Lab 5 – Distributed Key-Value Storage System

Assignment Overview

In this assignment you should build a Zookeeper cluster and implement a distributed key-value storage system demo based on Zookeeper cluster. You will use Zookeeper to do configuration management, cluster management, lock service, naming service in your implementation of distributed key-value system.

Building a Zookeeper cluster

You can refer to resources about how to build a Zookeeper cluster on the Internet. Here, your Zookeeper cluster should contain 3 or more nodes. You are required to record the details of every step on your lab report. Make sure it is well structured. You can deploy your Zookeeper node on a machine as different processes or different machines.

System Architecture

The distributed key-value storage system store and access data based in a client/server architecture. The servers include two type nodes: master node and data node. The master node uses Zookeeper to manages the metadata of distributed key-value storage system. The client communicates with master node to get the location of data node, and then access to the data node directly. The system uses Zookeeper to coordinate different node.

Requirements

- **Node**
  Your distributed key-value storage system should contain one master node and at least two data nodes. You can use different processes to simulate nodes in different machines. The node information is registered in Zookeeper and will used as metadata by master.

- **Communication**
  The communication between client, master node and data nodes use RPC. Choose an RPC framework you like.

- **Data Format**
  The distributed key-value storage system store data in the form of key-value, the type of key and value are String.

- **Partition & Scalability**
  The data stored in data node are distributed to different data node by key. Master node uses metadata and the key of data to determine the location of data node. You may consider the load balance of data. You can design your system to support add data node dynamically for scalability.

- **Operation**
  You should provide three operations for data accessing: PUT, READ, DELETE. PUT operation for creating or updating data. READ operation for reading data from cluster, and DELETE for removing data from cluster.

- **Concurrent Data Accessing**
  Your system should support multi-client. That is to say, you should consider concurrent data accessing. You can use Zookeeper to do concurrency control.

- **High Availability**
  Your system should support high availability for fault tolerance. You should build standby node for data node (at least two), and backup data in standby node. Choose a strategy to keep the consistency between primary data node and standby node you like. While the primary data node is down, using Zookeeper to determine the primary one. The standby node for master node is optional.

Submit Material

- **Source code**
- **Assignment report**
  The assignment report should describe your system environment, zookeeper install and configuration process, your demonstration of your system implementation.

Submit your assignment as a compressed file to Canvas platform, the requirement format for naming the submitted file: Lab5_StudentID_Name.zip

DEADLINE: 23:59 of July 2nd

Grading

50%: assignment report
50%: assignment examinations & defense