

MODULE *Broadcast*

The specification captures the *DAG* base reliable broadcast to disseminate shares over a peer to peer network.

The broadcast enables nodes to know which nodes have received the message by using implicit acknowledgements. The broadcast is not a *BFT* broadcast. We depend on the higher layers to provide that.

Does this open this broadcast to a *DDoS* attack? Yes, and our argument remains that *p2p* network can resist *DDoS* attacks by other means.

First pass - We assume no processes failures or messages lost.

EXTENDS *Naturals, Sequences*

CONSTANT

Proc, Set of processes
Data,
Nbrs

VARIABLES

sent_by, Set of messages sent by processes to their neighbours
recv_by Set of messages received by processes

$Message \triangleq [from : Proc, data : Data]$

$Init \triangleq$
 $\wedge sent_by = [m \in Message \mapsto \{\}]$
 $\wedge recv_by = [m \in Message \mapsto \{\}]$

$TypeInvariant \triangleq$
 $\wedge sent_by \in [Message \rightarrow Proc]$
 $\wedge recv_by \in [Message \rightarrow Proc]$

SendTo(*m*, *p*) - send message *m* to neighbour *p*

Sending to self is required as then the message is in the recv list as well.

$SendTo(m, p) \triangleq$
 $\wedge m.from \notin sent_by[m]$ Don't send again - we can add decay here
 $\wedge \langle m.from, p \rangle \in Nbrs$ Send only to neighbours
 $\wedge sent_by' = [sent_by \text{ EXCEPT } ![m] = @ \cup \{p\}]$
 $\wedge \text{UNCHANGED } \langle recv_by \rangle$

RecvAt(*m*, *q*) - receive message *m* at *q*. This can be received from forwards

$RecvAt(m, q) \triangleq$
 $\wedge \exists p \in Proc : p \in sent_by[m]$ Some process has sent the message
 $\wedge q \notin recv_by[m]$ Not already received by *q*
 $\wedge recv_by' = [recv_by \text{ EXCEPT } ![m] = @ \cup \{q\}]$
 $\wedge \text{UNCHANGED } \langle sent_by \rangle$

$Forward(m, p, q)$ - forward message m from p to q
 - enabling condition - m has been sent by some process, q has received the message, q is not the sender
 - effect - p forwards the message m to its nbrs

$Forward(m, p, q) \triangleq$
 $\wedge \exists r \in Proc : r \in sent_by[m]$ Some process has sent the message
 $\wedge p \neq q$ Don't forward to self
 $\wedge \langle p, q \rangle \in Nbrs$ Forward only to neighbour
 $\wedge p \in recv_by[m]$ p has received m
 $\wedge sent_by' = [sent_by \text{ EXCEPT } ![m] = @ \cup \{q\}]$
 $\wedge \text{UNCHANGED } \langle recv_by \rangle$

$Next \triangleq \exists p \in Proc, q \in Proc, m \in Message :$
 $\vee SendTo(m, p)$
 $\vee RecvAt(m, p)$
 $\vee Forward(m, p, q)$

$Spec \triangleq \wedge Init$
 $\wedge \square [Next]_{\langle sent_by, recv_by \rangle}$

THEOREM $Spec \Rightarrow \square TypeInvariant$

\ * Modification History
 \ * Last modified Tue Mar 07 22:50:27 CET 2023 by kulpreet
 \ * Created Sun Mar 05 15:04:04 CET 2023 by kulpreet