WinDbg Cheat Sheet - Data Structures, Commands and Extensions

Latest Version – 13/11/14
Author – Harry Miller (@CompSciMaths)
Blog – http://bsodtutorials.wordpress.com/

Contents:

Processes and Threads
Stack Traces and Frames
Pool Allocations and Troubleshooting Pools
Troubleshooting Memory
Objects and Handles
Processors, IRQLs and Interrupts
I/O and IRPs
Driver Verifier
System Hardware Information
Thread Scheduling
Registers and Exceptions
Memory Manipulation
Driver Information
Power Policy
SwishDbgExt (Custom Extensions)
ProcdumpExt (Custom Extensions)
Local Inter-Process Calls (LPCs)
Windows Registry
Windows Heap Manager
Windows Access Tokens
Miscellaneous

Processes and Threads:

!process (Lists _EPROCESS) – This extension displays similar information to the _EPROCESS data structure, and can be used for displaying all the active processes running on the system.

To list all active processes, then simply use !process 0 0. Otherwise, to display information about a specific process, then use !process [ImageName]/[Address].

Syntax: !process [/s Session] [/m Module] [Process [Flags]]

Flags:

0 – Time and Priority Statistics.

1 – List of threads, events and wait states for the process.
2 – List of threads which belong to the process. This bit combined with bit 1 will produce threads

with stack information provided too.

3 – Displays return address and stack pointer for each function.

!processfields (Omitted _EPROCESS) – This extension will display the offsets and field names within the _EPROCESS data structure.

.tlist – Lists all the processes running on the system.

!peb – Displays the process environment block

Data Structures:

_EPROCESS, _KPROCESS, _PEB

Global Variables:

PsActiveProcessHead – head of a doubly linked list of all active processes on a given processor.
PsInitialSystemProcess – the starting process (System) and the first process within the linked list.

!thread (Lists _ETHREAD) – This extension will display information similar to the _ETHREAD data structure. It is useful for gathering thread information, and setting the debugger context to the context of that thread. The .thread command is also used for setting to a local thread context.

Syntax: !thread [-p] [-t] [Address [Flags]]

Parameters:

The -p parameter can be added which will display information about the process which owns the thread.

The -t parameter can be used to specific to use the Thread ID (TID) when viewing a specific thread, instead of the thread's hexadecimal address.

Flags:

1 – Displays information about the thread's wait states

2 – This flag must be used with flag bit 1, otherwise it will not work; displays stack trace along with thread wait states.

3 – The return addresses and stack pointers will added to the stack trace.

4 – Sets the process context to the process which owns the threads being examined (Better stack traces)

!threadfields (Omitted _ETHREAD) – The concept is the same as the !processfields extension.
!tp – Displays Thread Pool Information; thread pools are used to manage worker threads on behalf of a process.

Syntax: !tp pool Address [Flags], !tp tqueue Address [Flags] (Check WinDbg for more options).

Flags:

1 – Display as a single-line

2 – Displays member information

3 – This flag will only work with the pool option. It will display pool work queues and/or pool work queues at normal priority and NUMA nodes.

!teb – Displays the Thread Environment Block

Data Structures:

_ETHREAD, _KTHREAD, _TEB

Stack Traces and Stack Frames:

k – Smallest amount of information (ChildEBP (Stack Frame Address), Return Address, Function Name)

knL – Stack Frame Number, Stack Frame Address, Return Address, Function Name

kb – Stack Frame Address, Return Address, Arguments to the Function, Function Name

kv – Stack Frame Address, Return Address, Arguments to the Function, Function Name , FPO (Frame Pointer Omission) Optimization, Trap Frames

.frame [/r] [Frame Number] – Displays the current stack frame and registers [/r]

!uniqstack [-b | -v | -p] [-n] – Displays all stacks for all threads within the current process

Parameters:

b – Displays first three parameters for each function

v – Displays FPO information if applicable

p – Displays all parameters for each function with data type

n – Displays the frame numbers

!findstack [Symbol] [Display Level] – Displays stacks with a specified module or symbol

Parameters (Display Level):

0 – Displays only the TID for each thread

1 – Displays both TID and stack frame for each thread

2 – Displays the entire stack for each thread

Pool Allocations and Troubleshooting Pools:

!frag [Flags] – Displays information about pool fragmentation in pool regions.

0 – Number and size of fragments for each index

1 – Pool allocation information with pool tags

!poolfind Tag String [Pool Type] – Searches paged and non-paged pool for pool allocations associated with the specified pool tag.

Pool Types: 0 = Non-Paged Pool, 1 = Paged Pool, 2 = Special Pool 4 = Session Pool

!pooltag Tag String – Displays information about a pool tag string.

