Reliable Local Broadcast in a Wireless Network Prone to Byzantine Failures *

Vartika Bhandari
Dept. of Computer Science, and
Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
vbhandar@uiuc.edu

Nitin H. Vaidya
Dept. of Electrical and Computer Eng., and
Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
nhv@uiuc.edu

ABSTRACT

Reliable broadcast can be a very useful primitive for many distributed applications, especially in the context of sensor-actuator networks. Recently, the issue of reliable broadcast has been addressed in the context of the radio network model that is characterized by a shared channel, and where a transmission is heard by all nodes within the sender's *neighborhood*. This basic defining feature of the radio network model can be termed as the *reliable local broadcast* assumption. However, in actuality, wireless networks do not exhibit such perfect and predictable behavior. Thus any attempt at distributed protocol design for multi-hop wireless networks based on the idealized radio network model requires the availability of a reliable local broadcast primitive that can provide guarantees of such idealized behavior. We present a simple proof-of-concept approach toward the implementation of a reliable local broadcast primitive with probabilistic guarantees, with the intent to highlight the potential for lightweight scalable solutions to achieve probabilistic reliable local broadcast in a wireless network.

Categories and Subject Descriptors

C.2.1 [**Computer Communication Networks**]: Network Architecture and Design—*Wireless communication*; C.2.4 [**Computer-Communication Networks**]: Distributed Systems; C.4 [**Performance of Systems**]: Fault Tolerance

General Terms

Algorithms, Reliability

Keywords

Wireless Networks, Byzantine failure, Local Broadcast, Fault Tolerance

*This research was supported in part by the National Science Foundation, US Army Research Office grant W911NF-05-1-0246, and a Vodafone Graduate Fellowship.

1. INTRODUCTION

As deployment and use of wireless networks for various roles, ranging from community mesh networks to sensor networks, becomes increasingly common, the reliability of communication in these networks is of growing concern. Reliable communication in wireless networks is fairly non-trivial, in large part due to significant time-variations in channel quality and the possibility of interference/collisions because of the shared nature of the medium.

One of the issues in this regard is that of achieving reliable broadcast in a wireless network, given that some nodes may exhibit Byzantine failure. The ability to perform this operation can be extremely useful, especially in the context of sensor-actuator networks, where nodes may need to perform some coordinated action based on a consistent view of events sensed by individual nodes.

Recent theoretical work on Byzantine fault-tolerant broadcast in radio networks [11], [1], [2], [3] assumes that if a node transmits a message it is received by each and every node within a designated neighborhood in its spatial vicinity. This eliminates the potential for *duplicity* and ensures local agreement as follows: when the sender is non-faulty, agreement is trivial, since all non-faulty neighbors of a non-faulty sender receive the message directly. If the sender is faulty and sends multiple conflicting copies of the message, all non-faulty neighbors receive all messages in the same order, and can agree on one (say the first). While this model reflects the shared nature of the wireless medium, it fails to capture its unreliability. The wireless medium can be extremely unreliable, and show highly variable channel quality over time, due to factors such as fading. This leads to significant fluctuation in received signal, and hence there is a non-negligible probability of unsuccessful reception, even in the absence of malicious collision-causing behavior.

Thus, any attempt at designing reliable broadcast protocols based on theoretical radio network results must begin with an effort to implement a *reliable local broadcast* primitive in a *scalable* manner.

One might envision implementing local broadcast by running a point-to-point Byzantine agreement protocol, with retransmissions over every lossy (point-to-point) link to handle channel errors. However, such a solution lacks scalability, as the underlying medium is shared and thus the operation of nearby (point-to-point) links must be serialized.

While the issue of reliable broadcast and consensus in the presence of a bounded number of collisions/spoofings has been addressed in previous work, such as [12] and [9], probabilistic channel losses have not been factored in. Random

transient Byzantine failures that include collision-causing is examined in [19]. Though also of a probabilistic nature, their model is different in that nodes either fail to transmit, transmit a wrong value or transmit out of turn, with a certain probability, in each round.

In this work we address channel unreliability, while assuming fault-free physical(PHY) and medium-access control(MAC) layers (i.e., nodes do not deliberately cause collision or spoof MAC addresses). We describe a simple proof-of-concept approach towards implementing reliable local broadcast with probabilistic guarantees in a *local broadcast domain*. We also briefly discuss how the proposed reliable local broadcast solution can be optimized further, and also be used as a sub-protocol in a global broadcast algorithm for multi-hop networks. Our primary intent in this paper is to highlight the potential for lighweight scalable solutions to achieve probabilistic reliable local broadcast in the face of a lossy wireless channel, by exploiting loose synchronization between the clocks of nearby nodes.

2. RELATED WORK

Since the seminal result of Pease, Shostak and Lamport [17], [14], there has been much work on Byzantine agreement. In recent times, there has been a focus on agreement/consensus problems in broadcast/multicast channels. Such models can be useful for reasoning about wireless networks.

Byzantine agreement in k-cast channels has been considered in [8]. However this model is not directly relevant to wireless networks. as it does not capture the spatially dependent connectivity that characterizes these networks. Reliable broadcast in radio networks deployed as an infinite regular grid was studied in [11]. A locally-bounded fault model was proposed wherein an adversary was allowed to place faults subject to the constraint that no neighborhood have more than t faults. It was shown that under a Byzantine failure model, reliable broadcast is not achievable for $t \geq \lceil \frac{1}{2} r(2r+1) \rceil$ in the L_∞ metric. This was later established as an exact threshold for the L_∞ metric in [1]. Additionally, an approximate threshold was established for the L_2 metric.

