Can process IDs be greater than 64000? Because we're seeing process IDs greater than 64000

devblogs.microsoft.com/oldnewthing/20140130-00

January 30, 2014

Raymond Chen

A customer asked what to me was a very strange question.

> Can process IDs be greater than 64000? Because we're seeing process IDs greater than 64000.

This is a strange question because the answer is right there: You're seeing process IDs greater than 64000 with your very own eyes. Do you doubt the evidence right there in front of you?

It's like asking, "Is it possible to have an orange with no seeds? Because I have an orange with no seeds."

We saw some time ago that process IDs can get very high indeed, although the kernel tries to keep the numbers small purely for cosmetic reasons.

The customer explained why they were asking this question.

> Our application is crashing when the process ID gets too large. Here is the code:
>
> ```
> int eventId = System.Diagnostics.Process.GetCurrentProcess().Id;
> EventLog.WriteEntry("Contoso",
>                     message,
>                     EventLogEntryType.Information,
>                     eventId);
> ```

Okay, um, that code makes no sense.

The code uses the process ID as the event ID. But event IDs are static; they are references to messages in your event source's message table. It's not like your message table contains 65535 entries like this:

```
MessageId=1
Severity=Informational
SymbolicName=MSG_PROCESSID_1
Language=English
An event occurred in Process ID 1: %1
.
MessageId=2
Severity=Informational
SymbolicName=MSG_PROCESSID_2
Language=English
An event occurred in Process ID 2: %1
.
MessageId=3
Severity=Informational
SymbolicName=MSG_PROCESSID_3
Language=English
An event occurred in process ID 3: %1
.
… and so on until …
MessageId=3
Severity=Informational
SymbolicName=MSG_PROCESSID_65535
Language=English
An event occurred in process ID 65535: %1
.
```

The event ID describes what happened, and the process ID and other information goes into the payload.

You can see from the format of event IDs that the event number can be at most 65535 because the upper bits of the event ID are used to encode other information.

And the code crashes because the `WriteEntry` method specifically checks for absurd event IDs and rejects them with an `ArgumentException`.

The fix is therefore to put the process ID in the payload of your event and let the event number describe what actually happened, like "A multi-part print job was created."

Raymond Chen

Follow