How can I generate a consistent but unique value that can coexist with GUIDs?

devblogs.microsoft.com/oldnewthing/20110324-00

March 24, 2011

Raymond Chen

A customer needed to generate a GUID for each instance of a hardware device they encounter:

> The serial number for each device is 20 bits long (four and a half bytes). We need to generate a GUID based on each device, subject to the constraints that when a device is reinserted, we generate the same GUID for it, that no two devices generate the same GUID, and that the GUIDs we generate not collide with GUIDs generated by other means. One of our engineers suggested just running `uuidgen` and substituting the serial number for the final nine hex digits. Is this a viable technique?

This is similar to the trap of thinking that half of a GUID is just as unique as the whole thing. Remember that all the pieces of a GUID work together to establish uniqueness. If you just take parts of it, then the algorithm breaks down. For this particular case, you're in luck. The classic Type 1 GUID uses 60 bits to encode the timestamp and 48 bits to identify the location (computer). You can take a network card, extract the MAC address, then smash the card with a hammer. Now you have a unique location. Put your twenty bits of unique data as the timestamp, and you have a Type 1 GUID that is guaranteed never to collide with another GUID.

If you have more than 60 bits of unique data, then this trick won't work. Fortunately, RFC4122 explains how to create a so-called *name-based UUID*, which is a UUID that can be reliably regenerated from the same source data. Section 4.3 explains how it's done. The result is either a type 3 or type 5 UUID, depending on which variant of the algorithm you chose.

Raymond Chen

Follow