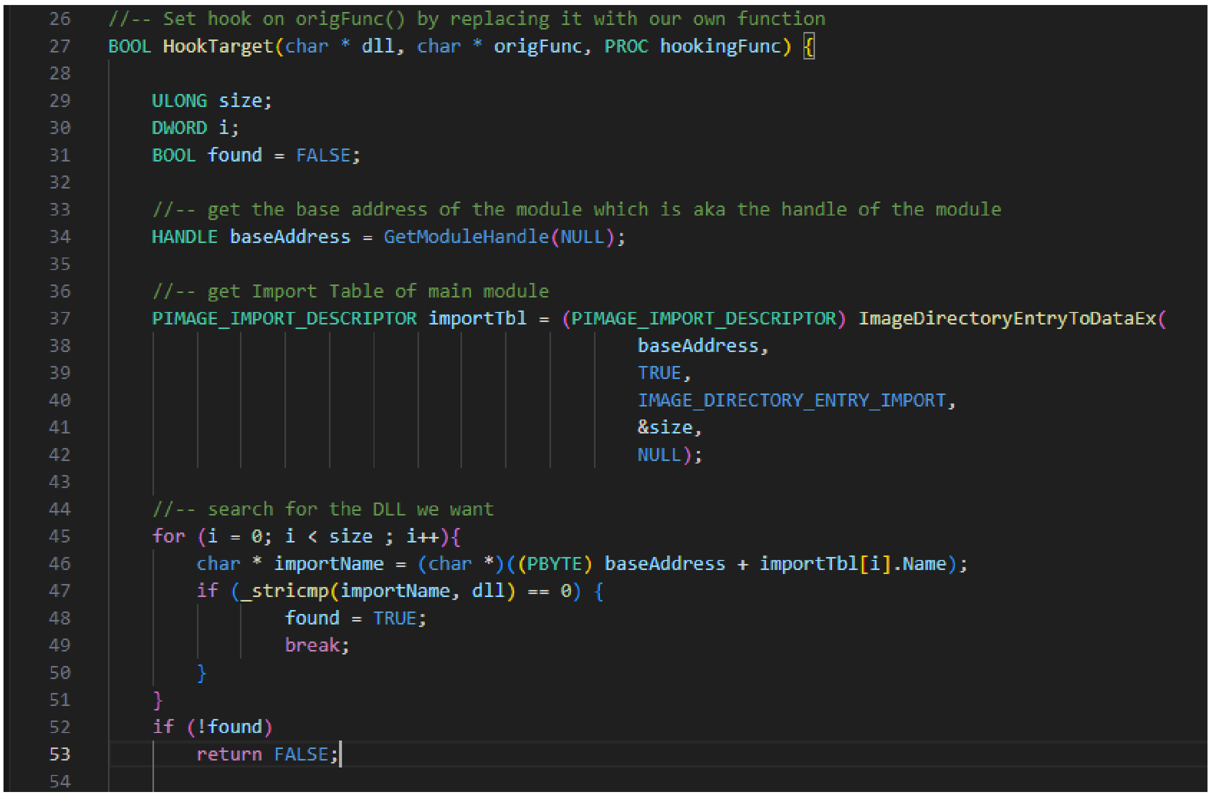
```c
#include <windows.h>
#include <stdio.h>
#include <stdlib.h>
#pragma comment(lib, "user32.lib")

int main(void){

    printf("Target For Hooker is Starting...\n");

    //-- ref:  https://docs.microsoft.com/en-us/windows/win32/api/winuser/nf-winuser-messagebox
    MessageBox(NULL, "This is the first message from THE FUTURE!", "1st MessageBox", MB_OK | MB_ICONINFORMATION);
    MessageBox(NULL, "This is the second message from THE FUTURE!", "2nd MessageBox", MB_OK | MB_ICONINFORMATION);
    MessageBox(NULL, "This is the third message from THE FUTURE!", "3rd MessageBox", MB_OK | MB_ICONINFORMATION);

    printf("Target For Hooker exiting...\n");

    return 0;
}
```

It is the same code, which we used earlier.

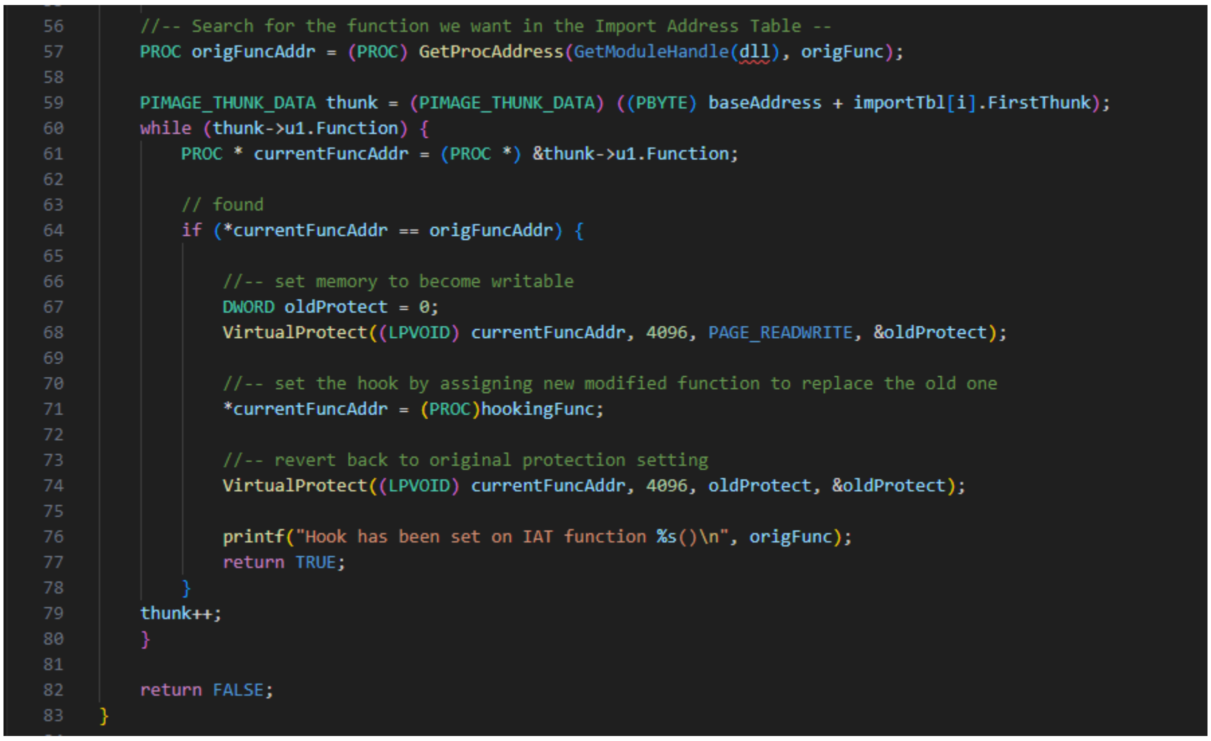
Here’s the code of the IAT hooker:



