A process shutdown puzzle

devblogs.microsoft.com/oldnewthing/20090129-00

January 29, 2009

Raymond Chen

In honor of National Puzzle Day, I leave you today with a puzzle based on an actual customer problem.

Part One: The customer explains the problem.

We have this DLL, and during its startup, it creates a thread with the following thread procedure:

```
DWORD CALLBACK ThreadFunction(void *)
{
  HANDLE HandleArray[2];
  HandleArray[0] = SetUpStuff();
  if (HandleArray[0]) {
    HandleArray[1] = ShutdownEvent;
    while (WaitForMultipleObjects(2, HandleArray,
                                  FALSE, INFINITE) == WAIT_OBJECT_0) {
      ProcessStuff();
    }
    CleanUpStuff(HandleArray[0]);
  }
  SetEvent(ThreadCompleteEvent);
  FreeLibraryAndExitThread(ThisLibrary, 0);
}
```

During process shutdown, the following function is called as part of `DLL_PROCESS_DETACH` handling:

```
void StopWorkerThread()
{
  // tell the thread to stop
  SetEvent(ShutdownEvent);


  // wait for it to stop
  WaitForSingleObject(ThreadCompleteEvent, INFINITE);


  // Clean up
  CloseHandle(ShutdownEvent);
  ShutdownEvent = NULL;


  CloseHandle(ThreadCompleteEvent);
  ThreadCompleteEvent = NULL;
}
```

The above function is hanging at the call to `WaitForSingleObject`. If we break in, we see that the thread that is supposed to be running the `ThreadFunction` is gone. I verified that the thread was successfully created, but by the time we get around to waiting for it, it's already gone.

I checked, and nobody sets the `ThreadCompleteEvent` except the `StopWorkerThread` function. I stepped through `SetUpStuff`, and it succeeded. However, a breakpoint on `CleanUpStuff` was never hit. No exceptions were thrown either.

> I am completely stumped as to how this thread disappeared.

You already know enough to explain how the thread disappeared.

Part Two: After providing your explanation, the customer came up with this solution.

> Thank you for your explanation. We've made the following changes to fix the problem. Again, thank you for your help.
>
> ```
> DWORD CALLBACK ThreadFunction(void *)
> {
>   HANDLE HandleArray[2];
>   HandleArray[0] = SetUpStuff();
>   if (HandleArray[0]) {
>     HandleArray[1] = ShutdownEvent;
>     while (WaitForMultipleObjects(2, HandleArray,
>                          FALSE, INFINITE) == WAIT_OBJECT_0) {
>       ProcessStuff();
>     }
>     CleanUpStuff(HandleArray[0]);
>   }
>   ~~// SetEvent(ThreadCompleteEvent);~~
>   FreeLibraryAndExitThread(ThisLibrary, 0);
> }
>
>
> void StopWorkerThread()
> {
>   // tell the thread to stop
>   SetEvent(ShutdownEvent);
>
>
>   // wait for the thread
>   WaitForSingleObject(ThreadHandle, INFINITE);
>
>
>   // Clean up
>   CloseHandle(ShutdownEvent);
>   ShutdownEvent = NULL;
> }
> ```

Criticize this proposed solution.

Part Three: Even though the proposed solution is flawed, explain why it doesn't cause a problem in practice. (I.e., explain why the customer is always lucky.)

Raymond Chen

Follow