So what is a Windows “critical process” anyway?

devblogs.microsoft.com/oldnewthing/20180216-00

February 16, 2018

Raymond Chen

I noted some time ago that Task Manager applies three somewhat arbitrary criteria for dividing processes into three categories: App, Background Process, and Windows Process. In particular, a Windows Process is one for which `IsProcessCritical` reports `TRUE`.

SimonRev quite rightly calls out the documentation for being useless due to the fact that it merely states a tautology.

> **IsProcessCritical**
>
> Determines whether the specified process is considered critical.
>
> ```
> BOOL WINAPI IsProcessCritical(
>   _In_  HANDLE hProcess,
>   _Out_ PBOOL  Critical
> );
> ```
>
> *hProcess* [in] A handle to the process to query. The process must have been opened with `PROCESS_LIMITED_QUERYINFORMATION` access.
>
> *Critical* [out] A pointer to the **BOOL** value this function will use to indicate whether the process is considered critical.
>
> **Return value**: This routine returns **FALSE** on failure. Any other value indicates success. Call **GetLastError** to query for the specific error reason on failure.

Great, so we learn that the `IsProcessCritical` function tells you whether the process is critical. But nowhere does it say what it means for a process to be critical or how a process becomes critical in the first place.

A critical process is one that forces a system reboot if it terminates. (More precisely, it forces a bluescreen error, which captures a memory dump before rebooting, so that the cause for termination can be investigated.)

How does a process get itself marked critical?

A few system processes do this on their own. For example, processes related to enforcing system security do this so that if one of them crashes, it stops the system immediately before any more damage can occur.

But most of the time, the way this happens if you create a service and set its recovery option to **Restart the Computer**.

Bonus chatter: Wait a second, there are some processes in the *Windows processes* list that aren't critical system processes. Like *Console Window Host*. How did they get there?

In addition to putting all critical system processes in the list, Task Manager also keeps a hard-coded list of processes that it puts in the *Windows processes* list whenever it sees them. That's why you see things like *Console Window Host* and *Desktop Window Manager*. So a more accurate list of what goes into *Windows processes* is

- A hard-coded list of specific processes, plus
- Processes marked as critical.

Raymond Chen

Follow