

Blue Screen of the Death is dead

Matthieu Suiche, MoonSols – <http://www.moonsols.com>



Introduction

Physical memory is one of the key elements of any computer. As a volatile memory container, we can retrieve everything we are looking for. But physical memory snapshot are not new, especially on Windows. Microsoft introduced more than 10 years ago with the Blue Screen of the death – which was one of the most “stable” way to get a physical memory snapshot on Windows. Moreover, the Microsoft crash dump file format was designed to work with Microsoft Windows Debugger which is probably the most advanced analyze utility for memory dump.

Because of the several advantages this format provides, I decided to create a Toolkit able to convert any Windows memory dump into a Microsoft crash dump. Moreover, I also created a live acquisition utility which is able to produce a physical memory snapshot in two different formats.

- Linear memory mump
- Microsoft crash dump

All these utilities are in one toolkit called: MoonSols Windows Memory Toolkit.

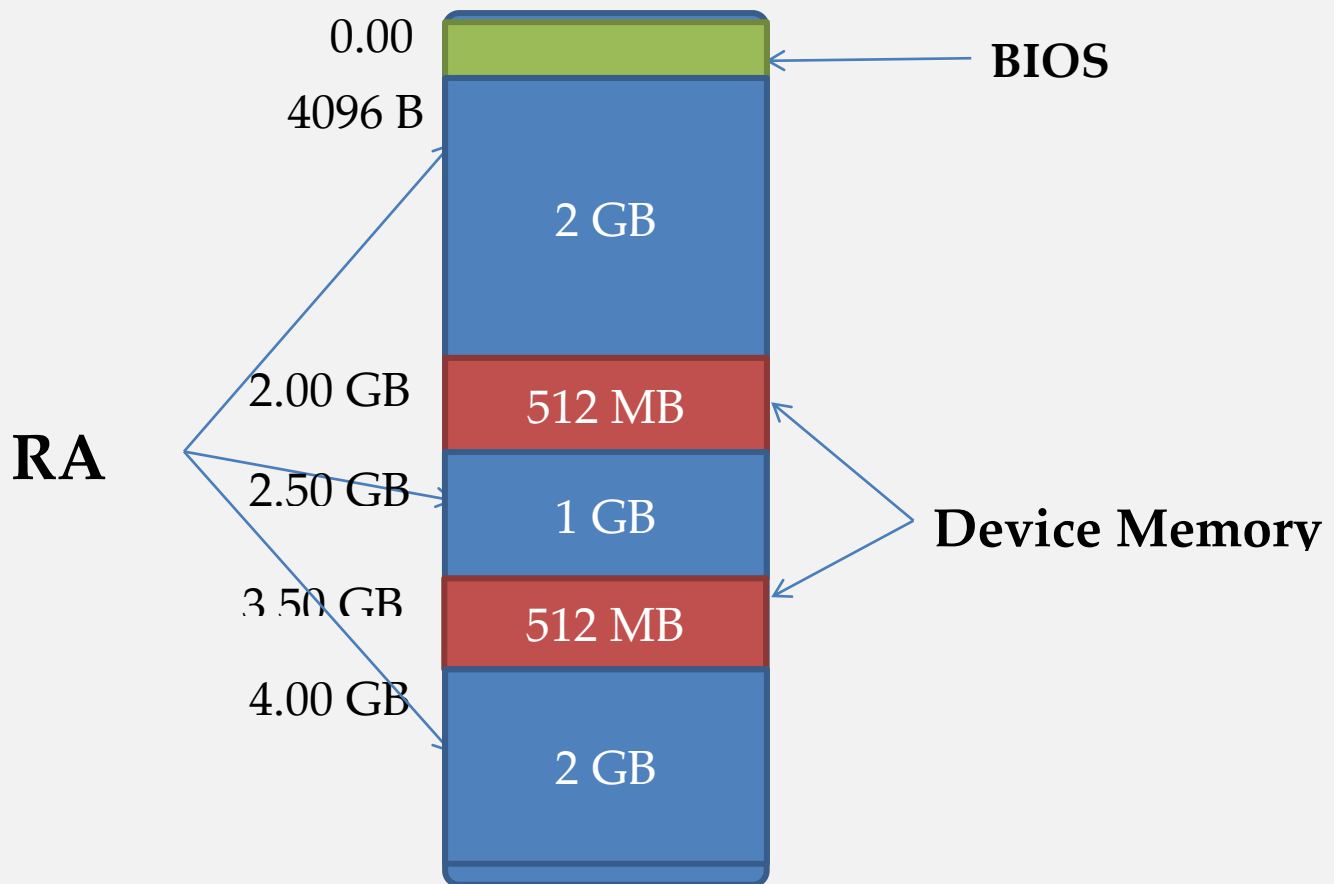
MoonSols Windows Memory Toolkit is the ultimate toolkit for memory dump conversion and acquisition on Windows. This toolkit had been designed to deal with various types of memory dumps such as VMWare memory snapshot, Microsoft crash dump and even Windows hibernation file.

Linear Memory Dump

A linear memory dump is how the processor views the physical memory, this is a raw snapshot. If you read the \\Device\\PhysicalMemory or any kernel API such as MmMapIoSpace()1 you should have access to a such view.

```
PVOID MmMapIoSpace (  
    IN PHYSICAL_ADDRESS PhysicalAddress,  
    IN ULONG NumberOfBytes,  
    IN MEMORY_CACHING_TYPE CacheType  
);
```

This must be differenced from a raw copy of the device memory itself. We are talking about the physical address space, which does not only include the physical memory but also MMIOs.



The figure above is a representation of the physical memory as seen by the processor. Blue blocks are the RAM.

When doing the acquisition, by default, our utility only copy the blue blocks – but there is still an option to copy all blocks. Windows does have a similar behavior when creating the hibernation file or the crash dump.

Blue Blocks are memory spaces used and reserved by the Operating System, these ranges are described by `MmPhysicalMemoryRange` variable, defined by `PHYSICAL_MEMORY_DESCRIPTOR` structure.

