

Reversing

DLL INJECTION



목차

Reversing_

01

DLL

02

DLL INJECTION

03

Dev DLL :
my_hack.cpp

04

DLL INJECTOR :
dll_injector.cpp

05

Training :
notepad.exe

06

Registry :
Applnit_DLLs

07

NEXT TIME ...

08

Q&A

01 | DLL

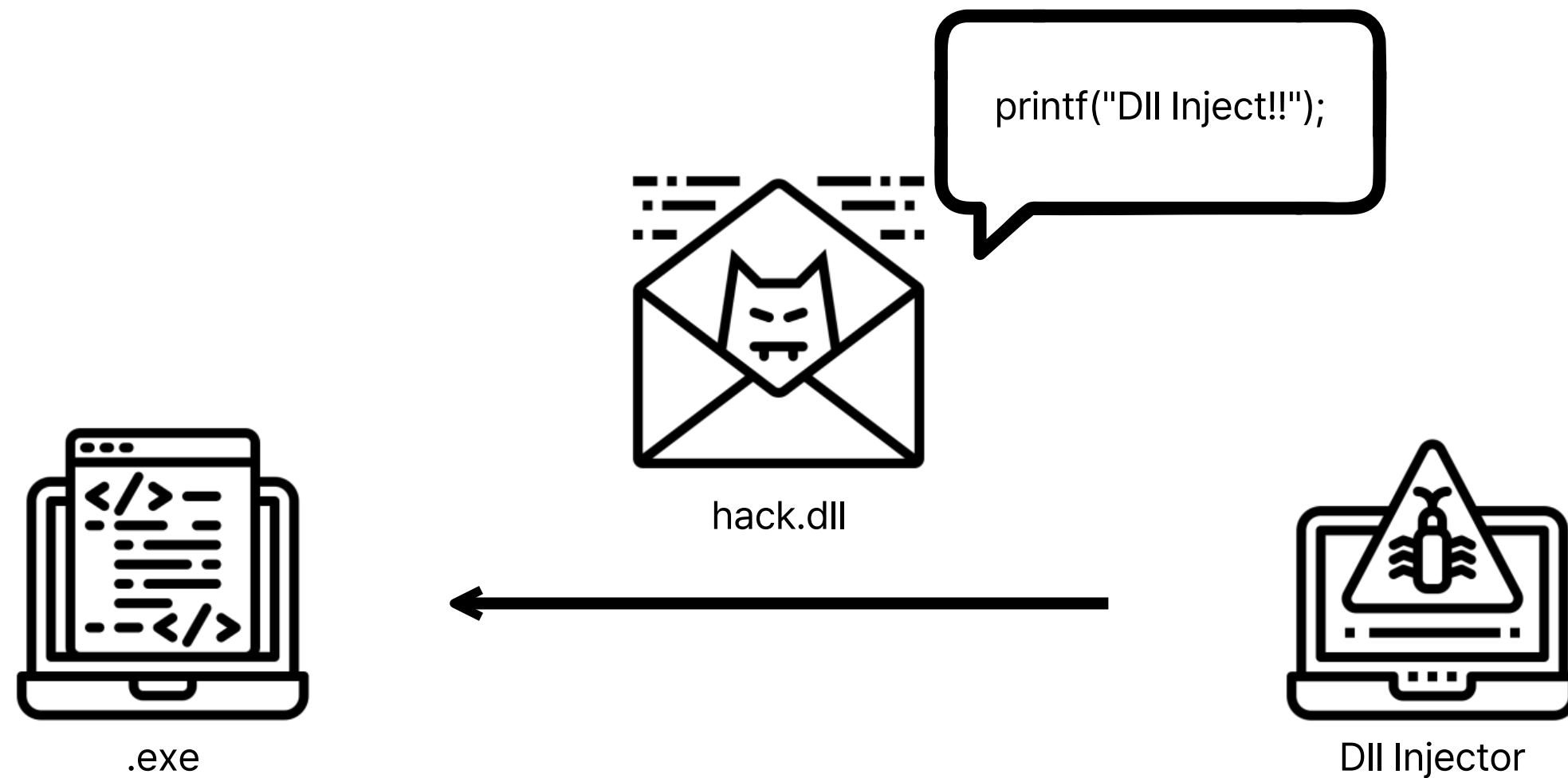


Dynamic Linked Library : 동적 링크 라이브러리

- 여러 프로그램에서 동시에 사용할 수 있는 코드와 데이터를 포함하는 "동적 라이브러리"
- Vs. Static Linked Library : 정적 링크의 경우 컴파일 이후에 실행 파일의 일부가 된다.

→ DLL을 이용하여 효율적인 개발 가능!