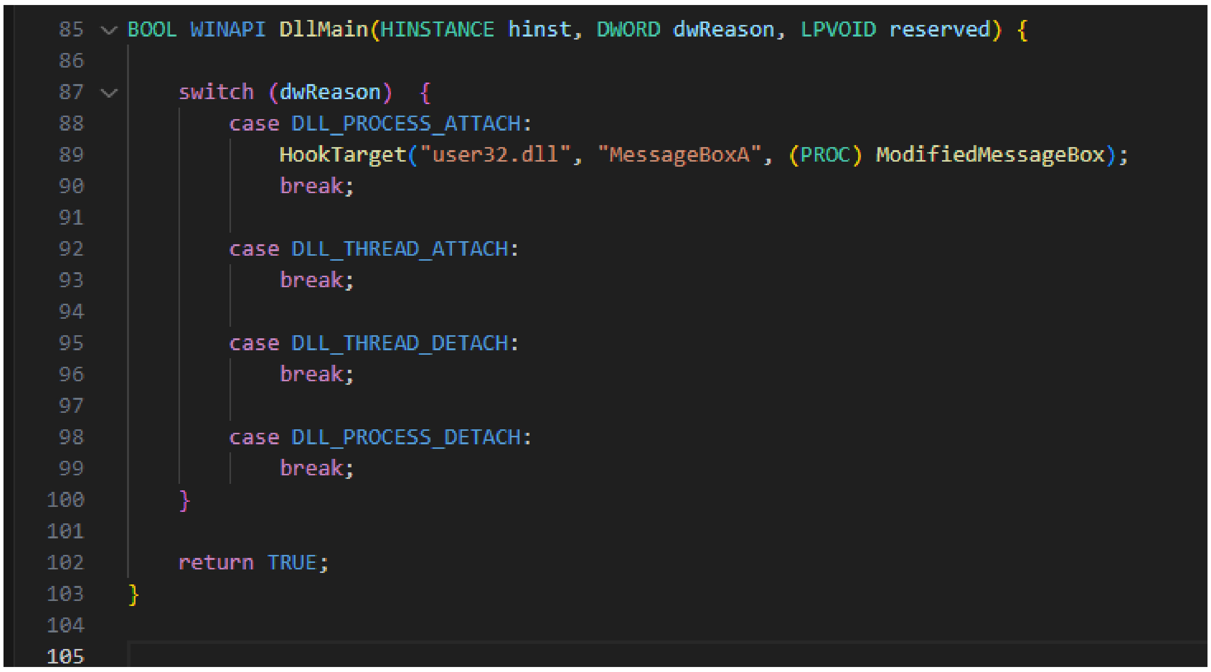
```c
#include <windows.h>
#include <stdio.h>
#include <dbghelp.h>

#pragma comment(lib, "user32.lib")
#pragma comment (lib, "dbghelp.lib")

//-- pointer to the original MessageBox function
int (WINAPI * pOrigMessageBox)(HWND hWnd, LPCTSTR lpText, LPCTSTR lpCaption, UINT uType) = MessageBox;


//-- the modified MessageBox function
int ModifiedMessageBox(HWND hWnd, LPCTSTR lpText, LPCTSTR lpCaption, UINT uType) {

    printf("ModifiedMessageBox() called. No MessageBox popup on screen!\n");

    //pOrigMessageBox(hWnd, lpText, lpCaption, uType);

    //if(strcmp(lpCaption, "2nd MessageBox")==0)
        //pOrigMessageBox(hWnd, "I have been modified", lpCaption, MB_OK | MB_ICONERROR);

    return IDOK;
}
```



```c
//-- Set hook on origFunc() by replacing it with our own function
BOOL HookTarget(char * dll, char * origFunc, PROC hookingFunc) {

    ULONG size;
    DWORD i;
    BOOL found = FALSE;

    //-- get the base address of the module which is aka the handle of the module
    HANDLE baseAddress = GetModuleHandle(NULL);

    //-- get Import Table of main module
    PIMAGE_IMPORT_DESCRIPTOR importTbl = (PIMAGE_IMPORT_DESCRIPTOR) ImageDirectoryEntryToDataEx(
                                                baseAddress,
                                                TRUE,
                                                IMAGE_DIRECTORY_ENTRY_IMPORT,
                                                &size,
                                                NULL);

    //-- search for the DLL we want
    for (i = 0; i < size ; i++){
        char * importName = (char *)((PBYTE) baseAddress + importTbl[i].Name);
        if (_stricmp(importName, dll) == 0) {
                found = TRUE;
                break;
        }
    }
    if (!found)
        return FALSE;