!poolused [Flags[Tag String]] – Displays pool allocations for a specified pool tag string

Flags:

0 – Displays more detailed information

1 – Sort by non-paged pool usage

2 – Sort by paged pool usage

3 – Displays session pool instead of paged/non-paged pool

!pool [Address [Flags]] – Displays information about a specific pool allocation or pool page.

Flags:

0 – Displays pool contents and pool headers

1 – Suppresses pool header information for all the pool pages, apart from the pool page which contains the pool address given

31 – Suppresses pool type and pool tag

!poolval Address [Display Level] – Checks pool headers for consistency and any possible pool corruption

Data Structures - _POOL_HEADER, _POOL_TYPE, _POOL_DESCRIPTOR

Troubleshooting Memory:

!vm [Flags] – Displays summary information about virtual memory

Flags:

0 – Omits process specific information

1 – Displays memory management stacks

2 – Displays Terminal Server Usage

3 – Displays Page File Write Log

4 – Displays Working Set Owner thread stacks

5 – Displays Kernel Address Space usage statistics

!vprot [Address] – Displays the virtual memory protection bits for a address.

!vadump [-v] – Displays the virtual memory protection bits for all address ranges. -v will display the original allocation information, since this can be changed by certain functions.

!pte [Virtual Address/PTE] – Displays the PTE and PDE for a given virtual address with PTE status bits.

!pfn [Page Frame] – Displays information about a given PFN entry.

!memusage [Flags] – Displays summary statistics about physical memory.

Flags:

8 (Most useful flag) – General summary information about memory use.

!sysptes [Flags] – Displays information about system PTEs.

!ptov [PFN] – Shows the physical to virtual mapping between pages for a given process.

PFN is the first four bits of the Directory Base for the process.

Data Structures: _MI_SYSTEM_VA_TYPE, _HARDWARE_PTE, _MI_SYSTEM_PTE_TYPE, _MMPTE_PROTOTYPE, _MMLISTS, _MMPFNENTRY, _MMPFN

Objects and Handles:

!object [Address/Name] – Displays information about a system object.

!obja [Address] – Displays object attributes information about a given object.

!handle [Handle [KMFlags [Process [Type Name]]]]

Handle = Index of the handle; -1 can be used to display all handles for a process.

Type Name = The name of the type of handle you wish to examine.

0 - Displays basic handle information

1 – Displays information about objects

2 – Displays free handle entries.

4 – Display handle from the kernel handle table instead of the process

5 – Interprets *Handle* as a TID/PID and then gives information about that object

!sd [Address] – Displays the security descriptor information for a given object.

Address: Use the address found in Security Descriptor field of the _OBJECT_HEADER

Data Structures: _OBJECT_HEADER, _OBJECT_TYPE, _OBJECT_ATTRIBUTES, _OBJECT_HANDLE_INFORMATION

Processors, Interrupts and IRQLs:

!irql – Displays the IRQL level of a given processor.

!idt [IDT] [-a] – Displays the IDT for a given processor(s). -a will displays the ISRs for each IDT entry too.

!ipi [Processor] – Displays information about IPIs for a given processor(s).

~ *[Processor]* – Changes to a specified processor.

!qlocks – Displays information about global queued spinlocks.

!locks [-v | -p | -d] [Address] – Displays information about ERESOURCE locks. *-v* will display detailed information, *-p* will include performance information and *-d* displays all locks including who which do not have any contention. Address is the address of the ERESOURCE structure, otherwise all locks are displayed.

!timer – Displays a list of timer objects.

!dpcs [Processor] – Displays the DPC Queue for a given processor.

!pcr [Processor] – Displays the PCR for a given processor.

!prcb [Processor] – Displays the PRCB for a given processor.

Data Structures: _KINTERRUPT, _KAPC, _KAPC_STATE, _ERESOURCE, _KSEMAPHORE, _DISPATCHER_HEADER, _FAST_MUTEX, _KEVENT, _KOBJECTS, _KWAIT_BLOCK, _KPCR, _KPRCB, _KDPC, _GROUP_AFFINITY, _KTIMER, _KTIMER_TABLE, _KTIMER_TABLE_ENTRY.

I/O and IRPs:

!irp [Address] [Detail] – Displays information about a I/O packet (IRP). *Address* is the address of the IRP. The *Detail* (any number) will display additional information such as owning thread, status of

the IRP, address of the corresponding MDL and stack locations of the IRP.

!irpfind [-v] [PoolType [Restart Address [Criteria] Data]]] – Searches for a specific IRP matching the criteria.

PoolType specifies which pool to search through; *Restart Address* specifies which address to start at; *Criteria* specifies which type of data to search for and *Data* specifies the data to look for e.g. specific driver name

-v = Displays additional information.

!devobj – Displays information about the _DEVICE_OBJECT data structure.