In [18], locally bounded faults were studied in arbitrary graphs. While the discussion mentions both radio and message-passing networks, there is an assumption that duplicity (sending different messages to different neighbors) is impossible, which seems to stem from the radio network model. Upper and lower bounds for achievability of reliable broadcast were presented, based on graph-theoretic parameters, for arbitrary graphs.

Probabilistic transient failures were considered in [19] which examines the case of both message-passing and radio networks. The considered model assumes arbitrary graph topologies.

Probabilistic permanent Byzantine failures were examined in [3], [4]. Two network models were considered: a regular grid and a randomly deployed network. Necessary and sufficient conditions were established for the critical transmission range (and hence node degree) to tolerate failure probabilities strictly less than $\frac{1}{2}$.

The case of an adversary capable of causing a bounded number of collisions in an infinite (or finite-toroidal) grid network was considered in [12], and it was shown that the ability to cause a bounded number of collisions or address-spoofings does not yield the adversary any additional power in thwarting broadcast, i.e., the tolerable fault threshold remains the same as for the no-collision case of [11] and [1]. However this result is based on the assumption that non-faulty nodes are not hindered by energy-limitations, and can retransmit messages as many times as needed. The impact of an energy-budget on consensus was studied for a single-hop setting in [9], and it was proved that non-faulty nodes would require at least incrementally larger budget than faulty nodes to arrive at a consensus. The transient failure behavior assumed in [19] also includes the possibility of nodes causing collision.

Much of the theoretical work mentioned earlier assumes that the wireless channel itself is perfectly reliable. The random lossy nature of the channel is not accounted for, and thus many of these results are not directly applicable to a real-world scenario. A proposal to reconcile the theory and practice of wireless broadcast has been made in [6]. They identify certain properties that a reliable local broadcast should have. They introduce some models to capture the nature of losses and collisions, viz., the No-Collisions(NC) model, the Eventual No-Collisions (ENC) model, the Total Collision (TC) model, and the Partial Collision (PC) Model. Of these the TC-model most closely resembles the reliable local broadcast assumption, in that if a message is received by one recipient, then it is received by all recipients. It was shown in [6] that in a single-hop network conforming to the TC-model consensus is achievable with any number of Byzantine/crash-stop failures. This is akin to the trivial local agreement property of the reliable local broadcast asumption. However, practical realization of the TC model is not delved into in detail in [6] (though they mention the possibility of using signal-jamming techniques to achieve TC properties). On a related note, consensus in single-hop wireless networks with crash-prone nodes is considered in [7].

Also relevant to our work is the notion of reliable multicast with probabilistic guarantees [5], [15] which also seeks to achieve a scalable solution with probabilistic guarantees.

3. HOW A LOSSY WIRELESS CHANNEL INHIBITS RELIABLE LOCAL BROADCAST

In this section we briefly discuss how an unreliable wireless channel can affect the achievability of reliable local broadcast.

Consider a source s that originates a message, which needs to be locally broadcast to its neighbors. However, as the channel is lossy, each neighbor successfully receives the message only with a certain probability. Resultantly, it is possible that a transmission may only be heard by some subset of s's neighbors. If s were non-faulty, this issue is readily resolved by having s retransmit the message sufficient times to ensure that each neighbor receives at least one copy with high probability (w.h.p.). However consider what might transpire if s is faulty, and seeks to leverage the channel's unreliability to create confusion amongst its neighbors.

Suppose that s initially sents a message m with value 0. Some of its neighbors do not receive it, i.e., it is received by some subset $\mathcal{N}_1$ of s's neighbors. It then sends another copy of the same message, containing a value 1. This message is received by some subset $\mathcal{N}_2$. If $\mathcal{N}_1 - \mathcal{N}_2$ is non-empty, there are certain nodes that will assume that s sent only one value,

i.e., 0. If $\mathcal{N}_2 - \mathcal{N}_1$ is non-empty, there are certain nodes that will assume that s sent only one value, i.e., 1. Nodes in $\mathcal{N}_1 \cap \mathcal{N}_2$ receive both values, and are in a position to detect s's duplicity. These nodes can choose a default value, e.g., the first value sent by s. However, there still remains the issue of ensuring that the other nodes do the same. One approach might consist in the raising of an alarm by nodes in $\mathcal{N}_1 \cap \mathcal{N}_2$, but would require a means for the other nodes to resolve whether the alarm(s) are to be trusted.

Thus, one may prefer to have a more lightweight approach to ensure agreement of all nodes on a common value (and potentially rely on the fact that after a number of duplicitous transmissions by s, all nodes would at some time detect its duplicity themselves, and s could be universally identified as untrustworthy).

4. CAUSAL ORDERING AND PHYSICAL CLOCKS

In this section, we briefly review notions of clocks and ordering that are relevant to the discussion in this paper.

We assume the existence of some frame of reference external to the system. The physical time in this frame of reference is considered to be an absolute measure of physical time, for the purpose of our discussion. Thus at time instant t, the external clock value is t.

Each node u in the system has its own physical clock. The clock value of a node u at time instant t is denoted by $C_u(t)$. When we refer to external synchronization within bound D, we imply synchronization to this ideal external clock within bound D, i.e., at each time instant t: $|C_u(t) - t| \leq D$. Clock drift is modeled as being linear, i.e., if the true time elapsed is T, the observed elapsed time lies in the range $[(1-\delta)T, (1+\delta)T]$, where δ is the drift per unit time (also referred to as drift-rate). When we refer to internal synchronization within bound D, we imply that at any time instant t, the clocks of two internally synchronized nodes u, w satisfy: $|C_u(t) - C_w(t)| \leq D$. When we refer to a node adjusting its clock, we imply that the node applies a correction to its clock value.

