

Forum: [Hack Forums](#)

Thread: [\[VB.NET\] PEB Patcher](#)

neofusion - August 10, 2015, 7:05 pm

Good day!

A while ago I made a PEB patcher in vb.net,
It uses pointers and structures, the code is not that simplified.
So instead of unmanaged pointers, I used managed bytes as buffers for ReadProcessMemory and WriteProcessMemory,

I can patch the PEB but this only erases the contents of it, not patch it (the command line). I suspect that I'm writing it with wrong or null values.

```
Code Dim bytNewPath As Byte() = Encoding.Unicode.GetBytes(strPath_)
```

```
    'Allocate memory
    Dim intNewAddr As Integer = VirtualAllocEx(objHandle, IntPtr.Zero, bytNewPath.Length,
    &H3000, &H40)

    'Write to allocated memory.
    If Not WriteProcessMemory(objHandle, intNewAddr, bytNewPath, bytNewPath.Length, 0) Then
    Throw New Exception

    'Write to PEB.
    Dim bytPointer As Byte() = BitConverter.GetBytes(intNewAddr)
    If Not WriteProcessMemory(objHandle, intUserProcParamsAddress + &H40, bytPointer,
    bytPointer.Length, 0) Then Throw New Exception

    'Closes the handle.
    CloseHandle(objHandle)
```

I'm sure the codes before that don't have problems since I can read the PEB paths right.

I'm sure I have the right values for intUserProcParamsAddress and strPath_

What might be the problem in here? Thanks for reading.

Menalix - August 10, 2015, 7:26 pm

I have really hard seeing what this has to do with patching the PEB structure of a process?

I can't see anything there, which should be related to the PEB struct:

[https://msdn.microsoft.com/en-us/library/windows/desktop/aa813706\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa813706(v=vs.85).aspx)

And what is: intUserProcParamsAddress ? supposed to be the baseaddress of this:

[https://msdn.microsoft.com/en-us/library/windows/desktop/aa813741\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa813741(v=vs.85).aspx) or?

Edit:

Oh you want to patch the Command Line parameter, I missed your paranthese, now I get it all, the NewImageBase confused me pretty much, seemed more like you were trying to write some pe to memory, wierd naming.

neofusion - August 10, 2015, 7:58 pm

(Aug 10, 2015 07:26 PM)Menalix Wrote: [\[CITE\]](#) Oh you want to patch the Command Line parameter, I missed your paranthese, now I get it all, the NewImageBase confused me pretty much, seemed more like you were trying to write some pe to memory, wierd naming.

Edited it. I was using your RunPE & I forgot to change the name of it, thank you.

Menalix - August 10, 2015, 8:55 pm

(Aug 10, 2015 07:58 PM)neofusion Wrote: [\[CITE\]](#) Edited it. I was using your RunPE & I forgot to change the name of it, thank you.

Ye okay.

Well I looked at your code you sent on PM.

The pointer address you retrieve for reading: RTL_USER_PROCESS_PARAMETERS structure, is very wrong. You read into the PEB_LDR_DATA structure, first 2 bytes being the end of a pointer address for baseaddress of the process mainmodule. So you are reading the wrong place.

You should use cheatengine for memory inspection, that would help a lot.

salmonesus - August 11, 2015, 12:29 pm

here u go fam, it's in own process but it should be like 3 lines of code change to do it in remote process, you can find peb of remote in runpe by CTX.EBX + 8 or thru NtQueryInformationProcess if u need an example of how to do it in remote cuz u cant figure out just let me know and i will help

