Towards Reliable Application Data Broadcast in Wireless Ad Hoc Networks

Bo Xing, Mayur Deshpande, Nalini Venkatasubramanian and Sharad Mehrotra
Donald Bren School of Information and Computer Sciences, University of California, Irvine
{bxing, mayur, nalini, sharad}@ics.uci.edu

Abstract— **Application data broadcast in ad hoc networks is an important primitive that has received little systematic research – the main focus of prior research being on control data broadcast. In this paper, we show why control data broadcast and even multicast techniques are insufficient for reliable application data broadcast; in fact their reliability degrades sharply with increasing application data size. We discover the root cause of this to be IP fragmenting the application data but not providing good reliability control on the fragments. We hence propose READ (Reliable and Efficient Application-data Dissemination), a protocol based on higher-layer fragmenatation with fragment-level reliability control. READ splits a data packet into fragments, and disseminates them separately at dynamically adaptive intervals. Receivers piggyback implicit NACKs when propagating the fragments, and retrieve missing fragments from neighbors. Through experiments, we show that READ consistently achieves high delivery ratio and short latency, outperforming all other examined protocols.**

I. Introduction

Network Wide Broadcast (NWB) in ad hoc networks is an important primitive for many applications – especially in crisis and military scenarios. Consider, for example, a group of first responders and incident commander at a rescue cite. They need to have a high level of situational awareness (accurate and up-to-date), which can help increase their safety and improve emergency management. They hence share situational information (e.g., locations, hazards, instructions, maps, etc.) among them [1]. Without the need for a network infrastructure, they use their handheld devices (e.g., PDAs) to form a wireless multi-hop network. For higher security and lower interference, the network is built up via a 802.11 variant protocol at the licensed 4.9GHz public safety frequency band [2]. Thus, the situational information acquired by one device is disseminated to other devices through the network, in the form of text, voice, image or video, etc.. In such *application data broadcast* scenarios, the goal is to deliver application-generated data from one node to maximum number of other nodes, as fast as possible.

Prior research on ad hoc NWB has mainly focused on *control data broadcast*, a primary use being route establishment in reactive ad hoc routing protocols. For example, in AODV [3], a source node broadcasts route-request messages to all other nodes and then waits for route-reply messages from the destination. However, control data broadcast differs from application data broadcast in several aspects: (1) Control data is usually small (e.g., tens of bytes), as compared to application data which can have a potentially large size (e.g., tens or hundreds of kilobytes). (2) In broadcasting routing-related data, the main goal is to reach specific destinations (whose locations are unknown), rather than to reach maximum number of nodes. (3) Control data is overhead instead of useful data; hence control data broadcast protocols try to generate least number of transmissions. Application data broadcast, on the other hand, might need to send redundant transmissions to increase reliability. (4) Control data broadcast is a frequent operation; hence, many protocols maintain an overlay to improve efficiency. In contrast, application data broadcast is relatively sporadic, thus overlay maintenance may be costly and unsuitable.

Multicast is another candidate that could be used for application data broadcast. However, multicast as a primitive is a mismatch for application data broadcast. Multicast protocols usually build an overlay (mesh/tree) for a specific multicast group. To be used for broadcast, all nodes will have to participate in constructing and maintaining the overlay. This is very expensive and unnecessary. Further, irrespective of the particular mesh-/tree-based multicast protocol, the constructed overlay is basically the same, i.e., the set of nodes that are not at the network perimeter. Hence, in essence, multicast protocols reduce to message-efficient broadcast protocols when used for broadcast, but with the added overhead for overlay maintenance and membership management. Additionally, we will show in a later section that multicast protocols also have significant reliability problems with large-size data.

This paper studies application data broadcast as a fundamental new problem. Our goal is to design protocols that deliver application data (e.g., a 32KB JPEG image) to all receivers with high reliability guarantee and in a timely manner. To establish application data broadcast as a new problem, we first thoroughly investigate the capability of existing broadcast/multicast protocols for delivering large-size data – we reexamine blind flooding and message-efficient approaches and find their performance degrades quickly with increasing data size. We apply various reliability-improving techniques at different layers, but these do not solve the problem either. By tracing experiments, we identify the major causes for the rapid decline in reliability: IP fragmentation and fragment drops in IP queues. To counter these, we then evaluate the effect of fragmenting large-size data before it is passed to IP. However, fragmentation has many inherent tradeoffs and we explore these tradeoffs in detail.

Finally, we propose a fragmentation-based protocol, READ (Reliable and Efficient Application-data Dissemination), which delivers application data with high reliability guarantee and short latency. READ splits a data packet into multiple fragments with "optimal" size, and disseminates them separately at intervals that dynamically adapt to the network traffic load. It employs implicit NACKs for receivers' local recovery of missing fragments. A node, when propagating a certain fragment, piggybacks which fragments it is currently missing, so as to request neighbors to retransmit the missing fragments. Additionally, READ applies stricter reliability control on source node transmissions. This prevents the pathological case of data propagation stopping at the first hop. Through experiments, we show that READ outperforms all other examined protocols.

