

Checking Your Memory Dump before Calling Support East

Hello, I am East with a quick review of what to do with your memory dumps before opening a support case. It's not uncommon for an entity to open a Microsoft support case with the intent of having a dump file examined by the escalation team. In some cases the memory dump is deemed corrupt which ultimately adds expense and additional hours to resolve the case. The purpose of this blog is to show you ways to potentially save money by doing a preliminary 'memory dump check' before opening the support case. This article is targeted at system administrators who don't necessarily spend a great chunk of time in the debugger, but may be tasked with fixing a crashing server.

DUMPCHK

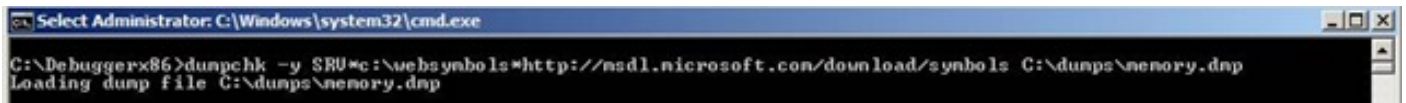
Microsoft provides a free debugging tools package to review memory dumps. After installing the tools, use Dumpchk.exe to validate both kernel and user memory dump files. It's a simple command line tool that doesn't even need the debugger.

The syntax for the command is

```
dumpchk [-y <sympath>] <dumpfile>
```

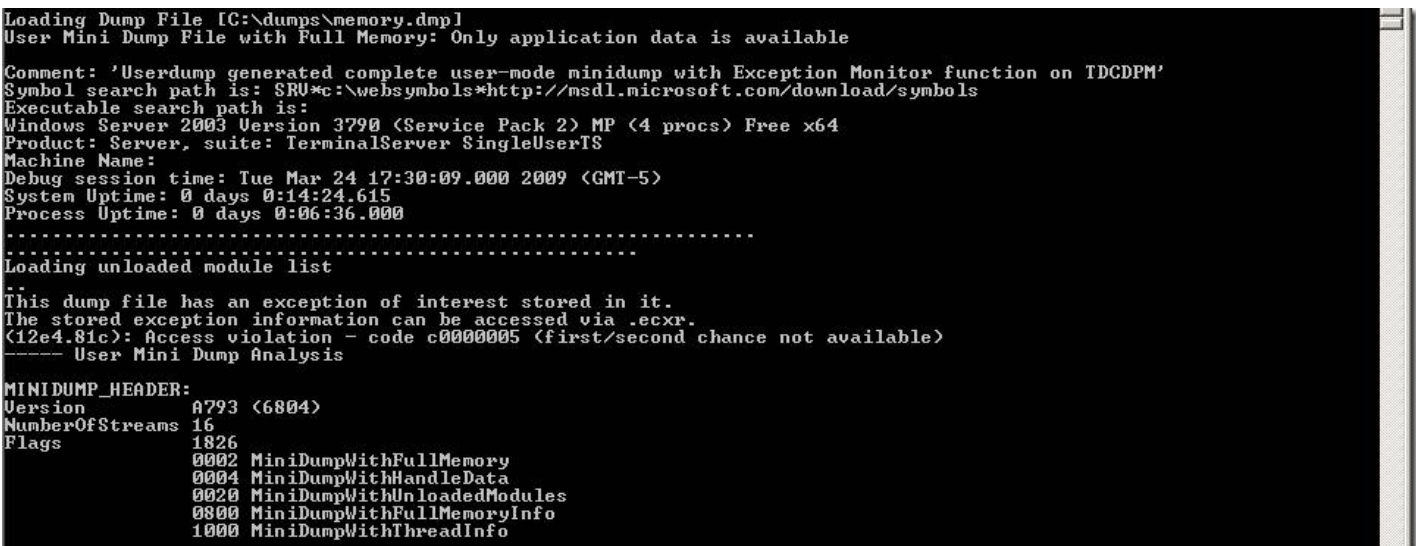
The -y parameter is used to specify a symbol path and is not always necessary for all dump files, however I recommend using it to show additional information. Simply point it to the Microsoft public symbol store at this path - <http://msdl.microsoft.com/download/symbols> (note: this link is not accessible through internet browsers).

