

Forum:[Hack Forums](#)

Thread: [Read PEB path \[Anti Emulator/Debugger\]](#)

0xBB - February 24, 2014, 4:06 am

Hi everyone, Ive been messing around lately with AVS debugger/emulator. I've noticed some change/remove the file path in PEB while it "hook everything". Beside using :

PHP Code

```
<?php
```

```
PEB.BeingDebugged
```

To check for a debugger you could use this method.

Note: I haven't tested on every recent debugger/emulators.

Anyway, there is my work :

Code Imports System.Runtime.InteropServices

Class PEB

```
<DllImport("Ntdll.dll", EntryPoint:="NtQueryInformationProcess", CharSet:=CharSet.Unicode,
SetLastError:=True, ExactSpelling:=True)> _
```

```
Public Shared Function NtQueryInformationProcess(ByVal processHandle As Integer, ByVal
ProcessInformationClass As Integer, ByRef ProcessInformation As
PROCESS_BASIC_INFORMATION, ByVal ProcessInformationLength As Integer, ByRef
ReturnLength As Integer) As Integer
End Function
```

```
<DllImport("Kernel32.dll", EntryPoint:="ReadProcessMemory", CharSet:=CharSet.Unicode,
SetLastError:=True, ExactSpelling:=True)> _
```

```
Public Shared Function ReadProcessMemory(ByVal processHandle As Integer, ByVal
BaseAddress As Integer, ByVal Buffer As IntPtr, ByVal Size As Integer, ByRef OutputSize As
Integer) As Boolean
End Function
```

```
<StructLayout(LayoutKind.Sequential)> _
```

```
Public Structure PROCESS_BASIC_INFORMATION
```

```
Public ExitStatus As Integer
```

```
Public PebBaseAddress As Integer
```

```
Public AffinityMask As Integer
```

```
Public BasePriority As Integer
```

```
Public UniqueProcessId As Integer
```

```
Public InheritedFromUniqueProcessId As Integer
```

```
End Structure
```

```
<StructLayout(LayoutKind.Sequential)> _
```

```
Public Structure LDR_MODULE
```

```
Public InLoadOrderModuleList As LIST_ENTRY
```

```
Public InMemoryOrderModuleList As LIST_ENTRY
```

```
Public InInitializationOrderModuleList As LIST_ENTRY
```

```

Public BaseAddress As Integer
Public EntryPoint As Integer
Public SizeOfImage As UInt32
Public FullDllName As UNICODE_STRING
Public BaseDllName As UNICODE_STRING
Public Flags As UInt32
Public LoadCount As Short
Public TlsIndex As Short
Public HashTableEntry As LIST_ENTRY
Public TimeDateStamp As UInt32
End Structure
<StructLayout(LayoutKind.Sequential)> _
Public Structure UNICODE_STRING
    Public Length As UInt16
    Public MaximumLength As UInt16
    Public Buffer As Integer
End Structure
<StructLayout(LayoutKind.Sequential)> _
Public Structure PEB32
    Public InheritedAddressSpace As Byte
    Public ReadImageFileExecOptions As Byte
    Public BeingDebugged As Byte
    Public Spare As Byte
    Public Mutant As Integer
    Public ImageBaseAddress As Integer
    Public LoaderData As Integer
End Structure
<StructLayout(LayoutKind.Sequential)> _
Public Structure PEB_LDR_DATA
    Public Length As Integer
    Public Initialized As Boolean
    Public SsHandle As Integer
    Public InLoadOrderModuleList As LIST_ENTRY
    Public InMemoryOrderModuleList As LIST_ENTRY
    Public InInitializationOrderModuleList As LIST_ENTRY
End Structure
<StructLayout(LayoutKind.Sequential)> _
Public Structure LIST_ENTRY
    Public Flink As Integer
    Public Blink As Integer
End Structure

Public Shared Function Read() As String

    Dim PH As IntPtr = Process.GetCurrentProcess.Handle
    Dim PI As New PROCESS_BASIC_INFORMATION
    Dim PEB As New PEB32
    Dim peb_data As New PEB_LDR_DATA
    Dim ldr_module As New LDR_MODULE

    NtQueryInformationProcess(PH, 0, PI, Marshal.SizeOf(PI), 0)

    Dim Ptr1 As IntPtr = Marshal.AllocHGlobal(Marshal.SizeOf(PEB))