In the next section, we describe background knowledge. Section III identifies the large-data reliability problem with existing techniques. We then explore fragmentation and its tradeoffs in Section IV. We propose the READ protocol in Section V. Finally, we conclude the paper in Section VI.

II. Preliminaries

A. Related Work

Since the identification of the broadcast storm problem [4], a large body of research work has been dedicated to reducing redundant transmissions in NWB [5][6] [7] [8] [9] [10] [11] [12] [13]. Aside from the efficiency issue, the reliability of NWB has drawn considerable attention from the research community. Reliable broadcast protocols fall into two categories - protocols that offer probabilistic delivery guarantees [14] [15] [16] [17] and protocols that provide deterministic reliability guarantees [18] [19]. However, all of the above protocols were designed with small-size data in mind, such as control data in ad hoc routing protocols. In contrast, our aim is at designing protocols that deal with large-size application data broadcast.

B. Problem Definition

In this paper, we consider the following broadcast problem. Given a wireless ad hoc network, in which a source node initiates the broadcast of an application data packet (denoted as d), the following criteria are optimized for:

(i) *Reliability*: Maximum number of nodes receive d.

(ii) *Timeliness*: Minimum time is taken for d to be delivered to all receiving nodes.

By definition, the protocol delivers application data in a best-effort manner (or, with probabilistic reliability guarantee), and within short time. Between the two primary objectives, reliability is of utmost importance, because timeliness is meaningless if only a small portion of receivers can be covered.

C. Experimental Study Methodology

On our road to a solution, we experiment with various NWB techniques including our proposed protocol. We mainly measure their reliability in disseminating data of varying sizes. Reliability is captured by *delivery ratio*, defined as the percentage of nodes that receive the entire application data. When comparing the protocols' timeliness properties, instead of using *end-to-end latency* (i.e., the time from when the source starts dissemination until all covered nodes receive the whole data) which is the absolute dissemination time, we use *normalized latency* (i.e., the ratio of end-to-end latency to delivery ratio) so as to take into account the protocols' differing reliability levels.

In our experiments, we use QualNet v3.9 [20] as the simulation framework. Nodes (with transmission ranges uniformly tuned to 300 meters) are randomly placed in a $1000m \times 1000m$ area. By default, 50 nodes are simulated (average neighbor degree: 13.2) with no mobility. All nodes employ IEEE802.11b MAC protocol on top of two-ray propagation path-loss model. The data rate is 2Mbps. UDP is used as the transport protocol. In the experiments involving mobility, nodes move following the random waypoint model, with the pause time set to zero and the minimum speed kept constant at 1m/s. Each simulation run has one node serve as the broadcast source, and runs sufficiently long for the broadcast traffic to completely vanish. Every result reported in this paper is averaged over simulation runs with *all* nodes serving as sources in 10 different topologies.

III. The Large-Size Data Reliability Problem

A number of broadcast techniques have been proposed in the literature. However, they primarily target control data broadcast. Consequently, their performance was evaluated using small data sizes. For instance, the broadcast storm problem paper [4] experimented with 280B packets; many other publications [13] [17] [21] [18] used 64B data. To the best of our knowledge, the largest data size that has been examined in NWB performance evaluations is 1KB [22]. Hence, it is unclear how well existing NWB protocols work in disseminating large-size data. In this section, we explore this aspect thoroughly with experiments.

A. Blind Flooding

Blind Flooding (BF) is the simplest NWB protocol – every node rebroadcasts every new message it receives. It is the de facto control data dissemination scheme for most ad hoc routing protocols. Further, prior research has shown that blind flooding provides high delivery guarantee, outperforming many other broadcast/multicast protocols [23] [24] [13]. Thus, we start with blind flooding as a baseline for comparing the performance of other protocols. Blind flooding, however, has certain drawbacks. The main drawback, referred to as the broadcast storm problem [4], is due to its inherent high redundancy. This leads to severe contentions and collisions, especially in dense networks.

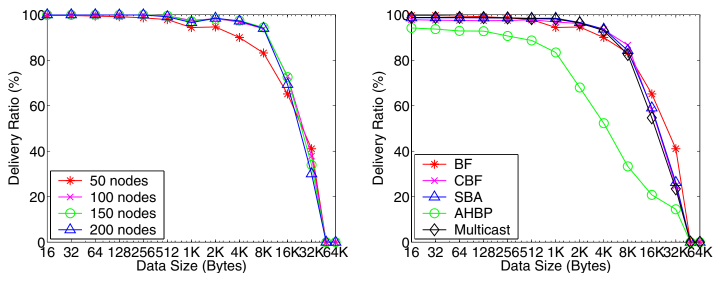
Our first experiment hence studies how the benefits and drawbacks of blind flooding play out with varying data size. The results are shown in Fig. 1(a). With small-size data, the delivery ratio of blind flooding is consistently high, regardless of the network density. However, with increasing data size, the coverage of blind flooding starts to drop – due to severer contentions and collisions. Finally, with data size over 50KB, blind flooding fails totally with no receiver receiving any data.