Code unsafe class Program

```
{
    // got this from pinvoke.net or w/e
    [StructLayout(LayoutKind.Sequential)]
    public struct UNICODE_STRING : IDisposable
    {
        public ushort Length;
        public ushort MaximumLength;
        private IntPtr buffer;

        public UNICODE_STRING(string s)
        {
            Length = (ushort)(s.Length * 2);
            MaximumLength = (ushort)(Length + 2);
            buffer = Marshal.StringToHGlobalUni(s);
        }

        public void Dispose()
        {
            Marshal.FreeHGlobal(buffer);
            buffer = IntPtr.Zero;
        }

        public override string ToString()
        {
            return Marshal.PtrToStringUni(buffer);
        }
    }

    static byte[] GetPEBFunc =
```

```

{
    0x55, /* push ebp */
    0x89, 0xEC, /* mov esp, ebp */
    0x31, 0xC0, /* xor eax, eax */
    0x64, 0xA1, 0x30, 0x00, 0x00, 0x00, /* mov eax, [fs:30h] */
    0xC9, /* leave */
    0xC3 /* ret */
};

[DllImport("Kernel32.dll")]
private static extern bool VirtualProtect(IntPtr lpAddress, uint dwSize, uint flNewProtect, ref uint
lpflOldProtect);

[DllImport("Kernel32.dll")]
private static extern bool WriteProcessMemory(IntPtr hProcess, IntPtr lpBaseAddress, IntPtr
lpBuffer, uint nSize, IntPtr x);

[UnmanagedFunctionPointer(CallingConvention.StdCall)]
private delegate IntPtr GetPEBAddress();

static void Main(string[] args)
{
    byte* lpAllocatedFunc = stackalloc byte[GetPEBFunc.Length];

    IntPtr lpAddressFunc = (IntPtr)((uint*)lpAllocatedFunc);
    Marshal.Copy(GetPEBFunc, 0, lpAddressFunc, GetPEBFunc.Length);

    uint dwOld = 0;
    VirtualProtect(lpAddressFunc, (uint)GetPEBFunc.Length, 0x40, ref dwOld);

    GetPEBAddress mtd =
(GetPEBAddress)Marshal.GetDelegateForFunctionPointer(lpAddressFunc,
typeof(GetPEBAddress));
    IntPtr pPEB = mtd();

    Console.WriteLine("PEB Address: 0x{0}", pPEB.ToString("X4"));

    /* +0x010 ProcessParameters : _RTL_USER_PROCESS_PARAMETERS */
    IntPtr lpRtlProcParams = (IntPtr)Marshal.ReadInt32(pPEB + 0x10);

    Console.WriteLine("RTL_USER_PROCESS_PARAMATERS Address: 0x{0}",
lpRtlProcParams.ToString("X4"));

    /* +0x040 CommandLine : _UNICODE_STRING */
    UNICODE_STRING strCmdLine =
(UNICODE_STRING)Marshal.PtrToStructure(lpRtlProcParams + 0x40,
typeof(UNICODE_STRING));

    Console.WriteLine("CommandLine Field: {0}", strCmdLine.ToString());

    string newField = "seaworld";

    UNICODE_STRING strNewCmdLine = new UNICODE_STRING(newField);

```

```

IntPtr lpNewField = Marshal.AllocHGlobal(Marshal.SizeOf(strNewCmdLine));
Marshal.StructureToPtr(strNewCmdLine, lpNewField, false);

/* Patch field using writeprocessmemory for sake of an example of how it could be done in a
remote process */
bool bRet = WriteProcessMemory(Process.GetCurrentProcess().Handle, lpRtlProcParams +
0x40, lpNewField, 8 /*(uint)Marshal.SizeOf(strNewCmdLine)*/, IntPtr.Zero);

Console.WriteLine(bRet);

/* +0x040 CommandLine : _UNICODE_STRING */
UNICODE_STRING strPatchedCmdLine =
(UNICODE_STRING)Marshal.PtrToStructure(lpRtlProcParams + 0x40,
typeof(UNICODE_STRING));

Console.WriteLine("CommandLine Field Patched Value: {0}", strPatchedCmdLine.ToString());

Marshal.FreeHGlobal(lpNewField);

Console.ReadLine();
}
}

```

neofusion - August 11, 2015, 8:32 pm

(Aug 11, 2015 12:29 PM)salmoneus Wrote: [CITE](#) here u go fam, it's in own process but it should be like 3 lines of code change to do it in remote process, you can find peb of remote in runpe by CTX.EBX + 8 or thru NtQueryInformationProcess

if u need an example of how to do it in remote cuz u cant figure out just let me know and i will help

Spoiler ([Click to View](#))

Code unsafe class Program

```

{
    // got this from pinvoke.net or w/e
    [StructLayout(LayoutKind.Sequential)]
    public struct UNICODE_STRING : IDisposable
    {
        public ushort Length;
        public ushort MaximumLength;
        private IntPtr buffer;

        public UNICODE_STRING(string s)
        {
            Length = (ushort)(s.Length * 2);
            MaximumLength = (ushort)(Length + 2);
            buffer = Marshal.StringToHGlobalUni(s);
        }

        public void Dispose()
        {
            Marshal.FreeHGlobal(buffer);
            buffer = IntPtr.Zero;
        }

        public override string ToString()
    }
}