Here's an example -



```
Select Administrator: C:\Windows\system32\cmd.exe
C:\Debugger>dumpchk -y SRU*c:\websymbols*http://msdl.microsoft.com/download/symbols C:\dumps\memory.dmp
Loading dump file C:\dumps\memory.dmp
```

With a user dump you would typically see output like this



```
Loading Dump File [C:\dumps\memory.dmp]
User Mini Dump File with Full Memory: Only application data is available

Comment: 'Userdump generated complete user-mode minidump with Exception Monitor function on TDCDPM'
Symbol search path is: SRU*c:\websymbols*http://msdl.microsoft.com/download/symbols
Executable search path is:
Windows Server 2003 Version 3790 (Service Pack 2) MP (4 procs) Free x64
Product: Server, suite: TerminalServer SingleUserTS
Machine Name:
Debug session time: Tue Mar 24 17:30:09.000 2009 (GMT-5)
System Uptime: 0 days 0:14:24.615
Process Uptime: 0 days 0:06:36.000

.....
Loading unloaded module list
..
This dump file has an exception of interest stored in it.
The stored exception information can be accessed via .ecxr.
(12e4.81c): Access violation - code c0000005 (first/second chance not available)
----- User Mini Dump Analysis

MINIDUMP_HEADER:
Version          A793 (6804)
NumberOfStreams  16
Flags            1826
                0002 MiniDumpWithFullMemory
                0004 MiniDumpWithHandleData
                0020 MiniDumpWithUnloadedModules
                0800 MiniDumpWithFullMemoryInfo
                1000 MiniDumpWithThreadInfo
```

Look for the "Finished dump check" message which usually indicates success. It is still possible for other forms of corruption in the dumps however Dumpchk is a great way to rule out obvious problems.

Checking Your Memory Dump before Calling Support East

```
USERPROFILE=C:\Documents and Settings\Default User
windir=C:\WINDOWS
Finished dump check
C:\Debugger>x86>
```

Note that other errors may be listed, some of which are actually benign. For example, the following error message does not represent a problem:

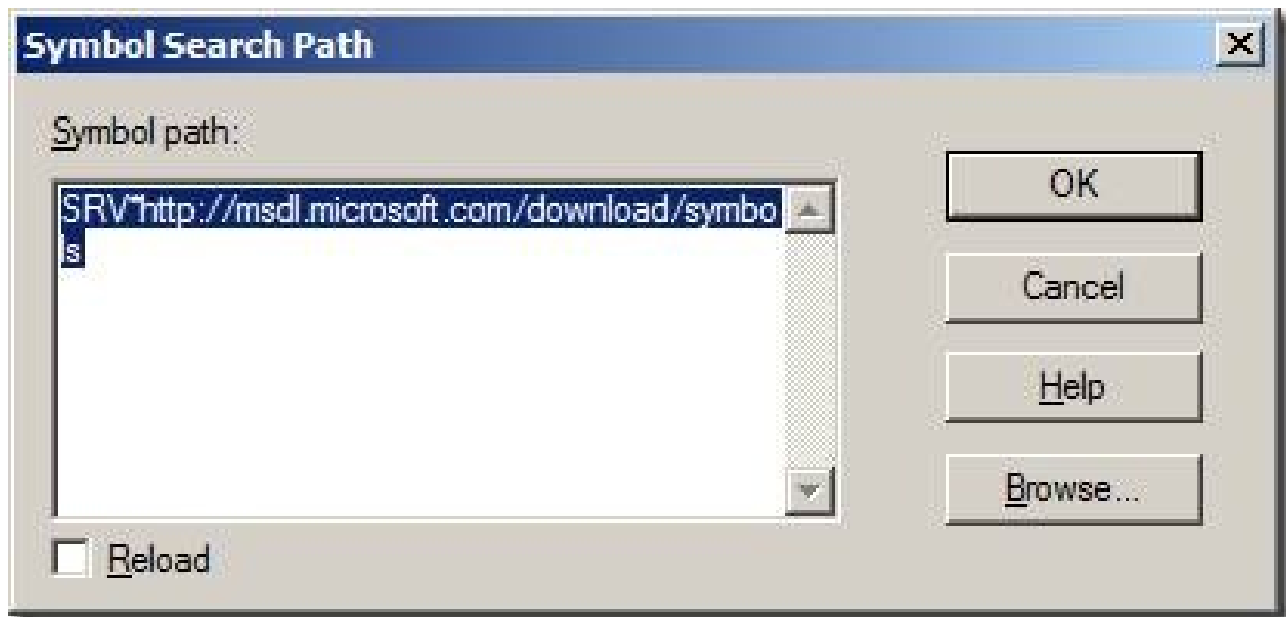
```
error 3 InitTypeRead( nt!_PEB at 7ffd5000)
```

As a general rule look for "Finished dump check".