PHYSICAL_MEMORY_DESCRIPTOR	
ULONG	NumberOfRuns
PFN_NUMBER	NumberOfPages
PHYSICAL_MEMORY_RUN []	Run

PHYSICAL_MEMORY_RUN	
PFN_NUMBER	BasePage
PFN_NUMBER	PageCount

Microsoft crash dump

There are several types of Microsoft crash dump file. This paragraph is only about the full memory crash dump file. It is also interesting to mention that Windows cannot generate a full memory dump (`\WINDOWS\MEMORY.dmp`) if there are more than 2GB of RAM. In this case Windows (by default) generate a Mini-dump (*64kb on 32-bits systems and 128kB on 64-bits systems*) in `\WINDOWS\Minidumps\`. See KB274598 and KB241046 for more information. But our utility does, without any problem – Moreover, it does not produce a BSOD – this is why the BSOD is dead.

Windows generate the Microsoft crash dump when the BugCheck function is called, this function calls `IoWriteCrashDump()` which will produce the crash dump.

```
VOID KeBugCheck (
    __in ULONG BugCheckCode
);
```

This generate has some weaknesses, because Microsoft implemented `KeRegisterBugCheckCallback()` which is executed before writing the dump on disk. This function had been used as anti-forensic method by some Rootkit (e.g. Rustock.C) to flush the driver memory.

```
BOOLEAN
KeRegisterBugCheckCallback (
    IN PKBUGCHECK_CALLBACK_RECORD CallbackRecord,
    IN PKBUGCHECK_CALLBACK_ROUTINE CallbackRoutine,
    IN PVOID Buffer,
    IN ULONG Length,
    IN PCHAR Component
);
```

But the Microsoft crash dump file format does have a lot of advantages.

- It is uncompressed
- Contains a rich file header
 - o With processes, drivers, debug data variables pointers.
 - o Directory Table Base (cr3)

Microsoft crash dump header

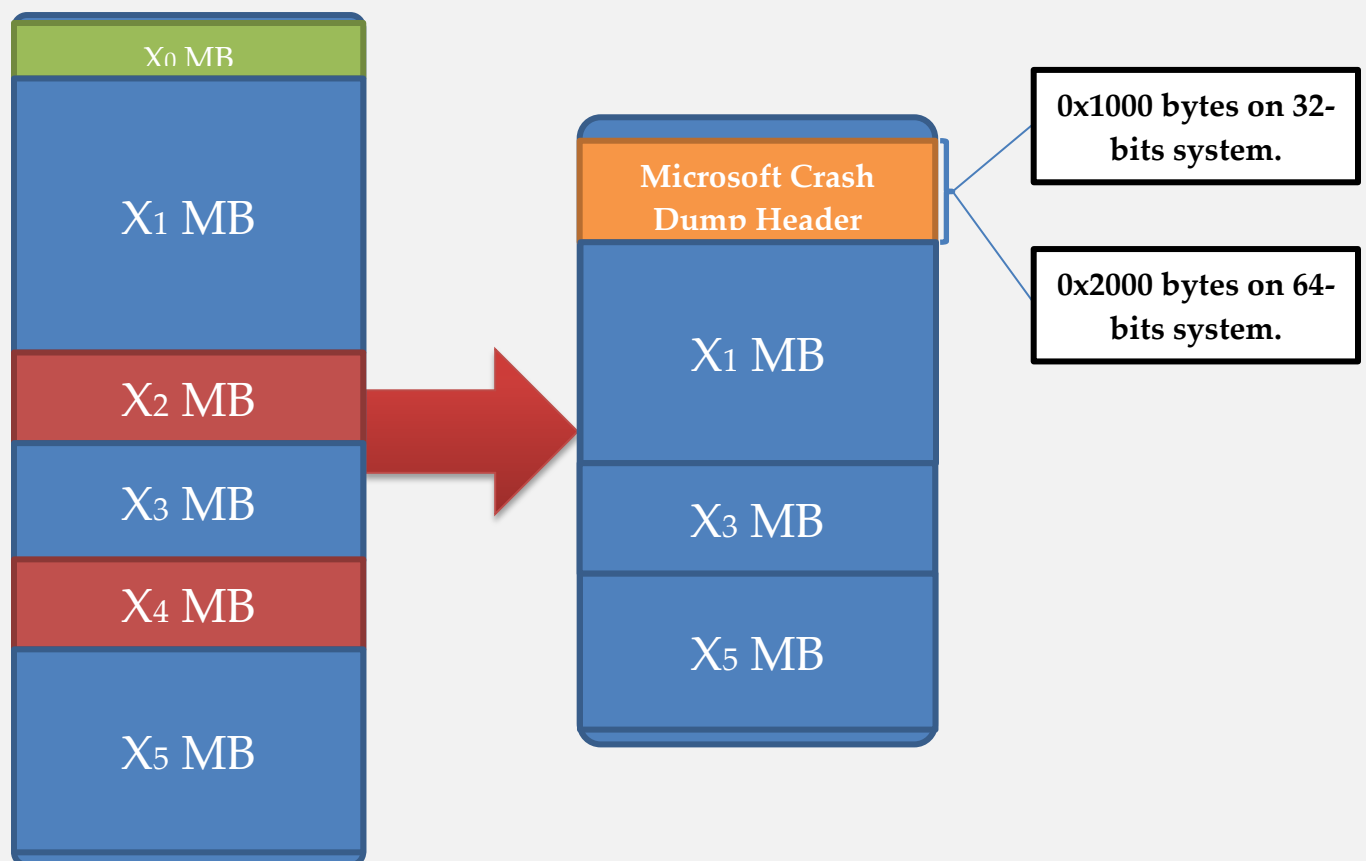
Microsoft crash dump header structure is available below of x86 versions of Windows. The x64 version is very similar, except the field “PaeEnabled” does not exist anymore because of the paging model of x64 processor.