```

```

{
return Marshal.PtrToStringUni(buffer);
}
}

static byte[] GetPEBFunc =
{
0x55, /* push ebp */
0x89, 0xEC, /* mov esp, ebp */
0x31, 0xC0, /* xor eax, eax */
0x64, 0xA1, 0x30, 0x00, 0x00, 0x00, /* mov eax, [fs:30h] */
0xC9, /* leave */
0xC3 /* ret */
};

[DllImport("Kernel32.dll")]
private static extern bool VirtualProtect(IntPtr lpAddress, uint dwSize, uint flNewProtect, ref uint
lpflOldProtect);

[DllImport("Kernel32.dll")]
private static extern bool WriteProcessMemory(IntPtr hProcess, IntPtr lpBaseAddress, IntPtr
lpBuffer, uint nSize, IntPtr x);

[UnmanagedFunctionPointer(CallingConvention.StdCall)]
private delegate IntPtr GetPEBAddress();

static void Main(string[] args)
{
byte* lpAllocatedFunc = stackalloc byte[GetPEBFunc.Length];

IntPtr lpAddressFunc = (IntPtr)((uint*)lpAllocatedFunc);
Marshal.Copy(GetPEBFunc, 0, lpAddressFunc, GetPEBFunc.Length);

uint dwOld = 0;
VirtualProtect(lpAddressFunc, (uint)GetPEBFunc.Length, 0x40, ref dwOld);

GetPEBAddress mtd =
(GetPEBAddress)Marshal.GetDelegateForFunctionPointer(lpAddressFunc,
typeof(GetPEBAddress));
IntPtr pPEB = mtd();

Console.WriteLine("PEB Address: 0x{0}", pPEB.ToString("X4"));

/* +0x010 ProcessParameters : _RTL_USER_PROCESS_PARAMETERS */
IntPtr lpRtlProcParams = (IntPtr)Marshal.ReadInt32(pPEB + 0x10);

Console.WriteLine("RTL_USER_PROCESS_PARAMATERS Address: 0x{0}",
lpRtlProcParams.ToString("X4"));

/* +0x040 CommandLine : _UNICODE_STRING */
UNICODE_STRING strCmdLine =
(UNICODE_STRING)Marshal.PtrToStructure(lpRtlProcParams + 0x40,
typeof(UNICODE_STRING));

```

```

Console.WriteLine("CommandLine Field: {0}", strCmdLine.ToString());

string newField = "seaworld";

UNICODE_STRING strNewCmdLine = new UNICODE_STRING(newField);
IntPtr lpNewField = Marshal.AllocHGlobal(Marshal.SizeOf(strNewCmdLine));
Marshal.StructureToPtr(strNewCmdLine, lpNewField, false);

/* Patch field using writeprocessmemory for sake of an example of how it could be done in a
remote process */
bool bRet = WriteProcessMemory(Process.GetCurrentProcess().Handle, lpRtlProcParams +
0x40, lpNewField, 8 /*(uint)Marshal.SizeOf(strNewCmdLine)*/, IntPtr.Zero);

Console.WriteLine(bRet);

/* +0x040 CommandLine : _UNICODE_STRING */
UNICODE_STRING strPatchedCmdLine =
(UNICODE_STRING)Marshal.PtrToStructure(lpRtlProcParams + 0x40,
typeof(UNICODE_STRING));

Console.WriteLine("CommandLine Field Patched Value: {0}", strPatchedCmdLine.ToString());

Marshal.FreeHGlobal(lpNewField);

Console.ReadLine();
}
}

```

That's a very unique approach! Thank you.

You did use ASM to find the offsets for the pebaddr.

I myself can't understand ASM but with the example that you gave, I kinda get the idea of it in relation to C#.

Tho, why can't I patch remote processes?

I added this so I can get their handles, but writeprocessmemory returns False.

```

Code static IntPtr GetPHandle(string strProcess_)
{
    foreach (Process objProc in Process.GetProcessesByName(strProcess_))
    {
        Console.WriteLine("Found!");
        return objProc.Handle;
    }
    return IntPtr.Zero;
}

```

salmonius - August 11, 2015, 10:44 pm

It's a bit different to do it in a remote process, but it's the same idea.

I made some working code for you and tried to comment it a little bit. Good luck.

Spoiler [\(Click to View\)](#)

<http://pastebin.com/Z9hdggtM>

neofusion - August 12, 2015, 3:15 am

(Aug 11, 2015 10:44 PM)salmoneus Wrote: [\[CITE\]](#) It's a bit different to do it in a remote process, but it's the same idea.

I made some working code for you and tried to comment it a little bit. Good luck.

Thank you so much for this!

Your code is like a book, It's clear and I can learn so much from it.

You did the trick far from what others did. Thank you again!

NTTITONV210 - August 12, 2015, 1:24 pm

(Aug 11, 2015 08:32 PM)neofusion Wrote: [\[CITE\]](#) That's a very unique approach! Thank you.

You did use ASM to find the offsets for the pebaddr.

I myself can't understand ASM but with the example that you gave, I kinda get the idea of it in relation to C#.

Tho, why can't I patch remote processes?

I added this so I can get their handles, but writeprocessmemory returns False.

```
Code static IntPtr GetPHandle(string strProcess_)
{
    foreach (Process objProc in Process.GetProcessesByName(strProcess_))
    {
        Console.WriteLine("Found!");
        return objProc.Handle;
    }
    return IntPtr.Zero;
}
```

It's disassembled code. It's retrieving the PEB pointer from the fs segment register. Used by the WINDOWS OS.

