TANGO

Connecting things together

Timescale Deployment & Implementation

EUROPEAN SYNCHROTRON RADIATION FACILITY

18/09/2019

Stuart James, Sept 2019 HDB++ Workshop

http://www.tango-controls.org/

Initial Deployment (First Phase)

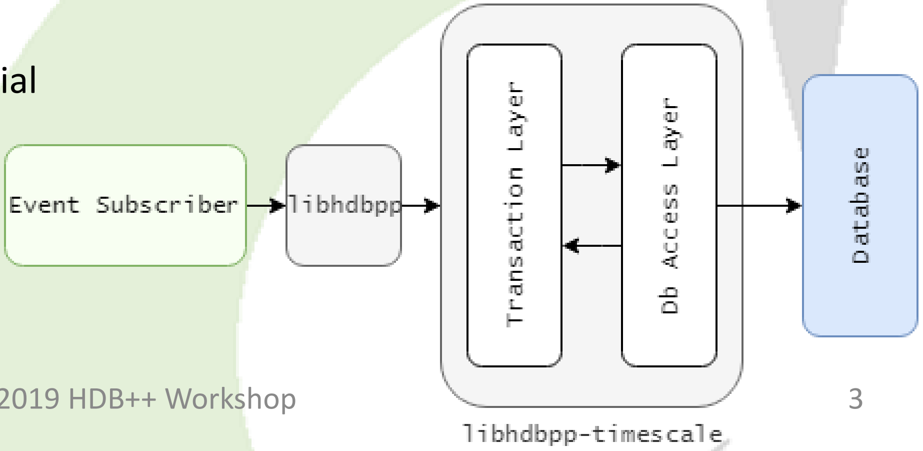
- Initial deployment – drop in replacement for Cassandra with no expanded capabilities.
  - Keep changes small and localized if possible
- We do not want to loose robustness of Cassandra
  - Fault tolerance an important factor
- Leverage performance of Timescale for insertion/queries.
- Address additional maintenance issues arising due to adoption of Timescale:
  - Index requires clustering to be viable
  - Timescale chunk interval times must be balanced against chunk size on disk.

Development Work (Shared Library)

- Initially based on libhdbpp-postgres
- Opted to rewrite library:
  - Allowed code base to be unit tested.
  - Use of modern C++
  - New internal architecture
- Internally split into two layers:
  - Transaction layer – prepares data/request for storage.
  - Transaction layer is generic and can be shared with other backends, or even merged into libhdbpp.
  - Db Access Layer – handles actual calls to the database.
  - Db Access layer can be quickly swapped out to try prototype new storage methods.
- Additional dependency: libpqxx
  - Why? High level library and offers potential future performance features.
  - Modern C++.
  - Currently statically linked into libhdbpp-timescale.

Issues Encountered

- PostgreSQL does not support unsigned types.
  - pgunit not compatible with latest PostgreSQL release. No Java support.
  - Currently using numeric via CREATE TYPE – but possible problem in Java extraction.
- We must escape arrays of strings for storage. On extraction these are not unescaped.
  - Under investigation



Development Work (Schema)

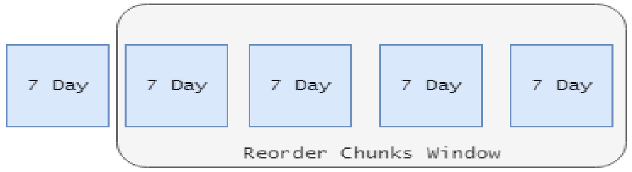
```
CREATE TABLE IF NOT EXISTS att_conf (
    att_conf_id serial NOT NULL,
    att_name text NOT NULL,
    att_conf_type_id smallint NOT NULL,
    att_conf_format_id smallint NOT NULL,
    att_conf_write_id smallint NOT NULL,
    table_name text NOT NULL,
    cs_name text NOT NULL DEFAULT '',
    domain text NOT NULL DEFAULT '',
    family text NOT NULL DEFAULT '',
    member text NOT NULL DEFAULT '',
    name text NOT NULL DEFAULT '',
    ttl int,
    hide boolean DEFAULT false,
    PRIMARY KEY (att_conf_id),
    FOREIGN KEY (att_conf_type_id) REFERENCES att_conf_
    FOREIGN KEY (att_conf_format_id) REFERENCES att_con
    FOREIGN KEY (att_conf_write_id) REFERENCES att_conf_
    UNIQUE (att_name)
);
CREATE TABLE IF NOT EXISTS att_scalar_devboolean (
    att_conf_id integer NOT NULL,
    data_time timestamp WITH TIME ZONE NOT NULL,
    value_r boolean,
    value_w boolean,
    quality smallint,
    att_error_desc_id integer,
    details json,
    PRIMARY KEY (att_conf_id, data_time),
    FOREIGN KEY (att_conf_id) REFERENCES att_conf (att_conf_
    FOREIGN KEY (att_error_desc_id) REFERENCES att_error_des
);
```