In his seminal paper [13], Lamport proposed that the key goal in a distributed system should be to ensure that causal relationships are respected. This causality could be captured in a *happened-before* relation, which imposes a partial order on system events. Thus $a \rightarrow b$ implies that *a happened-before* b, and b may be causally affected by a. Let $C(a)$ denote the time observed for an event a as per a clock C. A satisfactory clock C must then satisfy the following:

Clock Condition [13]. For any events a, b: $a \rightarrow b \implies C(a) < C(b)$.

To this effect, Lamport logical clocks were proposed in [13]. An anomalous scenario was also considered whereby out-of-system message exchanges could lead to violation of the Clock Condition. Thus one might consider a *Strong Clock Condition* whereby causal ordering is preserved even taking into account out-of-system messages. It was observed in [13] that if clock drift rate δ, maximum clock skew (or synchronization bound) D and minimum message transmission time T_l satisfy the relation: $T_l \geq \frac{D}{1-\delta}$, then the system of *physical* clocks satisfies the Strong Clock Condition. It was also shown that a simple synchronization algorithm suffices to ensure that clock skew is bounded by a suitable D.

The notion of leveraging physical clocks rather than logical clocks has wider significance. Consider a system where some processes may exhibit Byzantine behavior. Then their logical clock values cannot be trusted, and they may affix incorrect logical clock values to messages they send, in order to taint the logical clocks of other processes. If one could ensure that the physical clocks of non-faulty nodes satisfy certain ordering conditions, this could be quite beneficial. A similar intuition underlies our approach towards reliable local broadcast.

5. LOOSE SYNCHRONIZATION AND LOCAL BROADCAST

In this section we describe the basic assumptions and approach behind leveraging the existence of loose synchronization to facilitate a certain ordering condition between locally broadcast messages. In Section 6, we discuss how the ordering condition can be realized in a wireless network, and subsequently describe in Section 7 how it is leveraged to achieve reliable local broadcast with probabilistic guarantees.

Consider a system comprising a node v that is interested in sending messages, and a set of other nodes (neighbors of v) capable of receiving messages from v over a *shared* broadcast medium. Each node is equipped with a single half-duplex transceiver. Thus no node can send and receive messages simultaneously, and only one message can be successfully transmitted or received at a time by a node. Note that this is a reasonable model for wireless nodes equipped with a single half-duplex transceiver and an omnidirectional antenna, and operating on a single common channel.

Receive-Timestamp. A node is assumed capable of noting its local physical clock value just after it finishes receiving a message (this is a reasonable assumption; such a timestamping operation could be implemented in hardware). This is termed as the receive-timestamp observed by the node for the message.

The messages sent in this system have the following property: the minimum (absolute) time the packet transmission occupies the channel is T_l, and the actual total (absolute) time taken by a message in transit (between the time the sending node's physical layer starts sending the message, and the time the receiving node finishes receiving and takes its receive-timestamp) is upper-bounded by T_u. Hence $T_u - T_l$ subsumes the maximum propagation delay and upper bounds on any processing delays incurred upto the time of taking the timestamp.

Thus, the (absolute) time T taken by a message in transit from sender to receiver (between timestampings) satisfies $T_l \leq T \leq T_u$. Note that this condition is satisfied by all messages including those sent by faulty nodes. We explain in Section 6 why this is a reasonable assumption. We define the following condition:

Receipt-Order Condition. If a node v sends a message m_1, followed by a message m_2, then for all non-faulty nodes u, w (in v's neighborhood): the receive-timestamp observed by u for m_2 is greater than the receive-timestamp observed by w for m_1.

We identify two situations in which the Receipt-Order Condition holds. The first one relies on assumptions about

external clock synchronization, and the second one relies on assumptions about internal clock synchronization.

OBSERVATION 1. *(Externally Synchronized Nodes) If the physical clocks of all non-faulty nodes in the system are externally synchronized within bound D, and if $2T_l - T_u > 2D$, then the local physical timestamps observed by the non-faulty neighbors of v for messages sent by v satisfy the Receipt-Order Condition.*

PROOF. Suppose the sender starts sending the two messages m_1, m_2 at times t_1 and t_2 respectively (according to the ideal external clock). Then those non-faulty neighbors of v that received m_1 would have received it within the interval $(t_1 + T_l, t_1 + T_u]$ (as per the external clock), and their observed receive-timestamp would lie in the range $(t_1 + T_l - D, t_1 + T_u + D]$. Similarly, the observed receive-timestamp for the second message m_2 falls within $(t_2 + T_l - D, t_2 + T_u + D]$. Since the two messages are sent by v on the same medium, they are temporally ordered and separated in time i.e. $t_2 \geq t_1 + T_l$. Thus $(t_2 + T_l - D) - (t_1 + T_u + D) = t_2 - t_1 - T_u + T_l - 2D \geq 2T_l - 2D - T_u > 0$. Hence, any non-faulty node that receives the first message observes a receive-timestamp that is less than the receive-timestamp for the second message observed by those non-faulty nodes that see the second message. Hence the Receipt-Order Condition holds. □

OBSERVATION 2. *(Internally Synchronized Nodes) Consider an interval of time in the system in which no non-faulty node adjusts its physical clock, the physical clocks of all non-faulty nodes stay internally synchronized within bound D, and drift-rate is upper-bounded by δ. We are interested in messages sent and received entirely during this interval. If $2T_l - T_u - \delta(2T_l + T_u) > D$, then the local physical timestamps observed by the non-faulty neighbors of v for messages sent by v satisfy the Receipt-Order Condition.*

PROOF. The argument is almost the same as that used in [13] to argue that a system of physical clocks can be made to satisfy the *Strong Clock Condition*, except that we now apply it in the context of a broadcast medium with multiple recipients of the same message.