DUMP_HEADER32 (1/2)	
ULONG	Signature
ULONG	ValidDump
ULONG	MajorVersion
ULONG	MinorVersion
ULONG	DirectoryTableBase
ULONG	PfnDataBase
ULONG	PsLoadedModuleList
ULONG	PsActiveProcessHead
ULONG	MachineImageType
ULONG	NumberProcessors
ULONG	BugCheckCode
ULONG	BugCheckParameter1
ULONG	BugCheckParameter2
ULONG	BugCheckParameter3
ULONG	BugCheckParameter4
CHAR [32]	VersionUser
CHAR	PaeEnabled
CHAR	KdSecondaryVersion
CHAR [2]	Spare

DUMP_HEADER32 (2/2)	
ULONG	KdDebuggerDataBlock
PHYSICAL_MEMORY_DESCRIPTOR	PhysicalMemoryBlock
UCHAR [700]	

CONTEXT	Context
UCHAR [1200]	
EXCEPTION_RECORD	ExceptionRecord
CHAR [128]	Comment
UCHAR [1768]	Reserved0
ULONG	DumpType
ULONG	MiniDumpFields
ULONG	SecondaryDataState
ULONG	ProductType
ULONG	SuiteMask
UCHAR [4]	reserved1
LARGE_INTEGER	RequieredDumpSpace
UCHAR [16]	Reserved2
LARGE_INTEGER	SystemUpTime
LARGE_INTEGER	SystemTime
UCHAR [56]	reserved3

Generation of the Microsoft crash dump file



Like above, Blue blocks are the RAM. This is how Windows copies generate the physical memory. When doing the acquisition, by default, our utility only copy the blue blocks – but there is still an option to copy all blocks. Windows does have a similar behavior when creating the hibernation file or the crash dump.

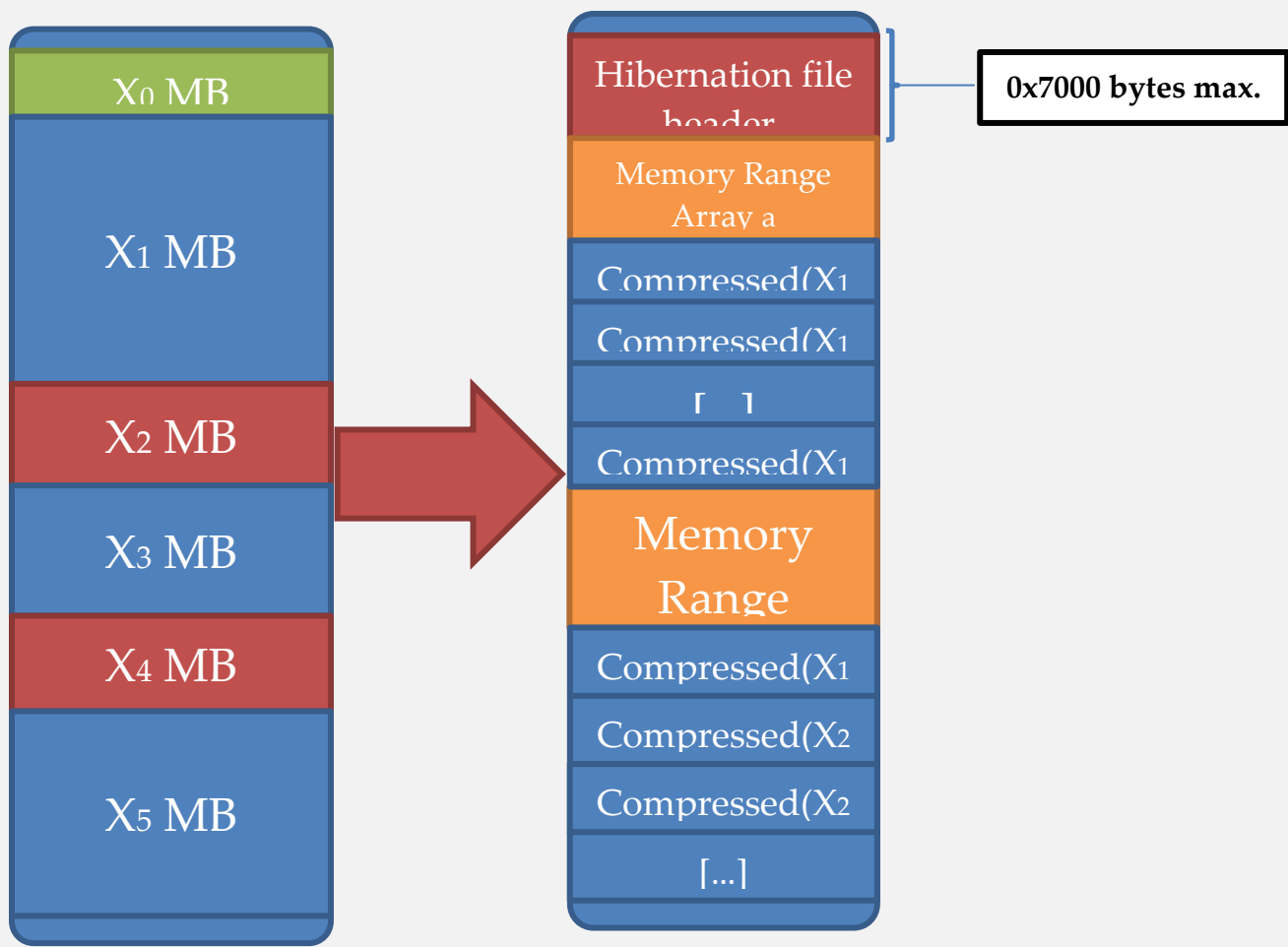
Microsoft Windows hibernation file

This feature also called “suspend to disk”, available since Windows 2000 also produces a memory snapshot on disk.

Unlike the BSOD, you do not need to generate an error to produce it. Moreover, it does not suffer of the 2GB limitation but of a 4GB limitation. And like the Microsoft crash dump file, it contains important information to make possible to the O.S. to resume its saved state.

The hibernation file can also be abused for “offensics” purposes, because it is writable but this is another story ☺

This format uses compression, and all pages are not in a linear order – which makes it



more difficult to be reused.

KPROCESSOR_STATE is in the hibernation file header (PO_MEMORY_IMAGE).

Name	Size	Description
Magic	8 bytes	"\x81\x81xpress" >= Win XP

PO_MEMORY_RANGE_TABLE		
PO_MEMORY_RANGE_TABLE	Next	Contains the page number of the next table.
ULONG32	NextTable	
ULONG32	EntryCount	Number of ranges.
PO_MEMORY_RANGE	Range	Array of range.