- Normalised the schema, added foreign keys.
- Consolidated scalar/array ro/rw tables into single table.
  - Allows ro/rw/wo type attributes to be stored in the same table.
  - Leverage Timescales ability to partition the data based on time.
- Data tables include a json field for future expansion. Example:
  - Insert time
- att_conf updated:
  - Include attribute type data and a reference to the data table. This may provide a method to handle attribute data type changes in future.
  - Additional field to hide inactive entries. For future use.

Development Work (Services)

Hdb Db Services Process (In Progress):

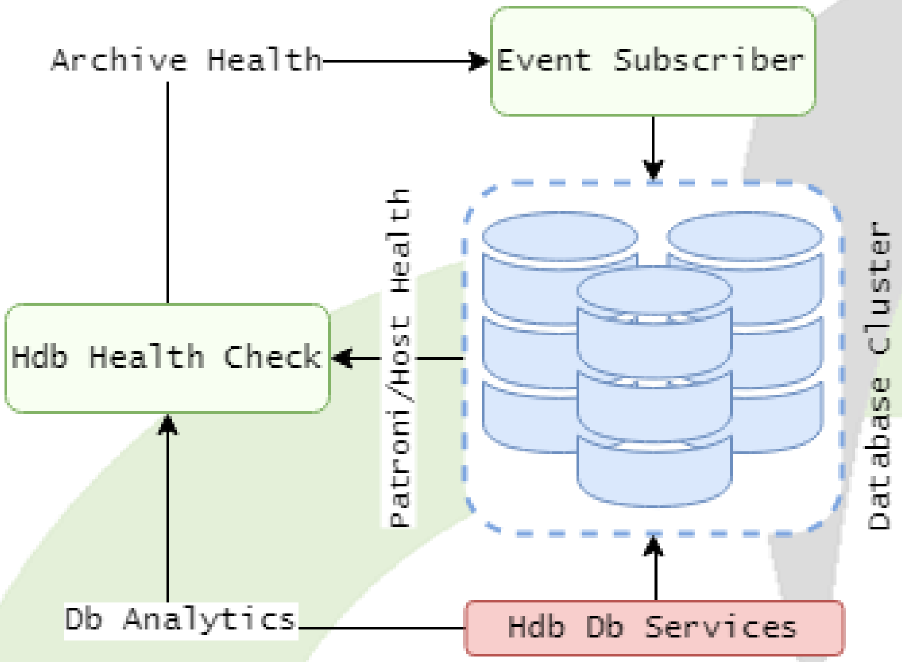
- Addresses the issue of index clustering: Reorder chunks in the data tables using Timescale lightweight commands. Allows creating an effcient rolling window of reordered chunks.



- Observe chunk intervals against chunk sizes using Timescale extension commands. Warn when we may need to alter chunk intervals.
- Run under systemd as part of the core database cluster stack on hdb-services.

Hdb Health Check Device Server (Design/Requirements):

- Central reporting mechanism, feeding health status into Tango Control system and archiving health events.
- Will draw metrics from Hdb Db Service, Patroni, hosts servers and more.