Only3 - August 12, 2015, 2:50 pm

?

Code unsafe class Program

```
{
    // got this from pinvoke.net or w/e
    [StructLayout(LayoutKind.Sequential)]
    public struct UNICODE_STRING : IDisposable
    {
        public ushort Length;
        public ushort MaximumLength;
        private IntPtr buffer;

        public UNICODE_STRING(string s)
        {
            Length = (ushort)(s.Length * 2);
            MaximumLength = (ushort)(Length + 2);
            buffer = Marshal.StringToHGlobalUni(s);
        }

        public void Dispose()
```

```

{
Marshal.FreeHGlobal(buffer);
buffer = IntPtr.Zero;
}

public override string ToString()
{
return Marshal.PtrToStringUni(buffer);
}
}?
```

Code [StructLayout(LayoutKind.Sequential)]

```

struct UNICODE_STRING : IDisposable
{
    public ushort Length;
    public ushort MaximumLength;
    public IntPtr buffer;

    public UNICODE_STRING(string s)
    {
        Length = (ushort)(s.Length * 2);
        MaximumLength = (ushort)(Length + 2);
        buffer = Marshal.StringToHGlobalUni(s);
    }

    public void Dispose()
    {
        Marshal.FreeHGlobal(buffer);
        buffer = IntPtr.Zero;
    }

    public override string ToString()
    {
        return Marshal.PtrToStringUni(buffer);
    }
}
```

salmonius - August 12, 2015, 3:36 pm

(Aug 12, 2015 02:50 PM)Only3 Wrote: [\[CITE\]](#) ?

Code unsafe class Program

```

{
    // got this from pinvoke.net or w/e
    [StructLayout(LayoutKind.Sequential)]
    public struct UNICODE_STRING : IDisposable
    {
        public ushort Length;
        public ushort MaximumLength;
        private IntPtr buffer;

        public UNICODE_STRING(string s)
        {
            Length = (ushort)(s.Length * 2);
            MaximumLength = (ushort)(Length + 2);
            buffer = Marshal.StringToHGlobalUni(s);
```

```

    }

    public void Dispose()
    {
        Marshal.FreeHGlobal(buffer);
        buffer = IntPtr.Zero;
    }

    public override string ToString()
    {
        return Marshal.PtrToStringUni(buffer);
    }
    }?
Code [StructLayout(LayoutKind.Sequential)]
struct UNICODE_STRING : IDisposable
{
    public ushort Length;
    public ushort MaximumLength;
    public IntPtr buffer;

    public UNICODE_STRING(string s)
    {
        Length = (ushort)(s.Length * 2);
        MaximumLength = (ushort)(Length + 2);
        buffer = Marshal.StringToHGlobalUni(s);
    }

    public void Dispose()
    {
        Marshal.FreeHGlobal(buffer);
        buffer = IntPtr.Zero;
    }

    public override string ToString()
    {
        return Marshal.PtrToStringUni(buffer);
    }
}

```

WOW YOU CAUGHT ME RED HANDED FOR COPYING A STRUCTURE FROM PINVOKE.NET!
 You're proving my point of how utterly clueless you are just by posting this.

Since mod removes everyone of my posts, I'll just leave this thread knowing I helped the OP and that any competent user realizes how stupid your point is.

Only3 - August 12, 2015, 3:46 pm

since u known as a liar its seems like COPYING FROM
<https://hackforums.net/showthread.php?tid=1762981>

salmonesus - August 12, 2015, 3:55 pm

(Aug 12, 2015 03:46 PM)Only3 Wrote: [\[CITE\]](#) since u known as a liar its seems like COPYING FROM <https://hackforums.net/showthread.php?tid=1762981>

lol whatever dude i can rest easy at night knowing i coded my own shit, not going to convince the user with a dancing money meme as their avatar that they are in fact wrong. especially when said

user can't speak english. you're just mad because i exposed your shit runpe thread.

Menalix - August 12, 2015, 6:25 pm

(Aug 12, 2015 03:46 PM)Only3 Wrote: [\[CITE\]](#) since u known as a liar its seems like COPYING FROM <https://hackforums.net/showthread.php?tid=1762981>

Well there's copy-pasters everywhere he's just one of them, if we should waste time, calling them out 24/7, we would have no time in life left.

salnoneus - August 12, 2015, 7:00 pm

(Aug 12, 2015 06:25 PM)Menalix Wrote: [\[CITE\]](#) Well there's copy-pasters everywhere he's just one of them, if we should waste time, calling them out 24/7, we would have no time in life left.

<https://en.wikipedia.org/wiki/Irony>