Validating a Corrupt Dump In the Debugger

In addition to using Dumpchk, I recommend opening the memory dump in the debugger to look for some of the more common problems. Remember we're not actually debugging to determine the root cause of the crash. Instead we're looking for errors to validate the health of the memory dump so you don't waste your time sending us a bad memory dump file. It's easy to open a memory dump in the debugger.

1. Install the latest debugging tools package from the web. Download the correct platform, either x86 or x64.
2. Install the package. I recommend using C:\Debuggers for the install directory.
3. After the installation is complete, open Windbg and set the symbol path by selecting "File", then "Symbol File Path"



4. After clicking OK to set your symbol path, open the memory dump by selecting "File", then "Open Crash Dump". After opening the dump file, look for any of the conditions below.

NOTE: the following techniques are only applicable to Kernel memory dumps.

Checking Your Memory Dump before Calling Support East

If you see the message, "KdDebuggerDataBlock is not present or unreadable" the dump is most likely corrupt. Several things can cause the corruption including a page file too small to accommodate the contents of memory, inadequate disk space to hold the dump, or perhaps a problem with the OS recognizing Dynamic Hardware Partitioning. This is only a partial list of things to check.

```
*****
THIS DUMP FILE IS PARTIALLY CORRUPT.
KdDebuggerDataBlock is not present or unreadable.
*****
Unable to read PsLoadedModuleList
*****
THIS DUMP FILE IS PARTIALLY CORRUPT.
KdDebuggerDataBlock is not present or unreadable.
*****
```

I need to point out one caveat about the error above. If it is accompanied by I/O errors then chances are you're opening the dump located on a network share, and the debugger is having problems accessing the share.

```
Ignored in-page I/O error
Ignored in-page I/O error
Ignored in-page I/O error
Ignored in-page I/O error
Ignored in-page I/O error
Exception 0xc0000006 while accessing file mapping
Unable to read PsLoadedModuleList
Ignored in-page I/O error
Ignored in-page I/O error
Ignored in-page I/O error
Ignored in-page I/O error
Ignored in-page I/O error
Exception 0xc0000006 while accessing file mapping
```

Copy the dump to the machine where the debugger is installed and reopen it in Windbg. If you continue to see the "KdDebuggerDataBlock is not present or unreadable" error locally then the dump is corrupt.

If you see this message it simply means the symbol file path is incorrect. Check for typos in the 'Symbol Search Path' dialog box. Remember to point your symbol path to <http://msdl.microsoft.com/download/symbols>

```
*****
*
*                               Bugcheck Analysis                               *
*
*****
```

Use !analyze -v to get detailed debugging information.

```
BugCheck 50, {fffffff0, 0, 8086d649, 0}
```

```
**** Kernel symbols are WRONG. Please fix symbols to do analysis.
```

Checking Your Memory Dump before Calling Support East

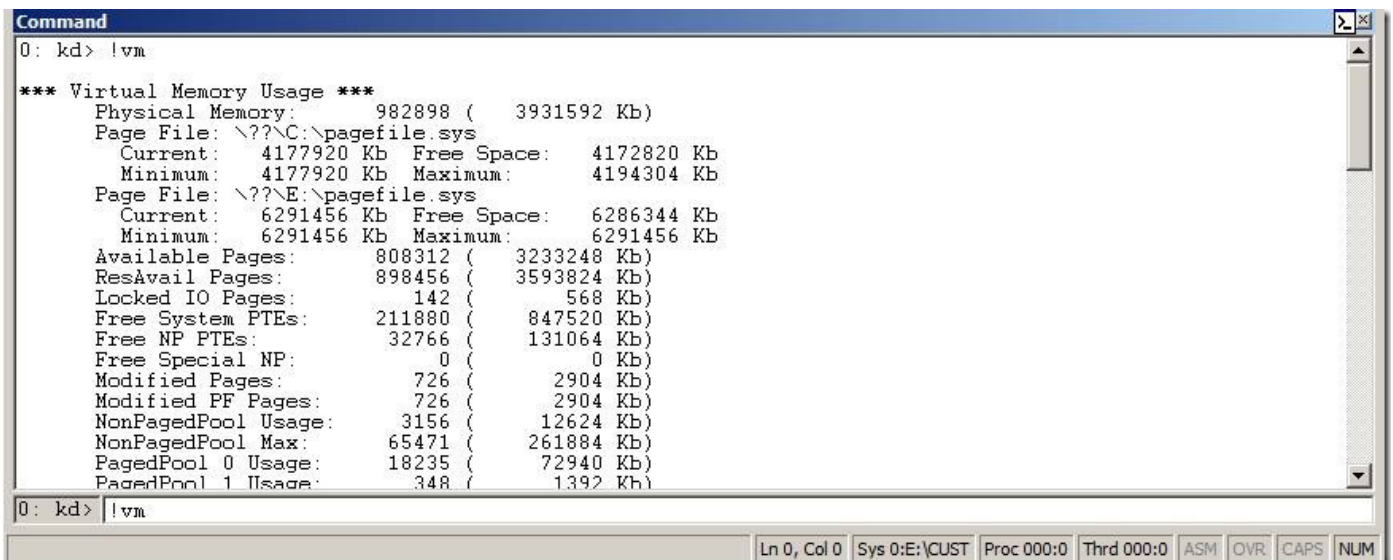
```
*****
***
***
*** Your debugger is not using the correct symbols ***
***
*** In order for this command to work properly, your symbol path ***
*** must point to .pdb files that have full type information. ***
***
*** Certain .pdb files (such as the public OS symbols) do not ***
*** contain the required information. Contact the group that ***
*** provided you with these symbols if you need this command to ***
*** work. ***
***
*** Type referenced: nt!_KPRCB ***
***
*****
```