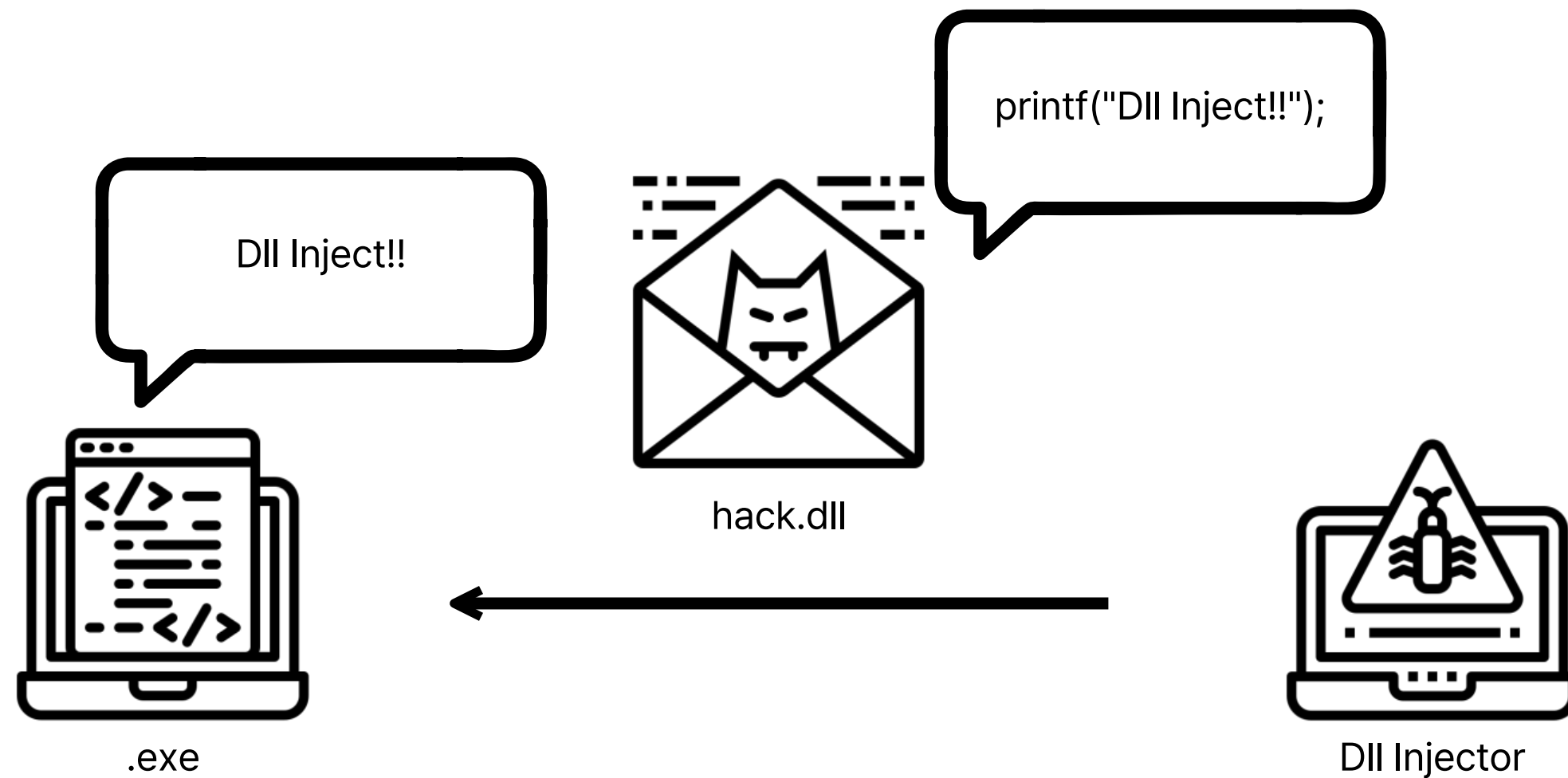
02 | DLL INJECTION



DLL INJECTION ?

- 일반적인 DLL 로딩 : 프로그램에서 사용할 명시된 DLL만 로딩
- DLL Injection : 실행중인 프로세스에 특정 DLL 파일을 강제로 삽입(로딩)하는 것
- DLL 인젝션 활용의 예시
 - 기능 개선 및 버그 패치
 - API 후킹
 - 기타 응용 프로그램
 - 악성코드

02 | DLL INJECTION



DLL INJECTION ?

- 일반적인 DLL 로딩 : 프로그램에서 사용할 명시된 DLL만 로딩
- DLL Injection : 실행중인 프로세스에 특정 DLL 파일을 강제로 삽입(로딩)하는 것
- DLL 인젝션 활용의 예시
 - 기능 개선 및 버그 패치
 - API 후킹
 - 기타 응용 프로그램
 - 악성코드

03

Dev DLL : my_hack.cpp

원하는 프로세스에 로딩 시킬 DLL 코드
URLDownloadToFile(), MessageBoxW()

```
#define DEF_URL      (L"http://www.naver.com/index.html")
#define DEF_FILE_NAME (L"index.html")

HINSTANCE g_hMod = NULL;

DWORD WINAPI ThreadProc(LPVOID lParam)
{
    WCHAR szPath[500] = { 0, };

    if (!GetModuleFileName(g_hMod, szPath, MAX_PATH))
        return FALSE;

    WCHAR* p = wcsrchr(szPath, '\\');
    if (!p)
        return FALSE;

    wcsncpy_s(p + 1, MAX_PATH, DEF_FILE_NAME);
    URLDownloadToFile(NULL, DEF_URL, szPath, 0, NULL);

    MessageBoxW(NULL, TEXT("Hello, World!"), TEXT("Test"), MB_OK);
    return 0;
}
```

