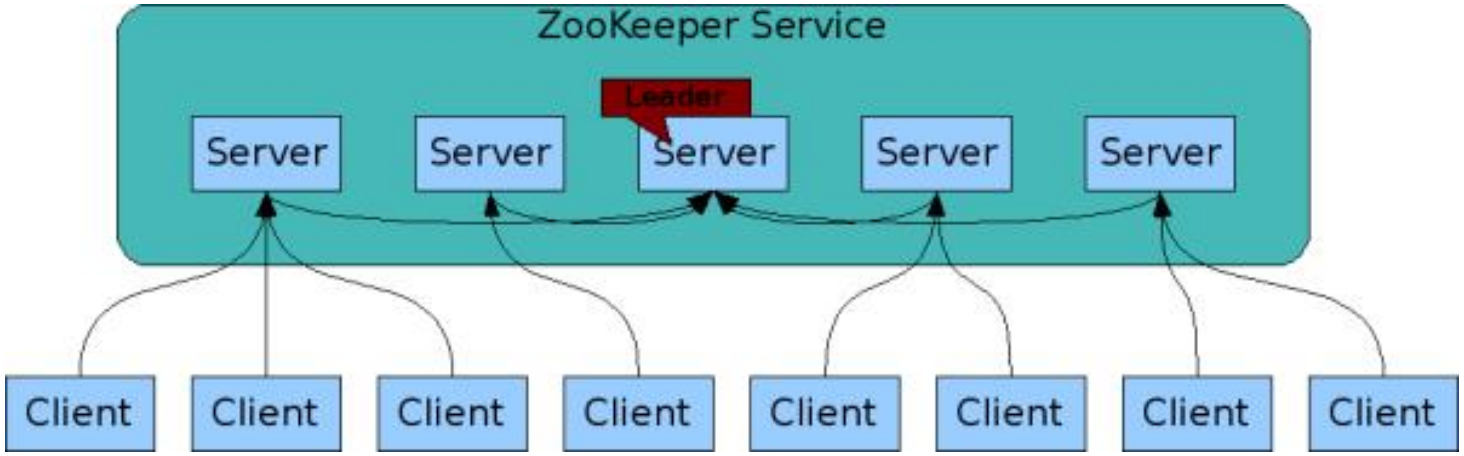
ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ
UNIVERSITY OF CRETE

HY-559
Infrastructure Technologies for Large-Scale Service-Oriented Systems

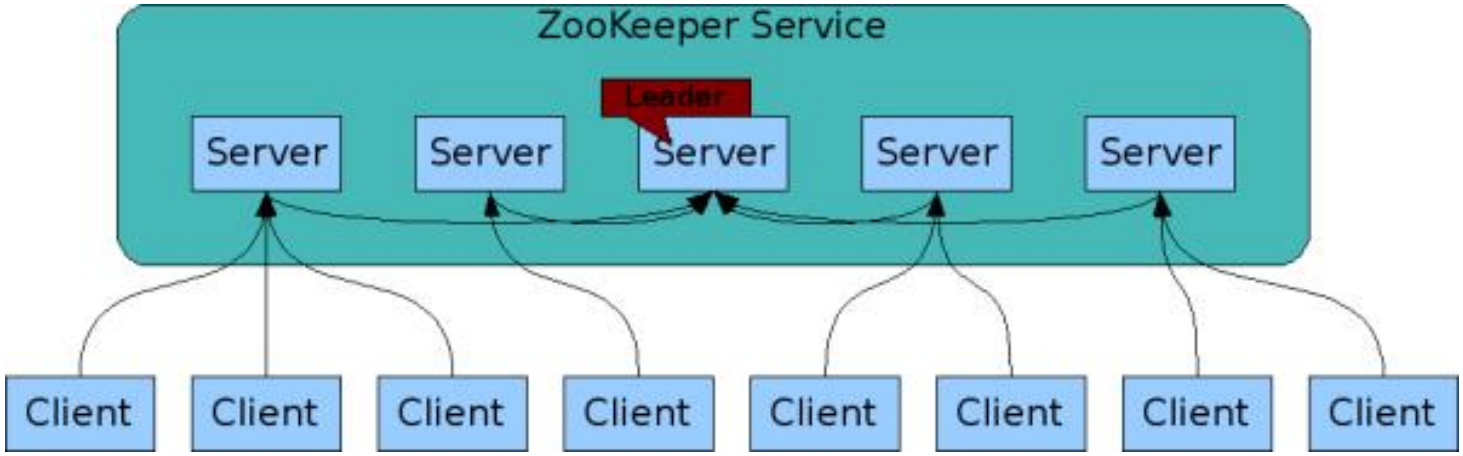
Kostas Magoutis
magoutis@csd.uoc.gr
http://www.csd.uoc.gr/~hy559