B. Message-Efficient Protocols

It can be conjectured that blind flooding's poor performance with large data is due to the broadcast storm problem; if contentions and collisions were to be reduced, data might have higher chances to get through. To test this conjecture, we examine message-efficient NWB protocols, which reduce redundant transmissions and hence mitigate the broadcast storm problem. In such protocols, only a portion of the receivers, i.e., *forward nodes*, rebroadcast the data. We pick several representative redundancy-reducing heuristics for our experimental study.

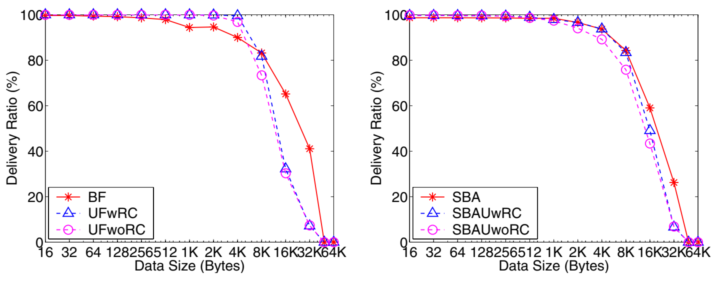
CBF (Counter-Based Flooding) [4] is a probabilistic approach – a node rebroadcasts only if fewer than n (3 in our implementation) other nodes' rebroadcasts are overheard within a predefined period. SBA (Scalable Broadcast Algorithm) [6] adopts the self-pruning technique. Nodes decide whether to rebroadcast according to the traveling route of the data. If a node, A, first receives a message from a node, B, whose transmission is able to cover all of A's neighbors, A refrains from rebroadcasting. AHBP (Ad Hoc Broadcast Protocol) [10] represents the neighbor-designating approach. A forward node explicitly specifies in its rebroadcast which of the 1-hop neighbors should rebroadcast, so that all 2-hop neighbors are covered.

In addition, we examine multicast protocols since they work similar to message-efficient broadcast protocols. When multicast is applied to broadcast, the nodes that are not at the network perimeter are forward nodes. In our experiments, the abstract multicast protocol (denoted as Multicast) preconstructs a shared spanning tree. Only non-spanning-tree-leaves rebroadcast upon receiving the data for the first time.



(a) Blind Flooding (b) Message-Efficient Protocols

Fig. 1. Flooding & Message-Efficient Protocols: Delivery Ratio Vs Data Size



(a) BF Based on MAC-Unicast (b) SBA Based on MAC-Unicast

Fig. 2. MAC-Unicast as NWB Primitive: Delivery Ratio Vs Data Size

Our second experiment studies how message-efficient protocols fare with increasing data size. As shown in Fig. 1(b), all the message-efficient protocols behave similarly to blind flooding. Though they mitigate the broadcast storm problem to a certain extent, their performance still degrades with increasing data size. The problem, therefore, seems deeper than just the broadcast storm problem. What is needed is a thorough reexamination of how broadcast operates at different layers of the network stack. We start with reexamining the MAC layer.

C. MAC Layer Rethink: MAC-Unicast as Primitive?

Traditionally, MAC-broadcast is the common primitive used in NWB protocols due to its one-to-all communication capability. However, MAC-unicast also offers certain enticing properties. With ACK/retransmission and RTS/CTS handshakes (optional), MAC-unicast provides enhanced reliability for single-hop communications. It can be conjectured that this lower-layer reliability may translate to increased reliability at upper layers, leading to higher NWB delivery ratio [25].

In order to examine the effect of using MAC-unicast as an NWB primitive, we designed MAC-unicast-based versions of blind flooding and SBA. Specifically, UF (Unicast Flooding) is the MAC-unicast-based version of blind flooding (UFwRC/UFwoRC denotes UF with/without RTS/CTS), whereas SBAU (SBA Unicast) is the MAC-unicast-based version of SBA (SBAUwRC/SBAUwoRC denotes SBAU with/without RTS/CTS). In all these protocols, routing is disabled and a broadcast message intended to multiple neighbors is sent individually to each neighbor using MAC-unicast.

In our third experiment, we test how MAC-unicast-based protocols work with varying data size. As Fig. 2 shows, MAC-unicast increases reliability slightly, and that too, only for small-size data. When disseminating large-size data, MAC-unicast-based protocols perform even worse than their MAC-broadcast-based counterparts. The reason for this is the increased amount of traffic that is put into the network as compared to using MAC-broadcast. The additional traffic is due to the RTS/CTS/ACK messages generated for each one-to-one communication between two neighbors. With increasing data size, this extra traffic congests an already congested network and reduces reliability.