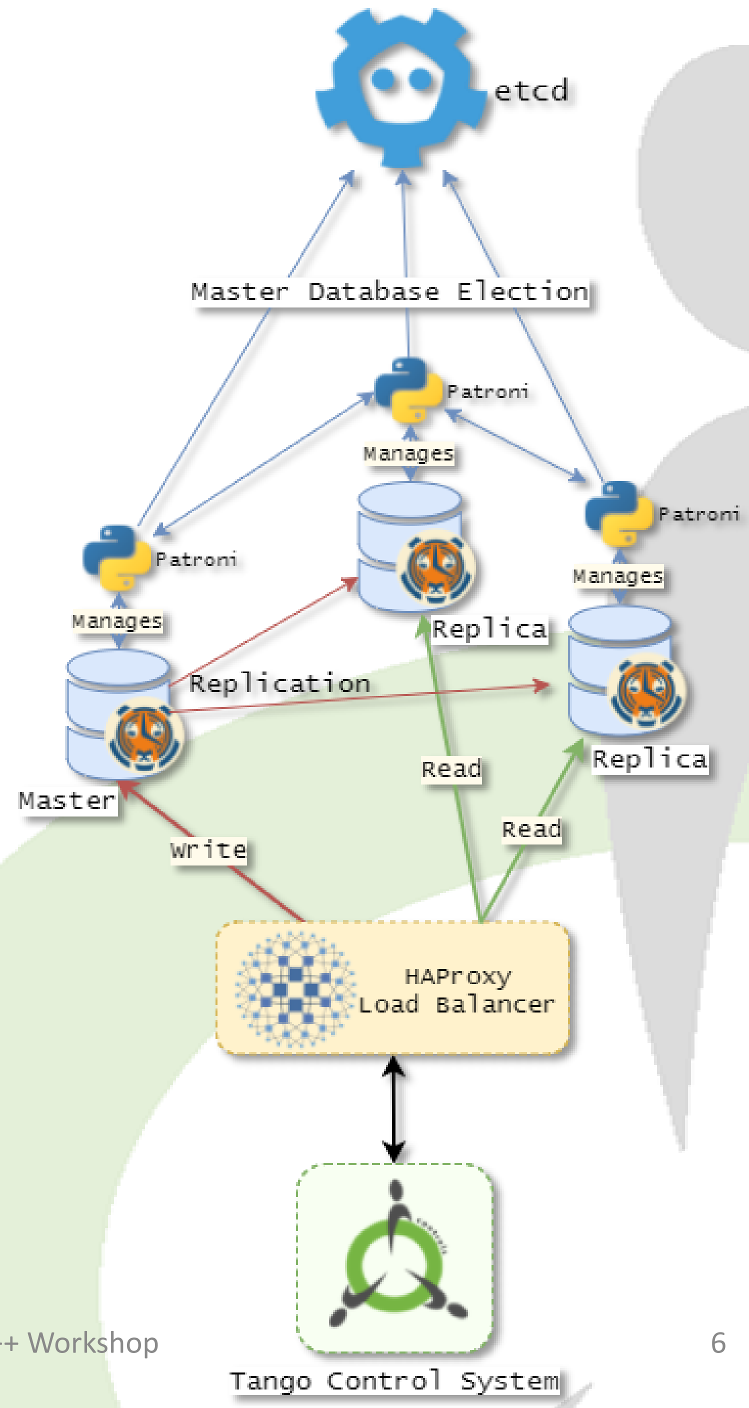


Cluster Design

- Based on recommendation by Timescale (see link)
- Also see Cluster Design document.