ZooKeeper: Wait-free coordination for Internet-scale systems



- Replicated state machine
  - Single leader, multiple followers
- Clients connect to service over a session (FIFO)
  - Service can check client health via heartbeats, deliver notifications to client
- Updates are ordered through leader
  - ZooKeeper atomic broadcast, majority based
- Reads proceed from any (closest) replica

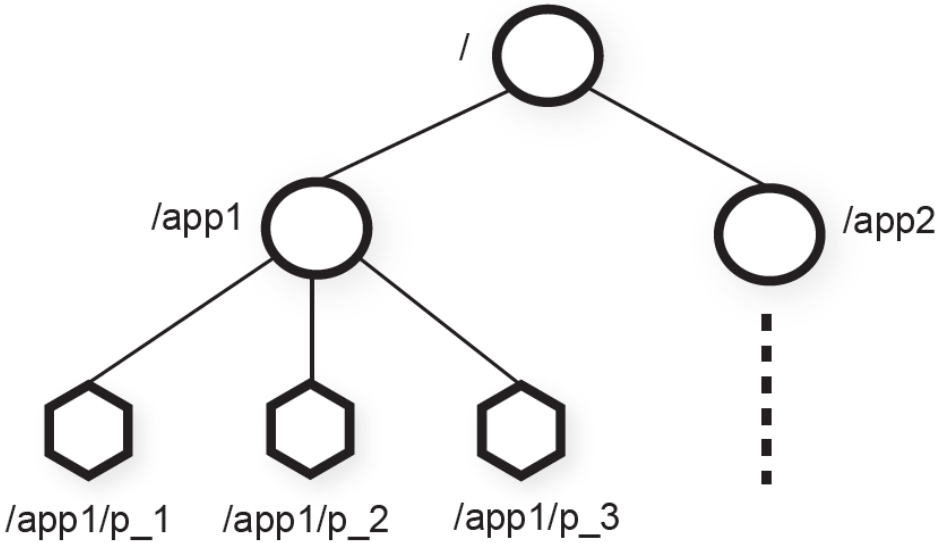
Overview



- Guarantees
  - Writes are linearizable
  - FIFO client ordering of all operations
- Reads can be stale

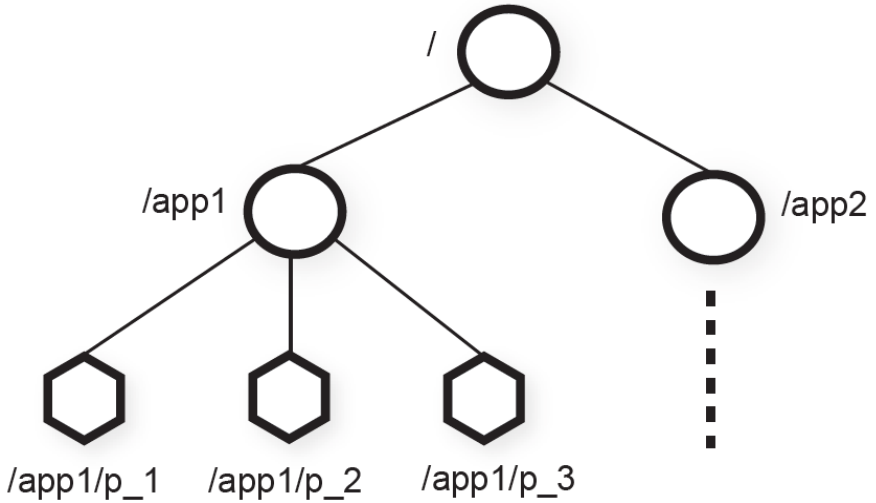
- Notifications (watches)
  - Client request them on updates
  - They do not block write requests
  - Clients notified, then read updated value
- One-time triggers