Denote by $E_v^s(m)$, the event of node v sending message m, and by $C_u(E_v^s(m))$ the local physical clock time at some non-faulty node u, at the time v started the transmission. Note that this does not imply that node u is aware of the instant at which transmission started. u may only detect the transmission after some minimum propagation delay. Denote by $E_u^r(m)$, the event of node u receiving message m, and by $C_u(E_u^r(m))$, the receive-timestamp observed by node u for a message m received by it (recall that receive timestamps are recorded when the reception has finished).

Suppose a node v starts sending a message m_1 at a time when local time at some non-faulty neighbor u is $C_u(E_v^s(m_1))$. Thus, from the assumption that clocks are internally synchronized within bound D, the local time at any other non-faulty neighbor w must be $C_w(E_v^s(m_1)) \leq C_u(E_v^s(m_1)) + D$, and w will observe a receive-timestamp $C_w(E_w^r(m_1)) \leq C_w(E_v^s(m_1)) + T_u(1+\delta) \leq (C_u(E_v^s(m_1)) + D) + T_u(1+\delta)$.

If v later starts sending a message m_2 when local-time at u is $C_u(E_v^s(m_2))$, then $C_u(E_v^s(m_2)) - C_u(E_v^s(m_1)) \geq T_l(1-\delta)$. Thus the receive-timestamp u observes for m_2 is at least $C_u(E_u^r(m_2)) \geq C_u(E_v^s(m_2)) + T_l(1-\delta) \geq C_u(E_v^s(m_1)) + 2T_l(1-\delta)$. Thus, for u and any other non-faulty node w: $C_u(E_u^r(m_2)) \geq C_u(E_v^s(m_1)) + 2T_l(1-\delta) = (C_u(E_v^s(m_1)) + D + T_u(1+\delta)) - T_u(1+\delta) - D + 2T_l(1-\delta) \geq C_w(E_w^r(m_1)) + (2T_l(1-\delta) - T_u(1+\delta) - D) = C_w(E_w^r(m_1)) + (2T_l - T_u - \delta(2T_l + T_u) - D) > C_w(E_w^r(m_1))$.

Thus the Receipt-Order Condition is satisfied. □

6. NETWORK MODEL

Consider a wireless multi-hop network. For the purpose of our discussion, we focus on a *local broadcast domain* within the wireless network, comprising a sender node s and nodes within its transmission-range, denoted by $nbd(s)$ (s is not included in $nbd(s)$), to which we wish to ensure reliable local broadcast delivery. We denote $|nbd(s)|$ by d, and define $d_o = \min_{x \in nbd(s)} nbd(x) \cap nbd(s)$. Thus d_o is the minimum number of common neighbors of s and any of its neighbors.

6.1 Fault Model

We assume the locally bounded fault model considered in [11], [1], [18] etc., wherein an adversary may place faults so long as the number of faults in any single neighborhood does not exceed a specified number b. Faulty nodes can exhibit Byzantine behavior at higher layers, i.e., they may change the values/semantics of messages. However all PHY/MAC layers are non-faulty and thus faulty nodes do not deliberately cause collisions or spoof MAC addresses. This is a reasonable assumption in situations where higher-layer or application code is much more prone to corruption or compromise.

6.2 Communication Model

We allow for an unreliable wireless channel where fading and other effects may lead to non-ideal transmission characteristics. Accidental collisions and interference are possible, due to an imperfect medium access mechanism. If a node transmits a message, the probability that a neighbor successfully receives it is p_s. Packet errors due to fading, or accidental interference etc. are subsumed in the error probability $(1 - p_s)$. The probability of successful reception p_s is assumed independent though identical for each transmission and each receiving node. A desired access probability $0 < p_a < 1$, and an accordingly large enough timeout T_a are chosen, such that if a packet was put into a node's outgoing queue at time t, then with probability at least p_a, by time $t + T_a$, the packet gets a chance to be transmitted by this node and received by neighbors. Both p_s and p_a are assumed independent of d, d_o. Note that T_a is a function of the target access probability p_a, as well as the lengths of packet-queues.

All nodes possess a single half-duplex transceiver with an omnidirectional antenna, and operate on a single channel. They also use a single transmission rate [1], and all valid messages are of a predetermined (and equal) size (as discussed later, this can be chosen to facilitate reliable local broadcast). Note that the use of a common transmission rate r bits/sec and a common message size l bits ensures that all messages occupy a certain minimum time $T_l \geq \frac{l}{r}$ on the

[1]Even in a multi-rate wireless network, it is possible to stipulate as part of the protocol specification that all nodes use a specific rate (say the lowest) for critical message types that require reliable dissemination.

channel. This extends to messages sent by faulty nodes, because non-faulty nodes can choose to ignore messages that do not conform to the rate/size specification, giving faulty nodes no incentive to deviate from this established behavior.

The maximum and minimum propagation delays are ${d_{prop}}^{max}$ and ${d_{prop}}^{min}$ respectively (note that ${d_{prop}}^{min} > 0$). Any additional delays in physical-layer timestamping are upper-bounded by t_{delay}, yielding a maximum delay bound of $T_d = {d_{prop}}^{max} + t_{delay}$. Thus $T_u = T_l + T_d$.

For the rest of our discussion, we assume that nodes are externally synchronized within bound D, so that we may leverage Observation 1.

We seek to ensure that the conditions of Observation 1 from Section 5 are satisfied. Thus we want $2T_l - T_u = T_l - T_d > D$, or $T_l > D + T_d$. Since T_d is independent of T_l, this is always achievable (albeit at the expense of inefficient bandwidth usage) by padding all messages with extra bits to achieve the desired packetsize l (and hence T_l) for the specified transmission rate r. Thus the Receipt-Order Condition can be made to hold.