MAC-layer primitives by themselves, therefore, do not address the reliability problem with large data. Upper layers in the stack have to be exploited. Next, we explore two heuristics that have been previously proposed to enhance reliability: (1) increasing redundancy and (2) using acknowledgements.

D. Higher Layer Optimizations (1): Increasing Redundancy

An intuitive idea to enhance reliability is to increase redundancy, with nodes rebroadcasting more than once. However, increasing redundancy blindly can be counter-productive due to increased network traffic. One idea that tries to balance this tradeoff is SR (Selective Rebroadcast) [15], in which a packet is rebroadcast an additional time if it is suspected to be lost. After rebroadcasting, if a node does not hear n other nodes' rebroadcasts within a certain period, it will rebroadcast one more time. Thus, the amount of increased redundancy naturally adapts to the interference level and density of the network.

In our fourth experiment, we test the performance of BFwSR (Blind Flooding with Selective Rebroadcast) with n equal to 3, 6 and 9. As Fig. 3(a) shows, the increased redundancy enhances reliability and the improvements are noticeable. However, the delivery ratio for large-size data (e.g., 32KB) does not improve over blind flooding, regardless of the counter threshold (n). This implies that with large data, simply increasing redundancy does not significantly increase the chance of packets being received.

E. Higher Layer Optimizations (2): Acknowledgements

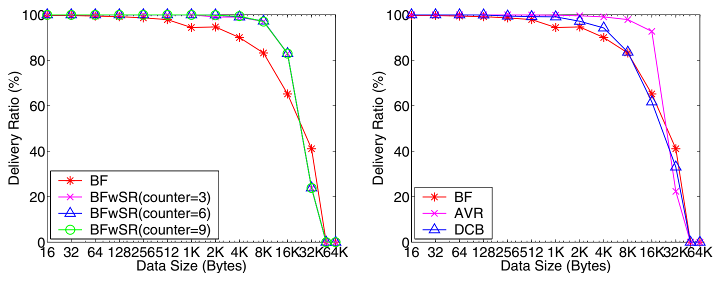
The next heuristic we test is the use of acknowledgements, i.e., a transmitting node waits to hear back from the receiver that it got the message (ACK), otherwise it retransmits the message (in this paper, 'retransmit' means a node MAC-broadcasting a message an additional time). We examine two reliable broadcast protocols based on ACKs, namely, AVR [14] and DCB (Double-Covered Broadcast) [17].

AVR employs explicit ACKs on top of blind flooding. A receiving node sends an ACK to the sender for every message it hears, and rebroadcasts those that are received for the first time. A sender retransmits a message if no ACKs from any neighbor is received after a predefined period. AVR has been shown in [16] and [17] to achieve the best delivery ratio among all compared protocols. However, AVR can potentially suffer from the "ACK implosion problem". DCB's goal is to avoid this while still achieving high reliability. DCB adopts the idea of neighbor-designating – only selected 1-hop neighbors of a sender rebroadcast. The 1-hop neighbors are selected in such a way that every non-designated node is covered by at least two rebroadcasts. The rebroadcasts of the designated nodes are overheard by the sender and treated as implicit ACKs.

Our fifth experiment tests these ACK-based protocols and Fig. 3(b) plots the results (the maximum number of retries is 2). With small data, both AVR and DCB achieve higher delivery ratio as compared to blind flooding; but again, when data size increases, delivery ratio drops. Since it is very hard to get a large data packet through to neighbors, multiple retries does not solve the problem, but rather further congests the network.

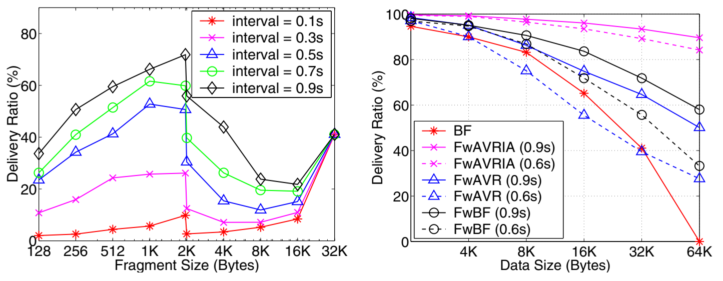
IV. The Road to Solutions: Fragmentation

So far, no protocol really solves the reliability problem with large-size data. However, as shown in the last section, most