```



```c
    //-- Search for the function we want in the Import Address Table --
    PROC origFuncAddr = (PROC) GetProcAddress(GetModuleHandle(dll), origFunc);

    PIMAGE_THUNK_DATA thunk = (PIMAGE_THUNK_DATA) ((PBYTE) baseAddress + importTbl[i].FirstThunk);
    while (thunk->u1.Function) {
        PROC * currentFuncAddr = (PROC *) &thunk->u1.Function;

        // found
        if (*currentFuncAddr == origFuncAddr) {

            //-- set memory to become writable
            DWORD oldProtect = 0;
            VirtualProtect((LPVOID) currentFuncAddr, 4096, PAGE_READWRITE, &oldProtect);

            //-- set the hook by assigning new modified function to replace the old one
            *currentFuncAddr = (PROC)hookingFunc;

            //-- revert back to original protection setting
            VirtualProtect((LPVOID) currentFuncAddr, 4096, oldProtect, &oldProtect);

            printf("Hook has been set on IAT function %s()\n", origFunc);
            return TRUE;
        }
    thunk++;
    }

    return FALSE;
}
```



```
BOOL WINAPI DllMain(HINSTANCE hinst, DWORD dwReason, LPVOID reserved) {

    switch (dwReason)  {
        case DLL_PROCESS_ATTACH:
            HookTarget("user32.dll", "MessageBoxA", (PROC) ModifiedMessageBox);
            break;

        case DLL_THREAD_ATTACH:
            break;

        case DLL_THREAD_DETACH:
            break;

        case DLL_PROCESS_DETACH:
            break;
    }

    return TRUE;
}

```

The IAT hooker has got the same structure, we have got the DllMain function, where the dll starts when it is attached to a process.

So, first, we will call the DLL_PROCESS_ATTACH function, and then we will execute this function HookTarget, passing three parameters:

1. The dll which contains the function(Here user32.dll, because it contains the message box)
2. Then the function(Here MessageBoxA)
3. Then the replaced function(Here it is ModifiedMessageBox)

In the code, we can see that we have defined the function pointer, as we did last time. It is a pointer to the original Message Box Function.

The ModifiedMessageBox function is also similar to the previous code.

Now, in the HookTarget function:

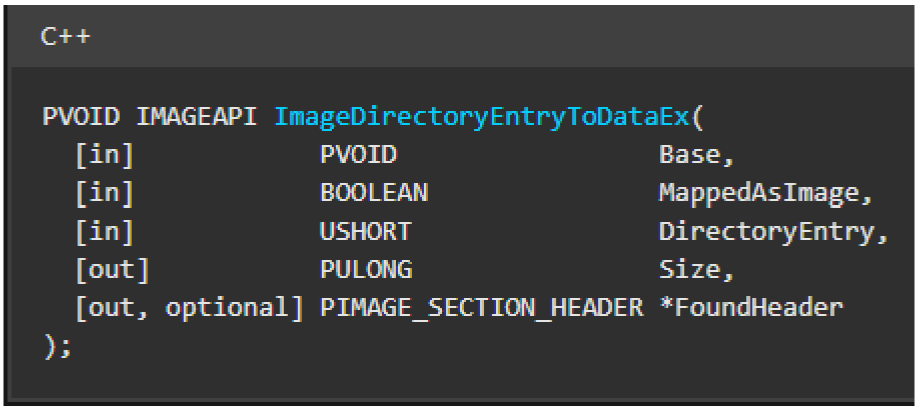
Here we send a hook to the original function and then replace it with our function. This hook function is called on the attachment of the function by the dll, and once the dll is attached to the target process this function is called, and we will receive three parameters: user32.dll, MessageBoxA, ModifiedMessageBox.

In the HookTarget function, we can see that we get the base address of the module, which is also known as the handle of the module, and it refers to the target process's base address. So, we save it to a HANDLE.

Then we get the Import Address Table, and we use a shortcut way to get the IAT, by using the ImageDiretoryEntryToDataEx function.

ImageDiretoryEntryToDataEx:

Locates a directory entry within the image header and returns the address of the data for the directory entry. This function returns the section header for the data located if one exists.



```C++
PVOID IMAGEAPI ImageDirectoryEntryToDataEx(
  [in]            PVOID                 Base,
  [in]            BOOLEAN               MappedAsImage,
  [in]            USHORT                DirectoryEntry,
  [out]           PULONG                Size,
  [out, optional] PIMAGE_SECTION_HEADER *FoundHeader
);
```

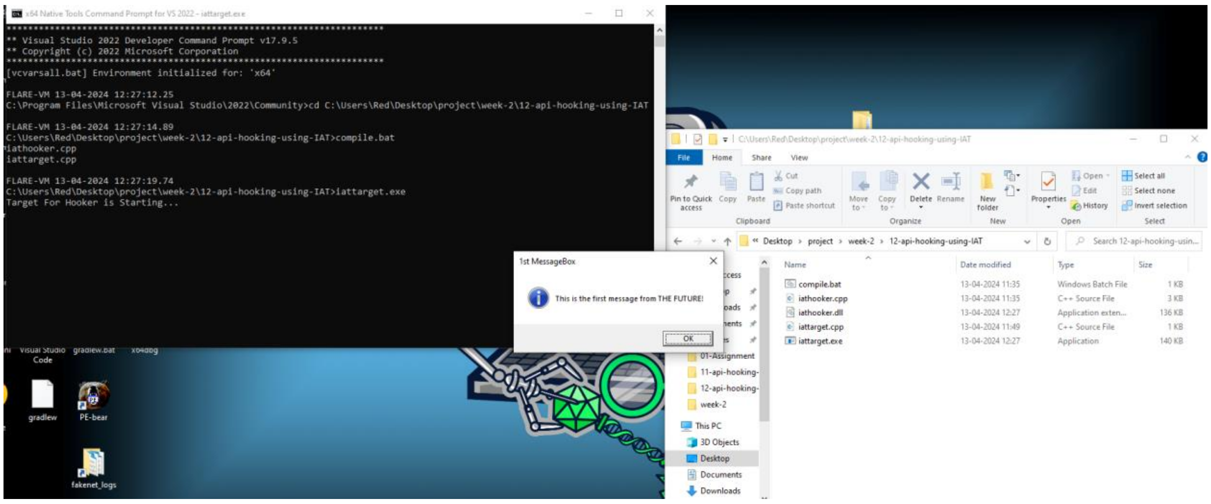
The 1st parameter is the base address, and the 2nd parameter refers to whether we want to map it to an image or not, and we put it as true in this case. The 3rd parameter is the type of DirectoryEntry that we want to retrieve, so in this case we want the Import Directory Table, and then the 4th parameter is the size, which is already declared. Once we get the import table, we will save it to a variable importTbl.

