How do I know that Resource Monitor isnt just retaining a handle to the terminated process?

devblogs.microsoft.com/oldnewthing/20180115-00

January 15, 2018

Raymond Chen

Not too long ago, I investigated whether the Resource Monitor retains an open handle to the process as its way to continue showing statistics after the process exited. I used Task Manager to determine whether there was still an open handle, on the theory that Task Manager shows all the valid process IDs, even for processes that have terminated.

A reader who needs to choose a better screen name pointed out that it's possible that Task Manager intentionally removes terminated processes from display.

This was a fair criticism, so let's test this theory.

```
#include <windows.h>
#include <stdio.h> // horrors! Mixing stdio and C++!

int __cdecl main()
{
 PROCESS_INFORMATION pi;
 STARTUPINFO si = { sizeof(si) };
 TCHAR command[] = TEXT("notepad.exe");
 CreateProcess(nullptr, command,
  nullptr, nullptr, FALSE, 0, nullptr, nullptr, &si, &pi);
 WaitForSingleObject(pi.hProcess, INFINITE);
 printf("Process has exited. "
        "Keeping handle open for a little while longer.\n");
 Sleep(10 * 1000);
 CloseHandle(pi.hProcess);
 CloseHandle(pi.hThread);
 return 0;
}
```

This program launches Notepad, waits for it to exit, then waits an additional ten seconds before terminating. During those ten seconds, the process handle is still open, and hopefully we'll see the process still listed in Task Manager.

Run this test program, close the Notepad window the opens, and then during the ten second delay, go to the Details page of Task Manager to see if Notepad is present.

Nope, it's not there. The commenter was correct. Task Manager *does* remove terminated processes from display, even though the process object has not been destroyed.

(This changed at some point, because the old Task Manager did keep zombie processes around. That was one way to find out that you had a process handle leak: The zombie processes started piling up in Task Manager. My guess is that the change occurred when Task Manager was rewritten for Windows 8.)

Okay, so our test was flawed. We'll have to try a different test. This time, let's write a program that launches Notepad, waits for it to exit, and then tries to reopen it by its process ID. This will succeed if there is still an outstanding handle to the process, and will fail if there are no outstanding handles.

```
#include <windows.h>
#include <stdio.h> // horrors! Mixing stdio and C++!

int __cdecl main()
{
 PROCESS_INFORMATION pi;
 STARTUPINFO si = { sizeof(si) };
 TCHAR command[] = TEXT("notepad.exe");
 CreateProcess(nullptr, command,
  nullptr, nullptr, FALSE, 0, nullptr, nullptr, &si, &pi);
 WaitForSingleObject(pi.hProcess, INFINITE);
 CloseHandle(pi.hProcess);
 CloseHandle(pi.hThread);
 printf("Process has exited. Try to open the process by ID.\n");
 Sleep(1000);
 auto h = OpenProcess(PROCESS_QUERY_LIMITED_INFORMATION,
   FALSE, pi.dwProcessId);
 printf("Resulting handle: %p\n", h);
 if (h) CloseHandle(h);
 return 0;
}
```

This second test program waits for Notepad to exit, and then closes all the handles and then tries to reopen it.

Note that there is a brief `Sleep` after the process exits. This gives a little time for anybody who was doing the "Close the handle as soon as the process terminate" thing to close their handle. Without it, you will find that even without Resource Manager running, you'll see a handle retained somewhere. My guess it's anti-malware software.

Okay, run the test program with no Resource Monitor running. Close Notepad, wait one second, and observe that it reports that the resulting handle is `00000000`, which means that the `OpenProcess` failed and there is no process with that ID, meaning that the process object has been destroyed, which means that there are no open handles to it.

Now run our experiment with Resource Monitor: Run the test program, let Resource Monitor observe the new Notepad process. Now connect a debugger to Resource Monitor and freeze it, so that if it had an open handle to Notepad, it won't be able to close it. Close the Notepad window, wait one second, and observe that the test program reports that the resulting handle is `00000000`. This demonstrates that Resource Monitor doesn't retain a handle to the Notepad process.

Raymond Chen

Follow