(a) Increasing Redundancy (b) Acknowledgements

Fig. 3. Higher Layer Optimizations: Delivery Ratio Vs Data Size



(a) Fragmentation with Blind Flooding (b) Fragmentation + Reliability Control

Fig. 4. Fragmentation: Delivery Ratio Vs Data Size

protocols work very well with small-size data. We thus arrive at the conclusion that the problem is independent of the protocol. Realizing this, we look into the experiment trace files for commonality among the protocols and it then becomes clear where the problem lies. When a data packet larger than the MAC layer MTU (Maximum Transmission Unit) is passed to the IP layer, the IP layer fragments the data. As a result, multiple IP fragments are generated (for a single data packet) and passed to the MAC layer separately. Only if all of them are received, does a receiver deliver the reassembled data to upper layers. This is a rare occurrence. Our experiment traces show that while blind flooding 32KB data to 50 nodes, 72.4% of the IP fragments are discarded during reassembling (due to missing fragments). To further exacerbate the problem, when data size is beyond the maximum size of IP queues, the packet is fragmented as usual but certain fragments are dropped at the sender IP queue itself. Consequently no receiver can get the whole data, leading to zero coverage. In summary, fragment-level reliability becomes increasingly important as application data size increases. Of all the protocols discussed in Section III, only the MAC-unicast-based approach does fragment-level reliability; all others attempt reliability only at the application data level. For small data sizes, this translates to direct correspondence of reliability at the IP and MAC layer. However, with large data sizes, this assumption no longer holds true and thus the poor performance of all the protocols.

The solution to this problem would be to prevent IP from fragmenting the data. A straightforward way is to perform fragmentation at the application layer and pass small pieces of data to the IP layer, i.e., move fragmentation away from IP to the application layer. However, to do so, we need to answer three questions: (1) How large should the fragments be? (2) What should be the interval between disseminating consecutive fragments (*inter-fragment interval*)? and (3) What NWB protocol should be used to disseminate individual fragments? These questions reflect the tradeoffs in designing an application data broadcast protocol: (1) Small fragments can be delivered with higher guarantee, but the large number of fragments lowers the probability of all fragments being received. (2) Small intervals lead to short broadcast latency, but are more likely to cause interferences between the dissemination of different fragments. (3) An NWB protocol with low redundancy or no feedback mechanism incurs low inter-fragment interferences but does not offer high delivery guarantee for individual fragments. In the following we explore these tradeoffs in more depth.

A. *Fragmentation with Blind Flooding*

We first study using blind flooding as the fragment dissemination scheme. The application layer at the source node fragments the data packet, and the fragments are disseminated separately with a certain interval in between. Other nodes rebroadcast every fragment they receive, once. Fig. 4(a) plots the delivery ratio of broadcasting 32KB data in a 50-node network with varying fragment sizes and inter-fragment intervals (note that when fragment size equals 32KB, no fragmentation is performed). We make the following observations: (a) When fragment size is small, the delivery ratio is low. This is because the large number of fragments makes the probability of receiving all of them low. (b) For fragment sizes over the IP fragmentation threshold, the original problem of IP fragmentation resurfaces. (c) Inter-fragment interval plays a pivotal role in determining broadcast performance. Short intervals cause severe inter-fragment interferences, leading to low delivery ratio. Long intervals alleviate this problem, but at the cost of linearly growing broadcast latency. (d) With appropriately selected fragment size and interval, fragmentation can greatly improve delivery ratio. Further, there exists an "optimal" fragment size, with which the best delivery ratio is achieved. For instance, when inter-fragment interferences are insignificant, fragments sized equal to the IP fragmentation threshold provide the highest reliability. Finally, in our experiments on disseminating 64KB data (in 2000B fragments, at 0.9s intervals) to 50 nodes, 58% of the nodes on average are covered after 29.5 seconds. In contrast, without fragmentation, no receiver would be covered at all in this case.

B. *Fragmentation with Fragment-Level Reliability Control*

Now that we have seen the reliability enhancement brought by fragmentation, our next goal is to test if delivery ratio can be further improved by adding fragment-level reliability control. For this, we first choose AVR for fragment dissemination, since it has the best delivery ratio among reliable broadcast techniques. The fragmentation protocol is called FwAVR (Fragmentation with AVR). We also design a scheme AVRIA (AVR with Implicit ACKs) that marries the best of AVR and DCB. AVRIA eliminates the explicit ACKs in AVR while exploiting neighbors' rebroadcasts as implicit ACKs (similar to DCB). In AVRIA, a node does not retransmit as long as a rebroadcast of any neighbor is overheard. The protocol based on AVRIA is called FwAVRIA (Fragmentation with AVRIA). We now examine FwAVR and FwAVRIA against FwBF (Fragmentation with Blind Flooding), all with the "optimal" fragment size.

As depicted in Fig. 4(b), FwAVR has lower delivery ratio than FwBF with the same inter-fragment interval, i.e., AVR's reliability control impairs reliability rather than boosting it. This is due to the high volume of ACK traffic FwAVR generates, which leads to severe inter-fragment interferences. On the other hand, FwAVRIA achieves higher delivery ratio than FwBF and FwAVR, and its reliability is less susceptible to shrinking inter-fragment intervals. This is because FwAVRIA adds reasonable fragment-level reliability control without incurring explicit ACK transmissions. Thus, we make a very important observation. Adding reliability control blindly to fragments does not increase overall reliability. The extra traffic generated for reliability control is a key determinant of the overall reliability. If the extra traffic is low (as in FwAVRIA), then overall reliability is greatly enhanced.