We now provide a brief description of message representation.

In order to distinguish between different messages, *distinct* messages sent by a particular source (originator) are distinguished via *identifiers*, that we shall denote as *id*. The *id* is a number in some range $[0, MAX]$, where MAX is a suitably large number. Individual nodes choose the sequence of *id*s for their messages in some *privately determined* pseudo-random manner (such that *id*s are re-used only after large intervals of time; thus identifiers may be considered unique for all practical purposes). This ensures that nodes have no easy way of anticipating what the sequence of *id*'s for a given source node will be.

If a node sends two conflicting versions of the same message, it implies that they both have the same *id*, but different values. Original messages are represented as $m(src, (id, value))$. Of these, the *src* field is obtained from the MAC header, and thus contains the true MAC address of the node that put the packet on air. The $(id, value)$ part is message-content. If a message m is relayed (repeated) by a neighbor, it is represented as $REPEAT(relay\_src, (m, timestamp))$. Once again, *relay_src* is the MAC address of the relay node, obtained from the MAC header. The $(m, timestamp)$ part is message-content (m denotes the $(src, (id, value))$ information for the message; however as this is now part of message *content*, a faulty relay node can modify the *src* information if it so chooses, though it cannot affect the correctness of the *relay_src* field in the MAC header).

7. THE ALGORITHM

The goal of the algorithm is to achieve the following agreement condition with high probability (w.h.p.):

Agreement Condition. If a local broadcast source s sends a message, then all its non-faulty neighbors should agree on a single value for this message. If s is non-faulty, this agreed-upon value should be the one actually sent by s. If s is faulty and sends multiple conflicting versions of the message, the protocol is designed to enable nodes to choose the *first* value that s sent.

For the sake of simplicity and without loss of generality (w.l.o.g.), we assume that the message m may take one of two values 0 or 1. The algorithm can however be easily generalized to more than two message values.

Suppose we have sender s. Each other node u follows the following algorithm:

- On receipt of a message $m(s, (i, p))$ from s directly with (local) receive-timestamp t:
  If no other earlier version of this message (i.e., of the form $m(s, (i, q))$) was received *directly* from s, make note of p as a candidate message value, and re-broadcast a copy of m as $REPEAT(u, (m(s, i, p), t))$. If an earlier version of the same message was received *directly* from s, discard this message.

- On receipt of a message $REPEAT(v, (m(s, i, p), t_v))$:
  If no previous $REPEAT(v, m(s, i, *), *)$ [2] has been received, make note of p as a candidate for message-id i from s, reported by v with timestamp t_v. Keep track of all such copies of m received via $REPEAT$ messages from different repeaters along with their reported timestamps.

  If this was the first message having the form $REPEAT(*, m(s, i, *), *)$ received by the node, start a timer (tagged by (s, i)) to expire after a duration $T+T_u$ (where $T = T_a + T_r$, T_a being the pre-defined access timeout, and T_r being an estimated upper bound on processing time from receiving a message m to time of generating a $REPEAT$ and enqueueing it in the outgoing packet queue).

- On expiration of the timer for (s, i):
  Perform a filtration procedure on the received $REPEAT$ messages containing repeated messages of the form $m(s, (i, *))$, and determine the value of m for which the highest number of *repeated* copies were received. Commit to this message value.

  Timestamp-based filtration and majority determination: The filtration and majority determination involves application of the following procedure: Let us refer to the value with highest repeated copy count as c_1, and the other one as c_2. If the number of copies of c_2 is less than or equal to b, choose c_1 as the correct value. If the number of copies of c_2 is greater than b: discard any messages with value c_1 whose timestamp t is greater than the timestamps of more than b copies of c_2. Commit to the majority value from amongst the remaining copies of c_1 and c_2.

THEOREM 1. *Consider a* local broadcast domain *in the wireless network comprising* $\{s\} \cup nbd(s)$ *for some node* s. *Assume that the physical clocks of all non-faulty nodes satisfy the Receipt-Order Condition. If at most* $b = \left(\frac{\alpha}{1+\alpha}\right) d_o$ *nodes in any single neighborhood are faulty (where* $\alpha \leq p_a p_s^2 - \epsilon$, *and* $\epsilon > 0$ *is a constant), then the above algorithm ensures that all non-faulty neighbors of* v *shall be able to achieve the previously described agreement condition for* v*'s message with error probability at most* $d \exp(-\frac{(1-\frac{\alpha}{p_a p_s^2})^2 p_a p_s^2 d_o}{2(1+\alpha)})$, *which is small if* d_o *is large, and* $d_o >> \ln d$.

PROOF. There are two cases: s is non-faulty or s is faulty:

[2] $*$ is a placeholder for any value.

- *s is non-faulty:* s transmits exactly one version of the message (call it $m_1 = m(s,(i,q_{m_1}))$). Since any $u \in nbd(s)$ has at most b faulty nodes in $nbd(u)$, it may receive up to a maximum of b spurious repeats of s's message. If the number of $REPEAT$ copies of the message received from non-faulty nodes (and thus containing the correct value) is greater than b, this suffices to distinguish the legitimate value from a spurious one.

- *s is faulty:* If s is faulty, it may leverage the unreliability of the channel, and attempt to create confusion by sending more than one version of the message, each containing different values. We show that despite this, under the assumed conditions, reliable broadcast will still be achieved.

