

This is a high-level specification of *SCP* focusing on the nomination protocol.

Currently, as implemented, before voting for a txset hash, nodes wait to obtain its preimage. Delaying the point at which we wait for the pre-image would leave more room for disseminating the txset in parallel to nomination. However this has to be done carefully to maintain the main property of nomination: assuming that there is a nomination round with a good leader and during which the network is fast enough, at least a Tier-1 quorum must eventually enter balloting.

In the version specified in this document, we do not wait on the pre-image to vote for a txset hash, but we do wait for the pre-image before accepting it.

In the previous version of this document, we even accepted without a pre-image. There is a problem with this: it could create a situation in which not enough nodes can start balloting (*i.e.* not a full quorum) and the whole system is stuck.

The problem stems from the fact that, in the nomination protocol, nodes that confirm a candidate then stop voting for new values (otherwise nomination is not guaranteed to converge). So if a blocking set B confirms a candidate but somehow other nodes cannot get the pre-images they need to do so, more nomination rounds will not help because the members of B have stopped voting, which blocks the progress of any new candidate. Depending on how pre-images are disseminated, this can potentially be exploited by an attacker to halt the system.

So accepting without a pre-image is only workable if there is some way to guarantee that, once a Tier-1 blocking set has a pre-image, then everybody in Tier-1 eventually gets it.

Another problem is that we want it to be likely that a quorum starts balloting already in agreement and roughly at the same time. If we delay checking pre-images to the confirm stage, an attacker could first send the pre-image to a set A of nodes, which then enter balloting at time T_A , but not send the pre-image to another set B of nodes, which then enter balloting at time $T_B > T_A$ because they need to get the pre-image from A before starting balloting. For example, if it takes $500ms$ for members of B to get the pre-image from members of A , then $T_B = T_A + 500ms$. This can cause the first ballot to end without a decision. Members of B could also start a new nomination round before T_B and then enter balloting not only late but also with a different value than members of A .

EXTENDS *Naturals, FiniteSets*

CONSTANTS

V , validators
 $TxSet$, blocks
 Bot , default value
 $Quorum(-)$, $Quorum(v)$ is the set of quorums of validator v
 $Blocking(-)$, $Blocking(v)$ is the set of blocking sets of validator v
 $Combine(-)$, the functions that combines candidates to produce a txset for balloting
 H , domain of hashes
 $Hash(-)$ hash function

VARIABLES $txSetForBalloting$, $voted$, $accepted$, $round$, $candidates$,
 $preImage$, $leader$

$vars \triangleq \langle txSetForBalloting, voted, accepted, round, candidates,$
 $preImage, leader \rangle$

$$\begin{aligned}
Init &\triangleq \\
&\wedge txSetForBalloting = [v \in V \mapsto Bot] \\
&\wedge voted = [v \in V \mapsto \{\}] \\
&\wedge accepted = [v \in V \mapsto \{\}] \\
&\wedge round = [v \in V \mapsto 0] \\
&\wedge candidates = [v \in V \mapsto \{\}] \\
&\wedge preImage = [v \in V \mapsto [h \in H \mapsto Bot]] \\
&\wedge leader = [v \in V \mapsto Bot] \\
\\
StartRound(v) &\triangleq \\
&\wedge round' = [round \text{ EXCEPT } ![v] = round[v] + 1] \\
&\wedge \exists l \in V : \\
&\quad \wedge leader' = [leader \text{ EXCEPT } ![v] = l] \\
&\quad \wedge \text{IF } l = v \\
&\quad \quad \text{THEN } \wedge \exists txs \in TxSet : \\
&\quad \quad \quad \wedge preImage' = [preImage \text{ EXCEPT } ![v][Hash(txs)] = txs] \\
&\quad \quad \quad \wedge voted' = [voted \text{ EXCEPT } ![v] = voted[v] \cup \{Hash(txs)\}] \\
&\quad \quad \text{ELSE UNCHANGED } \langle voted, preImage \rangle \\
&\wedge \text{UNCHANGED } \langle txSetForBalloting, accepted, candidates \rangle \\
\\
Vote(v) &\triangleq \\
&\wedge \text{IF } candidates[v] = \{\} \\
&\quad \text{THEN } \wedge leader[v] \neq Bot \\
&\quad \quad \wedge \text{LET } hs \triangleq voted[leader[v]] \text{ IN} \\
&\quad \quad \quad \wedge hs \neq \{\} \\
&\quad \quad \quad \wedge voted' = [voted \text{ EXCEPT } ![v] = voted[v] \cup hs] \\
&\quad \text{ELSE UNCHANGED } voted \\
&\wedge \text{UNCHANGED } \langle txSetForBalloting, accepted, round, candidates, preImage, leader \rangle \\
\\
VotedHashes &\triangleq \text{UNION } \{voted[v] : v \in V\} \\
\\
GetTxSet(v, txs) &\triangleq \\
&\wedge Hash(txs) \in VotedHashes \\
&\wedge preImage' = [preImage \text{ EXCEPT } ![v][Hash(txs)] = txs] \\
&\wedge \text{UNCHANGED } \langle txSetForBalloting, voted, accepted, round, candidates, leader \rangle \\
\\
Accept(v, h) &\triangleq \\
&\wedge preImage[v][h] \neq Bot \\
&\wedge \vee \exists Q \in Quorum(v) : \forall w \in Q : h \in voted[w] \vee h \in accepted[w] \\
&\quad \vee \exists Bl \in Blocking(v) : \forall w \in Bl : h \in accepted[w] \\
&\wedge accepted' = [accepted \text{ EXCEPT } ![v] = accepted[v] \cup \{h\}] \\
&\wedge \text{UNCHANGED } \langle txSetForBalloting, voted, round, candidates, preImage, leader \rangle \\
\\
Confirm(v, h) &\triangleq \exists Q \in Quorum(v) : \\
&\quad \wedge preImage[v][h] \neq Bot \\
&\quad \wedge \forall w \in Q : h \in accepted[w] \\
&\quad \wedge candidates' = [candidates \text{ EXCEPT } ![v] = candidates[v] \cup \{preImage[v][h]\}]
\end{aligned}$$

