ProcDump v11.0

Article • 12/12/2022 • 8 minutes to read

By Mark Russinovich and Andrew Richards

Published: 11/03/2022

Download ProcDump **(714 KB)**

Download ProcDump for Linux (GitHub)

https://www.microsoft.com/en-us/videoplayer/embed/RE591St?autoplay=true&loop=true&controls=false&postJsllMsg=true

Created with ZoomIt

Introduction

ProcDump is a command-line utility whose primary purpose is monitoring an application for CPU spikes and generating crash dumps during a spike that an administrator or developer can use to determine the cause of the spike. ProcDump also includes hung window monitoring (using the same definition of a window hang that Windows and Task Manager use), unhandled exception monitoring and can generate dumps based on the values of system performance counters. It also can serve as a general process dump utility that you can embed in other scripts.

Using ProcDump

Capture Usage:

```cmd
procdump.exe [-mm] [-ma] [-mt] [-mp] [-mc <Mask>] [-md <Callback_DLL>] [-mk]
            [-n <Count>]
            [-s <Seconds>]
            [-c|-cl <CPU_Usage> [-u]]
            [-m|-ml <Commit_Usage>]
            [-p|-pl <Counter> <Threshold>]
            [-h]
            [-e [1] [-g] [-b] [-ld] [-ud] [-ct] [-et]]
            [-l]
            [-t]
            [-f  <Include_Filter>, ...]
            [-fx <Exclude_Filter>, ...]
            [-dc <Comment>]
            [-o]
            [-r [1..5] [-a]]
            [-at <Timeout>]
            [-wer]
            [-64]
            {
                {{[-w] <Process_Name> | <Service_Name> | <PID>} [<Dump_File>
| <Dump_Folder>]}
            |
                {-x <Dump_Folder> <Image_File> [Argument, ...]}
            }
```

Install Usage:

cmd

```
procdump.exe -i [Dump_Folder]
            [-mm] [-ma] [-mt] [-mp] [-mc <Mask>] [-md <Callback_DLL>] [-mk]
            [-r]
            [-at <Timeout>]
            [-k]
            [-wer]
```

Uninstall Usage:

cmd

```
procdump.exe -u
```

Dump Types:

Dump Type	Description
-mm	Write a 'Mini' dump file. (default) - Includes directly and indirectly referenced memory (stacks and what they reference). - Includes all metadata (Process, Thread, Module, Handle, Address Space, etc.).
-ma	Write a 'Full' dump file. - Includes all memory (Image, Mapped and Private). - Includes all metadata (Process, Thread, Module, Handle, Address Space, etc.).
-mt	Write a 'Triage' dump file. - Includes directly referenced memory (stacks). - Includes limited metadata (Process, Thread, Module and Handle). - Removal of sensitive information is attempted but not guaranteed.

Dump Type	Description
-mp	Write a 'MiniPlus' dump file. - Includes all Private memory and all Read/Write Image or Mapped memory. - Includes all metadata (Process, Thread, Module, Handle, Address Space, etc.). - To minimize size, the largest Private memory area over 512MB is excluded. A memory area is defined as the sum of same-sized memory allocations. The dump is as detailed as a Full dump but 10%-75% the size. - Note: CLR processes are dumped as Full (-ma) due to debugging limitations.
-mc	Write a 'Custom' dump file. - Includes the memory and metadata defined by the specified `MINIDUMP_TYPE` mask (Hex).
-md	Write a 'Callback' dump file. - Includes the memory defined by the `MiniDumpWriteDump` callback routine named `MiniDumpCallbackRoutine` of the specified DLL. - Includes all metadata (Process, Thread, Module, Handle, Address Space, etc.).
-mk	Also write a 'Kernel' dump file. - Includes the kernel stacks of the threads in the process. - OS doesn't support a kernel dump (`-mk`) when using a clone (`-r`). - When using multiple dump sizes, a kernel dump is taken for each dump size.

Conditions:

Condition	Description
-a	Avoid outage. Requires `-r`. If the trigger will cause the target to suspend for a prolonged time due to an exceeded concurrent dump limit, the trigger will be skipped.
-at	Avoid outage at Timeout. Cancel the trigger's collection at `N` seconds.
-b	Treat debug breakpoints as exceptions (otherwise ignore them).
-c	CPU threshold above which to create a dump of the process.
-cl	CPU threshold below which to create a dump of the process.
-dc	Add the specified string to the generated Dump Comment.
-e	Write a dump when the process encounters an unhandled exception. Include the `1` to create dump on first chance exceptions. Add `-ld` to create a dump when a DLL (module) is loaded (filtering applies). Add `-ud` to create a dump when a DLL (module) is unloaded (filtering applies). Add `-ct` to create a dump when a thread is created. Add `-et` to create a dump when a thread exits.