03

Dev DLL : my_hack.cpp

DLL이 실행되는 동안 디버그창에 ??? 출력
DLL이 로딩되는 순간에 이전 코드들 실행

```
BOOL WINAPI DllMain(HINSTANCE hinstDLL, DWORD fdwReason, LPVOID lpvReserved)
{
    OutputDebugString(L"???");
    HANDLE hThread = NULL;

    g_hMod = (HINSTANCE)hinstDLL;

    switch (fdwReason)
    {
    case DLL_PROCESS_ATTACH:
        OutputDebugString(L"<64my_hack.dll> Injection!!!");
        hThread = CreateThread(NULL, 0, ThreadProc, NULL, 0, NULL);
        CloseHandle(hThread);
        break;
    }

    return TRUE;
}
```

04

DLL INJECTOR: dll_injector.cpp

```
int _tmain(int argc, TCHAR* argv[])
{
    if (argc != 3)
    {
        _tprintf(L"USAGE : %s pid dll_path\n", argv[0]);
        return 1;
    }
    DWORD dwPID = (DWORD)_tstol(argv[1]);
    LPCTSTR szDllPath = argv[2];
    HANDLE hProcess = NULL, hThread = NULL;
    HMODULE hMod = NULL;
    LPVOID pRemoteBuf = NULL;
    DWORD dwBufSize = (DWORD)(_tcslen(szDllPath) + 1) * sizeof(TCHAR);
    LPTHREAD_START_ROUTINE pThreadProc;

    if (!(hProcess = OpenProcess(PROCESS_ALL_ACCESS, FALSE, dwPID)))
    {
        _tprintf(L"OpenProcess(%d) failed!!! [%d]\n", dwPID, GetLastError());
        return FALSE;
    }

    pRemoteBuf = VirtualAllocEx(hProcess, NULL, dwBufSize, MEM_COMMIT, PAGE_READWRITE);

    WriteProcessMemory(hProcess, pRemoteBuf, (LPVOID)szDllPath, dwBufSize, NULL);

    hMod = GetModuleHandle(L"kernel32.dll");
    pThreadProc = (LPTHREAD_START_ROUTINE)GetProcAddress(hMod, "LoadLibraryW");

    hThread = CreateRemoteThread(
        hProcess, //hProcess
        NULL,     // lpThreadAttributes
        0,       // dwStackSize
        pThreadProc, // lpStartAddress
        pRemoteBuf, // lpParameter
        0,       // dwCreationFlags
        NULL);   // lpThreadId
    WaitForSingleObject(hThread, INFINITE);
    _tprintf(L"OpenProcess(%d) success!!! [%d]\n", dwPID, GetLastError());
    CloseHandle(hThread);
    CloseHandle(hProcess);

    return 0;
}
```


04

DLL INJECTOR : dll_injector.cpp

dwPID : 프로세스 ID

OpenProcess() : 원하는 프로세스의 핸들을 구
해와서 제어

```
int _tmain(int argc, TCHAR* argv[])
{
    if (argc != 3)
    {
        _tprintf(L"USAGE : %s pid dll_path\n", argv[0]);
        return 1;
    }
    DWORD dwPID = (DWORD)_tstol(argv[1]);
    LPCTSTR szDllPath = argv[2];
    HANDLE hProcess = NULL, hThread = NULL;
    HMODULE hMod = NULL;
    LPVOID pRemoteBuf = NULL;
    DWORD dwBufSize = (DWORD)(_tcslen(szDllPath) + 1) * sizeof(TCHAR);
    LPTHREAD_START_ROUTINE pThreadProc;

    if (!(hProcess = OpenProcess(PROCESS_ALL_ACCESS, FALSE, dwPID)))
    {
        _tprintf(L"OpenProcess(%d) failed!!! [%d]\n", dwPID, GetLastError());
        return FALSE;
    }
}
```

04

DLL INJECTOR : dll_injector.cpp

VirtualAllocEx() : 인젝션할 DLL의 경로를 써줄 버퍼를 할당

```
pRemoteBuf = VirtualAllocEx(hProcess, NULL, dwBufSize, MEM_COMMIT, PAGE_READWRITE);

WriteProcessMemory(hProcess, pRemoteBuf, (LPVOID)szDllPath, dwBufSize, NULL);

hMod = GetModuleHandle(L"kernel32.dll");
pThreadProc = (LPTHREAD_START_ROUTINE)GetProcAddress(hMod, "LoadLibraryW");

hThread = CreateRemoteThread(
    hProcess,    //hProcess
    NULL,        // lpThreadAttributes
    0,           // dwStackSize
    pThreadProc, // lpStartAddress
    pRemoteBuf,  // lpParameter
    0,           // dwCreationFlags
    NULL);      // lpThreadId
WaitForSingleObject(hThread, INFINITE);
_tprintf(L"OpenProcess(%d) success!!! [%d]\n", dwPID, GetLastError());
CloseHandle(hThread);
CloseHandle(hProcess);

return 0;
}
```