Hierarchical namespace

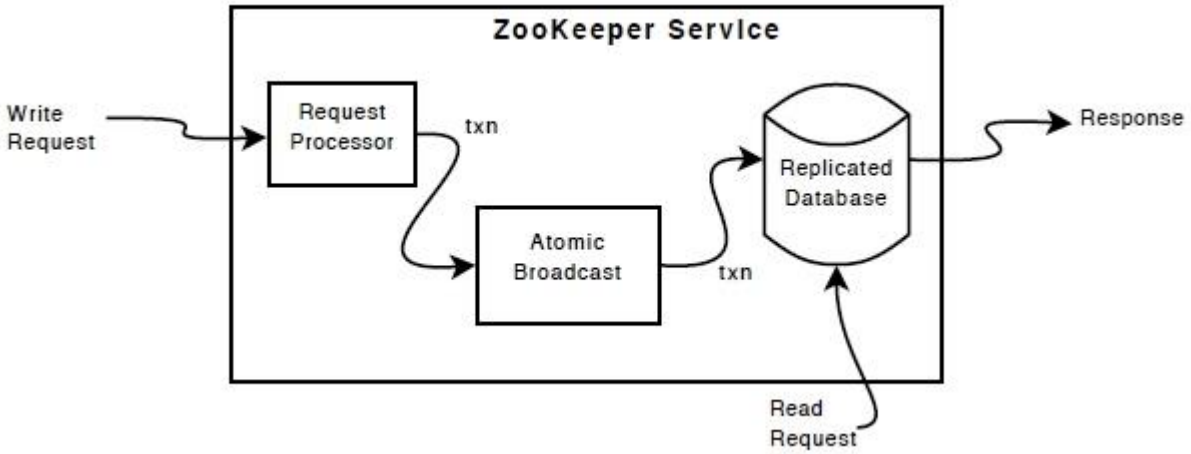


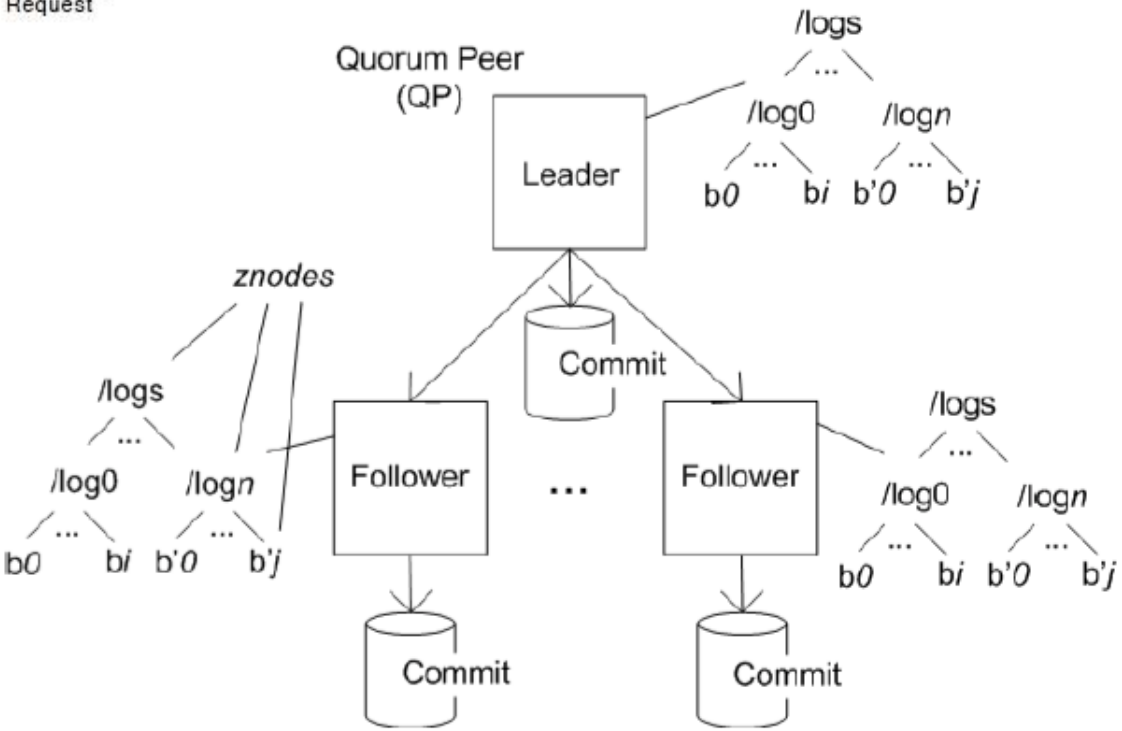
Znode types

- Regular
- Ephemeral
- Sequential



ZooKeeper atomic broadcast (writes only)





Wait-free client API

`create(path, data, flags):` Creates a znode with path name `path`, stores `data[]` in it, and returns the name of the new znode. `flags` enables a client to select the type of znode: regular, ephemeral, and set the sequential flag;

`delete(path, version):` Deletes the znode `path` if that znode is at the expected version;

`exists(path, watch):` Returns true if the znode with path name `path` exists, and returns false otherwise. The `watch` flag enables a client to set a watch on the znode;

`getData(path, watch):` Returns the data and meta-data, such as version information, associated with the znode. The `watch` flag works in the same way as it does for `exists()`, except that ZooKeeper does not set the watch if the znode does not exist;

`setData(path, data, version):` Writes `data[]` to znode `path` if the version number is the current version of the znode;

`getChildren(path, watch):` Returns the set of names of the children of a znode;

`sync(path):` Waits for all updates pending at the start of the operation to propagate to the server that the client is connected to. The path is currently ignored.

Configuration management

- Configuration info stored in a znode named `config`
- Starting processes read `config` with watch flag set
  - `getData ( path=…/app/config, watch = true)`
- If updated …
  - `setData ( path=…/app/config, newConfig, …)`