!drvobj – Displays information about the _DRIVER_OBJECT data structure.

!devnode – Displays information about a device node within the device tree. *!devnode 1* will display all pending removals of device objects, *!devnode 2* will show all pending ejects of a device object and *!devnode 0 1* will display the entire device tree.

!devstack [Device Object] – Displays the device stack for a device object. *Device Object* is the address of the device object data structure.

Data Structures: _DEVICE_OBJECT, _DRIVER_OBJECT, _IRP, _MDL, _IO_STATUS_BLOCK, _IO_STACK_LOCATION

Driver Verifier:

!deadlock – Displays information about a deadlock detected by Driver Verifier. Using *!deadlock 1* will display the stacks of the deadlocked threads.

!verifier – Shows the status of Driver Verifier. For the flags consult the WinDbg index.

System Hardware Information:

!sysinfo [machineid | cpuinfo | cpuspeed] - Displays hardware information about a given processor. Please note I've only added the three most useful parameters for this extension.

Machineid – Displays general BIOS information and motherboard information.

Cpuinfo – Displays general information about the processor.

Cpuspeed – Displays the current and stock clockspeed of the processor.

!whea – Displays a formatted view of the _WHEA_ERROR_RECORD_HEADER.

!errpkt [*Address*] – Displays a formatted view of the _WHEA_ERROR_PACKET.

!errrec [*Address*] – Displays a formatted view of the _WHEA_ERROR_RECORD.

!tz/!tzinfo – Displays hardware temperature information in Kelvins. The information is gathered

from ACPI tables. Check the ACPI documentation for more details.

!cpuinfo – Displays information about the processor such as family, model and stepping.

!cpuid – Displays similar information to !cpuinfo, but will include processor clock speed.

Data Structures: _WHEA_ERROR_RECORD, _WHEA_ERROR_RECORD_HEADER, _WHEA_ERROR_RECORD_HEADER_VALIDBITS, _WHEA_TIMESTAMP, _WHEA_ERROR_SOURCE_TYPE, _WHEA_ERROR_PACKET,

Thread Scheduling:

!running -ti – Displays a list of all the running threads on all processors.

!runaway [Flags] – Displays how long a thread has been running for. This can only be used for live debugging and crash dumps created by **.dump /mt**.

Flags:

0 – Displays amount User-Mode time which has been consumed.

1 – Displays the amount of Kernel-Mode time which has been consumed.

2 – Displays the amount of time consumed since the creation of the thread.

Registers and Exceptions:

r – Displays the contents of a register.

.trap – Displays a formatted view of the _KTRAP_FRAME data structure with registers.

u – Produces assembly code translation for program code. Use *ub* for a specific memory range.

Memory Manipulation:

da – Displays the memory contents of an array.

dd – Displays the memory contents within a given address range.

dps – Displays memory with symbol information if possible.

dds – Similar to dps but with Double Word data lengths.

Driver Information:

lmvm [Module Name] – Displays detailed information about a driver module. This includes a timestamp.

lm – Displays loaded modules at the time of the crash.

lmstm – Displays detailed information about modules at the time of the crash.

Power Policy:

!popolicy – Displays power related information about the current user. This is the settings configured within the SYSTEM_POWER_POLICY data structure.

!pocaps - Displays information in relation to the power capabilities of the system, this is ideal for checking if drivers are attempting to use a unsupported sleep state.

!poaction – Displays a list of outstanding Power IRPs called using the *PoRequestPowerIrp* function. The function will create a Power IRP and send it to the top of device stack for a given device object.

The list of power IRPs will be shown under the *FieldOffset* field. The extension will provide the device object, driver object and the nature of the power IRP.

!podev – Provides power related information about a PnP device object.

!poaction - The *!poaction* extension will provide the current power action, and a list of devices which are currently being powered off or down. It also provides a list of completed IRPs.

SwishDbgExt – Custom Extensions (written by Matt Suishe):

!ms_timers – It has a similar nature to the *!timer* extension, however, it is able to provide information about hooking in the _KTIMER_TABLE.
!ms_idt – Provides the same information as *!idt* with the additional of hooking detection.

!ms_ssdt – This will dump the SSDT and provide any information on if functions have been hooked or patched.

!ms_gdt – This will dump the GDT and LDT (LDT is stored within the GDT).

!ms_drivers - The *!ms_drivers* extension is basically the same as the *lm* or *lmnst* command. There are some additional parameters you can add to the *!ms_drivers* extension to spice up the command. The *!ms_drivers /scan* extension can be used to find drivers using IRP Hooking.

IRP Hooking involves a hook within the array stored within the DRIVER_OBJECT structure, this array or table of IRP_MJ_ functions is hooked and the code responsible for the IRP is redirected to malicious code. Please note hooking is used for legitimate processes such as debugging and patch releases.