Then it will search for the dll it wants, so in this case, it will look for a particular dll, user32.dll, so here we iterate through the whole importTbl and find the user32.dll

Once it has found the dll, it will look for the function locally, and searches for every function in the dll, and once it finds it will change the protection to readable and writable because we want to substitute it with our function, so we use VirtualProtect, and then we hook the function by assigning our hooking function, to replace on that is formed.

Then the hooking function is passed through the ModifiedMessageBox function, so at this point, the hook is set, then we perform whatever's inside the ModifiedMessageBox function. Then we will revert to the original protection setting, so once again we use VirtualProtect.

Now, let's compile and run the file:



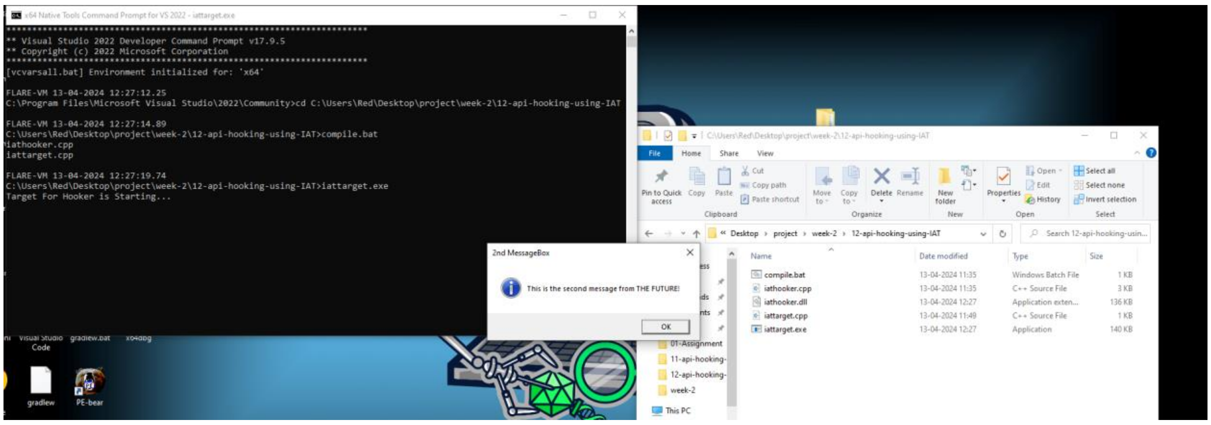
```
**********************************************************************
** Visual Studio 2022 Developer Command Prompt v17.9.5
** Copyright (c) 2022 Microsoft Corporation
**********************************************************************
[vcvarsall.bat] Environment initialized for: 'x64'

FLARE-VM 13-04-2024 12:27:12.25
C:\Program Files\Microsoft Visual Studio\2022\Community>cd C:\Users\Red\Desktop\project\week-2\12-api-hooking-using-IAT

FLARE-VM 13-04-2024 12:27:14.89
C:\Users\Red\Desktop\project\week-2\12-api-hooking-using-IAT>compile.bat
iathooker.cpp
iattarget.cpp

FLARE-VM 13-04-2024 12:27:19.74
C:\Users\Red\Desktop\project\week-2\12-api-hooking-using-IAT>iattarget.exe
Target For Hooker is Starting...
```

1st MessageBox

This is the first message from THE FUTURE!

OK

```
Name              Date modified      Type                    Size
compile.bat       13-04-2024 11:35   Windows Batch File      1 KB
iathooker.cpp     13-04-2024 11:35   C++ Source File         3 KB
iathooker.dll     13-04-2024 12:27   Application exten...    136 KB
iattarget.cpp     13-04-2024 11:49   C++ Source File         1 KB
iattarget.exe     13-04-2024 12:27   Application             140 KB
```



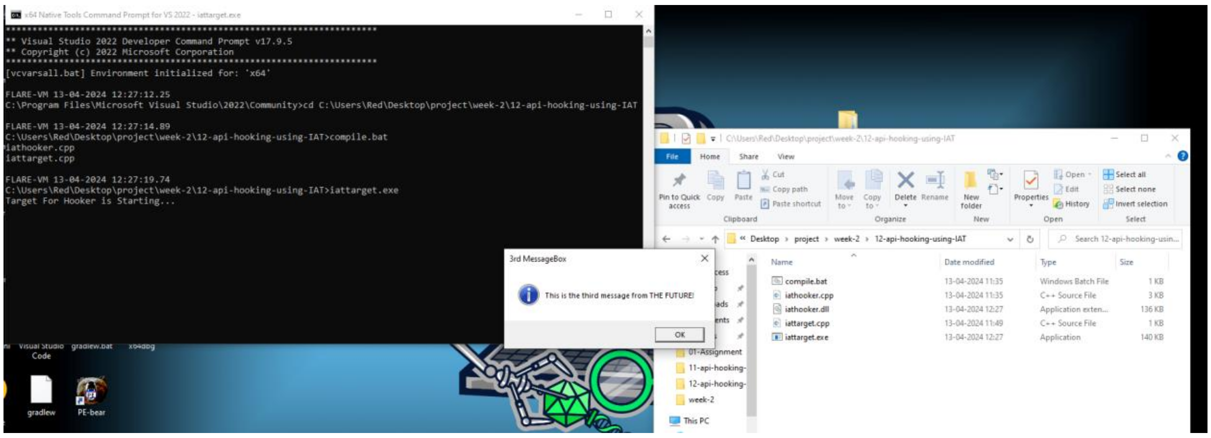
```
**********************************************************************
** Visual Studio 2022 Developer Command Prompt v17.9.5
** Copyright (c) 2022 Microsoft Corporation
**********************************************************************
[vcvarsall.bat] Environment initialized for: 'x64'

FLARE-VM 13-04-2024 12:27:12.25
C:\Program Files\Microsoft Visual Studio\2022\Community>cd C:\Users\Red\Desktop\project\week-2\12-api-hooking-using-IAT

FLARE-VM 13-04-2024 12:27:14.89
C:\Users\Red\Desktop\project\week-2\12-api-hooking-using-IAT>compile.bat
iathooker.cpp
iattarget.cpp

FLARE-VM 13-04-2024 12:27:19.74
C:\Users\Red\Desktop\project\week-2\12-api-hooking-using-IAT>iattarget.exe
Target For Hooker is Starting...
```