PO_MEMORY_RANGE		
ULONG32	StartPage	Start of page range in memory.
ULONG32	EndPage	End of page range in memory.

Info	4 bytes	Contains compressed size + uncompressed size in bit field format. Compressed size is used to get a pointer to the next compressed block.
		$\text{CompressedSize} = ((\text{Info} \gg 10) + 1);$
		$\text{NumberOfUncompressedPages} = ((\text{Info} \& 0x3ff) + 1);$
Padding	16	Filled with null bytes.

As you can see, the structure is totally more complex than the two previous file format.

MoonSols Windows Memory Toolkit

MoonSols Windows Memory Toolkit is the ultimate toolkit for memory dump conversion and acquisition on Windows. This toolkit had been designed to deal with various types of memory dumps such as VMWare memory snapshot, Microsoft crash dump and even Windows hibernation file.

Toolkit works with every version of Windows from Windows XP to Windows 7, with x86 and x64 architectures.

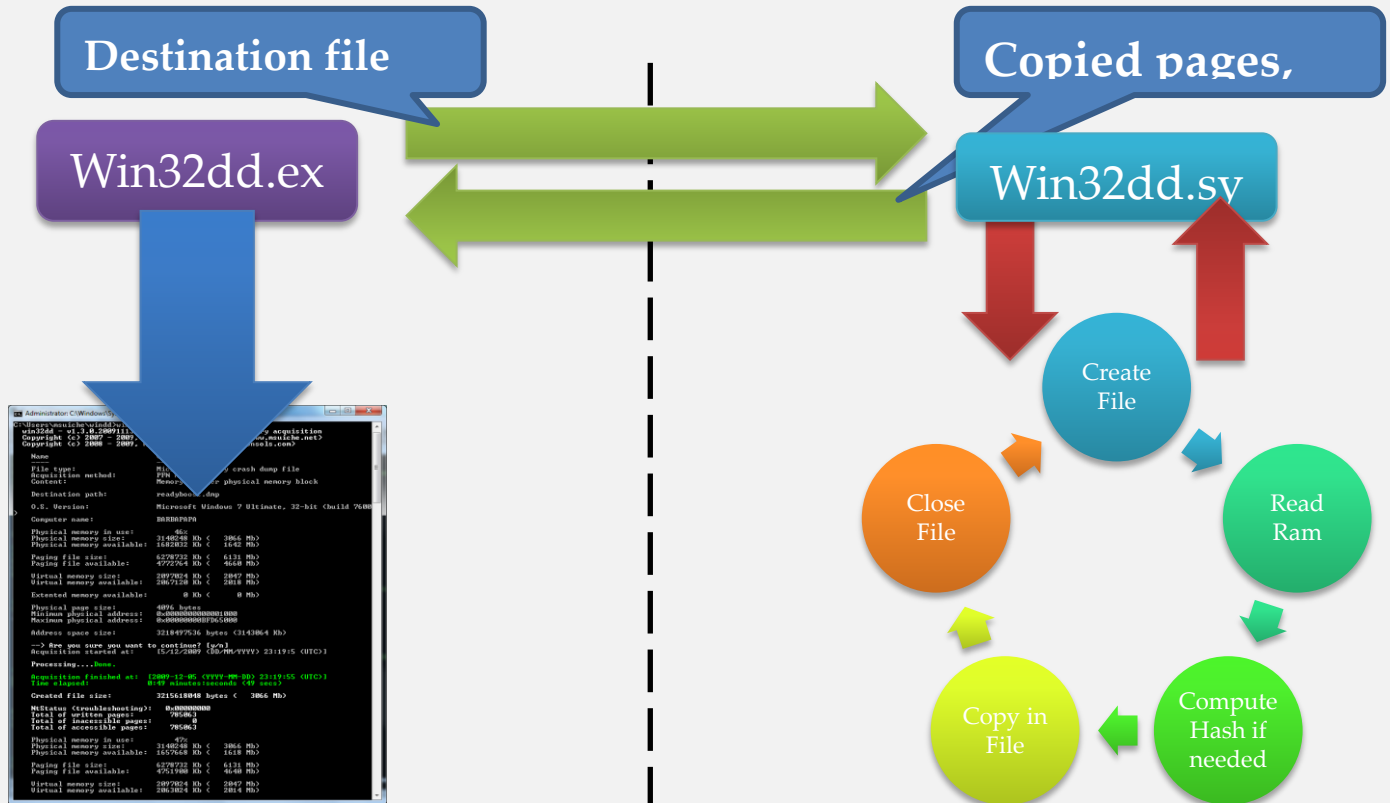
The toolkit includes :

- **Win32dd**
- **Win64dd**
- **Hibr2bin**
- **Hibr2dmp**
- **Dmp2bin**
- **Bin2dmp**

Win32dd Win64dd

These utilities aims at creating a full physical memory dump of a machine, either a linear format or in a Microsoft crash dump.