Condition	Description
-f	Filter (include) on the content of exceptions, debug logging and filename at DLL load/unload. Wildcards (*) are supported.
-fx	Filter (exclude) on the content of exceptions, debug logging and filename at DLL load/unload. Wildcards (*) are supported.
-g	Run as a native debugger in a managed process (no interop).
-h	Write dump if process has a hung window (does not respond to window messages for at least 5 seconds).
-k	Kill the process after cloning (`-r`), or at end of dump collection.
-l	Display the debug logging of the process.
-m	Memory commit threshold in MB at which to create a dump.
-ml	Trigger when memory commit drops below specified MB value.
-n	Number of dumps to write before exiting.
-o	Overwrite an existing dump file.
-p	Trigger when the Performance Counter is at, or exceeds, the specified Threshold. Some Counters and/or Instance Names can be case-sensitive.
-pl	Trigger when the Performance Counter falls below the specified Threshold.
-r	Dump using a clone. Concurrent limit is optional (default 1, max 5). OS doesn't support a kernel dump (`-mk`) when using a clone (`-r`). **CAUTION:** a high concurrency value may impact system performance. - Windows 7: Uses Reflection. OS doesn't support `-e`. - Windows 8.0: Uses Reflection. OS doesn't support `-e`. - Windows 8.1+: Uses PSS. All trigger types are supported.
-s	Consecutive seconds before dump is written (default is 10).
-t	Write a dump when the process terminates.
-u	Treat CPU usage relative to a single core (used with `-c`).
-v	**DEBUG ONLY:** Verbose output.
-w	Wait for the specified process to launch if it's not running.
-wer	Queue the (largest) dump to Windows Error Reporting.
-x	Launch the specified image with optional arguments. If it is a Store Application or Package, ProcDump will start on the next activation (only).

Condition	Description
-y	**HIDDEN:** Store Application activation.
-64	By default ProcDump will capture a 32-bit dump of a 32-bit process when running on 64-bit Windows. This option overrides to create a 64-bit dump. Only use for WOW64 subsystem debugging.

License Agreement:

Use the `-accepteula` command line option to automatically accept the Sysinternals license agreement.

Automated Termination:

`-cancel <Target Process PID>`

Using this option or setting an event with the name `ProcDump-<PID>` is the same as typing Ctrl+C to gracefully terminate ProcDump. Graceful termination ensures the process is resumed if a capture is active. The cancellation applies to ALL ProcDump instances monitoring the process.

Filename:

Default dump filename: `PROCESSNAME_YYMMDD_HHMMSS.dmp`

The following substitutions are supported:

Substitution	Explanation
PROCESSNAME	Process Name
PID	Process ID
EXCEPTIONCODE	Exception Code
YYMMDD	Year/Month/Day
HHMMSS	Hour/Minute/Second

Examples

- Write a mini dump of a process named 'notepad' (only one match can exist):

```cmd
C:\>procdump notepad
```

- Write a Full dump of a process with PID '4572':

```cmd
C:\>procdump -ma 4572
```

- Write a Mini first, and then a Full dump of a process with PID '4572':

```cmd
C:\>procdump -mm -ma 4572
```

- Write 3 Mini dumps 5 seconds apart of a process named 'notepad':

```cmd
C:\>procdump -n 3 -s 5 notepad
```

- Write up to 3 Mini dumps of a process named 'consume' when it exceeds 20% CPU usage for five seconds:

```cmd
C:\>procdump -n 3 -s 5 -c 20 consume
```

- Write a Mini dump for a process named 'hang.exe' when one of its windows is unresponsive for more than 5 seconds:

```cmd
C:\>procdump -h hang.exe
```

- Write a Full and Kernel dump for a process named 'hang.exe' when one of its windows is unresponsive for more than 5 seconds:

```cmd
C:\>procdump -ma -mk -h hang.exe
```

- Write a Mini dump of a process named 'outlook' when total system CPU usage exceeds 20% for 10 seconds:

cmd

```
C:\>procdump outlook -s 10 -p "\Processor(_Total)\% Processor Time" 20
```

- Write a Full dump of a process named 'outlook' when Outlook's handle count exceeds 10,000:

```cmd
C:\>procdump -ma outlook -p "\Process(Outlook)\Handle Count" 10000
```