04

DLL INJECTOR : dll_injector.cpp

WriteProcessMemory() : 할당된 버퍼에
dll 경로 문자열 저장

```
pRemoteBuf = VirtualAllocEx(hProcess, NULL, dwBufSize, MEM_COMMIT, PAGE_READWRITE);

WriteProcessMemory(hProcess, pRemoteBuf, (LPVOID)szDllPath, dwBufSize, NULL);

hMod = GetModuleHandle(L"kernel32.dll");
pThreadProc = (LPTHREAD_START_ROUTINE)GetProcAddress(hMod, "LoadLibraryW");

hThread = CreateRemoteThread(
    hProcess,    //hProcess
    NULL,        // lpThreadAttributes
    0,           // dwStackSize
    pThreadProc, // lpStartAddress
    pRemoteBuf,  // lpParameter
    0,           // dwCreationFlags
    NULL);       // lpThreadId
WaitForSingleObject(hThread, INFINITE);
_tprintf(L"OpenProcess(%d) success!!! [%d]\n", dwPID, GetLastError());
CloseHandle(hThread);
CloseHandle(hProcess);

return 0;
}
```

szDllPath = argv[2];

04

DLL INJECTOR : dll_injector.cpp

GetProcAddress() : 인젝터에 로딩된
kernel32.dll의 LoadLibrary() 주소를 구한다.

```
pRemoteBuf = VirtualAllocEx(hProcess, NULL, dwBufSize, MEM_COMMIT, PAGE_READWRITE);

WriteProcessMemory(hProcess, pRemoteBuf, (LPVOID)szDllPath, dwBufSize, NULL);

hMod = GetModuleHandle(L"kernel32.dll");
pThreadProc = (LPTHREAD_START_ROUTINE)GetProcAddress(hMod, "LoadLibraryW");

hThread = CreateRemoteThread(
    hProcess,    //hProcess
    NULL,        // lpThreadAttributes
    0,           // dwStackSize
    pThreadProc, // lpStartAddress
    pRemoteBuf,  // lpParameter
    0,           // dwCreationFlags
    NULL);       // lpThreadId
WaitForSingleObject(hThread, INFINITE);
_tprintf(L"OpenProcess(%d) success!!! [%d]\n", dwPID, GetLastError());
CloseHandle(hThread);
CloseHandle(hProcess);

return 0;
}
```

04 | Why Injector's Address?



.exe



Dll Injector

user32.dll	0x77CF0000		user32.dll	0x77CF0000
...
kernel32.dll	0x7C7D0000		kernel32.dll	0x7C7D0000
shell32.dll	0x7D5A0000		shell32.dll	0x7D5A0000

Windows 운영체제에서 kernel32.dll은 프로세스마다 같은 주소에 로딩된다!

- 물론 부팅할 때마다의 주소는 바뀐다. (ASLR 기능)
- 하지만 한 번 로딩이 된 이후에는 다른 프로세스가 해당 메모리 주소를 매핑하여 메모리를 효율적으로 사용한다.
- 즉, 인젝터에서 가져오는 LoadLibrary() 의 메모리 주소와 인젝션할 프로세스의 LoadLibrary() 주소는 같다!

04

DLL INJECTOR : dll_injector.cpp

CreateRemoteThread() : 대상 프로세스에 스레드를 원격 실행한다.