2nd MessageBox

This is the second message from THE FUTURE!

OK

```
Name              Date modified      Type                    Size
compile.bat       13-04-2024 11:35   Windows Batch File      1 KB
iathooker.cpp     13-04-2024 11:35   C++ Source File         3 KB
iathooker.dll     13-04-2024 12:27   Application exten...    136 KB
iattarget.cpp     13-04-2024 11:49   C++ Source File         1 KB
iattarget.exe     13-04-2024 12:27   Application             140 KB
```



```
**********************************************************************
** Visual Studio 2022 Developer Command Prompt v17.9.5
** Copyright (c) 2022 Microsoft Corporation
**********************************************************************
[vcvarsall.bat] Environment initialized for: 'x64'

FLARE-VM 13-04-2024 12:27:12.25
C:\Program Files\Microsoft Visual Studio\2022\Community>cd C:\Users\Red\Desktop\project\week-2\12-api-hooking-using-IAT

FLARE-VM 13-04-2024 12:27:14.89
C:\Users\Red\Desktop\project\week-2\12-api-hooking-using-IAT>compile.bat
iathooker.cpp
iattarget.cpp

FLARE-VM 13-04-2024 12:27:19.74
C:\Users\Red\Desktop\project\week-2\12-api-hooking-using-IAT>iattarget.exe
Target For Hooker is Starting...
```

3rd MessageBox

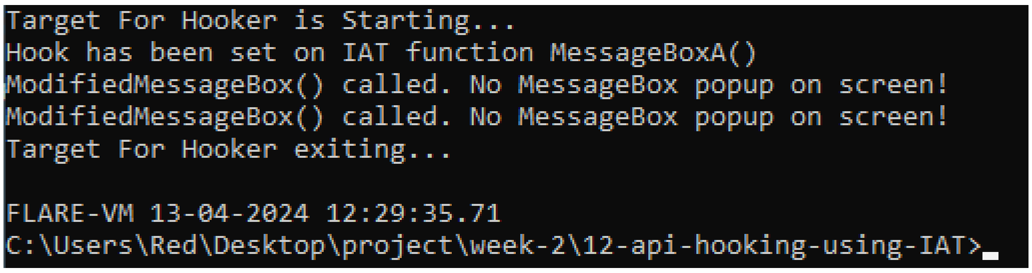
This is the third message from THE FUTURE!

OK

```
Name              Date modified      Type                    Size
compile.bat       13-04-2024 11:35   Windows Batch File      1 KB
iathooker.cpp     13-04-2024 11:35   C++ Source File         3 KB
iathooker.dll     13-04-2024 12:27   Application exten...    136 KB
iattarget.cpp     13-04-2024 11:49   C++ Source File         1 KB
iattarget.exe     13-04-2024 12:27   Application             140 KB
```

We can see that the .exe file is working properly, so now we will inject the .dll file using the Process hacker:

Once we inject the .dll file and press ok, it exits the process, and prints some line in the command prompt, and it doesn't even give another pop-up.

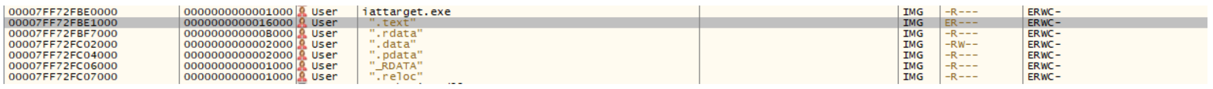


```
Target For Hooker is Starting...
Hook has been set on IAT function MessageBoxA()
ModifiedMessageBox() called. No MessageBox popup on screen!
ModifiedMessageBox() called. No MessageBox popup on screen!
Target For Hooker exiting...

FLARE-VM 13-04-2024 12:29:35.71
C:\Users\Red\Desktop\project\week-2\12-api-hooking-using-IAT>
```

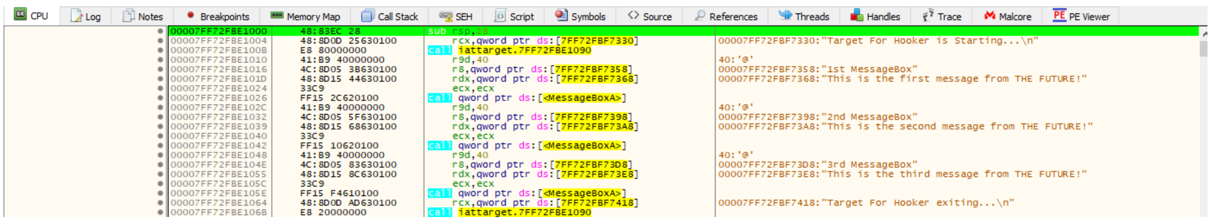
So, if want to change the message box, or see all the message boxes, then just change the code, like we had done the last time.