$$\begin{aligned} &\wedge \text{ txSetForBalloting}' = [\text{txSetForBalloting} \text{ EXCEPT } ![v] = \text{Combine}(\text{candidates}'[v])] \\ &\wedge \text{ UNCHANGED } \langle \text{voted}, \text{accepted}, \text{round}, \text{preImage}, \text{leader} \rangle \end{aligned}$$

$$\begin{aligned} \text{Next} &\triangleq \exists v \in V, \text{ txs} \in \text{TxSet}, h \in H : \\ &\quad \vee \text{ StartRound}(v) \\ &\quad \vee \text{ Vote}(v) \\ &\quad \vee \text{ GetTxSet}(v, \text{ txs}) \\ &\quad \vee \text{ Accept}(v, h) \\ &\quad \vee \text{ Confirm}(v, h) \end{aligned}$$

Here we assume that all agree on a leader in round 3 and stay in round 3 forever (for liveness)

$$\begin{aligned} \text{LeaderAgreement} &\triangleq \\ &\quad \wedge \exists l \in V : \forall v \in V : \text{round}[v] = 3 \Rightarrow \text{leader}[v] = l \\ &\quad \wedge \forall v \in V : \text{round}[v] \leq 3 \end{aligned}$$

$$\begin{aligned} \text{Spec} &\triangleq \\ &\quad \wedge \text{ Init} \\ &\quad \wedge \Box[\text{Next} \wedge \text{LeaderAgreement}']_{\text{vars}} \\ &\quad \wedge \forall v \in V, \text{ txs} \in \text{TxSet}, h \in H : \\ &\quad \quad \wedge \text{WF}_{\text{vars}}(\text{StartRound}(v) \wedge \text{round}[v] \leq 2) \\ &\quad \quad \wedge \text{WF}_{\text{vars}}(\text{GetTxSet}(v, \text{ txs})) \\ &\quad \quad \wedge \text{WF}_{\text{vars}}(\text{Vote}(v)) \\ &\quad \quad \wedge \text{WF}_{\text{vars}}(\text{Accept}(v, h)) \\ &\quad \quad \wedge \text{WF}_{\text{vars}}(\text{Confirm}(v, h)) \end{aligned}$$

The type-safety invariant:

$$\begin{aligned} \text{TypeOkay} &\triangleq \\ &\quad \wedge \text{ txSetForBalloting} \in [V \rightarrow \text{TxSet} \cup \{\text{Bot}\}] \\ &\quad \wedge \text{ voted} \in [V \rightarrow \text{SUBSET } H] \\ &\quad \wedge \text{ accepted} \in [V \rightarrow \text{SUBSET } H] \\ &\quad \wedge \text{ round} \in [V \rightarrow \text{Nat}] \\ &\quad \wedge \text{ candidates} \in [V \rightarrow \text{SUBSET } \text{TxSet}] \\ &\quad \wedge \text{ preImage} \in [V \rightarrow [H \rightarrow \text{TxSet} \cup \{\text{Bot}\}]] \\ &\quad \wedge \text{ leader} \in [V \rightarrow V \cup \{\text{Bot}\}] \end{aligned}$$

Liveness: if a validator enters balloting, then eventually all do.

$$\begin{aligned} \text{Liveness} &\triangleq \\ &\quad \forall v \in V : \Box(\text{txSetForBalloting}[v] \neq \text{Bot} \\ &\quad \Rightarrow \exists t \in \text{TxSet} : \Diamond(\forall w \in V : \text{txSetForBalloting}[w] = t)) \end{aligned}$$

Liveness: eventually, all converge on a txset for balloting.

$$\begin{aligned} \text{Liveness2} &\triangleq \\ &\quad \exists t \in \text{TxSet} : \Diamond(\forall v \in V : \text{txSetForBalloting}[v] = t) \end{aligned}$$

Definition for model-checking:

Concrete hashing for the model-checker:

$TestH \triangleq 1 \dots Cardinality(TxSet)$

$TestHash(b) \triangleq$

LET $f \triangleq$ CHOOSE $f \in [TxSet \rightarrow H] : \forall txs1, txs2 \in TxSet : txs1 \neq txs2 \Rightarrow f[txs1] \neq f[txs2]$
IN $f[b]$

Debugging canaries:

$Canary2 \triangleq \forall v \in V : Cardinality(candidates[v]) \leq 1$

$Canary3 \triangleq \forall v \in V : txSetForBalloting[v] = Bot$

$TestQuorums \triangleq \{Q \in SUBSET V : 2 * Cardinality(Q) > Cardinality(V)\}$

$TestBlocking \triangleq \{Bl \in SUBSET V : Cardinality(Bl) > 1\}$

\ * Modification History

\ * Last modified *Fri Mar 31 11:17:40 PDT 2023* by *nano*

\ * Created *Fri Jan 13 09:09:00 PST 2023* by *nano*