```

```

If ReadProcessMemory(PH, PI.PebBaseAddress, Ptr1, Marshal.SizeOf(PEB), 0) = False Then
Throw New System.ComponentModel.Win32Exception
End If
PEB = Marshal.PtrToStructure(Ptr1, GetType(PEB32))
Marshal.Release(Ptr1)

Dim Ptr2 As IntPtr = Marshal.AllocHGlobal(Marshal.SizeOf(peb_data))
If ReadProcessMemory(PH, PEB.LoaderData, Ptr2, Marshal.SizeOf(peb_data), 0) = False Then
Throw New System.ComponentModel.Win32Exception
End If
peb_data = Marshal.PtrToStructure(Ptr2, GetType(PEB_LDR_DATA))
Marshal.Release(Ptr2)

Dim Ptr3 As IntPtr = Marshal.AllocHGlobal(Marshal.SizeOf(ldr_module))
If ReadProcessMemory(PH, peb_data.InLoadOrderModuleList.Flink, Ptr3,
Marshal.SizeOf(ldr_module), 0) = False Then
Throw New System.ComponentModel.Win32Exception
End If
ldr_module = Marshal.PtrToStructure(Ptr3, GetType(LDR_MODULE))
Marshal.Release(Ptr3)

Dim Ptr4 As IntPtr = Marshal.AllocHGlobal(ldr_module.FullDllName.MaximumLength)

If ReadProcessMemory(PH, ldr_module.FullDllName.Buffer, Ptr4,
ldr_module.FullDllName.MaximumLength, 0) = False Then
Throw New System.ComponentModel.Win32Exception
End If
Return Marshal.PtrToStringUni(Ptr4, ldr_module.FullDllName.MaximumLength)

End Function

End Class

```

Thanks to Aeonhack for helping me sorting out little bugs.

Usage :

PHP Code

<?php

```

If Not PEB.Read() = Reflection.Assembly.GetExecutingAssembly.Location Then

```

```

    MsgBox("OMAHGUD")

```

```

End If

```

?

Madara-Uchiha - February 24, 2014, 5:42 am

Nice work you produced there.

Does this work for all debuggers?

frontenhack - February 24, 2014, 6:34 am

Nice work ibennz! But as you are reading your process memory use Marshal.Copy . Will work better :)

0xBB - February 24, 2014, 7:06 am

(Feb 24, 2014 07:34 AM)frontenhack Wrote: [\[CITE\]](#) Nice work ibennz! But as you are reading your

process memory use Marshal.Copy . Will work better :)

Yeah I know , but I've used Readproc api in case users want to monitor other processes. Thanks btw ? !

(Feb 24, 2014 06:42 AM)Madara-Uchiha Wrote: [\[CITE\]](#) Nice work you produced there.

Does this work for all debuggers?

Haven't tested with alot of em. Probably fews, not sure.

Only3 - February 24, 2014, 12:54 pm

there simple way to check can be use as another option

nL2TP - February 24, 2014, 12:59 pm

This is an old idea but still effective I think ...:)

This could be done easier in c# with unsafe code , I am just saying...

Thank You for the share ...

Banks? - February 24, 2014, 1:25 pm

(Feb 24, 2014 01:59 PM)nL2TP Wrote: [\[CITE\]](#) This is an old idea but still effective I think ...:)

This could be done easier in c# with unsafe code , I am just saying...

Thank You for the share ...

Faster? Marginally, at max a few milliseconds. Easier? Not really.

NETExplorer - February 24, 2014, 2:00 pm

(Feb 24, 2014 02:25 PM)Banks? Wrote: [\[CITE\]](#) Faster? Marginally, at max a few milliseconds.

Easier? Not really.

I don't believe he mentioned faster

Menalix - February 24, 2014, 2:52 pm

Would it be possible to use WriteProcessMemory to just write the original PEB back again?

Banks? - February 24, 2014, 2:54 pm

(Feb 24, 2014 03:00 PM)PatPositron Wrote: [\[CITE\]](#) I don't believe he mentioned faster

Yes but that would be the main reason to use unsafe code.

0xBB - March 10, 2014, 3:35 am

(Feb 24, 2014 03:52 PM)Menalix Wrote: [\[CITE\]](#) Would it be possible to use WriteProcessMemory to just write the original PEB back again?

Sorry didn't saw your question. Yes it is. Just allocate the new path's space and use wpm to write it.