Note: If you wish to use the ProcDumpExt DLL for WinDbg, and also view the help information for the extensions provided in SwishDbgExt, then you'll need to unload ProcDumpExt first since ProcDumpExt will overload the *!help* extension with it's own version. You can simply load ProcDumpExt again afterwards. Alternatively, if you do not wish to unload the ProcDumpExt DLL, then simply use the longhand method of *!SwishDbgExt.help <SwishDbgExt Extension>*.

ProcDumpExt – Custom Extensions (written by Andrew Richards):

!dpx – Displays the entire stack for the thread, with combination of dps, dpu, dpp and dpa

commands.

!dtr – A slightly more detailed version of *r,* with the addition of idtr and gdtr.

!msr – Displays model specific registers

!procdumpext.help – Displays all available extensions with a short description

Local Inter-Process Calls:

Please note that !lpc applies to Windows XP and earlier operating systems, whereas, *!alpc* only works with later versions.

!alpc /lpp [Address of Process] – Displays all connections for that process. This includes ports created by the process, and ports which the process is currently connected to.

!alpc /p [Port Address] – Displays information regarding the specified port. This includes port, server communication port, client communication port, connection port and message queue information.

!alpc /m [Message Address] – Displays information in relation to the specified message.

Data Structures - _CLIENT_ID, _LPCP_PORT_OBJECT, _LPCP_PORT_QUEUE, _LPCP_NONPAGED_PORT_QUEUE, _KALPC_MESSAGE, _LPCP_MESSAGE, _PORT_MESSAGE

More information about LPCs can be found on my blog.

Windows Registry:

!reg hivelist – Displays HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\hivelist

!reg dumppool – Displays paged pool consumption of registry hives.

!reg viewlist – Displays the mapped views for a selected hive.

!reg freebins – Displays free hive bins.

!reg openkeys – Displays open keys.

!reg kcb – Displays the Key Control Block for a registry key, this will require the use of *!reg findkcb*.

!reg findkcb – This will give the address of the KCB for the specified file path of the registry key.
!reg cellindex – Gives information corresponding to a cell index.

Data Structures: _CMHIVE, _HHIVE, _CM_CELL_DATA, _CM_KEY_CONTROL_BLOCK, _CM_KEY_HASH, _HMAP_DIRECTORY, _HMAP_TABLE, _HMAP_TABLE_ENTRY

Windows Heap Manager:

!heap – Displays currently active heaps.

Parameters:

stat – Provides greater quick summary information about active heaps.

m – Shows segment entries for specified heap address.

s – Shows memory consumption statistics for all heaps within the process address space.

i – Displays heap debugging flags such as _HEAP_ENTRY_BUSY.

l – Detects memory leaks within the heap.

!address – summary: Displays the process address space and statistics about memory consumption related to that process address space. This includes heap statistics.

Data Structures: _HEAP, _HEAP_LOCK, _HEAP_TUNING_PARAMETERS, _DPH_BLOCK_INFORMATION.

Windows Access Tokens:

!token – Gives information about a specified access token.

Data Structures: _TOKEN_TYPE, _TOKEN, _SID, _SID_NAME_USE, _TOKEN_SOURCE, _SEP_TOKEN_PRIVILEGES, _SECURITY_IMPERSONATION_LEVEL, _TOKEN_CONTROL, _SID_AND_ATTRIBUTES_HASH

Miscellaneous:

.chain – Displays all loaded .DLLs for the dump file.

.time – Shows the system time for the dump file.

vertarget – Displays operating system information for the dump file.

!exchain – Lists all the exception handlers within the stack with the stack frame number.

!validatelist – Shows if a linked list is corrupt.

dl – Transverses through a doubly linked list. The *!dblink* and *!dflink* provide similar functionality; flink transverses forward, whereas, blink will transverse backwards. This command/extension will require the address of a _LINKED_LIST structure.

Data Structures: _LINKED_LIST, _LIST_ENTRY

!chkimg – Verifies the symbol of an binary image file (.EXE) within the dump file against the symbol stored within the Microsoft Symbols store or local symbol store. This is used to detect if a binary file is corrupt.

Parameters:

f – This will fix any corruption found within the image, by transferring the symbols from the symbol store to the dump file.

v – Displays verbose information.

d – Shows the number of mismatch errors.

ln – Displays symbol information at or near the given address.

.reload – Reloads the symbols from the symbol store, this is useful if there are symbol errors or you have added additional symbol files (.pdb)

!sym noisy/quiet – By specifying the noisy or quiet parameter, you can control if symbol prompts are shown when loading symbols.

!load/!unload – Loads or unloads a .DLL for the dump file.

For additional information please consult the WinDbg documentation.