Unique ID Generator

[This problem may have been done in the class. Whatever the instructor taught you supersedes the following]

Design a random/unique ID [or Design a Ticket Server]

* You have to assume that the interviewer is asking for you to generate an ID on a distribution of servers. But you have to start your answer with a single server.

* Usecases can be many. Any place that gets millions of requests generally gets an ID e.g. logs, or analytics etc.

* Assume that IDs need to be unique, they need to be generated fast, and they need to be somewhat comparable i.e. when you see two IDs, you need to be able to tell which one came first. The last one is not a hard requirement, but is nice to have.

* Single server: Just unique numbers are enough. If they exceed a Long value, then you can implement your own big number. At one point, you'll have too many requests and thus you won't be able to handle all generation on a single machine.

* Discuss how databases can generate auto-increment IDs, which are unique. But those are not distributed.

* When generating from multiple machines, attach a time-stamp to each unique id from each machine, and also the machine id. That will guarantee sufficient uniqueness.

* Discuss the need to keep all servers on a single time. NTP is a standard way to do that.

* ID is a frequent operation, and many others are dependent on it, so discuss the need for low network latency, low disk latency (or zero disk latency).

Reading pointers:

A history of UUID generation: https://segment.com/blog/a-brief-history-of-the-uuid/

Stackoverflow Discussion: http://stackoverflow.com/questions/2671858/distributed-sequence-number-generation

Snowflake: https://blog.twitter.com/2010/announcing-snowflake

Less useful but still good: https://www.slideshare.net/davegardnerisme/unique-id-generation-in-distributed-systems