A feature for remote acquisition over TCP is also present.



```
C:\Windows\system32\cmd.exe - win64dd.exe /f /f D:\Dumps\Windows\Crash\remote.dmp
I:\MoonSols\Products>win64dd.exe /f /f D:\Dumps\Windows\Crash\remote.dmp

I:\MoonSols\Products>win64dd.exe
win64dd - 1.3.1.20100405 - (Professional Edition - Single User Licence)
Kernel land physical memory acquisition
Copyright (C) 2007 - 2010, Matthieu Suiche <http://www.msuiche.net>
Copyright (C) 2009 - 2010, MoonSols <http://www.moonsols.com>

Remote server: 0.0.0.0:1337
Remote client: 127.0.0.1
Acquisition started at: [2/6/2010 (DD/MM/YYYY) 11:36:17 (UTC)]
Processing...Done.
Acquisition finished at: [2010-06-02 (YYYY-MM-DD) 11:36:17 (UTC)]
Time elapsed: 1:21 minutes:seconds (81 secs)
--> 4147847168 bytes received
```

```
C:\Windows\system32\cmd.exe - win64dd.exe /t localhost /d
I:\MoonSols\Products>win64dd.exe /t localhost /d

I:\MoonSols\Products>win64dd.exe
Paging file size: 8099348 Kb ( 7909 Mb)
Paging file available: 5967432 Kb ( 5827 Mb)

Virtual memory size: 8589934464 Kb (8388607 Mb)
Virtual memory available: 8589882020 Kb (8388556 Mb)

Extended memory available: 0 Kb ( 0 Mb)

Physical page size: 4096 bytes
Minimum physical address: 0x0000000000001000
Maximum physical address: 0x00000000137FFF000

Address space size: 5234491392 bytes (5111808 Kb)

--> Are you sure you want to continue? [y/n] y
Acquisition started at: [2/6/2010 (DD/MM/YYYY) 11:36:17 (UTC)]
Processing...Done.
Acquisition finished at: [2010-06-02 (YYYY-MM-DD) 11:36:39 (UTC)]
Time elapsed: 1:21 minutes:seconds (81 secs)
Created file size: 4147847168 bytes ( 3955 Mb)

NtStatus (troubleshooting): 0x00000000
Total of written pages: 1012658
Total of inaccessible pages: 0
Total of accessible pages: 1012658

MD5: 5B2044C42650FB0715DAF769F737A485

Physical memory in use: 41%
Physical memory size: 4050624 Kb ( 3955 Mb)
Physical memory available: 2384772 Kb ( 2328 Mb)

Paging file size: 8099348 Kb ( 7909 Mb)
Paging file available: 5993368 Kb ( 5852 Mb)

Virtual memory size: 8589934464 Kb (8388607 Mb)
Virtual memory available: 8589882020 Kb (8388556 Mb)

Extended memory available: 0 Kb ( 0 Mb)

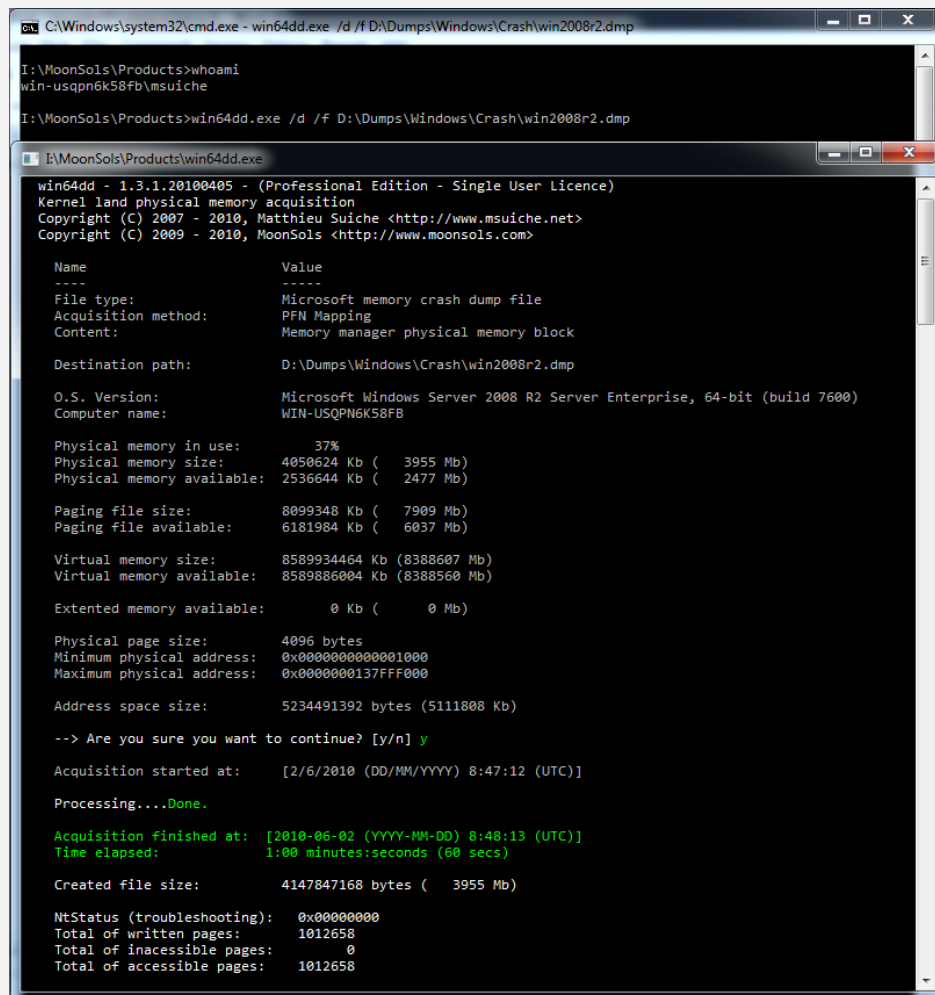
Physical page size: 4096 bytes
Minimum physical address: 0x0000000000001000
Maximum physical address: 0x00000000137FFF000

Address space size: 5234491392 bytes (5111808 Kb)
```

Server Side

Client

Comman



The screenshot shows two overlapping windows. The top window is a command prompt with the following text:

```
C:\Windows\system32\cmd.exe - win64dd.exe /d /f D:\Dumps\Windows\Crash\win2008r2.dmp
I:\MoonSols\Products>whoami
win-usqp6k58fb\msuiche
I:\MoonSols\Products>win64dd.exe /d /f D:\Dumps\Windows\Crash\win2008r2.dmp
```