여태 구해온 프로세스 핸들, LoadLibrary() 주소 인젝션할 dll 문자열 주소 등등을 넘겨준다.

```
pRemoteBuf = VirtualAllocEx(hProcess, NULL, dwBufSize, MEM_COMMIT, PAGE_READWRITE);

WriteProcessMemory(hProcess, pRemoteBuf, (LPVOID)szDllPath, dwBufSize, NULL);

hMod = GetModuleHandle(L"kernel32.dll");
pThreadProc = (LPTHREAD_START_ROUTINE)GetProcAddress(hMod, "LoadLibraryW");

hThread = CreateRemoteThread(
    hProcess,    //hProcess
    NULL,        // lpThreadAttributes
    0,           // dwStackSize
    pThreadProc, // lpStartAddress
    pRemoteBuf,  // lpParameter
    0,           // dwCreationFlags
    NULL);       // lpThreadId
WaitForSingleObject(hThread, INFINITE);
_tprintf(L"OpenProcess(%d) success!!! [%d]\n", dwPID, GetLastError());
CloseHandle(hThread);
CloseHandle(hProcess);

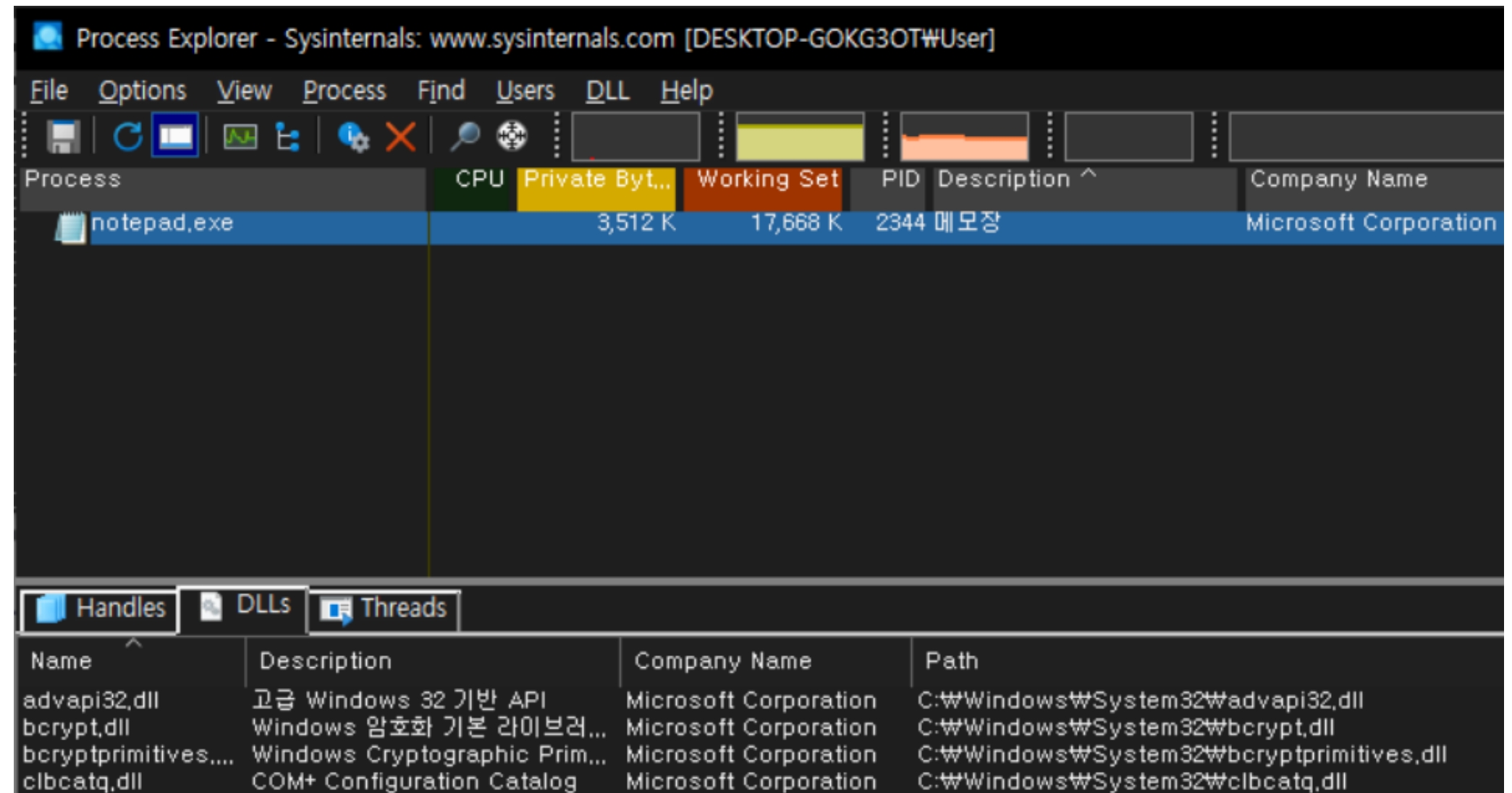
return 0;
}
```

05

Training : notepad.exe

실습과정 : cmd, process explorer

```
C:\test>64dll_injection.exe 2344 C:\test\64_real_my_hack.dll
```