Now, let's reverse engineer it, we will be using xdbg, then inject the dll file:

And now attach a file, then synchronize it, then go to memory map, in the iatarget.exe, go to the .txt section:



```
00007FF72FBE0000  0000000000001000  User  iattarget.exe   IMG  -R---  ERWC-
00007FF72FBE1000  0000000000016000  User   ".text"        IMG  ER---  ERWC-
00007FF72FBF7000  000000000000B000  User   ".rdata"       IMG  -R---  ERWC-
00007FF72FC02000  0000000000002000  User   ".data"        IMG  -RW--  ERWC-
00007FF72FC04000  0000000000002000  User   ".pdata"       IMG  -R---  ERWC-
00007FF72FC06000  0000000000001000  User   "_RDATA"       IMG  -R---  ERWC-
00007FF72FC07000  0000000000001000  User   ".reloc"       IMG  -R---  ERWC-
```

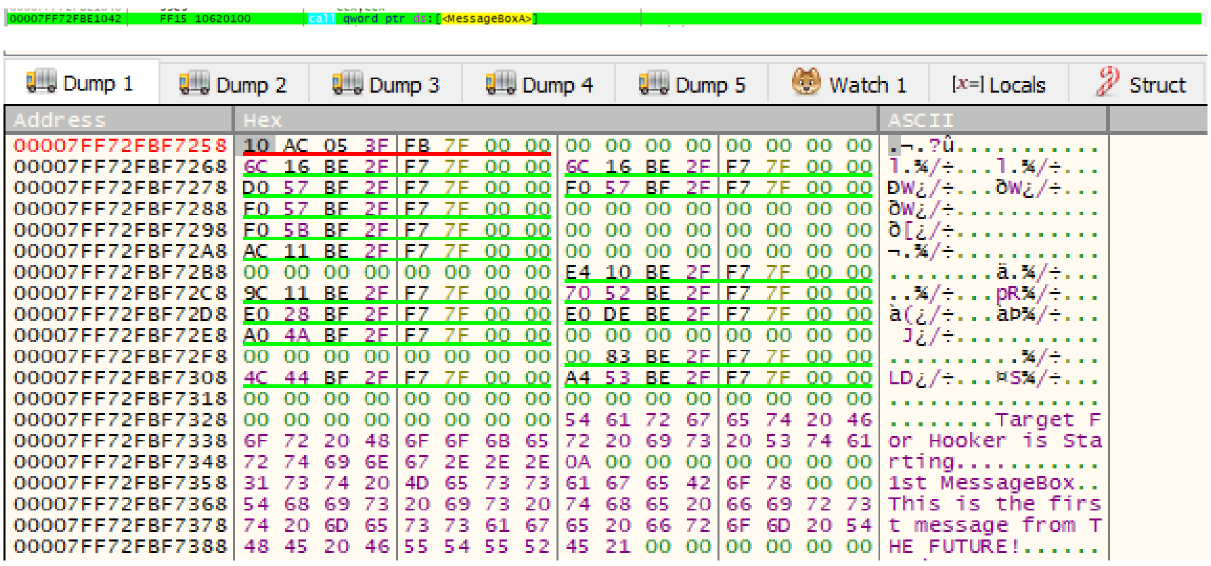


```
00007FF72FBE1000  48:83EC 28          sub rsp,28
00007FF72FBE1004  48:8D0D 25630100        rcx,qword ptr ds:[7FF72FBF7330]      00007FF72FBF7330:"Target For Hooker is Starting...\n"
00007FF72FBE100B  E8 80000000         call iattarget.7FF72FBE1090
00007FF72FBE1010  41:B9 40000000          r9d,40                               40:'@'
00007FF72FBE1016  4C:8D05 3B630100        r8,qword ptr ds:[7FF72FBF7358]       00007FF72FBF7358:"1st MessageBox"
00007FF72FBE101D  48:8D15 44630100        rdx,qword ptr ds:[7FF72FBF7368]      00007FF72FBF7368:"This is the first message from THE FUTURE!"
00007FF72FBE1024  33C9                    ecx,ecx
00007FF72FBE1026  FF15 2C620100       call qword ptr ds:[<MessageBoxA>]
00007FF72FBE102C  41:B9 40000000          r9d,40                               40:'@'
00007FF72FBE1032  4C:8D05 5F630100        r8,qword ptr ds:[7FF72FBF7398]       00007FF72FBF7398:"2nd MessageBox"
00007FF72FBE1039  48:8D15 68630100        rdx,qword ptr ds:[7FF72FBF73A8]      00007FF72FBF73A8:"This is the second message from THE FUTURE!"
00007FF72FBE1040  33C9                    ecx,ecx
00007FF72FBE1042  FF15 10620100       call qword ptr ds:[<MessageBoxA>]
00007FF72FBE1048  41:B9 40000000          r9d,40                               40:'@'
00007FF72FBE104E  4C:8D05 83630100        r8,qword ptr ds:[7FF72FBF73D8]       00007FF72FBF73D8:"3rd MessageBox"
00007FF72FBE1055  48:8D15 8C630100        rdx,qword ptr ds:[7FF72FBF73E8]      00007FF72FBF73E8:"This is the third message from THE FUTURE!"
00007FF72FBE105C  33C9                    ecx,ecx
00007FF72FBE105E  FF15 F4610100       call qword ptr ds:[<MessageBoxA>]
00007FF72FBE1064  48:8D0D AD630100        rcx,qword ptr ds:[7FF72FBF7418]      00007FF72FBF7418:"Target For Hooker exiting...\n"
00007FF72FBE106B  E8 20000000         call iattarget.7FF72FBE1090
```

We can see the three-message box here in the .txt section.

So, according to our code, it is going to intercept the 2nd message box and stop it from popping up.

