Esper HA

High-Availability for Complex Event Processing
Esper HA – Overview

- Provides resiliency for CEP Runtime State
- As a foundation for both:
  - The horizontal scale-out architecture that is part of Enterprise Edition (see EE), providing linear horizontal scalability, elastic scaling and balancing of load, fault tolerance
  - The classic “standalone” architecture with high-availability in a cold-standby and hot-standby (aka. active-passive) configuration
- State management features
EsperHA as related to Esper

- Resilience for Event Processing Language (EPL) state

- Resilient statement state - Swapping of state in and out of memory
  - Incremental checkpointing - Memory management

Event Stream connectors & adapters

Historical data access layer

Historical data

High-speed high-volume real-time data streams

Event Processing Statements

Named Windows

Event Query & Causality Pattern Language

Esper core container

Esper

Continuous processing

Subscribers

Output adapters

- EsperHA

Copyright EsperTech Inc. 2006 - 2019
Esper HA – State Management (1/3)

- EsperHA manages state residence
  - State can be in memory, or not in memory, as needed
  - State does not need to fit into heap memory
  - State except metadata can be released from memory
  - Reduces the chance of out-of-memory errors
  - Effectively addresses state hot spots
    - Fast-changing or frequently-accessed state can be in memory
    - Slow-changing or infrequently-accessed state need not all be in memory
  - Allows resource use prioritization of statements
  - Allows fast recovery
  - The runtime brings state back into memory when needed (state is not lost)
  - The runtime, after a checkpoint, can remove state from memory to relieve memory pressure
State is organized into pages

- Pages can be swapped in+out of memory
- Pages are a convenient, efficient unit of IO
- Minimizes pause time of checkpoints as we can serialize pages in parallel

When EsperHA is configured for horizontal scale-out:

- The source-of-truth state resides in a Kafka partitioned topic, the `changelog`
- The local file system acts as a cache, for each partition
- Horizontal scale-out uses Zookeeper, Kafka, Kafka Streams
- Kafka topic `changelog` compaction keeps the newest version of a page
- Pages are efficient units for appending to a Kafka partitioned topic (the changelog) or writing+reading for a state store
- Therefore Kafka+Kafka Streams as the source-of-truth for horizontal scale-out is a very efficient and fail-safe distributed state store for EsperHA
- See the separate presentation on Enterprise Edition Scale-Out
When EsperHA is configured in a standalone-server, the state lives in:

- File system
- or Redis
- or JDBC/relational database (MySQL, Oracle, SQLServer, Postgres, SQLite)

EsperHA provides detailed statistics
Thank you

http://www.espertech.com

info@espertech.com