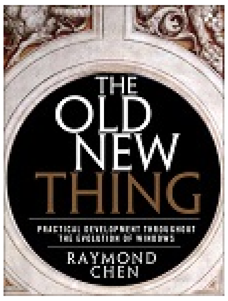
Why does EnumProcessModules report no modules on a process that was created suspended?

devblogs.microsoft.com/oldnewthing/20250526-00/?p=111218

May 26, 2025



A customer had a test that created a process suspended, and without resuming it, they called `EnumProcessModules` to see what modules are in it. The `EnumProcessModules` reported no modules. Why is that? Shouldn't it at least report the primary executable?

Recall that in Windows, processes are self-loading, which means that they manage their own module list. When the kernel creates a process, it sets the initial instruction pointer to an internal function inside `ntdll.dll`, provides information about what the new process should do (for example, the command line arguments), and then lets the process start executing. The function inside `ntdll.dll` does the work of loading all the modules, adding entries to the module list as it goes.

If the process is created suspended, then it hasn't started loading itself, which means that there is nothing in the module list.

This is called out in the documentation for `EnumProcessModules`:

> The **EnumProcessModules** function is primarily designed for use by debuggers and similar applications that must extract module information from another process. If the module list in the target process is corrupted or not yet initialized, or if the module list changes during the function call as a result of DLLs being loaded or unloaded, **EnumProcessModules** may fail or return incorrect information.