The bottom window is titled "E:\MoonSols\Products\win64dd.exe" and displays the following information:

```
win64dd - 1.3.1.20100405 - (Professional Edition - Single User Licence)
Kernel land physical memory acquisition
Copyright (C) 2007 - 2010, Matthieu Suiche <http://www.msuiche.net>
Copyright (C) 2009 - 2010, MoonSols <http://www.moonsols.com>

Name          Value
----          -
File type:     Microsoft memory crash dump file
Acquisition method: PFM Mapping
Content:       Memory manager physical memory block

Destination path: D:\Dumps\Windows\Crash\win2008r2.dmp

O.S. Version:   Microsoft Windows Server 2008 R2 Server Enterprise, 64-bit (build 7600)
Computer name:  WIN-USQP6K58FB

Physical memory in use: 37%
Physical memory size: 4050624 Kb ( 3955 Mb)
Physical memory available: 2536644 Kb ( 2477 Mb)

Paging file size: 8099348 Kb ( 7909 Mb)
Paging file available: 6181984 Kb ( 6037 Mb)

Virtual memory size: 8589934464 Kb (8388607 Mb)
Virtual memory available: 8589886004 Kb (8388560 Mb)

Extended memory available: 0 Kb ( 0 Mb)

Physical page size: 4096 bytes
Minimum physical address: 0x0000000000001000
Maximum physical address: 0x00000000137FFF000

Address space size: 5234491392 bytes (5111808 Kb)

--> Are you sure you want to continue? [y/n] y

Acquisition started at: [2/6/2010 (DD/MM/YYYY) 8:47:12 (UTC)]

Processing....Done.

Acquisition finished at: [2010-06-02 (YYYY-MM-DD) 8:48:13 (UTC)]
Time elapsed: 1:00 minutes:seconds (60 secs)

Created file size: 4147847168 bytes ( 3955 Mb)

NtStatus (troubleshooting): 0x00000000
Total of written pages: 1012658
Total of inaccessible pages: 0
Total of accessible pages: 1012658
```

The figure above shows how windd is organized.

Create a raw memory dump:

windd.exe /f dump.bin

Create a Microsoft crash memory dump:

windd.exe /d /f dump.dmp

Put windd in server mode:

win64dd /l /f F:\moonsols.dmp

To send the file (Microsoft crash dump) to the server:

win64dd /t sample.moonsols.com /d

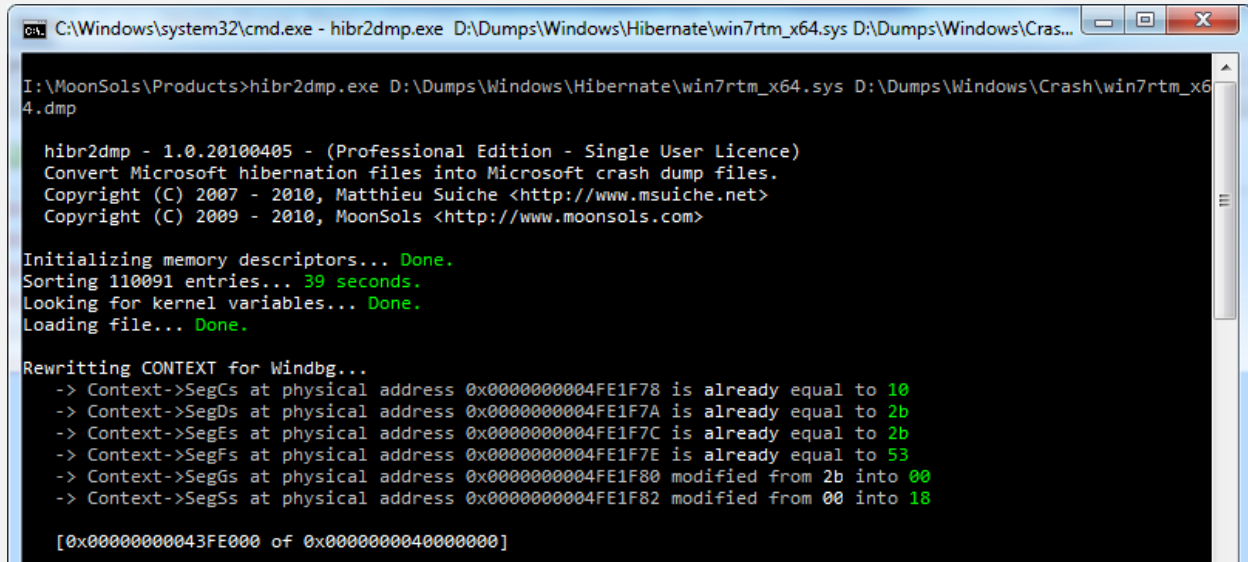
Hibr2dmp Hibr2bin

These utilities aims at converting Microsoft hibernation file either in a linear memory dump or in a Microsoft crash dump file with a single command line.

Hibr2dmp.exe hibernation_file_input.sys crashdump_output.dmp

Hibr2bin.exe hibernation_file_input.sys linear_memory_dump_output.bin

Both commands uncompress and reconstruct the hibernation file to either a Microsoft crash dump file or a linear memory dump.



```
C:\Windows\system32\cmd.exe - hibr2dmp.exe D:\Dumps\Windows\Hibernat\win7rtm_x64.sys D:\Dumps\Windows\Cras...
I:\MoonSols\Products>hibr2dmp.exe D:\Dumps\Windows\Hibernat\win7rtm_x64.sys D:\Dumps\Windows\Cras\win7rtm_x64.dmp

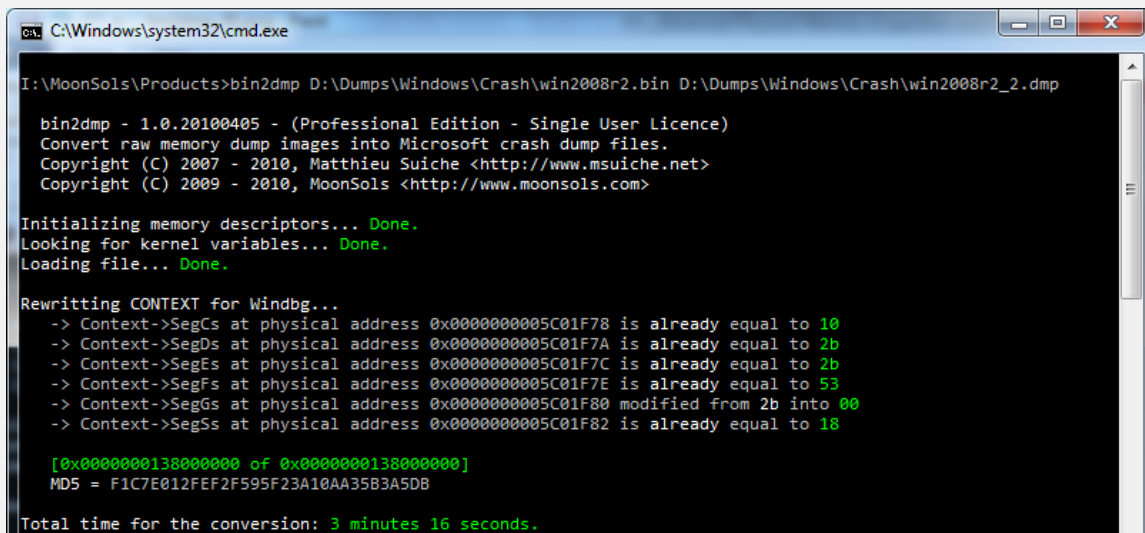
hibr2dmp - 1.0.20100405 - (Professional Edition - Single User Licence)
Convert Microsoft hibernation files into Microsoft crash dump files.
Copyright (C) 2007 - 2010, Matthieu Suiche <http://www.msuiche.net>
Copyright (C) 2009 - 2010, MoonSols <http://www.moonsols.com>

Initializing memory descriptors... Done.
Sorting 110091 entries... 39 seconds.
Looking for kernel variables... Done.
Loading file... Done.

Rewriting CONTEXT for Windbg...
-> Context->SegCs at physical address 0x0000000004FE1F78 is already equal to 10
-> Context->SegDs at physical address 0x0000000004FE1F7A is already equal to 2b
-> Context->SegEs at physical address 0x0000000004FE1F7C is already equal to 2b
-> Context->SegFs at physical address 0x0000000004FE1F7E is already equal to 53
-> Context->SegGs at physical address 0x0000000004FE1F80 modified from 2b into 00
-> Context->SegSs at physical address 0x0000000004FE1F82 modified from 00 into 18

[0x0000000043FE000 of 0x0000000040000000]
```

