

## Forum:[Hack Forums](#)

Thread: [WriteProcessMemory works in Visual Studio but not when run separately](#)

**Logik** - July 10, 2020, 12:36 pm

Hi, I was using gigajew's RunPE code when I came across this weird behavior. When I inject a 64bit .NET application into another 64bit .NET application or when I inject a 64bit Native application into another 64bit Native application, it only successfully injects and runs when I use Visual Studio to execute the code. When I try to execute the code through the compiled executable, the code hits an early return and exits out. The piece of code it exits out on is this:

```
Code if (!WriteProcessMemory(processInfo.hProcess, plImageBase, pBuffer,
ntHeaders->OptionalHeader.SizeOfHeaders, IntPtr.Zero))
{
    if (!TerminateProcess(processInfo.hProcess, -1))
    {
        throw new Win32Exception(Marshal.GetLastWin32Error());
    }
    else
    {
        CloseHandle(processInfo.hProcess);
        CloseHandle(processInfo.hThread);

        return false;
    }
}
```

Here are all the obvious steps I tried to do:Run executable as administrator

Set Platform Target to x64

Checked the "Allow unsafe code box"

My guess is that when I run it through Visual Studio, it obtains special debugging rights and is able to read and write process information. If this is the case, is there a way I can have the compiled executable access the same rights?

**gigajew** - July 10, 2020, 2:27 pm

Platform target can vary when you change Debug/Release mode. Ensure that you've set it properly for both.

Edit: Just realized you're using some of my RunPE code. It has a weird bug like that. I cannot figure it out.

**Logik** - July 10, 2020, 3:07 pm

(Jul 10, 2020 02:27 PM)gigajew Wrote: [CITE](#) Platform target can vary when you change Debug/Release mode. Ensure that you've set it properly for both.

Edit: Just realized you're using some of my RunPE code. It has a weird bug like that. I cannot figure

it out.

This really is a strange bug. if I find a fix I'll be sure to let you know!