CS 6378: Project III

Instructor: Ravi Prakash

Assigned on: April 9, 2024
Due date: May 3, 2024

This is a project to be done in groups of two and you are expected to demonstrate its operation either to the instructor or the TA. All team members must be present during the demonstration. Along with your assignment files, please submit a file clearly stating the contribution of each team member, including who designed/implemented/tested each functionality.

Project Description

Let there be seven data servers, $S_0,\ S_1, \ldots S_6$ and five clients, $C_0,\ C_1, \ldots, C_4$. There exist communication channels between all servers. When a client wishes to perform a read or write, it establishes a communication channel with a server chosen as per the description below. This client-server channel lasts for the duration it takes to complete the read or write operation. All communication channels are FIFO and reliable when they are operational: implemented using sockets. Occasionally, a channel may be disrupted in which case no message can be communicated across that channel. There exists a hash function, H, such that for each object, O_k, H(O_k) yields a value in the range $0 - 6$.

- When a client, C_i has to insert/update an object, O_k, this operation must be performed at three servers numbered: H(O_k), H(O_k)+2 modulo 7, and H(O_k)+4 modulo 7. Instead of contacting all three replicas directly, the client establishes a channel with server identified by H(O_k) and sends its task to that server. Then, that server, acting on behalf of the client ensures that the insert/update operation is performed at the three replicas. If server H(O_k) is not accessible from the client, the client then tries to access server H(O_k)+2 modulo 7, asking it to perform the update on the two live replicas.
- When a client, C_j has to read an object, O_k, it can read the value from any one of the three servers: H(O_k), H(O_k)+2 modulo 7, or H(O_k)+4 modulo 7.

In this project you are required to implement the following (and be able to demonstrate the implementation):

1. A client should be able to randomly choose any of the three replicas of an object when it wishes to read the value of the object. If a client tries to read an object that is not present (has not been inserted earlier by any client), the operation should return with an error code.
2. When a client wishes to update/insert an object into the data repository, it should be able to successfully perform the operation on at least two, and if possible all the three servers that are required to store the object.
3. If a client is unable to access at least two out of the three servers that are required to store an object, then the client does not perform updates to any replica of that object.
4. If two or more clients try to concurrently write to the same object and at least two replicas are available, the writes must be performed in the same order at all the replicas of the object.
5. You must also selectively disable some channels so that the servers get partitioned into two components, each with a subset of servers such that some updates are permitted in one partition, while other updates are permitted in the other partition.
6. After network partitions are merged and it is found that some writes have been performed at two out of the three replicas of an object, the third replica should be able to synchronize with the other two replicas before any further update to that object is performed and before any read of that object is performed from the third replica.

You will need to demonstrate that you have implemented all the requirements mentioned above. For instance, you may need to selectively disable some channel(s) for a brief period of time so that access two or one replica of an object is disrupted temporarily. If at least two replicas of the object are accessible, updates to that object should be allowed. However, if only one replica of the object is accessible, the update should be aborted and a corresponding message should be sent to the client.

Point Distribution

Design Document (20%): Employ appropriate software engineering practices to prepare this document.

Implementation (30%): Source code of your well structured and well documented program. You are required to write your code in C, C++ or Java.

Correctness (35%): Description of the various design choices you have made, for example the hash function, the protocol for permitting writes, etc. and what are the safety and liveness conditions you set about satisfying. The description should indicate how the safety and liveness conditions are satisfied.

Report (15%): Results of all your experiments presented in an understandable form.

Submission Information

The project is to be submitted via eLearning.