

Got High CPU Usage - ProcDump Em

Michael Morales

(Reprinted from WindowsItPro Magazine)

Executive Summary

Microsoft provides several useful free tools for troubleshooting high-CPU-usage issues on Windows systems: adplus.vbs, Xperf, and Process Explorer. The latest addition to this list of tools is ProcDump (procdump.exe), a Windows Sysinternals tool written by Microsoft Technical Fellow Mark Russinovich, directly in response to requests from Microsoft's Global Escalation Services team for a tool to capture a dump file of a process. ProcDump lets you configure how much CPU a process should consume and for how long a time period before creating a dump of the process—so you don't have to be physically at a console issuing commands to run the process and capture the dump. Learn how to use ProcDump in a typical high-CPU situation to flag and get detailed information about a CPU-hogging process.

On the Microsoft support team, one of the most common customer problems we encounter is systems experiencing high CPU usage. Solving this type of problem is often challenging because you must first determine which process or activity is responsible for consuming so much CPU time, then determine the best approach for capturing the process's activity during the problem period so that it can be analyzed for root cause. Fortunately, Microsoft provides tools available to assist with high-CPU issues. I'll give a brief rundown of these tools, then introduce you to a brand-new free tool called ProcDump that will save you much time and hassle the next time you run into a high-CPU problem.

High-CPU Usage Troubleshooting Tools

Until now, we've relied mainly upon these tools to help troubleshoot high-CPU problems on Windows systems:

Adplus.vbs

This VBScript tool comes with the Debugging Tools for Windows and is a great resource for administrators to use for dumping out a process during a high CPU occurrence. However, one of the drawbacks of Adplus is that a person usually has to be at the console to physically issue the Adplus command to dump out the process when the CPU spike occurs.

Xperf

This is a super tool for collecting process activity during a high CPU spike, and it doesn't require anyone to be physically at the console to monitor for high -CPU occurrences. (You can download Xperf at msdn.microsoft.com/en-us/performance/default.aspx.) Although Xperf isn't fully supported on Windows Server 2003, our experience with collecting stackwalk data on Windows 2003 (which is the critical piece of data for analyzing high-CPU problems) has been very positive, as long as you have the hotfix download available at support.microsoft.com/kb/938486 or a later-dated kernel installed.

One of the things to consider with XPERF is that the tool collects data about all processes and activity on the system, then lets you narrow your focus post-mortem, which means there's no way to specify, say, "I just want stackwalking for XYZ.EXE"; instead you have to turn it on for the entire system. So collecting and logging all of a system's activity for a problem that may occur once in 24 hours could be too much overhead depending on the typical workload of the systems you're monitoring.

Got High CPU Usage - ProcDump Em

Michael Morales

(Reprinted from WindowsItPro Magazine)

Process Explorer (procepx.exe). I highly recommend that you use Process Explorer, which you can download at technet.microsoft.com/en-us/sysinternals/bb896653.aspx, to at least look at the thread that's spiking the CPU to determine what components are involved, so that you can update them before calling tech support. If you need to investigate the problem further, though, you'll need a tool that actually dumps out the process during the high-CPU spike; Process Explorer can't do this. But ProcDump can.

Introducing ProcDump

ProcDump (procdump.exe) is a new Windows Sysinternals tool from Mark Russinovich, which you can download at technet.microsoft.com/en-us/sysinternals/dd996900.aspx. Procdump.exe was created after one of the escalation engineers in my group asked Mark if he would consider adding functionality to Process Explorer to allow for capturing a dump file of a process to help troubleshoot those pesky high-CPU problems. After some thought, it was determined that the best approach was to write a new tool, and ProcDump was born.

ProcDump lets you configure how much CPU a process should consume and for how long a time period before ProcDump creates a dump of the process. What this means is that you don't have to be at the console ready to issue commands the next time the process spikes the CPU. And you get to determine at what threshold the process can consume the CPU before ProcDump captures a dump of the spiking process.