By assumption, the physical clocks of all non-faulty nodes satisfy the Receipt-Order Condition. Then, in the algorithm described earlier, copies of the second message received from non-faulty neighbors get filtered out as follows: Suppose the sender s sends the two message-versions $m_1 = m(s,(i,q_{m_1}))$ and $m_2 = m(s,(i,q_{m_2}))$ at absolute times t_1 and t_2 respectively.

Hence, any non-faulty node that receives the first message observes a receive-timestamp that is less than the receive-timestamp for the second message observed by those non-faulty nodes that see the second message. All non-faulty nodes attach the correct observed timestamp to any $REPEAT$ messages they send, and non-faulty nodes that receive the $REPEAT$ messages record the timestamp along with the message encapsulated in the $REPEAT$.

Recall that the first message-version sent out by s is m_1 and the second is m_2. Also, the message-version with highest pre-filtration count is referred to as c_1 and the other one is referred to as c_2.

We show that if more than b $REPEAT$ copies of m_1 were received from non-faulty nodes, the agreement condition is achieved. Thereafter we show that more than b copies of m_1 are received from non-faulty nodes w.h.p.

Suppose more than b copies of m_1 were received from non-faulty nodes, i.e., more than b *correct* copies of m_1 were received.

Then the following cases may arise:

- If $c_1 = m_1$, and at most b copies of m_2 were received: m_1 will win the majority vote, and get chosen immediately.

- If $c_1 = m_1$, i.e., m_1 has the highest pre-filtration count, and greater than b copies of m_2 were received:
  A non-faulty node will only send a $REPEAT$ of m_2 if it receives the message m_2 directly from s, and it will affix a correct receive-timestamp to its $REPEAT$. Since the Receipt-Order Condition holds, the timestamp reported in any such $REPEAT$ copy of m_2 will be greater than the timestamp reported in any of the correct $REPEAT$ copies of m_1. Thus, no more than b copies of $c_2 = m_2$ can bear a false earlier timestamp. Resultantly, no copy of m_1 sent by a non-faulty node will get filtered out erroneously, and m_1 will win the majority vote.

- If $c_1 = m_2$ i.e. m_2 has the highest pre-filtration count: Since greater than b copies of m_1 were received from non-faulty nodes, then from the Receipt-Order Condition, any copy ($REPEAT$) of m_2 sent by a non-faulty node has a reported timestamp greater than the reported timestamps on the greater-than-b correct copies of m_1, and the timestamp filtration rule ensures that all copies of m_2 sent by non-faulty nodes get filtered out. This leaves only upto b copies of m_2 sent by faulty nodes. Thus, if the correct $REPEAT$ copies of m_1 are greater than b, m_1 will win the majority vote.

Hence, the algorithm definitely makes the correct decision if more than b copies of m_1 were received from non-faulty nodes. This is the same as the sufficient condition we earlier stated for correct decision with a non-faulty source.

When b or fewer copies of m_1 are received from non-faulty nodes, the decision may be correct or wrong, depending on how many copies of m_2 were received. To bound the error probability, we assume the worst, i.e., it is always wrong if b or fewer copies of m_1 are received from non-faulty nodes.

We represent the copies of m_1 repeated by non-faulty nodes that were received by a node u as a random variable Z. Then, the requirement is that $Z > b$ for both the cases (recall that in the first case, the source is non-faulty, and so it sends only one message-version m_1, but upto b spurious $REPEAT$ messages containing wrong values may still be received from faulty nodes).

Let the number of non-faulty mutual neighbors of s and u be g. Then $g \geq d_o - b$. Z is the sum of g i.i.d. $Bernoulli(p_a p_s^2)$ random variables, since a repeated copy of m_1 is received from a non-faulty neighbor if that neighbor received m_1 directly from s (probability p_s), it was able to transmit the $REPEAT$ packet before timeout (probability p_a), and the $REPEAT$ was successfully received by u (probability p_s). This allows us to apply the following special form of the Chernoff bound [16]:

$$Pr[Z \leq (1-\beta)E[Z]] \leq \exp(\frac{-\beta^2 E[Z]}{2}), 0 < \beta < 1 \quad (1)$$

Thus, knowing that $b = \frac{\alpha}{1+\alpha} d_o \leq \alpha g$, we can set $\beta = 1 - \frac{\alpha}{p_a p_s^2}$ to obtain $b \leq (1-\beta)E[Z]$. Thus application of the Chernoff bound yields:

$$\begin{aligned} Pr[Z > b] &\geq 1 - Pr[Z \leq (1-\beta)E[Z]] \\ &\geq 1 - \exp(-\frac{(1-\frac{\alpha}{p_a p_s^2})^2 p_a p_s^2 g}{2}) \\ &\geq 1 - \exp(-\frac{(1-\frac{\alpha}{p_a p_s^2})^2 p_a p_s^2 d_o}{2(1+\alpha)}) \quad (2) \end{aligned}$$

Since $0 < \beta < 1$, the constraint on α is that $\alpha \leq p_a p_s^2 - \epsilon$ with $\epsilon > 0$. Thus α (which gives a measure of the proportion of tolerable faults) can be large when the probability of successful receipt ($p_a p_s^2$) is large, and can only be small when $p_a p_s^2$ is small. Applying the union bound over all d neighbors of sender s, probability that any node makes an error is less than $d \exp(-\frac{(1-\frac{\alpha}{p_a p_s^2})^2 p_a p_s^2 d_o}{2(1+\alpha)})$, which is small for large d_o, and $d_o >> \ln d$. □

Note that, as d increases, the timeout component T_a must also increase to maintain a sufficiently high value of p_a (due to increased contention for the shared channel). However, in

most cases of practical interest, d will not be unduly large, and a moderate value for T can suffice. Besides, the protocol is still reasonably scalable, as it only requires one message to be sent by each node.