V. The Proposed Protocol: READ

We base the design of the READ protocol on the following observations and insights obtained from our experiments: (1) In FwBF, if a fragment fails at the first hop, i.e., does not reach any neighbor of the source node, then no receiver will get the whole data. To prevent this pathological case, transmissions at the source node need stricter reliability control than others. (2) Implicit ACKs are superior to explicit ACKs, as seen from the performance of FwAVRIA against FwAVR. Implicit ACKs reduce inter-fragment interferences, thus improving overall reliability. (3) A drawback of FwAVRIA is that when all neighbors of a node have received a fragment, there will be no implicit ACKs generated. The fragment hence will be rebroadcast multiple times, which unnecessarily adds to network congestion. This can be avoided using NACKs if nodes are equipped to detect missing fragments [26] [27]. (4) Due to the uncertainty in wireless transmissions, the propagation of a fragment may last an unpredictably long time. Consequently, when fragments are disseminated at fixed intervals, a fragment is subject to being overwhelmed by a previous fragment's broadcast traffic. Thus, broadcasting fragments at dynamic intervals that adapt to the dissemination progress may be necessary.

The main characteristics of READ are: (a) READ splits a data packet into multiple fragments. All the fragments except the last one are of "optimal" size – a data fragment along with READ/UDP/IP headers exactly fit in the MAC MTU. Thus, IP fragmentation is avoided, while least number of fragments are generated. (b) READ disseminates the fragments separately using blind flooding. Blind flooding is employed as the fragment-level dissemination scheme because of its high reliability and simplicity. (c) READ exploits nodes' fragment transmissions as implicit NACKs. These implicit NACKs enable receivers to locally recover missing fragments without incurring extra ACK traffic. (d) READ performs stricter reliability control on source node transmissions. This is to ensure that fragments do not stop propagating at the first hop. (e) READ dynamically adjusts the starting time of a fragment's dissemination, depending on traffic load. Hence, a fragment is less likely to be overwhelmed by the dissemination traffic of preceding fragments.

Every message transmitted in READ carries a data fragment, and is encapsulated with a READ header. The READ header of a message sent by node N_i contains the following information: (1) the source-node-address/port-number pair (which uniquely identifies the dissemination of the data packet), (2) the number of fragments of the data packet (nf), (3) the sequence number of the carried fragment (0, 1, ..., $nf-1$), (4) the sequence number of the highest fragment N_i has received (hf_i), (5) the number of lost fragments N_i has detected due to out-of-order receptions (nlf_i), (6) the sequence number of the lowest lost fragment at N_i (llf_i), (7) the number of consecutive lost fragments starting from llf_i ($nclf_i$). The last four fields are used to advertise N_i's reception progress to neighboring nodes and request retransmissions of N_i's missing fragments.

```
CASE the transmission timer for $f_k$ ($tmr_k$) expires:
    MAC-broadcast $f_k$, piggybacking $hf_i$, $nlf_i$, $llf_i$, $nclf_i$;
    IF I am the source node AND $f_{k+1}$ is not out yet THEN
        IF $f_k$ has been transmitted less than retry limit THEN
            set up $tmr_k$ with a random delay;
        ELSE set up $tmr_{k+1}$ with zero delay;
CASE a message ($M$) carrying fragment $f_k$ is received:
    FOR each missing fragment $f_m$ requested in $M$
        IF I have $f_m$ AND $tmr_m$ is off THEN
            set up $tmr_m$ with a random delay based on $nlf_i$;
    IF I am the source node AND $tmr_k$ is on THEN
        cancel $tmr_k$; set up $tmr_{k+1}$ with zero delay;
    ELSE IF I am a non-source node THEN
        IF $f_k$ is received for the first time THEN
            set up $tmr_k$ with a random delay;
        ELSE
            IF $tmr_k$ is on THEN cancel $tmr_k$;
            IF $M$ is from the source node THEN
                set up $tmr_k$ with a random delay;
```

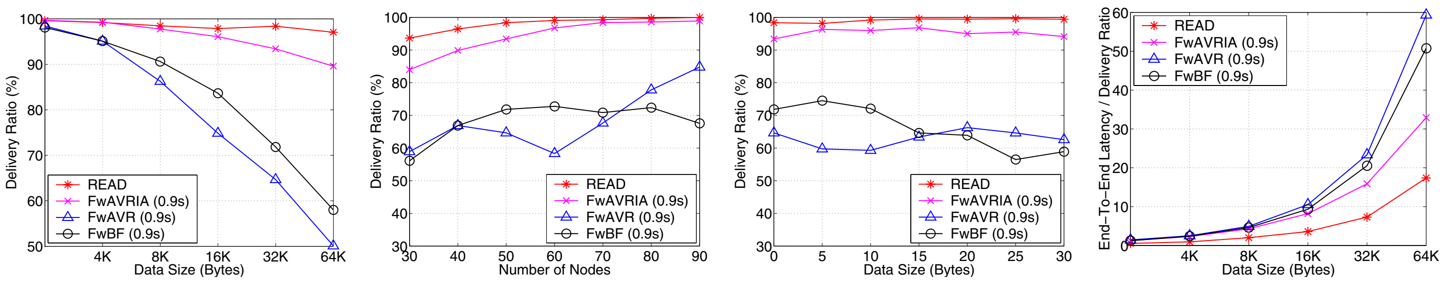
Fig. 5. READ: Protocol Pseudo-Code Executed on Node N_i

Protocol Description In detail, READ works as follows (the protocol pseudo-code is shown in Fig. 5).