- … processes receive notification, read `config` again
- They need to set watch flag again

Group membership

- Designate a znode (`workers`) to represent the group
- When a member starts, creates ephemeral znode under `workers`, either with unique name or sequential ID
  - `create ( path=…/workers/w1, data, EPHEMERAL )`
- Processes may put info (IP, port, etc.) in child znode
- If processes crashes, child znode is automatically removed
- Obtain group info
  - `getChildren ( path=…/workers, watch=true )`
- Set watch flag to monitor changes

Simple locks

- Lock files
  - Each lock is represented by a znode L
- To acquire a lock, create ephemeral znode L
  - If it succeeds, you hold the lock
- If lock already held, set a watch flag
  - Holder can release it by deleting L
- If you are notified that L was released, try to acquire
  - Herd effect, only exclusive locking

Locks without Herd Effect

Lock

```
1 n = create(l + "/lock-", EPHEMERAL|SEQUENTIAL)
2 C = getChildren(l, false)
3 if n is lowest znode in C, exit
4 p = znode in C ordered just before n
5 if exists(p, true) wait for watch event
6 goto 2
```

Unlock

```
1 delete(n)
```

Shared locks

Write Lock

```
1  n = create(l + "/write-", EPHEMERAL|SEQUENTIAL)
2  C = getChildren(l, false)
3  if n is lowest znode in C, exit
4  p = znode in C ordered just before n
5  if exists(p, true) wait for event
6  goto 2
```

Read Lock

```
1  n = create(l + "/read-", EPHEMERAL|SEQUENTIAL)
2  C = getChildren(l, false)
3  if no write znodes lower than n in C, exit
4  p = write znode in C ordered just before n
5  if exists(p, true) wait for event
6  goto 3
```