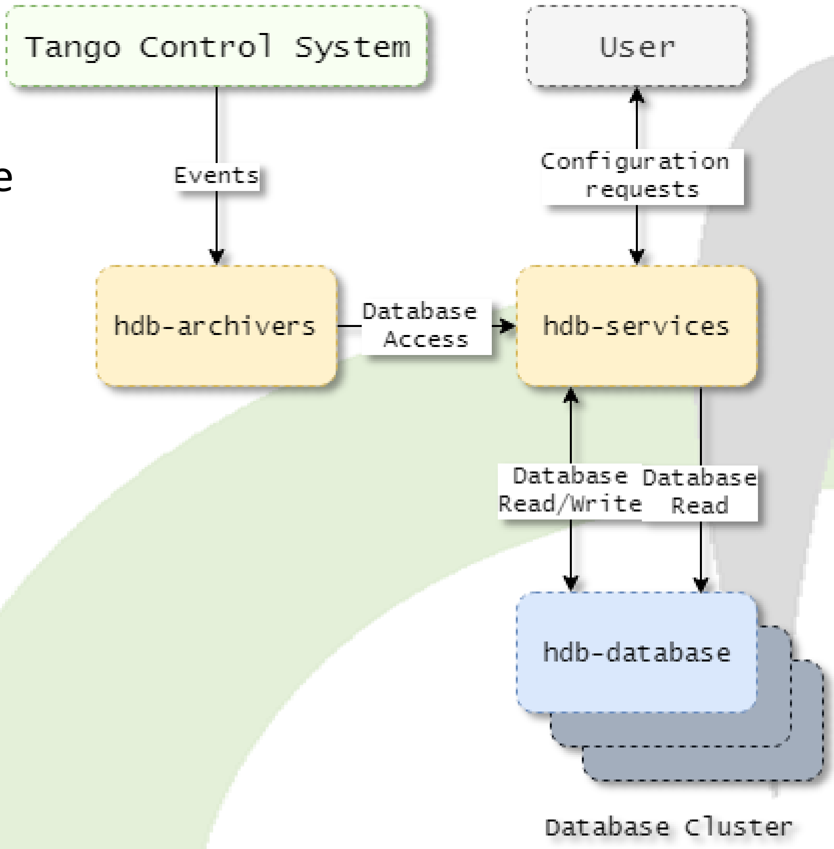
Notes

- Single point of access via proxy.
- Our design utilities a Master and one or more Replica nodes.
  - All writes to Master via proxy.
  - All reads to Replica's via proxy.
  - Reduces load on Master when users query for data.
- On Master failure, one Replica is promoted to Master.
  - Proxy is dynamically updated via Confd (see ClusterDesign document) on promotion.
- Each database node is deployed with an instance of Patroni.
  - Patroni manages PostgreSQL process itself.
  - Replication is also handled via Patroni automatically
- New failure points, etcd, HAProxy and Patroni managed by systemd to partially mitigate problems.



Deployment

- Deployment is on virtual/real servers to allow us to utilise high end ex Cassandra servers for database nodes.
- Each database is deployed on its down server with its Patroni process. Allows the node to be down without taking out other critical software.
- Opted for a single instance of etcd initially, with option to cluster at a later point. (recommended)
- Deployment was split logically as follows:
  - hdb-archivers aggregates the Tango Device Servers into one or more virtual servers.
  - hdb-services aggregates all the service software onto a single server. Can be scaled if required.
- We can not scale write performance, but read performance can be scaled with additional Replica nodes.

