The InfluxDB data model

The InfluxDB data model is quite different from other time series solutions like Graphite, RRD, or OpenTSDB. InfluxDB has a line protocol for sending time series data which takes the following form:

```
<measurement name>,<tag set> <field set> <timestamp>
```

The measurement name is a string, the tag set is a collection of key/value pairs where all values are strings, and the field set is a collection of key/value pairs where the values can be int64, float64, bool, or string. The measurement name and tag sets are kept in an inverted index which make lookups for specific series very fast.

For example, if we have CPU metrics:

```
cpu,host=serverA,region=uswest idle=23,user=42,system=12 1549063516
```

Timestamps in InfluxDB can be by second, millisecond, microsecond, or nanosecond precision. The micro and nanosecond scales make InfluxDB a good choice for use cases in finance and scientific computing where other solutions would be excluded. Compression is variable depending on the level of precision the user needs.

On disk, the data is organized in a columnar style format where contiguous blocks of time are set for the measurement, tagset, fieldset. So, each field is organized sequentially on disk for blocks of time, which make calculating aggregates on a single field a very fast operation.

There is no limit to the number of tags and fields that can be used. Other time series solutions don't support multiple fields, which can make their network protocols bloated when transmitting data with shared tag sets. Most other time series solutions only support float64 values, which means the user is unable to encode additional metadata along with the time series data.

Even OpenTSDB and Kairos, which support tags (unlike Graphite and RRD) have limitations on the number of tags that can be used. At around five to six tags, the user will start seeing hot spots within their cluster of HBase or Cassandra machines. InfluxDB, on the other hand, doesn't have this limitation.

The InfluxDB data model is purpose-built for time series, specifically. It pushes the developer in the right direction to get good performance out of the database by indexing tags and keeping fields unindexed. It's flexible in that many data types are supported and the user can have many fields and tags.