Enumerating All Modules For a Process

Article • 08/20/2020

To determine which processes have loaded a particular DLL, you must enumerate the modules for each process. The following sample code uses the EnumProcessModules function to enumerate the modules of current processes in the system.

C++

```cpp
#include <windows.h>
#include <tchar.h>
#include <stdio.h>
#include <psapi.h>

// To ensure correct resolution of symbols, add Psapi.lib to TARGETLIBS
// and compile with -DPSAPI_VERSION=1

int PrintModules( DWORD processID )
{
    HMODULE hMods[1024];
    HANDLE hProcess;
    DWORD cbNeeded;
    unsigned int i;

    // Print the process identifier.

    printf( "\nProcess ID: %u\n", processID );

    // Get a handle to the process.

    hProcess = OpenProcess( PROCESS_QUERY_INFORMATION |
                            PROCESS_VM_READ,
                            FALSE, processID );
    if (NULL == hProcess)
        return 1;
```

```
   // Get a list of all the modules in this process.

    if( EnumProcessModules(hProcess, hMods, sizeof(hMods), &cbNeeded))
    {
        for ( i = 0; i < (cbNeeded / sizeof(HMODULE)); i++ )
        {
            TCHAR szModName[MAX_PATH];

            // Get the full path to the module's file.

            if ( GetModuleFileNameEx( hProcess, hMods[i], szModName,
                                      sizeof(szModName) / sizeof(TCHAR)))
            {
                // Print the module name and handle value.

                _tprintf( TEXT("\t%s (0x%08X)\n"), szModName, hMods[i] );
            }
        }
    }

    // Release the handle to the process.

    CloseHandle( hProcess );

    return 0;
}

int main( void )
{

    DWORD aProcesses[1024];
    DWORD cbNeeded;
    DWORD cProcesses;
    unsigned int i;

    // Get the list of process identifiers.
    if ( !EnumProcesses( aProcesses, sizeof(aProcesses), &cbNeeded ) )
        return 1;

    // Calculate how many process identifiers were returned.

    cProcesses = cbNeeded / sizeof(DWORD);

    // Print the names of the modules for each process.

    for ( i = 0; i < cProcesses; i++ )
    {
        PrintModules( aProcesses[i] );
    }

    return 0;
}
```

The main function obtains a list of processes by using the **EnumProcesses** function. For each process, the main function calls the PrintModules function, passing it the process identifier. PrintModules in turn calls the **OpenProcess** function to obtain the process handle. If **OpenProcess** fails, the output shows only the process identifier. For example, **OpenProcess** fails for the Idle and CSRSS processes because their access restrictions prevent user-level code from opening them. Next, PrintModules calls the **EnumProcessModules** function to obtain the module handles function. Finally, PrintModules calls the **GetModuleFileNameEx** function, once for each module, to obtain the module names.