In our analysis, we have assumed that whenever the number of copies of m_1 received from non-faulty nodes is less than b, a wrong decision is made. In actuality, if the number of copies of m_1 received from non-faulty nodes is less than b, there may still be situations where a correct decision may be made (it is possible that the total number of received copies containing q_{m_2} (from faulty or non-faulty nodes) be much less than b, since these transmissions are also subject to reception errors). Thus the presented analysis establishes a rather conservative upper bound on the error probability.

8. POSSIBLE OPTIMIZATIONS

From a practical perspective, one can consider many possible enhancements/optimizations to the basic algorithm.

1. Each node can be made to retransmit its REPEAT messages k times. This can help improve loss-resilience, without causing duplication problems, as (in absence of address spoofing) two receipts of the same message are easily identified by the repeater's address.

2. One could consider triggering the reliable local broadcast algorithm only if at least one warning message is heard from a node claiming to have heard two inconsistent messages sent by s (this would work only if it is very likely that a fair number of nodes will receive both variants of s's message). Also, while faulty nodes can raise false alarms, that is no worse that proactively using the algorithm each time.

9. DISCUSSION ON SYNCHRONIZATION REQUIREMENTS

The synchronization assumptions required to ensure the Receipt-Order Condition holds may actually be practically feasible in many settings.

It is possible that in the near-future, wireless nodes may be equipped with on-chip atomic clocks [10] with very low drift. Thus, if the clocks are synchronized with an external time source at time of deployment, then one might bound the total skew over the entire operational lifetime of the network, and this would not be overly large. Alternatively, nodes might be GPS-equipped, thus providing an out-of-band means of external synchronization. In such scenarios, the conditions for Observation 1 can be made to hold.

In the absence of on-chip atomic clocks or GPS-equipped devices, it may not be possible to ensure that all nodes in the network be synchronized to an external clock within some constant bound D. However, it is still quite feasible to ensure that each node is internally synchronized within constant bound D with its two-hop neighbors. One could envisage a situation where nodes are initially synchronized at time of deployment, and thereafter periodically run a re-synchronization protocol, to ensure that any any two nodes within two-hops of each other always stay internally synchronized within the bound D. A lightweight Byzantine time synchronization protocol should potentially suffice for this. In the period between two consecutive resynchronizations, the conditions of Observation 2 can thus be made to hold for every local broadcast domain in the network.

10. USING THE PRIMITIVE FOR MULTI-HOP BROADCAST

We briefly discuss how the proposed primitive could potentially be used as a building block in a protocol to achieve broadcast in a multi-hop setting. It was observed in [12] that the algorithm of [1] requires neighbors of the original sender to agree on the value it sent, even if the original sender is faulty; for other nodes in the network, correctness only requires that neighbors of non-faulty nodes agree on the messages they sent. Thus, if one is using a global broadcast protocol with similar properties, one could consider using the reliable local broadcast primitive in the neighborhood of the original sender, and merely stipulate that other nodes retransmit their messages a sufficient number of times.

Otherwise, if the protocol requires that neighbors of all nodes agree on what they sent, one could proceed as follows: Let us consider a multi-hop network of n nodes, where the minimum node degree is d_{min}, maximum node degree is d_{max}, and $d_o = \min_x \min_{y \in nbd(x)} |nbd(x) \cap nbd(y)|$. Thus d_o is the minimum number of common neighbors shared by any two neighbors. The number of faulty nodes in any single neighborhood is at most $b = \frac{\alpha}{1+\alpha} d_o$ where $\alpha \leq p_a p_s^2 - \epsilon (\epsilon > 0)$. Through exchange of periodic hello messages, nodes maintain a list of neighbors. Neighbors are added/removed only if more than a certain number of HELLO messages have been consecutively received/lost. This helps maintain a degree of stability in the neighborhood information, in the face of short-term signal fluctuations. Suppose we have a global multi-hop broadcast protocol that assumes reliable local broadcast, and requires a total of $O(n^m)$ messages to be sent (m is a constant), i.e. has message complexity polynomial in n. Then, for each step of the protocol that requires a node to perform a local broadcast, the reliable local broadcast primitive protocol is run in the *local broadcast domain* comprising the node and its neighbors. Following the proof argument of Theorem 1, we can obtain that the probability local broadcast is achieved reliably is at least $1 - d_{max} \exp(-\frac{(1-\frac{\alpha}{p_a p_s^2})^2 p_a p_s^2 d_o}{2(1+\alpha)}) = 1 - \exp(-\frac{(1-\frac{\alpha}{p_a p_s^2})^2 p_a p_s^2 d_o}{2(1+\alpha)} + \ln d_{max})$. Since n^m such successful local broadcasts are needed, if $d_o = c_1 m \log n$ for a suitably chosen constant $c_1 > \frac{2(1+\alpha)}{(1-\frac{\alpha}{p_a p_s^2})^2 p_a p_s^2}$, and $d_{max} \leq c_2 \log n$ for another suitable constant c_2 (note that $c_2 \geq c_1 m$ by definition), then by applying the union bound, one may see that the global broadcast will also succeed with probability at least $1 - n^m \exp(-\frac{(1-\frac{\alpha}{p_a p_s^2})^2 p_a p_s^2 d_o}{2(1+\alpha)} + \ln d_{max})$, which approaches 1 for large n.

The tolerable number of per-neighborhood faults is given by the minimum of the tolerance threshold for the global protocol, and the local broadcast primitive.