(1) Source node S: The dissemination of each fragment, say f_k, starts by S MAC-broadcasting it. S then waits for its neighbors' rebroadcasts of f_k (implicit ACKs). S retransmits if no rebroadcast is heard within a certain period. It repeatedly retries until it hears a rebroadcast. Only after hearing a rebroadcast of f_k, or the maximum retry limit (10 in our implementation) is reached, does S sends out the next fragment, f_{k+1}. Thus, the inter-fragment interval dynamically adapts to the network traffic load – if the network is so congested that S cannot hear any rebroadcast of f_k, S postpones the dissemination of succeeding fragments so as to mitigate interferences.

(2) A non-source node N_i: On receiving an unseen fragment, say f_k, from a neighbor, say N_j, N_i schedules rebroadcasting f_k with a random delay (sets up the transmission timer for f_k). Meanwhile, N_i inspects N_j's reception progress piggybacked on the message (implicit NACK), and schedules retransmitting the fragments that N_j is currently missing. These fragments include N_j's consecutive lost fragments starting from llf_j, and the fragments N_j is expecting (the fragments succeeding hf_j). For each of these fragments (f_m), if N_i has it, N_i sets up its transmission timer with a random delay. This retransmission delay usually is larger than a rebroadcast delay, and is inversely proportional to nlf_i. In effect, a node with more lost fragments will retransmit earlier, thus having higher chances to request lost fragments. N_i cancels its pending retransmission of f_m if it overhears f_m thereafter (either a retransmission by another neighbor of N_j triggered by the same NACK, or a rebroadcast by N_j after N_j receives f_m). Thus, retransmission traffic is minimized. If N_i is a neighbor of S and receives f_k more than once from S, it is likely that S has not heard the rebroadcast of f_k from any of its neighbors. In this case, N_i retransmits f_k to acknowledge S and trigger the dissemination of f_{k+1}.

Performance Evaluation To demonstrate the advantages of READ, we examine its performance and compare it against



(a) Delivery Ratio Vs Data Size (b) Delivery Ratio Vs Num of Nodes (c) Delivery Ratio Vs Max Speed (d) Normalized Latency Vs Data Size

Fig. 6. READ: Performance Evaluation and Comparisons

other fragmentation protocols (FwAVRIA, FwAVR and FwBF, all with 0.9s inter-fragment intervals) though experiments. We measure their delivery ratios across several dimensions: (i) varying data size from 2KB to 64KB (results shown in Fig. 6(a)), (ii) varying network density (the number of nodes in the area varies from 30 to 90, while the data size is fixed at 32KB) (results shown in Fig. 6(b)), (iii) varying mobility (the maximum nodal speed varies from 5m/s to 30m/s, while the data size is fixed at 32KB) (results shown in Fig. 6(c)). Moreover, we compare the protocols' normalized latency with different sizes of data (results shown in Fig. 6(d)).

As depicted by Fig. 6(a), while other protocols' reliability declines with increasing data sizes, READ consistently achieves high delivery ratio (over 97%). Fig. 6(b) and 6(c) show that READ outperforms the other protocols across varying network density and mobility. Its delivery ratio tends to further increase when the network becomes denser or topological changes are more frequent. READ's high reliability is attributed to several factors: First, implicit NACKs help receivers recover the fragments they are missing. Second, stricter first-hop reliability control ensures the dissemination task is distributed from the single source to multiple other nodes. Furthermore, unlike the other protocols whose delivery ratios are greatly dependent on the inter-fragment interval parameter, READ uses dynamically adaptive intervals, and hence does not suffer as much from the interferences between different fragments' broadcast traffic.

As Fig. 6(d) shows, READ achieves shorter normalized latency as compared to the other protocols. In READ, a particular fragment is disseminated as soon as the preceding fragment is acknowledged. In effect, this expedites the dissemination process when the network traffic is light, and reduces data loss when the network is congested. In contrast, in the other protocols, the source node sends out fragments at fixed intervals. This either makes inefficient use of resources (leading to long latency), or causes severe interferences and hence significant amount of data loss (leading to low delivery ratio).

Though READ's performance is superior to the other protocols, it does not achieve hundred percent coverage. This is because of the following: in order to minimize extra traffic, READ uses only implicit NACKs for the recovery of missing fragments. However, the number of implicit NACKs a node can send is limited by the number of fragment transmissions it performs. Resolving this issue and seeking deterministic reliability is our plan of future research.