You can ignore these messages-

```
"Page ????? not present in the dump file. Type ".hh dbgerr004" for details "  
"Page ????? not present in the dump file. Type ".hh dbgerr004" for details "  
"Page ????? not present in the dump file. Type ".hh dbgerr004" for details "  
"Page ????? not present in the dump file. Type ".hh dbgerr004" for details "
```

This indicates a page of memory needed by the debugger is missing from the dump. This does not mean the dump is corrupt and can be ignored, for the most part.

Page File Issues - One of the most common causes of corruption is an improperly configured page file. I highly recommend reviewing "How to generate a kernel or a complete memory dump file in Windows Server 2008" to properly configure the page file. If you don't have immediate access to the machine that generated the dump, and want to check for page file problems in the debugger, you can use the !vm command to weed out obvious issues.



Checking Your Memory Dump before Calling Support East

If you see something similar to the output below, it typically indicates the pagefile was not large enough, or the pagefile could not be found (notice the values given are so large as to not make sense).

```
kd> !vm
```

```
*** Virtual Memory Usage ***
Physical Memory:      3145107 ( 12580428 Kb)
Paging File Name paged out
  Current: 6385980911134738644 Kb  Free Space: 870495645396584060 Kb
  Minimum: 2351575501587540372 Kb  Maximum: 3053207781125429928 Kb
```

It's a good idea to set the page file at least 1 MB larger than the size of physical memory. Additionally the page file must be present on the system drive. This is the type of output you would see if the page file size is wrong or no page file was specified.

```
kd> !vm
```

```
*** Virtual Memory Usage ***
Physical Memory: 1903435 ( 7613740 Kb)
00000000: Unable to get page file
00000000: Unable to get paged pool info 0
unable to get nt!MmAllocatedNonPagedPool
unable to get nt!MmExtendedCommit
unable to get nt!MmPageFileFullExtendPages
unable to get nt!MmPageFileFullExtendCount
unable to get nt!MmTotalFreeSystemPtes
unable to get nt!MmSystemLockPagesCount
unable to get nt!MiSpecialPagesNonPaged
unable to get nt!MiSpecialPagesNonPagedMaximum
00000004: Unable to get number of free nonpaged PTEs
unable to get nt!MmSpecialPagesInUse
Available Pages: 1301896 ( 5207584 Kb)
ResAvail Pages: 1815514 ( 7262056 Kb)
Modified Pages: 426 ( 1704 Kb)
NonPagedPool Usage: 22058 ( 88232 Kb)
NonPagedPool Max: 65536 ( 262144 Kb)
8c5f7000: Unable to get pool descriptor
```

If the system has no page file you might see the message below when loading the dump

```
*****
THIS DUMP FILE IS PARTIALLY CORRUPT.
KdDebuggerDataBlock is not present or unreadable.
*****
Unable to read PsLoadedModuleList
.....
.....
Unable to get program counter
GetContextState failed, 0xD0000147
Unable to get current machine context, NTSTATUS 0xC0000147 <- this machine may not
have a pagefile.
GetContextState failed, 0xD0000147
GetContextState failed, 0xD0000147
```

Checking Your Memory Dump before Calling Support East

For additional information on memory dumps check the KB. Hopefully the information in this article will help you make better business decisions by evaluating whether it's too early to open a support call due to a corrupt dump. Thanks for the read!