Mandatory Exercise 3

thhk, jglr, emja, krbh (Group AD)

November 24th 2020

A

1) Propose a communication protocol for the system

TCP would be optimal, as it is crucial that the user knows if a bid did not go through. If UDP was used, the user would not know if an error occured when he/she made a bid.

2) How many nodes should be setup?

A cluster of a minimum of three nodes. Where a Topic is created per item available for bidding, which is then partitioned and distributed among the node (brokers).

Kafka claims to be fault-tolerant, because if one server fails, other servers can take on the work of the failed server. The more servers there are, the more servers can fail without causing issues.

The number of clients is insignificant to the operation of the service, as they only access the service.

3) What network topology you are recommending?

A tree structure would be a good fit. Clients don't need to be connected to each other, so a tree structure where all clients are leaves, connected to servers, would make sense. This way the servers would also primarily be connected to other servers, such that they can keep data about bids up to date.

4) How to configure the value Minimum In Sync Replicas

The value, Minimum In Sync Replicas, can be set on two levels. Setting the value on Broker-level, that is to set a default value that any new Topic will inherit if not overridden on a Topic-level.

B

1) Describe the leader election process

A Broker is, by the ZooKeeper, appointed as the Controller. The Controller watches for Broker failures and is resposible for electing new partition leader, if a leader Broker fails [1]. Also a ISR(In Sync Replicant)-list, that is a list of all Follower replicants for a Topic which are up to date on events, is maintained [2]. The ZooKeeper is informed if this list changes, and is thus informed of which Replicants are capable of taking on the role as leader. The first follower in the list will be picked as leader [3].

2) Is it an active or passive replication architecture?

It is a passive replication architecture. It functions such that the leader server essentially handles all traffic, but the other ones are there to take over, in case of a failure.

3) Is your setup sequentially consistent?

Yes, this is a guaranteed by using Kafka. " ... Kafka guarantee the order [of the events] and it's one of the reasons for choosing kafka."[4]

4) Is your setup linearizable?

Again, this is a guaranteed by using Kafka. "... Kafka guarantees that any consumer of a given topic-partition will always read that partition's events in exactly the same order as they were written."[5]

[1] *4.7: Replica Management* in https://kafka.apache.org/documentation/
[2] *4.7: Replicated Logs: Quorums, ISRs, and State Machines (Oh my!)* in https://kafka.apache.org/documentation/
[3] *6.1: Balancing leadership* in https://kafka.apache.org/documentation/
[4] https://medium.com/@felipedutratine/kafka-ordering-guarantees-99320db8f87f
[5] *1.1: Main Concepts and Terminology* in https://kafka.apache.org/documentation/

C

1) What fault tolerance does your system setup guarantee

In A.2 we declared to have 3 nodes, and if we follow the recommended replication setup of 3 replicas [6], then that system will maintain full functionality after 2 of the 3 nodes have failed, because each node will keep a replicate of any topic.

2) What happens in case of the network failure?

Without a network connection, the system is unable to receive client requests and therefore availability fails.

3) What happens in case of a node failure?

Reelection must occur, if the failed node was a leader of some Topic partition.

4) How many nodes can fail before availability of the system is impacted?

If there are $f + 1$ nodes, then f failures can happen and the system will still have functionality.

5) How many nodes can fail before fault tolerance of the system is impacted?

If there are $f + 1$ replicants pr. Topic partition, then the topic partition can survive f failures.

[6] *1.1: Main Concepts and Terminology* in https://kafka.apache.org/documentation/