Introduction to Blockchain Ecosystem: Assignment 2

Sara Baradaran

Question 1

Recall the PBFT implementation in Assignment 1. Now implement the following the protocol described below. Compare the prefix-closedness of this protocol against PBFT in terms of actual working code by constructing at least 2 executions in which this protocol presents different outcomes from PBFT. Please prepare two scripts for me to execute that will present the output of these two executions. Then, explain in text, in your own words, the sequence of steps in the executions that lead to the safety violations.

There are several issues with the protocol described. First, no messages would be broadcasted in this protocol. The nodes directly send their responses to the primary without considering the fact that a primary might be malicious. They also receive the next block directly from the primary without even knowing what is the next block received by the other nodes and without ensuring agreement on the next block. In this protocol, telling different things to different non-faulty nodes easily breaks the safety property.

Scenario 1: Assume an execution scenario where the primary in round l is malicious. It send a value v_l to all other nodes and receives $f+1$ reply messages ($S = \{r_1, r_2, r_3, ...r_{f+1}\}$) from nodes. In this step, all the nodes expect to receive $O_1, ...O_l$ from the primary, where $O_l = min(S \cup v_l)$. Now, assume $\exists 1 \leq i \leq f+1; min(S \cup v_l) = r_i$. Here, the primary can do equivocation by sending $O_1, ...r_i$ to f non-faulty nodes and a different chain, for example, $O_1, ...r_j$ (where $r_j < r_i$), to $f+1$ other non-faulty nodes, causing a fork in chains and safety violation. This happens because the honest nodes do not know the actual minimum value.

Scenario 2: Assume another execution scenario where the primary in round l is malicious. It proposes a value v_l to f non-faulty nodes, while it proposes a different value v_t to $f+1$ remaining non-faulty nodes. In such a case, if all the nodes propose values bigger than both v_l and v_t, then $min(S \cup v_l) = v_l$ and $min(S \cup v_r) = v_r$. Here, f non-faulty nodes receive $O_1, ...v_l$ as the next chain, while $f+1$ honest nodes receive $O_1, ...v_t$ as the next chain, causing a fork and safety violation.

However, the scenarios above are not possible in PBFT protocol since all the messages would be broadcasted, and $2f+1$ commit messages are required for each node to execute an operation proposed. If a node does not receive $2f+1$ commit messages, it ignores the operation proposed after a timeout, and the nodes try to change the primary by sending view change messages (Note: I have not implemented view change as a part of PBFT protocol. Thus, in my implementation, the nodes only ignore the operation proposed after a timeout when they could not reach a consensus).