Building a Name Service with ZooKeeper

Albert Kim
Motivation

**Question:** Why do we want a name service?

• How do we find services in a cluster?
  – Clusters have 1000s of machines running

• Registering and querying needs to be dynamic

• Services may change in location or availability
What is ZooKeeper?

- Configuration, synchronization, and naming service for distributed systems
- Provides abstraction of a set of data nodes called *znodes*
- Clients communicate by operating on these *znodes*
Using ZooKeeper

```
/  
|-- cluster1  -- cluster2
    |        |
  /  
|-- service1  -- service2
   |        |
 /  
|-- machine1  -- machine2
```
Why Use ZooKeeper?

• Solved problem of dynamically registering and query for services

• Why not something like DNS?
  – Problem with TTL

• ZooKeeper provides watches
  – Clients can be notified when znode changes
Related Work

• Netflix Curator
  – Provides exactly the service described
  – Even has Service Cache for watching nodes

• zk_watcher
  – Provides client API to use ZooKeeper as a name service for services
What am I doing?

• Weakness of ZooKeeper
  – Everything is fundamentally synchronized at the cluster level

• Some services may be on the same machine
  – Want to use local resources to communicate
  – Want to be notified when configuration changes within a machine
    • Process A was pinned on cpu 1 but moves to cpu 2
Goal

- Measure response times with Curator
- Build a hierarchical name service
- Pluggable into Curator
- Local name services will be more responsive to local configuration changes
- Name service will have more info about how to communicate using other channels than just TCP/IP