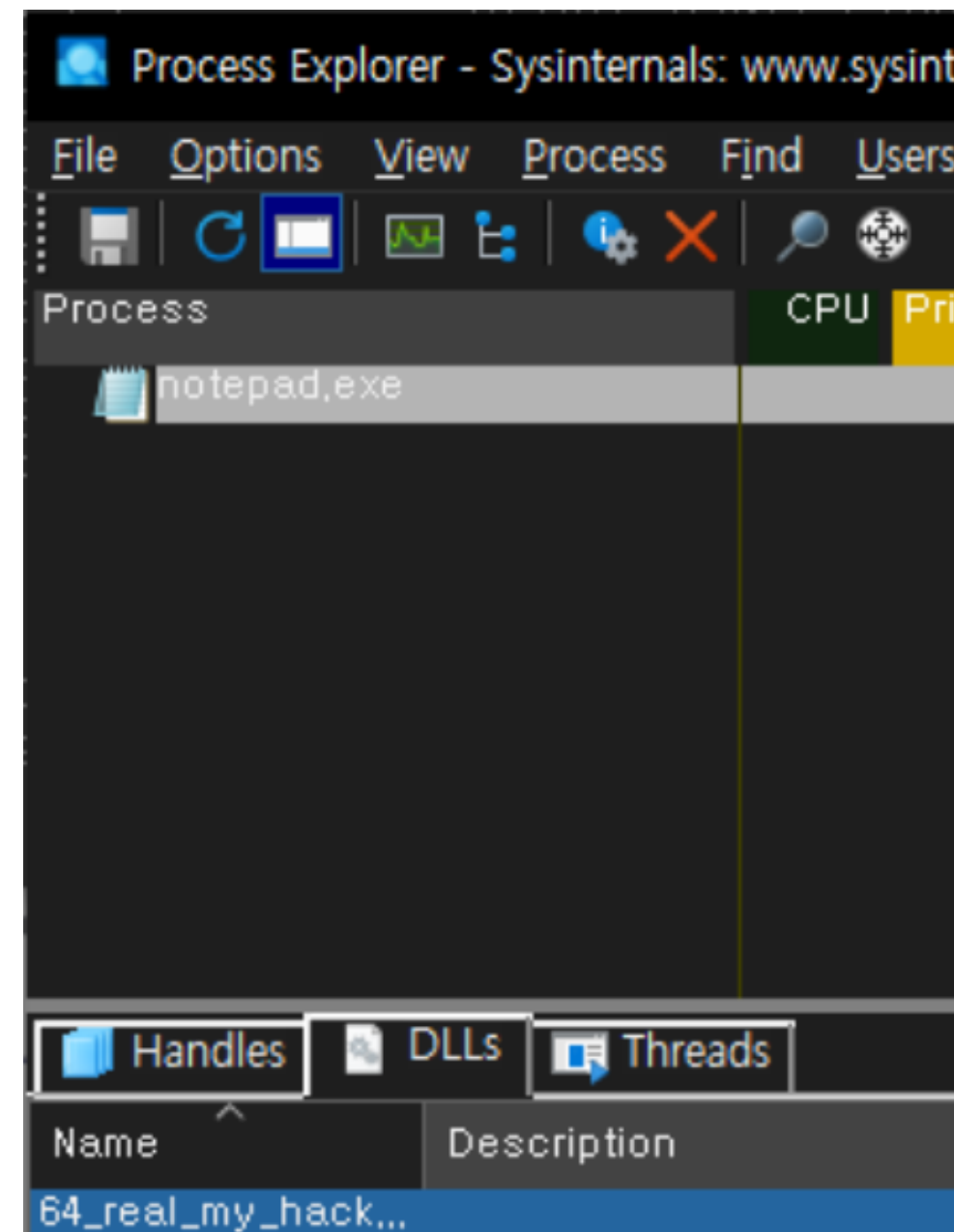
05

Training : notepad.exe

실습과정 : 인젝션 이후의 모습

```
C:\#test>64dll_injection.exe 2344 C:\#test\64_real_my_hack.dll
OpenProcess(2344) success!!! [0]

C:\#test>
```

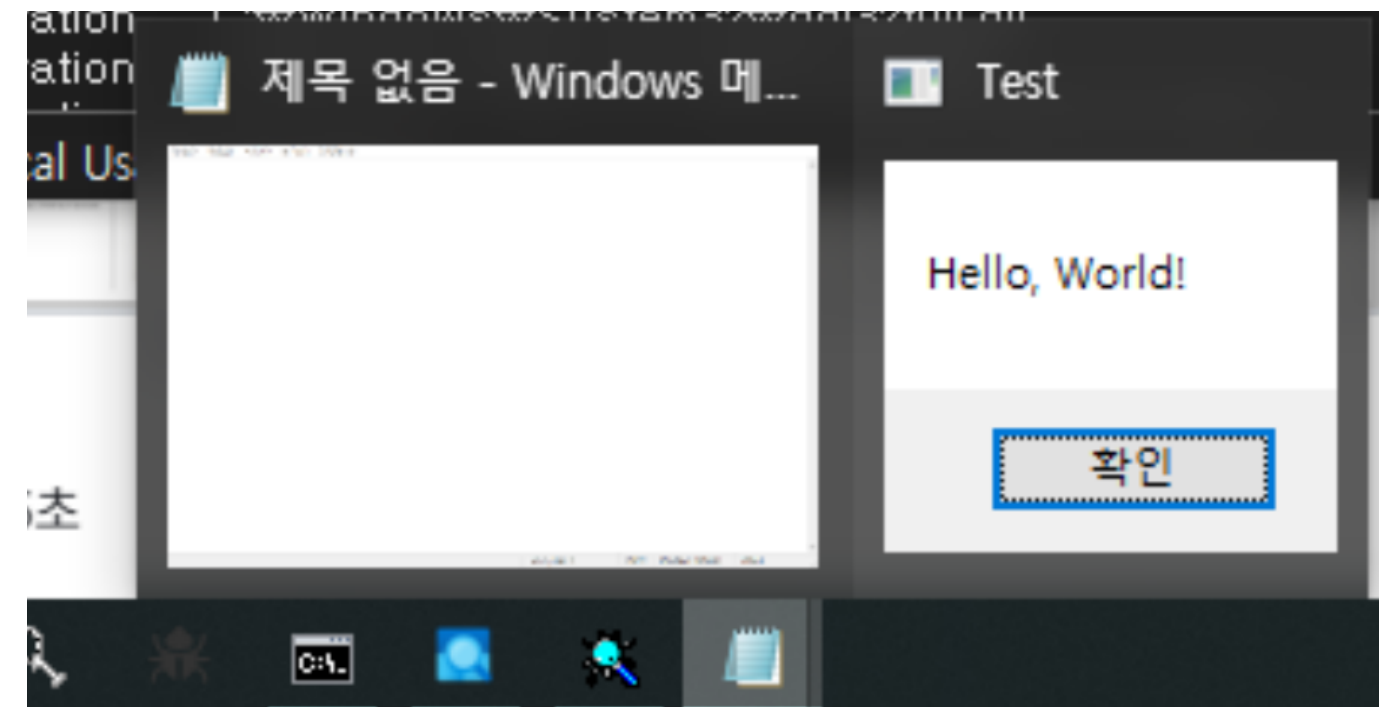


05

Training : notepad.exe

실습과정 : notepad.exe

```
C:\#test>64dll_injection.exe 2344 C:\#test\64_real_my_hack.dll  
OpenProcess(2344) success!!! [0]  
  
C:\#test>
```

