

ScribeToCassandra

Getting Scribe to write to Cassandra can be very easy by just implementing a custom scribe server.

Here's a super simple example written in java and using Hector. It uses a column family according to the `LogEntry`'s category, and puts the payload directly into a column name "scribePayload".

```
class ScribeToCassandraService implements com.facebook.scribe.scribe.Iface {

    private static final String CLUSTER_NAME = "myCluster";
    private static final String HOST_PORT = "localhost:9160";
    private static final String KEYSPACE = "keyspace1";
    private static final String COLUMN = "scribePayload";
    ...
    public ResultCode Log(final List<LogEntry> logEntries) throws TException {

        Cluster cluster = HFactory.getOrCreateCluster(CLUSTER_NAME, HOST_PORT);
        Mutator<UUID> mutator = HFactory.createMutator(
            HFactory.createKeyspace(KEYSPACE, cluster),
            UUIDSerializer.get());

        for (LogEntry logEntry : logEntries) {
            try{
                String payload = logEntry.getMessage();
                String category = msg.getCategory();

                HColumn<String,String> column = HFactory.createColumn(
                    COLUMN,
                    payload,
                    StringSerializer.get(),
                    StringSerializer.get());

                UUID uuid = UUID.fromString(UUIDGenerator.getInstance().generateTimeBasedUUID().toString());
                mutator.addInsertion(uuid, category, column);

            }catch(Exception je){
                // example catch code
                LOG.error("invalid LogEntry\n"
                    + "category=" + logEntry.getCategory()
                    + "\nmessage is\n" + logEntry.getMessage() + "\n",
                    je);
            }
        }
        mutator.execute();
        return ResultCode.OK;
    }
    ...
}
```

If you are running an embedded cassandra you can directly use the [StorageProxy](#) for faster performance,

```
// replace above lines 11-13 with
List<RowMutation> mutations = new ArrayList<RowMutation>();

// replace above lines 20-27 with
UUID uuid = UUID.fromString(UUIDGenerator.getInstance().generateTimeBasedUUID().toString());
RowMutation change = new RowMutation(keyspaceName, UUIDSerializer.get().toByteBuffer(uuid));
ColumnPath cp = new ColumnPath(category).setColumn(COLUMN.getBytes());
change.add(new QueryPath(cp), ByteBuffer.wrap(payloadBytes), HFactory.createClock());
mutations.add(change);

// replace above line 37 with
StorageProxy.mutate(mutations, ConsistencyLevel.ONE);
```