11. OPEN ISSUES

The algorithm we have outlined in this paper is primarily a proof-of-concept approach, whereby we seek to highlight that one can leverage the shared nature of the medium, and information from lower-layers (in this case, timestamps), to design scalable probabilistic solutions to the local broadcast problem. However, there are still numerous outstanding issues that need to be addressed.

One issue is that of using a suitable Byzantine time syn-

chronization protocol to ensure internal synchronization between neighboring nodes (see Section 9). It might be possible to leverage existing work in this area, e.g., [20]. Another issue is that one might wish to eliminate the requirement in Observation 2 that during the interval in which the local broadcast is occurring, nodes do not adjust their clocks. This would require a synchronization algorithm that can run simultaneously with the local broadcast algorithm without affecting the Receipt-Order Condition. Additionally, the described algorithm assumes i.i.d. loss probabilities. If channel losses exhibit spatial correlation, the algorithm may need to be modified to handle such situations.

A major shortcoming of the algorithm is the need to estimate the timeout T based on access probability p_a, average length of outgoing packet-queues, and processing time to generate a $REPEAT$. It would be preferable to have an algorithm where nodes decide to invoke the filtration and majority determination procedure based on some event, e.g., receipt of certain messages.

12. DISCUSSION

Many of the assumptions in this paper are justified by assuming a network with a single channel and omnidirectional antennas. One might wish to consider alternative scenarios where multiple channels or beamforming antennas are available. We remark that usage of multiple channels or directional antennas tends to alter the broadcast nature of the wireless medium, and makes the network look increasingly like a point-to-point network. Thus, algorithms based on the point-to-point abstraction may increasingly seem suitable in such scenarios.

13. CONCLUSION

In this paper, we have considered the issue of implementing reliable local broadcast in a wireless network with a lossy channel. We have proposed a simple proof-of-concept approach towards this end, with the intent to highlight the potential for obtaining lightweight probabilistic solutions to the problem.

14. REFERENCES

[1] V. Bhandari and N. H. Vaidya. On reliable broadcast in a radio network. In *PODC '05: Proceedings of the twenty-fourth annual ACM SIGACT-SIGOPS symposium on Principles of distributed computing*, pages 138–147. ACM Press, 2005.

[2] V. Bhandari and N. H. Vaidya. On reliable broadcast in a radio network: A simplified characterization. Technical Report, CSL, UIUC, May 2005.

[3] V. Bhandari and N. H. Vaidya. Reliable Broadcast in Wireless Networks with Probabilistic Failures. In *Proceedings of IEEE INFOCOM*, pages 715–723, Anchorage, Alaska, May 2007.

[4] V. Bhandari and N. H. Vaidya. Reliable broadcast in wireless networks with probabilistic failures. Technical Report, CSL, UIUC, Jan. 2007.

[5] K. P. Birman, M. Hayden, O. Oskasap, Z. Xiao, M. Budiu, and Y. Minsky. Bimodal multicast. *Transactions on Computer Systems (TOCS)*, 17(2):41–88, May 1999.

[6] G. Chockler, M. Demirbas, S. Gilbert, N. Lynch, C. Newport, and T. Nolte. Reconciling the theory and practice of (un)reliable wireless broadcast. In *ICDCSW '05: Proceedings of the Fourth International Workshop on Assurance in Distributed Systems and Networks (ADSN) (ICDCSW'05)*, pages 42–48. IEEE Computer Society, 2005.

[7] G. Chockler, M. Demirbas, S. Gilbert, C. Newport, and T. Nolte. Consensus and collision detectors in wireless ad hoc networks. In *PODC '05: Proceedings of the twenty-fourth annual ACM symposium on Principles of distributed computing*, pages 197–206. ACM Press, 2005.

[8] J. Considine, L. A. Levin, and D. Metcalf. Byzantine agreement with faulty majority using bounded broadcast. *CoRR*, cs.DC/0012024, 2000.

[9] S. Gilbert, R. Guerraoui, and C. Newport. Of malicious motes and suspicious sensors. In *Proc. of OPODIS*, 2006.

[10] S. Knappe, L. Liew, V. Shah, P. Schwindt, J. Moreland, L. Hollberg, and J. Kitching. A microfabricated atomic clock. *Appl. Phys. Lett.*, 85, 2004.

[11] C.-Y. Koo. Broadcast in radio networks tolerating byzantine adversarial behavior. In *PODC '04: Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing*, pages 275–282. ACM Press, 2004.

[12] C.-Y. Koo, V. Bhandari, J. Katz, and N. H. Vaidya. Reliable broadcast in radio networks: The bounded collision case. In *Proceedings of ACM PODC 2006*, 2006.

[13] L. Lamport. Time, clocks, and the ordering of events in a distributed system. *Commun. ACM*, 21(7):558–565, 1978.

[14] L. Lamport, R. Shostak, and M. Pease. The byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, 1982.

[15] J. Luo, P. Eugster, and J.-P. Hubaux. Route driven gossip: Probabilistic reliable multicast in ad hoc networks. In *Proc. of INFOCOM 2003*, 2003.

[16] M. Mitzenmacher and E. Upfal. *Probability and computing*. Cambridge University Press, 2005.

[17] M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *J. ACM*, 27(2):228–234, 1980.

[18] A. Pelc and D. Peleg. Broadcasting with locally bounded byzantine faults. *Information Processing Letters*, 93(3):109–115, Feb 2005.

[19] A. Pelc and D. Peleg. Feasibility and complexity of broadcasting with random transmission failures. In *PODC '05: Proceedings of the twenty-fourth annual ACM SIGACT-SIGOPS symposium on Principles of distributed computing*, pages 334–341, 2005.

[20] T. K. Srikanth and S. Toueg. Optimal clock synchronization. *J. ACM*, 34(3):626–645, 1987.