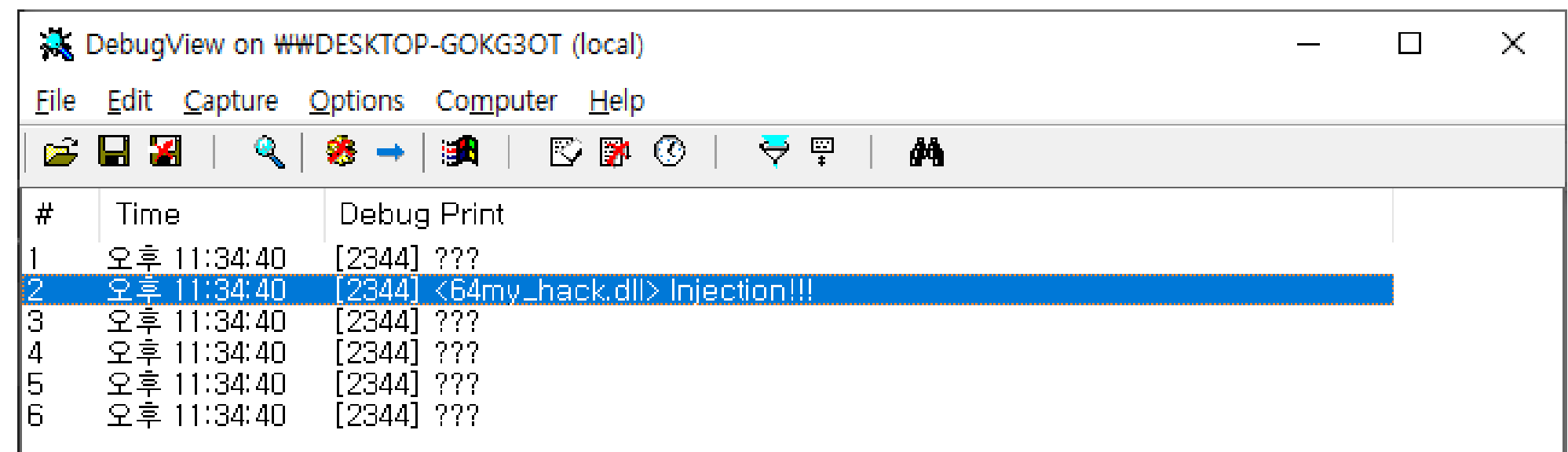


05

Training : notepad.exe

실습과정 : DebugView

```
C:\test>64dll_injection.exe 2344 C:\test\64_real_my_hack.dll  
OpenProcess(2344) success!!! [0]  
  
C:\test>
```



#	Time	Debug Print
1	오후 11:34:40	[2344] ???
2	오후 11:34:40	[2344] <64my_hack.dll> Injection!!!
3	오후 11:34:40	[2344] ???
4	오후 11:34:40	[2344] ???
5	오후 11:34:40	[2344] ???
6	오후 11:34:40	[2344] ???