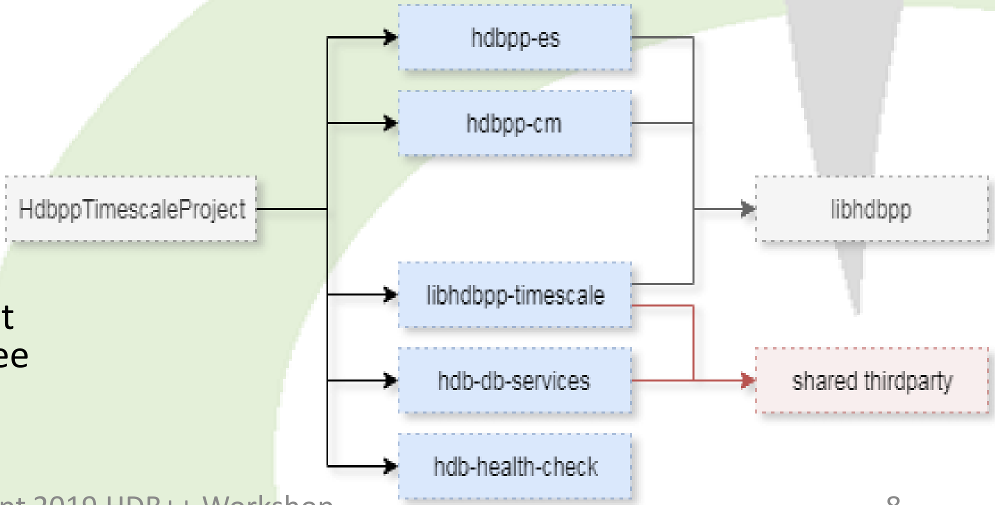


Hdbpp Timescale Project

- Proposal to centralize all components for the Hdb++ Timescale project into a single repository.
  - Additionally move libhdbpp-timescale shared library component into the project.
  - To do this, requires rebuild of component build systems
- As project increases in complexity, then managing a coherent release gets harder.
- Timescale project will add at least 2 more components (hdb-db-services and hdb-health-check).
- All components put under a single integrated build system, to build entire release. Advantages
  - New user friendly
  - Maintain any version dependencies
  - Potentially allow easier packaging
- Easy to copy project structure and build files for other database back-ends.
- Shared components would either become git submodules or downloaded at build time (see HdbppMetrics for an example).

Component Rework:

- New CMake build system for hdbpp-es, hdbpp-cm, improved build for libhdbpp.
- Refactored libhdbpp (Breaking changes and imply a 2.0.0 library release)
- Standardized install location for headers as include/hdb++



Other Hdbpp Development + Deployment

Development

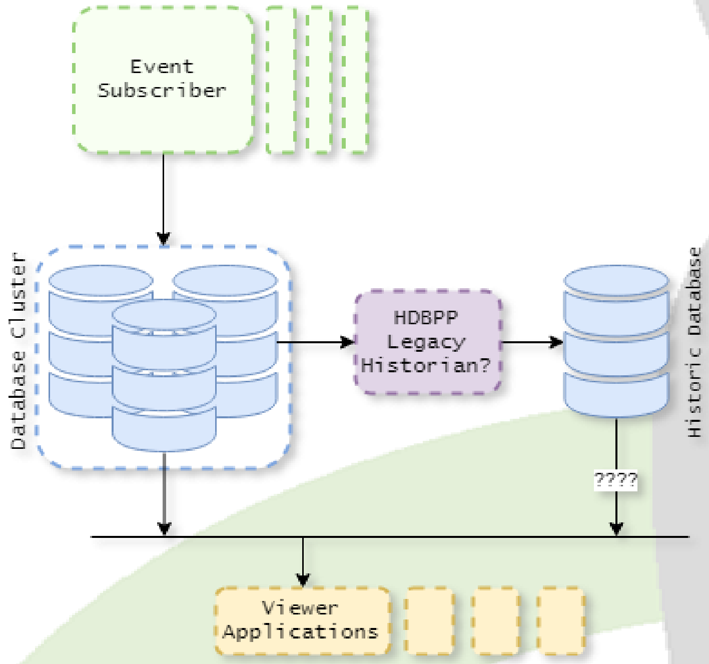
- Performance Tuning. libhdbpp-timescale is not optimised, instead the initial version is built for stability + correctness.
  - libhdbpp-timescale – micro-benchmarking library integrated. Some initial tests complete, more to do.
  - HdbppMetrics project – stub for future metrics work. Instruments Event Subscriber. Possible metrics:
    - Back end comparison.
    - Event time break down.
    - More?

Deployment

- Backup solution.
- Additional cluster monitoring tools.

Multi Db Support (Second Phase)

- What to do with legacy Cassandra data?
  - We would like to make this available via a legacy database.
- How much data can be stored in a single database?
  - We may need to offload data in the live database to a legacy database after a set period of time.
- Do we want decimate data in long term storage?
  - Or throw some data away?
- How do we access these additional databases?
  - Hdbpp viewer applications need to be aware of the additional databases.
- Potential solutions being evaluated:
  - Custom common extraction API able to query all databases.
  - GraphQL based solution.
  - Charting solutions with built in PostgreSQL support.



- Looking at requirements of the second phase now.
- Once we have outline requirements, a short study if the available solutions can be completed and shared with the community.

Connecting things together

Thank you!

Any questions?

Proxy PgAdmin4

http://www.tango-controls.org/