So, for example, you notice the wmiprvse.exe (the WMI Provider Host process) spikes the CPU to 90 percent at random times throughout the day, and you'd like to capture a few dumps for analysis. The following command will dump out the spooler process three times when the CPU for wmiprvse.exe is at or exceeds 90 percent for 3 seconds and store the dumps in the c:\procdumps directory that you've already created:

```
c:\procdump.exe -c 80 -s 3 -n 3 wmiprvse.exe c:\procdumps
```

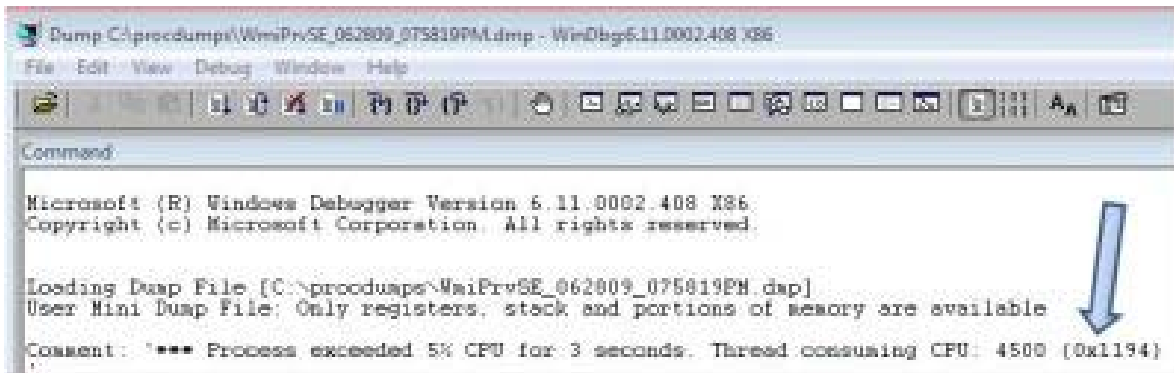
The -c option is the CPU threshold parameter that you can configure. The -s option tells ProcDump how long the service needs to consume the CPU at the threshold you configured before a dump is generated. The -n option tells ProcDump how many dumps to create, and wmiprvse.exe is the process name you're asking ProcDump to monitor.

So, for the previous command line, the WMI Provider Host service will be dumped out each time the process exceeds 80 percent CPU for three seconds or more and store the dump files in the c:\procdumps directory. The name of the dump file will be in the format PROCESSNAME_DATE_TIME.dmp; the included timestamp makes it easy to identify files captured over a period of several days. The other great feature of ProcDump is that the thread that consumed the highest amount of CPU is baked into the dump file, so that when the dump file is opened in the debugger, you get a message indicating which thread consumed the CPU, as Figure 1 shows.

Got High CPU Usage - ProcDump Em

Michael Morales

(Reprinted from WindowsItPro Magazine)



```
Dump C:\procdumps\WmiPrvSE_062809_075819PM.dmp - WinDbg6.11.0002.408 X86
File Edit View Debug Window Help
Microsoft (R) Windows Debugger Version 6.11.0002.408 X86
Copyright (c) Microsoft Corporation. All rights reserved.
Loading Dump File [C:\procdumps\WmiPrvSE_062809_075819PM.dmp]
User Mini Dump File: Only registers, stack and portions of memory are available
Comment: '*** Process exceeded 5% CPU for 3 seconds. Thread consuming CPU: 4500 (0x1194)'
```

Figure 1: ProcDump output showing high-CPU-consuming thread

Now there's no guesswork as to which thread was doing the work. From the screen in Figure 1, you can then issue the ~ (tilde) command in the debugger to find out what thread number corresponds to 0x1194. Figure 2 shows the command line and its output. As you can see, thread 2 (which includes 1194 in the line) is the thread that corresponds to 0x1194.

