

## Finding some non-exported kernel variables in Windows XP

by Edgar Barbosa

In the great majority of kernel modules that needs to get the value of some non-exported variable of the kernel, the solution was scan the kernel in the memory to find specific opcodes or specific signatures. But I'd founded a new way to get some of those hidden variables in the kernel by a new undocumented field in an old structure called **Processor Control Region**.

Let's see it:

KPCR STRUCT

```
; Start of the architecturally defined section of the PCR. This section
; may be directly addressed by vendor/platform specific HAL code and will
; not change from version to version of NT.
```

```
NtTib                NT_TIB <>
SelfPcr              PVOID    ?      ; 1Ch
Prcb                 PVOID    ?      ; 20h
Irql                 BYTE     ?      ; 24h
                    db        3 dup(?) ; padding
IRR                  DWORD    ?      ; 28h
IrrActive            DWORD    ?      ; 2Ch
IDR                  DWORD    ?      ; 30h
Reserved2           DWORD    ?      ; 34h
IDT                  PVOID    ?      ; 38h
GDT                  PVOID    ?      ; 3Ch
TSS                  PVOID    ?      ; 40h PTR KTSS
MajorVersion         WORD     ?      ; 44h
MinorVersion         WORD     ?      ; 46h
SetMember            KAFFINITY ?      ; 48h
StallScaleFactor     DWORD    ?      ; 4Ch
DebugActive          BYTE     ?      ; 50h
Number              BYTE     ?      ; 51h
                    db        2 dup(?) ; 052 padding
```

KPCR ENDS

While playing with the kernel of Windows XP, and looking for the KPCR area, I'd perceived that 0xffdff034 (Reserved2 field) was not equal 0x0, as usual in Windows 2000.

And while dumping this new pointer in XP, I'd founded some very important hidden kernel variables like:

```
PsActiveProcessHead
PsLoadedModuleList
MmPfnDatabase
PspCidTable
ObpRootDirectoryObject and more...
```

Then, I saw the following change in WinDbg:

```
lkd> dt _KPCR
+0x000 NtTib           : _NT_TIB
+0x01c SelfPcr        : Ptr32 _KPCR
+0x020 Prcb           : Ptr32 _KPRCB
+0x024 Irql           : UChar
+0x028 IRR            : Uint4B
+0x02c IrrActive      : Uint4B
+0x030 IDR            : Uint4B
```

```

+0x034 KdVersionBlock      : Ptr32 Void
+0x038 IDT                 : Ptr32 _KIDTENTRY
+0x03c GDT                 : Ptr32 _KGDENTRY
+0x040 TSS                 : Ptr32 _KTSS
+0x044 MajorVersion       : Uint2B
+0x046 MinorVersion       : Uint2B
+0x048 SetMember          : Uint4B
+0x04c StallScaleFactor   : Uint4B
+0x050 DebugActive        : UChar
+0x051 Number             : UChar
+0x052 Spare0             : UChar
+0x053 SecondLevelCacheAssociativity : UChar
+0x054 VdmAlert           : Uint4B
+0x058 KernelReserved    : [14] Uint4B
+0x090 SecondLevelCacheSize : Uint4B
+0x094 HalReserved       : [16] Uint4B
+0x0d4 InterruptMode     : Uint4B
+0x0d8 Spare1            : UChar
+0x0dc KernelReserved2   : [17] Uint4B
+0x120 PrcbData          : _KPRCB

```

What is the **KdVersionBlock** ?

While looking in the Google for some page that have all the kernel variables listed above, I founded the following include file:

<http://dotnet.di.unipi.it/Content/sscli/docs/doxygen/tools/sos/global.s.html>

And the structure that I'd founded appears to be the same of the **KDDEBUGGER\_DATA32** structure. Then I created the **getvar** (get in the [www.rootkit.com](http://www.rootkit.com)) program to make a test and it worked perfectly in Windows XP

Now, for example, to get the **PsActiveProcessHead**, is just a question of :

```

mov eax, 0ffdff034h
mov eax, [eax]
mov eax, [eax+078h]

```

Now, you have the **PsActiveProcessHead** with just 3 instructions!!! Well, I hope this text to be useful to the Rootkit community! Any errors or questions to: [embarbosa AT yahoo DOT com](mailto:embarbosa AT yahoo DOT com)

Regards,  
Opc0de