So, follow this address in the dump:

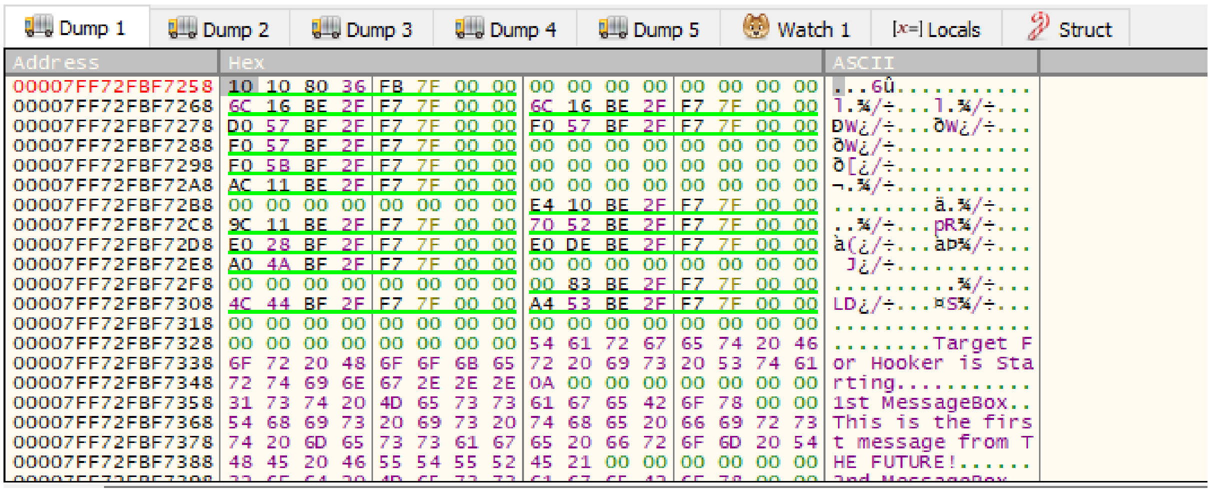


```
00007FF72FBE1042  FF15 10620100   call qword ptr ds:[<MessageBoxA>]
```

```
Address           Hex                                               ASCII
00007FF72FBF7258  10 AC 05 3F FB 7F 00 00 00 00 00 00 00 00 00 00  .¬.?û...........
00007FF72FBF7268  6C 16 BE 2F F7 7F 00 00 6C 16 BE 2F F7 7F 00 00  l.¾/÷...l.¾/÷...
00007FF72FBF7278  D0 57 BF 2F F7 7F 00 00 F0 57 BF 2F F7 7F 00 00  ÐW¿/÷...ðW¿/÷...
00007FF72FBF7288  F0 57 BF 2F F7 7F 00 00 00 00 00 00 00 00 00 00  ðW¿/÷...........
00007FF72FBF7298  F0 5B BF 2F F7 7F 00 00 00 00 00 00 00 00 00 00  ð[¿/÷...........
00007FF72FBF72A8  AC 11 BE 2F F7 7F 00 00 00 00 00 00 00 00 00 00  ¬.¾/÷...........
00007FF72FBF72B8  00 00 00 00 00 00 00 00 E4 10 BE 2F F7 7F 00 00  ........ä.¾/÷...
00007FF72FBF72C8  9C 11 BE 2F F7 7F 00 00 70 52 BE 2F F7 7F 00 00  ..¾/÷...pR¾/÷...
00007FF72FBF72D8  E0 28 BF 2F F7 7F 00 00 E0 DE BE 2F F7 7F 00 00  à(¿/÷...àÞ¾/÷...
00007FF72FBF72E8  A0 4A BF 2F F7 7F 00 00 00 00 00 00 00 00 00 00   J¿/÷...........
00007FF72FBF72F8  00 00 00 00 00 00 00 00 00 83 BE 2F F7 7F 00 00  ..........¾/÷...
00007FF72FBF7308  4C 44 BF 2F F7 7F 00 00 A4 53 BE 2F F7 7F 00 00  LD¿/÷...¤S¾/÷...
00007FF72FBF7318  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00007FF72FBF7328  00 00 00 00 00 00 00 00 54 61 72 67 65 74 20 46  ........Target F
00007FF72FBF7338  6F 72 20 48 6F 6F 6B 65 72 20 69 73 20 53 74 61  or Hooker is Sta
00007FF72FBF7348  72 74 69 6E 67 2E 2E 2E 0A 00 00 00 00 00 00 00  rting...........
00007FF72FBF7358  31 73 74 20 4D 65 73 73 61 67 65 42 6F 78 00 00  1st MessageBox..
00007FF72FBF7368  54 68 69 73 20 69 73 20 74 68 65 20 66 69 72 73  This is the firs
00007FF72FBF7378  74 20 6D 65 73 73 61 67 65 20 66 72 6F 6D 20 54  t message from T
00007FF72FBF7388  48 45 20 46 55 54 55 52 45 21 00 00 00 00 00 00  HE FUTURE!......
```

So, here it contains the address of the function MessageBoxA.