Bin2dmp



```
C:\Windows\system32\cmd.exe
I:\MoonSols\Products>bin2dmp D:\Dumps\Windows\Crash\win2008r2.bin D:\Dumps\Windows\Crash\win2008r2_2.dmp

bin2dmp - 1.0.20100405 - (Professional Edition - Single User Licence)
Convert raw memory dump images into Microsoft crash dump files.
Copyright (C) 2007 - 2010, Matthieu Suiche <http://www.msuiche.net>
Copyright (C) 2009 - 2010, MoonSols <http://www.moonsols.com>

Initializing memory descriptors... Done.
Looking for kernel variables... Done.
Loading file... Done.

Rewriting CONTEXT for Windbg...
-> Context->SegCs at physical address 0x0000000005C01F78 is already equal to 10
-> Context->SegDs at physical address 0x0000000005C01F7A is already equal to 2b
-> Context->SegEs at physical address 0x0000000005C01F7C is already equal to 2b
-> Context->SegFs at physical address 0x0000000005C01F7E is already equal to 53
-> Context->SegGs at physical address 0x0000000005C01F80 modified from 2b into 00
-> Context->SegSs at physical address 0x0000000005C01F82 is already equal to 18

[0x0000000138000000 of 0x0000000138000000]
MD5 = F1C7E012FEF2F595F23A10AA35B3A5DB

Total time for the conversion: 3 minutes 16 seconds.
```

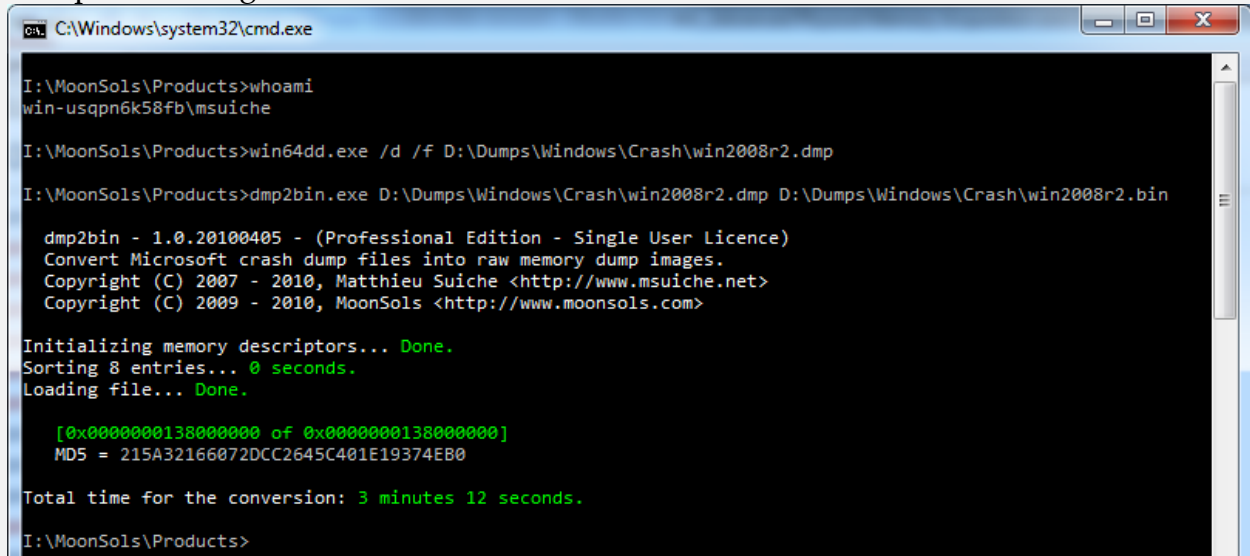
Bin2dmp aims at converting a linear memory dump into a Microsoft crash dump with a single command line.

bin2dmp.exe linear_memory_dump_input.bin Microsoft_crash_dump_output.dmp

KPCR may requires to be fixed when converting a memory snapshot into a Microsoft crash dump for a 100% compatibility with WinDbg.

