Apache ZooKeeper and orchestration in distributed systems

Andrew Kondratovich
andrew.kondratovich@gmail.com

«A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable»

— Leslie Lamport

Fallacies of distributed computing

1. The network is reliable.
2. Latency is zero.
3. Bandwidth is infinite.
4. The network is secure.
5. Topology doesn't change.
6. There is one administrator.
7. Transport cost is zero.
8. The network is homogeneous.

Apache ZooKeeper

What is Apache ZooKeeper?

*«**Apache ZooKeeper** is a centralized service for maintaining **configuration information**, **naming**, providing **distributed synchronization**, and providing **group services**»*

— https://zookeeper.apache.org/

History

November 2006 — Google's "Chubby"

December 2006 — first commit

Yahoo Research

November 2007 — version 0.0.1

June 2008 — moved to Apache

subproject of Hadoop

January 2011 — Apache top-level project

Motivation

Coordination is usually given less attention

— *more bugs*

Coordination is usually reinvented

— *higher cost*

ZooKeeper provides tools for create correct distributed applications

Widely used

— Yahoo
— Facebook
— LinkedIn
— Twitter
— Cloudera

…

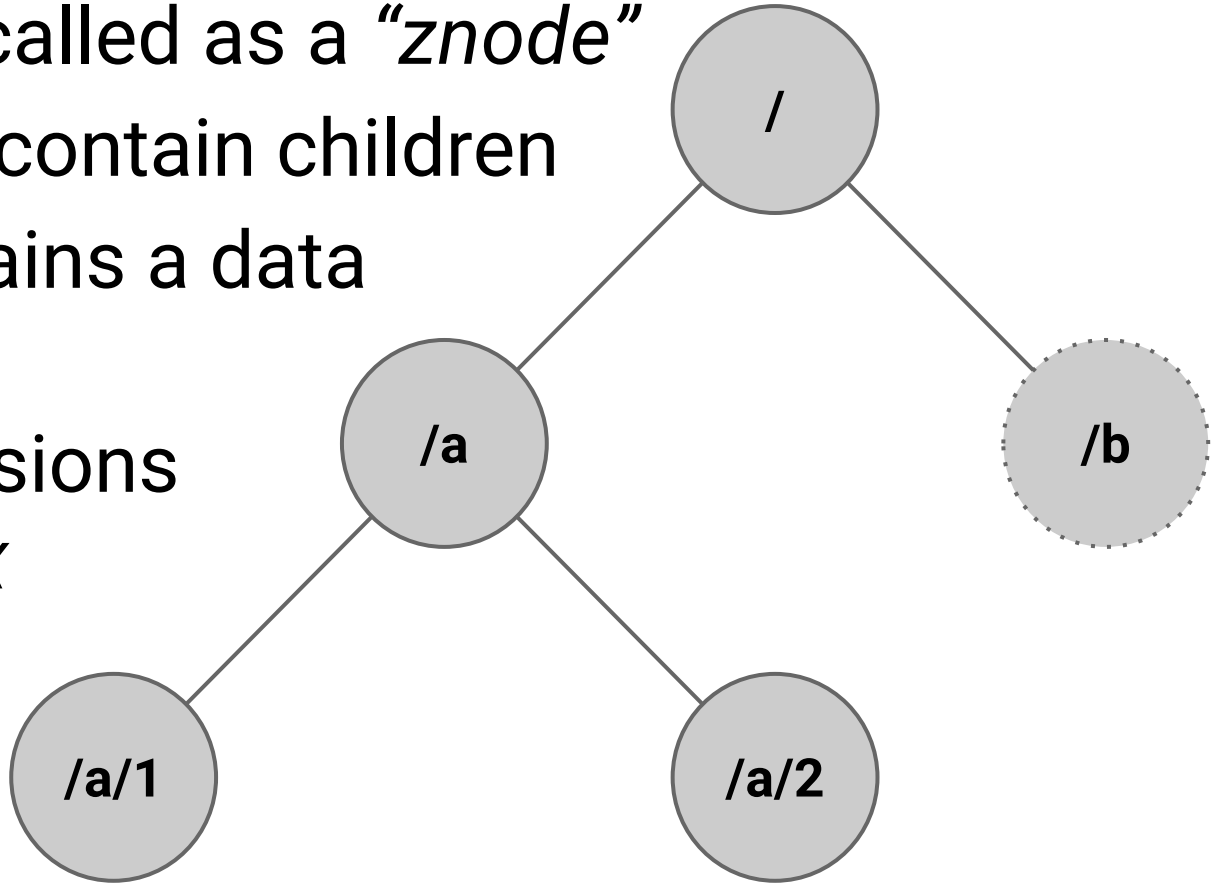
— *Everywhere*

Data model 1/2

- Hierarchical tree of nodes
- ZK node is called as a *“znode”*
- Znode may contain children
- Znode contains a data