06

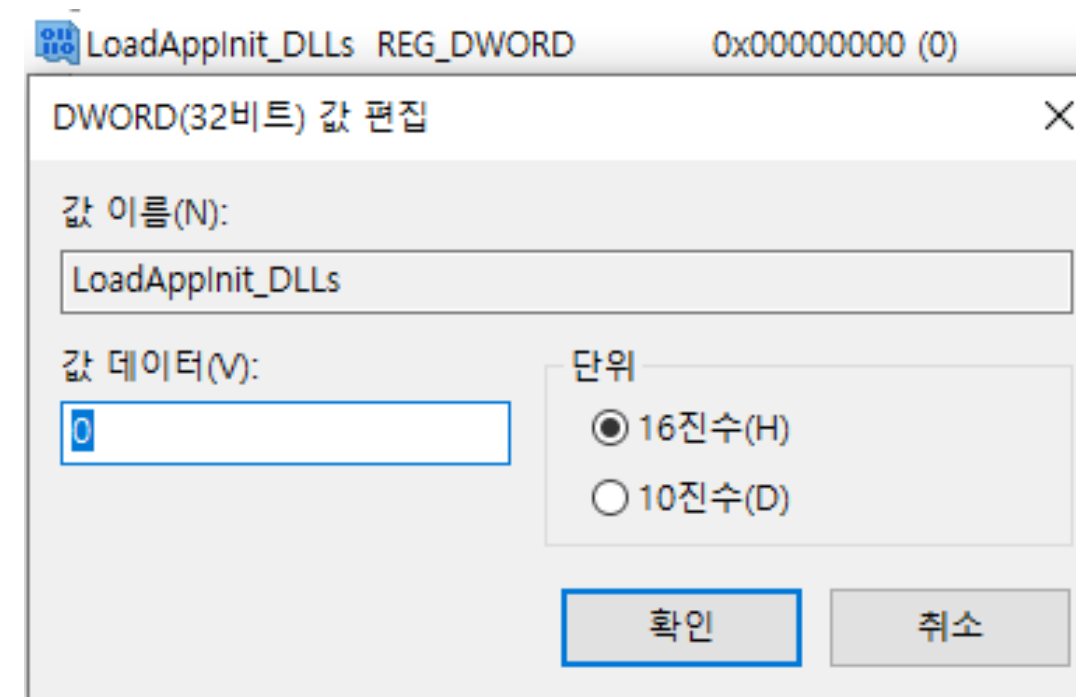
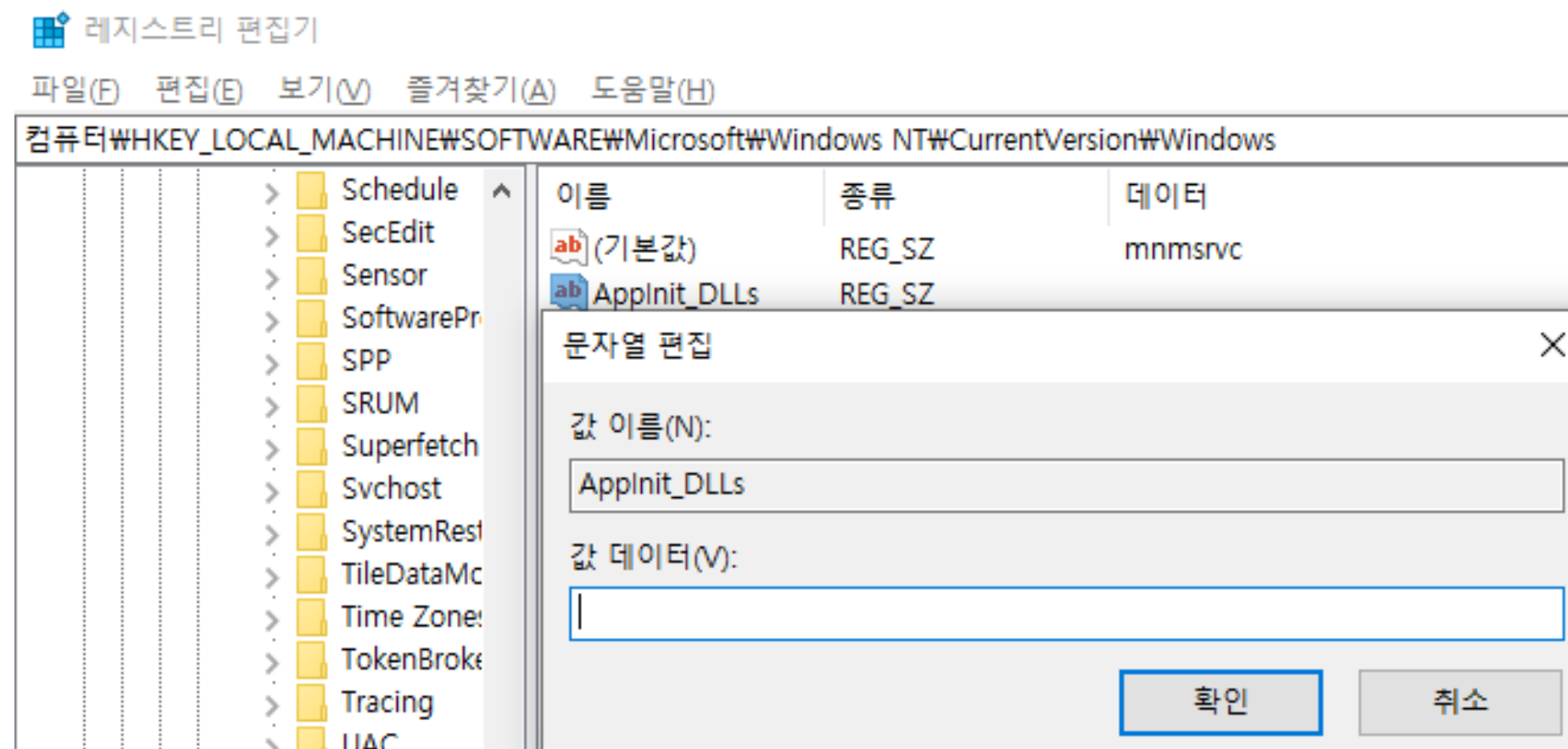
Registry : AppInit_DLLs

윈도우에서 기본으로 제공하는 레지스트리 항목을 이용하는 방법

AppInit_DLLs : 원하는 dll 경로 작성

LoadAppInit_DLLs : 1로 설정

→ 실행되는 모든 프로세스에 dll이 로딩된다!



07

NEXT TIME . .

DLL INJECTION



DLL EJECTION

PE File Patch?

Thank you!

Q & A

4주간의 삽질.. 날 더 강하게 만들었을까..?