Dmp2bin

Dmp2bin aims at converting a full memory Microsoft crash dump into a linear memory dump with a single command line.



```
C:\Windows\system32\cmd.exe

I:\MoonSols\Products>whoami
win-usqpn6k58fb\msuiche

I:\MoonSols\Products>win64dd.exe /d /f D:\Dumps\Windows\Crash\win2008r2.dmp

I:\MoonSols\Products>dmp2bin.exe D:\Dumps\Windows\Crash\win2008r2.dmp D:\Dumps\Windows\Crash\win2008r2.bin

dmp2bin - 1.0.20100405 - (Professional Edition - Single User Licence)
Convert Microsoft crash dump files into raw memory dump images.
Copyright (C) 2007 - 2010, Matthieu Suiche <http://www.msuiche.net>
Copyright (C) 2009 - 2010, MoonSols <http://www.moonsols.com>

Initializing memory descriptors... Done.
Sorting 8 entries... 0 seconds.
Loading file... Done.

[0x0000000138000000 of 0x0000000138000000]
MD5 = 215A32166072DCC2645C401E19374EB0

Total time for the conversion: 3 minutes 12 seconds.

I:\MoonSols\Products>
```

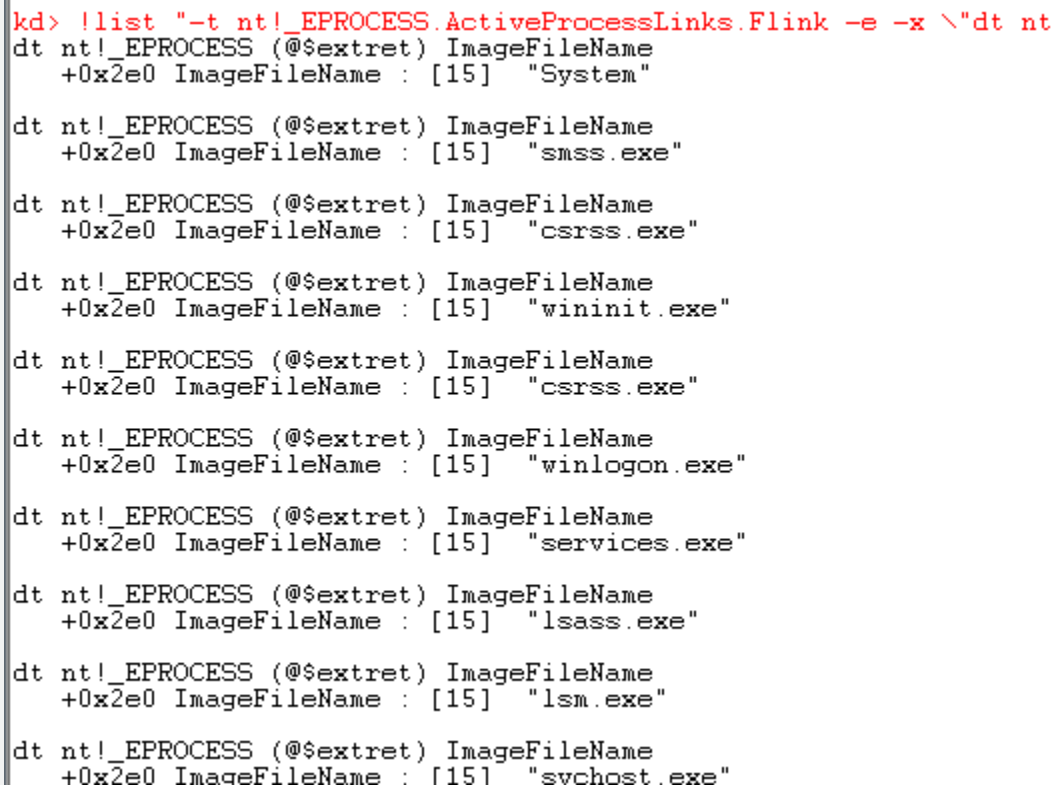
`dmp2bin.exe Microsoft_crash_dump_input.dmp linear_memory_dump_output.bin`

WinDbg

The advantage to use Microsoft crash dump is that this format is readable by Microsoft WinDbg, and this one also provides advanced features including a rich scripting language, plus several existing functions to do advanced manipulation like reading the Registry.

```
!list "-t nt!_EPROCESS.ActiveProcessLinks.Flink -e -x \"dt nt!_EPROCESS (@$extret) ImageFileName\" (poi(nt!PsActiveProcessHead)-@@c++(#FIELD_OFFSET(nt!_EPROCESS, ActiveProcessLinks)))"
```

The command above does exactly the same job that the command “!process 0 0” does. This command will work with any version of the target memory dump if symbols are loaded.



```
kd> !list "-t nt!_EPROCESS.ActiveProcessLinks.Flink -e -x \"dt nt
dt nt!_EPROCESS (@$extret) ImageFileName
+0x2e0 ImageFileName : [15] \"System\"

dt nt!_EPROCESS (@$extret) ImageFileName
+0x2e0 ImageFileName : [15] \"smss.exe\"

dt nt!_EPROCESS (@$extret) ImageFileName
+0x2e0 ImageFileName : [15] \"csrss.exe\"

dt nt!_EPROCESS (@$extret) ImageFileName
+0x2e0 ImageFileName : [15] \"wininit.exe\"

dt nt!_EPROCESS (@$extret) ImageFileName
+0x2e0 ImageFileName : [15] \"csrss.exe\"

dt nt!_EPROCESS (@$extret) ImageFileName
+0x2e0 ImageFileName : [15] \"winlogon.exe\"

dt nt!_EPROCESS (@$extret) ImageFileName
+0x2e0 ImageFileName : [15] \"services.exe\"

dt nt!_EPROCESS (@$extret) ImageFileName
+0x2e0 ImageFileName : [15] \"lsass.exe\"

dt nt!_EPROCESS (@$extret) ImageFileName
+0x2e0 ImageFileName : [15] \"lsm.exe\"

dt nt!_EPROCESS (@$extret) ImageFileName
+0x2e0 ImageFileName : [15] \"svchost.exe\"
```

The figure above is a screenshot of the command used on a Windows 7 x64 hibernation file converted into a Microsoft crash dump via hibr2dmp.exe utility.

Reference

http://msdn.moonsols.com/win7rtm_x64/PO_MEMORY_IMAGE.php

http://msdn.moonsols.com/winvistasp2_x64/PO_MEMORY_RANGE_TABLE.php

<http://msdn.moonsols.com>

<http://moonsoles.com/component/jdownloads/view.download/3/2>