Figure 2: Output of ~ command

```
0:000> ~
. 0 Id: 1260.e74 Suspend: 0 Teb: 7ffdf000 Unfrozen
  1 Id: 1260.6d0 Suspend: 0 Teb: 7ffde000 Unfrozen
  2 Id: 1260.1194 Suspend: 0 Teb: 7ffdd000 Unfrozen
  3 Id: 1260.11f8 Suspend: 0 Teb: 7ffdc000 Unfrozen
  4 Id: 1260.1780 Suspend: 0 Teb: 7ffdb000 Unfrozen
  5 Id: 1260.13d4 Suspend: 0 Teb: 7ffda000 Unfrozen
  6 Id: 1260.1544 Suspend: 0 Teb: 7ffd9000 Unfrozen
  7 Id: 1260.1164 Suspend: 0 Teb: 7ffd7000 Unfrozen
```

This was just an example that I created to demonstrate the tool, but now we can change the focus to thread 2 to find out what was going on at the time the CPU was consumed. At the command prompt, run the following command to change the context to thread 2:

```
0:000> ~2s
```

The command's output in Figure 3 shows that the wmioprse.exe process enumerated through various directories (EnumDirsNT) at the time this test was done, which makes sense since the WMI query I issued required the enumeration of all directories on my system.

Figure 3: Wmioprse.exe process thread 2 details

```
eax=013bd900 ebx=00004021 ecx=00000004 edx=00000044 esi=76fb49f4 edi=00100001
eip=76fb5cb4 esp=013bd548 ebp=013bd840 iopl=0 nv up ei pl nz na po nc
```

Got High CPU Usage - ProcDump Em

Michael Morales

(Reprinted from WindowsItPro Magazine)

```
cs=001b ss=0023 ds=0023 es=0023 fs=003b gs=0000 efl=00000202
ntdll!KiFastSystemCallRet:
76fb5cb4 c3          ret
0:002> k
ChildEBP RetAddr
013bd544 76fb4a00 ntdll!KiFastSystemCallRet
013bd548 75810c0a ntdll!ZwOpenFile+0xc
013bd840 75810def kernel32!FindFirstFileExW+0x1c9
013bd860 60c44cbb kernel32!FindFirstFileW+0x16
013bdd5c 60c4585e cimwin32!CImplement_LogicalFile::EnumDirsNT+0x5b2
013be254 60c4585e cimwin32!CImplement_LogicalFile::EnumDirsNT+0x1151
013be74c 60c4585e cimwin32!CImplement_LogicalFile::EnumDirsNT+0x1151
013bec44 60c7b7e9 cimwin32!CImplement_LogicalFile::EnumDirsNT+0x1151
013beec8 666ff3dd cimwin32!CShortcutFile::EnumerateInstances+0x157
013beedc 666ff82f framedynos!Provider::CreateInstanceEnum+0x21
```

ProcDump will also dump a process if any of the process's windows are hung (-h option); as with the other ProcDump functions, you don't need to be physically at the console to initiate this task.

Launching a Process Under the Debugger

An especially useful ProcDump option is the ability to launch a process directly under the debugger using the -x option. The -x option works with the Image File Execution Options registry entry. The command example in Figure 4, which specifies -x with the lsass.exe process, will take three dumps of lsass.exe when the process spikes the CPU to 90 percent.

Figure 4: Using ProcDump with the -x option

```
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution
Options\LSASS.EXE
Debugger = c:\procdump\procdump.exe -c 90 -n 3 -ma -x
```

Now the next time lsass.exe is started, ProcDump will monitor the process with the configured parameters. Why is this so cool? Because there are processes that could spike immediately on startup and freeze your whole system, and you can't log on to the console until the CPU has settled down—but by that time, there's nothing to dump out because the high CPU has gone down. Using ProcDump with the -x option lets you capture information about these spikes when they happen.

More Help for High-CPU Issues

I predict the ProcDump will be the tool of choice for most high-CPU issues and will change the way we attack such problems and how fast they're resolved. ProcDump was built as a grassroots effort initiated by Microsoft's Global Escalation Services team. A special thanks to Ming Chen, the senior escalation engineer who first approached Mark and got the ball rolling; Jeff Daily, a principal escalation engineer, for his leadership and guidance; and of course, a huge thanks to Mark Russinovich, a Microsoft technical fellow, for taking our input so frequently and making changes so fast.