- Write a Full dump of 'svchost' PID 1234, Instance #87, when the handle count exceeds 10,000:

```cmd
C:\>procdump -ma 1234 -p "\Process(svchost#87)\Handle Count" 10000
```

Note: Multiple Instance Counters
If there are multiple instances of the counter, you'll need to include the Name and/or Instance number.

```txt
\Processor(NNN)\% Processor Time
\Thermal Zone Information(<name>)\Temperature
\Process(<name>[#NNN])\<counter>
```

Older OSes require you to append the PID for `\Process` counters.

```txt
\Process(<name>[_PID])\<counter>
```

Tip: Use Performance Monitor to view the counters (esp. case sensitivity).
Tip: For `\Process(*)` based counters, use PowerShell to map a PID to its `#NNN`.

```pwsh
Get-Counter -Counter "\Process(*)\ID Process"
```

- Write a Full dump for a 2nd chance exception:

cmd

```
C:\>procdump -ma -e w3wp.exe
```

- Write a Full dump for a 1st or 2nd chance exception:

```cmd
C:\>procdump -ma -e 1 w3wp.exe
```

- Write a Full dump for a debug string message:

```cmd
C:\>procdump -ma -l w3wp.exe
```

- Write up to 10 Full dumps of each 1st or 2nd chance exception of w3wp.exe:

```cmd
C:\>procdump -ma -n 10 -e 1 w3wp.exe
```

- Write up to 10 Full dumps if an exception's code/name/msg contains '`NotFound`':

```cmd
C:\>procdump -ma -n 10 -e 1 -f NotFound w3wp.exe
```

- Write up to 10 Full dumps if a debug string message contains '`NotFound`':

```cmd
C:\>procdump -ma -n 10 -l -f NotFound w3wp.exe
```

- Wait for a process called 'notepad' (and monitor it for exceptions):

```cmd
C:\>procdump -e -w notepad
```

- Launch a process called 'notepad' (and monitor it for exceptions):

```cmd
C:\>procdump -e -x c:\dumps notepad
```

- Register for launch, and attempt to activate, a store 'application'. A new ProcDump instance will start when it is activated:

```cmd
C:\>procdump -e -x c:\dumps Microsoft.BingMaps_8wekyb3d8bbwe!AppexMaps
```

- Register for launch of a store 'package'. A new ProcDump instance will start when it is (manually) activated:

```cmd
C:\>procdump -e -x c:\dumps
Microsoft.BingMaps_1.2.0.136_x64__8wekyb3d8bbwe
```

- Write a MiniPlus dump of the Microsoft Exchange Information Store when it has an unhandled exception:

```cmd
C:\>procdump -mp -e store.exe
```

- Display without writing a dump, the exception codes/names of w3wp.exe:

```cmd
C:\>procdump -e 1 -f "" w3wp.exe
```

- Windows 7/8.0; Use Reflection to reduce outage for 5 consecutive triggers:

```cmd
C:\>procdump -r -ma -n 5 -s 15 wmplayer.exe
```

- Windows 8.1+; Use PSS to reduce outage for 5 concurrent triggers:

```cmd
C:\>procdump -r 5 -ma -n 5 -s 15 wmplayer.exe
```

- Install ProcDump as the (AeDebug) postmortem debugger:

cmd

```
C:\>procdump -ma -i c:\dumps
```

..or..

```cmd
C:\Dumps>procdump -ma -i
```

- Uninstall ProcDump as the (AeDebug) postmortem debugger:

```cmd
C:\>procdump -u
```

See a list of example command lines (the examples are listed above):

```cmd
C:\>procdump -? -e
```

Related Links

- Windows Internals Book The official updates and errata page for the definitive book on Windows internals, by Mark Russinovich and David Solomon.
- Windows Sysinternals Administrator's Reference The official guide to the Sysinternals utilities by Mark Russinovich and Aaron Margosis, including descriptions of all the tools, their features, how to use them for troubleshooting, and example real-world cases of their use.

Download ProcDump (714 KB)

Download ProcDump for Linux (GitHub)

Runs on:

- Client: Windows 8.1 and higher.
- Server: Windows Server 2012 and higher.

Learn More

- Defrag Tools: #9 - ProcDump This episode of Defrag Tools covers what the tool captures and expected outage durations
- Defrag Tools: #10 - ProcDump - Triggers This episode covers trigger options in particular 1st & 2nd chance exceptions
- Defrag Tools: #11 - ProcDump - Windows 8 & Process Monitor This episode covers modern application support and Process Monitor logging support