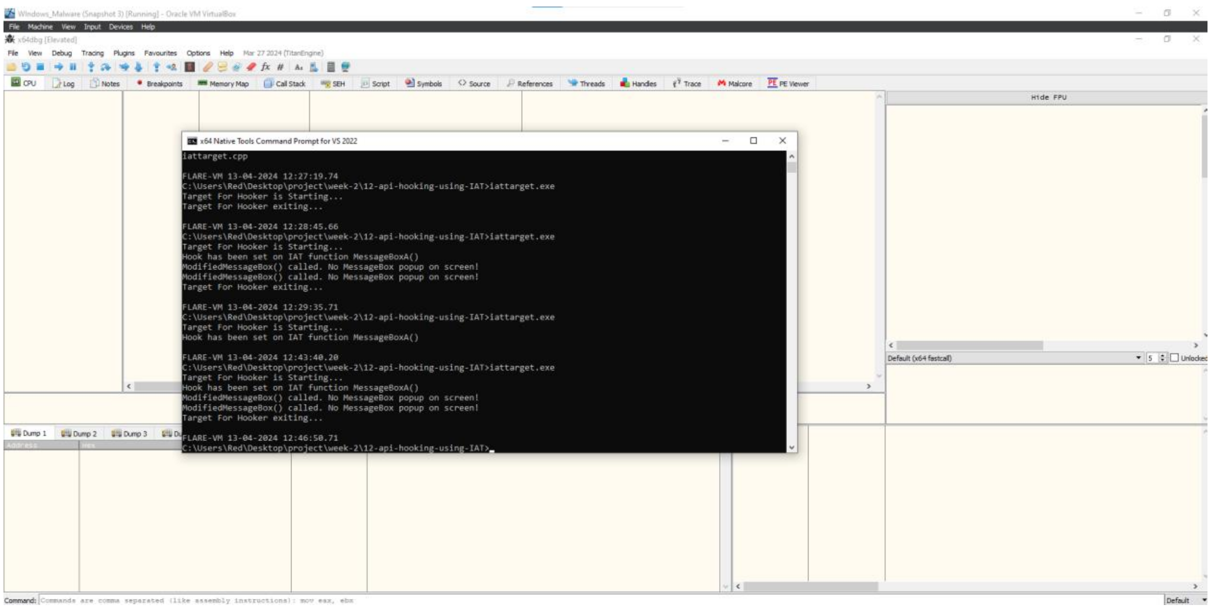
So, now attach the dll file, and see that change in the dump:



```
Address           Hex                                              ASCII
00007FF72FBF7258  10 10 80 36 FB 7F 00 00 00 00 00 00 00 00 00 00  ...6û..........
00007FF72FBF7268  6C 16 BE 2F F7 7F 00 00 6C 16 BE 2F F7 7F 00 00  l.¾/÷...l.¾/÷...
00007FF72FBF7278  D0 57 BF 2F F7 7F 00 00 F0 57 BF 2F F7 7F 00 00  ÐW¿/÷...ðW¿/÷...
00007FF72FBF7288  F0 57 BF 2F F7 7F 00 00 00 00 00 00 00 00 00 00  ðW¿/÷...........
00007FF72FBF7298  F0 5B BF 2F F7 7F 00 00 00 00 00 00 00 00 00 00  ð[¿/÷...........
00007FF72FBF72A8  AC 11 BE 2F F7 7F 00 00 00 00 00 00 00 00 00 00  ¬.¾/÷...........
00007FF72FBF72B8  00 00 00 00 00 00 00 00 E4 10 BE 2F F7 7F 00 00  ........ä.¾/÷...
00007FF72FBF72C8  9C 11 BE 2F F7 7F 00 00 70 52 BE 2F F7 7F 00 00  ..¾/÷...pR¾/÷...
00007FF72FBF72D8  E0 28 BF 2F F7 7F 00 00 E0 DE BE 2F F7 7F 00 00  à(¿/÷...àÞ¾/÷...
00007FF72FBF72E8  A0 4A BF 2F F7 7F 00 00 00 00 00 00 00 00 00 00   J¿/÷...........
00007FF72FBF72F8  00 00 00 00 00 00 00 00 00 83 BE 2F F7 7F 00 00  .........¾/÷...
00007FF72FBF7308  4C 44 BF 2F F7 7F 00 00 A4 53 BE 2F F7 7F 00 00  LD¿/÷...¤S¾/÷...
00007FF72FBF7318  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00007FF72FBF7328  00 00 00 00 00 00 00 00 54 61 72 67 65 74 20 46  ........Target F
00007FF72FBF7338  6F 72 20 48 6F 6F 6B 65 72 20 69 73 20 53 74 61  or Hooker is Sta
00007FF72FBF7348  72 74 69 6E 67 2E 2E 2E 0A 00 00 00 00 00 00 00  rting...........
00007FF72FBF7358  31 73 74 20 4D 65 73 73 61 67 65 42 6F 78 00 00  1st MessageBox..
00007FF72FBF7368  54 68 69 73 20 69 73 20 74 68 65 20 66 69 72 73  This is the firs
00007FF72FBF7378  74 20 6D 65 73 73 61 67 65 20 66 72 6F 6D 20 54  t message from T
00007FF72FBF7388  48 45 20 46 55 54 55 52 45 21 00 00 00 00 00 00  HE FUTURE!......
```

So, we can see the change in the dump memory, so this means we have successfully injected our function into this address, to replace the original one.

So, now if we continue to run by pressing ok on the message box, we can see that it exits out.



```
iattarget.cpp

FLARE-VM 13-04-2024 12:27:19.74
C:\Users\Red\Desktop\project\week-2\12-api-hooking-using-IAT>iattarget.exe
Target For Hooker is Starting...
Target For Hooker exiting...

FLARE-VM 13-04-2024 12:28:45.66
C:\Users\Red\Desktop\project\week-2\12-api-hooking-using-IAT>iattarget.exe
Target For Hooker is Starting...
Hook has been set on IAT function MessageBoxA()
ModifiedMessageBox() called. No MessageBox popup on screen!
ModifiedMessageBox() called. No MessageBox popup on screen!
Target For Hooker exiting...

FLARE-VM 13-04-2024 12:29:35.71
C:\Users\Red\Desktop\project\week-2\12-api-hooking-using-IAT>iattarget.exe
Target For Hooker is Starting...
Hook has been set on IAT function MessageBoxA()

FLARE-VM 13-04-2024 12:43:40.20
C:\Users\Red\Desktop\project\week-2\12-api-hooking-using-IAT>iattarget.exe
Target For Hooker is Starting...
Hook has been set on IAT function MessageBoxA()
ModifiedMessageBox() called. No MessageBox popup on screen!
ModifiedMessageBox() called. No MessageBox popup on screen!
Target For Hooker exiting...

FLARE-VM 13-04-2024 12:46:50.71
C:\Users\Red\Desktop\project\week-2\12-api-hooking-using-IAT>
```

This happened because we intercepted the 2nd and the 3rd message box.

So, it is working properly.