  1mb limit
- ACL permissions

  similar to UNIX



Data model 2/2

- Keep-alive connection

  Sessions and heartbeats
- Sync and async API
- Atomic data access
- Versions and conditional updates

  Compare-and-set operations
- Watches

  Notify me on change

API

`create`	Creates a znode
`delete`	Deletes a znode
`exists`	Tests whether a znode exists
`getACL`	Gets the ACL
`setACL`	Sets the ACL
`getChildren`	Gets a list of the children
`getData`	Gets the data associated
`setData`	Sets the data associated
`sync`	Synchronizes with leader

ZNode types

- Persistent
  exists till explicitly deleted
- Ephemeral
  exists as long as the session is active and can’t have children
- Sequential
  append a monotonically increasing counter to the end of path

Watches

— Created by read operations:

exists, *getChildren*, *getData*

— Triggered by write operations:

create, *delete*, *setData*

— Watch event includes modified znode path

* Watch event is a one-time trigger

Usage

— Name service
— Configuration management
— Distributed locking
— Leader election
— Group membership

Apache Curator

«Curator is a set of Java libraries that make using Apache ZooKeeper much easier»

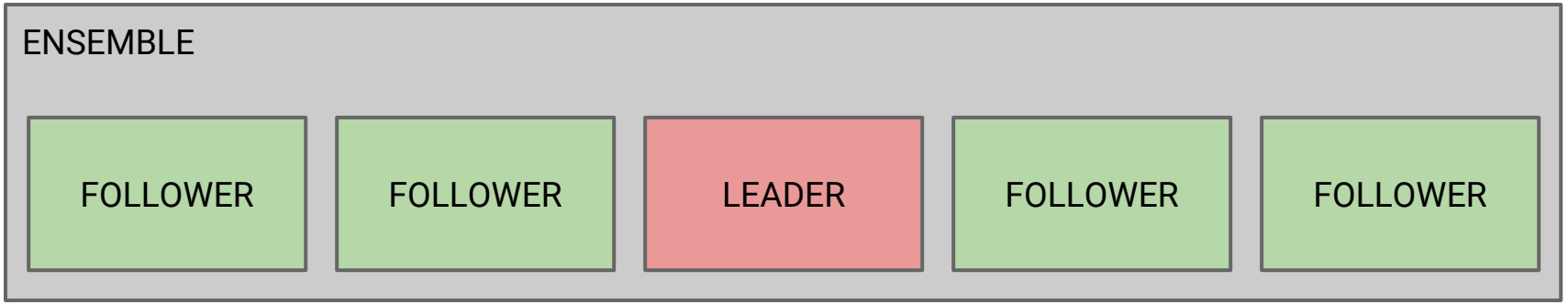
— http://curator.apache.org/

Simplifies using ZooKeeper and offers some common “recipes”.

Common problems

- Too many connections
- Session timeout misconfiguration
- Herd effect
- Transaction log performance
- Missing of data change
- Ephemeral node owner

Architecture



— All servers have a copy of state

— All updates go through leader

— Update responses are sent when a majority* of servers have persisted the change

* We need 2n+1 servers to tolerate n failures

The CAP Theorem

«You can’t sacrifice partition tolerance.

In the event of failures, which will this system sacrifice? Consistency or availability?»

— Coda Hale

Guarantees 1/2

- Sequential Consistency

  Updates from a client will be applied in the order that they were sent.

- Atomicity

  Updates either succeed or fail. No partial results.

- Single System View

  A client will see the same view of the service regardless of the server that it connects to.

- Reliability

  Once an update has been applied, it will persist until a client overwrites the update.

- Timeliness

  The client's view of the system is guaranteed to be up-to-date within a certain time bound.

Guarantees 2/2

Writes are linear

Once a write completes, all later reads should return the value of that write or the value of a later write.

Reads can be stale

The “view” of a client may be outdated, since the master updates the corresponding server with a bounded but undefined delay.

Alternatives

- Etcd https://github.com/coreos/etcd
- Serf https://serfdom.io/
- Consul https://consul.io/
- Eureka https://github.com/netflix/eureka

- Doozer, Noah - pretty much dead

Q/A

http://kondratovich.me/talks/august-2015-zk.pdf