VI. Concluding Remarks

This paper addresses reliable application data broadcast in wireless ad hoc networks. We identify and explore the large-data reliability problem with existing broadcast techniques. We propose a protocol, READ, which exploits higher-layer fragmentation, implicit NACKs, first-hop reliability control and other mechanisms to provide high delivery guarantee for application data. Experiment results show that the READ protocol consistently achieves high delivery ratio and short latency, outperforming all other examined protocols.

References

[1] B. J. Betts and et al., "Improving situational awareness for first responders via mobile computing, Tech. Rep. NASA/TM-2005-213470, 2005.

[2] (2004) Broadband public safety data networks in the 4.9 ghz band. [Online]. Available: http://www.tropos.com/pdf/Spectrum_Whitepaper.pdf

[3] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *IEEE WMCSA*, Feb. 1999, pp. 90 – 100.

[4] S.-Y. Ni and et al., "The broadcast storm problem in a mobile ad hoc network," in *Mobicom*, 1999, pp. 151 – 162.

[5] H. Lim and C. Kim, "Multicast tree construction and flooding in wireless ad hoc networks," in *ACM MSWIM*, 2000.

[6] W. Peng and X.-C. Lu, "On the reduction of broadcast redundancy in mobile ad hoc networks," in *IEEE Mobihoc*, 2000.

[7] J. Sucec and I. Marsic, "An efficient distributed network-wide broadcast algorithm for mobile ad hoc networks, Tech. Rep. 248, Sept. 2000.

[8] J. Wu and F. Dai, "Broadcasting in ad hoc networks based on self-pruning," in *IEEE INFOCOM*, vol. 3, Mar. 2003.

[9] W. Lou and J. Wu, "On reducing broadcast redundancy in ad hoc wireless networks," *IEEE Trans. Mobile Computing*, vol. 1, no. 2, Apr. 2002.

[10] W. Peng and X.-C. Lu, "Ahbp: An efficient broadcast protocol for mobile ad hoc networks," *Journal of Computer Science and Technologies*, 2001.

[11] J. Wu and F. Dai, "A generic distributed broadcast scheme in ad hoc wireless networks," in *IEEE ICDCS*, May 2003.

[12] E. Pagani and G. P. Rossi, "Reliable broadcast in mobile multihop packet networks," in *ACM MOBICOM*, Sept. 1997.

[13] B. Williams and T. Camp, "Comparison of broadcasting techniques for mobile ad hoc networks," in *IEEE MOBICOM*, 2002.

[14] S. Alagar, S. Venkatesan, and J. Cleveland, "Reliable broadcast in mobile wirless networks," in *IEEE MILCOM*, Nov. 1995, pp. 236 – 240.

[15] P. Rogers and N. Abu-Ghazaleh, "Towards reliable network wide broadcast in mobile ad hoc networks, Tech. Rep. tr-cs-02-04, Dec. 2004.

[16] W. Lou and J. Wu, "A reliable broadcast algorithm with selected acknowledgements in mobile ad hoc networks," in *GLOBECOM*, 2003.

[17] ——, "Double-covered broadcast (dcb): A simple reliable broadcast algorithm in manets," in *IEEE INFOCOM*, vol. 3, Mar. 2004.

[18] E. Vollset and P. Ezhilchelvan, "An efficient reliable broadcast protocol for mobile ad-hoc networks, Tech. Rep. CS-TR-822, 2003.

[19] K. Singh, A. Nedos, G. Gaertner, and S. Clarke, "Message stability and reliable broadcasts in mobile ad-hoc networks," in *ADHOC-NOW*, 2005.

[20] (2005) Qualnet 3.9 user's guide, scalable network technologies inc. [Online]. Available: http://www.scalable-networks.com/

[21] J. Wu and F. Dai, "Efficient broadcasting with guaranteed coverage in mobile ad hoc networks," *IEEE Trans on Mobile Computing*, 2005.

[22] V. Drabkin and et al., "Efficient byzantine broadcast in wireless ad-hoc networks," in *DSN*, 2005, pp. 160 – 169.

[23] K. Obraczka, K. Viswanath, and G. Tsudik, "Flooding for reliable multicast in multi-hop ad hoc networks," *Wireless Networks*, Nov. 2001.

[24] K. Viswanathand and et al., "Exploring mesh and tree-based multicast routing protocols for manets," *IEEE Trans on Mobile Computing*, 2006.

[25] J. Lipman, P. Boustead, and J. Chicharo, "Reliable optimised flooding in ad hoc networks," in *IEEE Symp. on Emerging Techonologies*, 2004.

[26] S. Y. Cho, J. H. Sin, and B. I. Mun, "Reliable broadcast scheme initiated by receiver in ad hoc networks," in *IEEE LCN*, Oct. 2003.

[27] T. Kunz, "Multicasting in mobile ad-hoc networks: Achieving high packet delivery ratios," in *CASCON*, 2003.