PIM-WSN: Efficient Multicast for IPv6 Wireless Sensor Networks

Alan Marchiori and Qi Han
Department of Mathematical and Computer Sciences
Colorado School of Mines
Email: {amarchio,qhan}@mines.edu

***Abstract*—We present PIM-WSN, a protocol independent multicast (PIM) protocol tailored for IPv6 wireless sensor networks (WSNs). Existing solutions for multicast in WSNs are limited because they either support multicast only from a single source node (usually the root node) or they limit the multicast group size to constrain memory usage. Our design allows any node to be a mulitcast source with an unlimited number of subscribers. We constrain the memory usage by approximating multicast group membership using a fixed sized Bloom filter. The efficiency of the protocol degrades as the false positive rate of the Bloom filter increases; however, correct operation is always maintained. Using detailed simulations we show that PIM-WSN achieves 1) high packet delivery rate (over 97%), 2) low latency per hop (less than 5 ms), and 3) lower radio utilization than all other comparable protocols (by more than 50%). Using a ten-hop testbed of TelosB motes we have verified our implementation of PIM-WSN for TinyOS 2.x with the Blip IPv6 networking stack which uses only 5,978 bytes of ROM and 235 bytes of RAM.**

***Keywords*-Sensor networks, Network Protocols, Distributed applications**

I. Introduction

Sensor network applications are increasingly being developed that not only passively sense the physical environment but also actively interact with it. In order for this interaction to be more responsive, intuitive, and scalable a distributed approach is essential. Multicast communication is useful for this application because it allows sensed data to be transmitted directly to multiple receiving nodes. Multicast is heavily studied in IP-based networks, while multicast in wireless sensor networks (WSNs) has focused on specialized protocols with limited functionality (e.g. [3, 8, 9, 15, 18, 19]). These limitations constrain the number of nodes in the multicast and which nodes can source multicast traffic to ease implementation on resource constrained sensor nodes. Support is growing for IPv6/6lowpan support in WSNs, however, current implementations do not provide support for multicast. This is because no existing WSN multicast implementations provide the broad support needed for general purpose multicast. Our contribution is a general purpose multicast protocol able to provide multicast for emerging IPv6/6lowpan WSN solutions.

The most common form of multicast is Protocol Independent Multicast (PIM). "Protocol independent" refers to the fact that the it relies on the underlying network routing protocol, rather than implementing its own routing protocol [6]. This is a good match for sensor networks because many routing protocols already exist that have been validated and tuned for a certain application or network topologies. We elect to follow the same, protocol independent, approach for our multicast design and implementation.

In this paper, we design, implement, and evaluate PIM-WSN, a protocol independent multicast protocol customized for WSNs. The distinguishing features of PIM-WSN are:

- Unlimited number of source and subscriber nodes
- Constant memory usage
- Reliable multicast packet delivery
- No periodic messaging or route maintenance

Through extensive performance studies using carefully designed simulations we demonstrate that PIM-WSN delivers packet delivery comparable to other multicast approaches (over 95%), low latency (under 5 ms), and significantly outperforms the three other protocols we tested in terms of radio utilization (by more than 50%). We have also implemented and verified PIM-WSN using TinyOS 2.x with the Blip IPv6 networking stack [4]; our implementation requires only 5,978 bytes of ROM and 235 bytes of RAM on the TelosB platform.

II. Related Work

Numerous multicast protocols have been proposed for wireless sensor networks. Existing approaches can be classified as: adaptations of mobile ad-hoc network (MANET) protocols, WSN-specific, or geographic. We will not consider geographic protocols (such as [9, 18]) because they rely on extra information (geographic) that we do not. In fact, PIM-WSN used in conjunction with a geographic routing protocol, would effectively create a geographic multicast. We also do not consider the numerous adaptations of link-state protocols (such as OSPF, OLSR, and FRM [10, 14, 17]) because this routing paradigm is generally not supported in WSNs.

The authors of [3] have evaluated the MANET routing protocol, Adaptive Demand-driven Multicast Routing (ADMR), in wireless sensor networks. The main problem with this (and other) MANET routing protocols is they require storing state for up to every node in the network. In a WSN nodes cannot store this much state information, so the authors explored various policies to reduce memory usage by pruning low quality paths. However, it is not clear if this could result in a disconnection of the multicast. Our solution to this problem may deliver packets to extra nodes but does ensure no node is disconnected (if a path exists). Exploring other MANET multicast protocols does not look like a promising approach. Lee et al. has explored four MANET multicast protocols, however, they all depend on frequent periodic messaging to maintain multicast routes and do not take any steps to limit memory usage [11] . Both of these issues must be resolved to be useful in WSNs where periodic messaging consumes too much energy and memory is constrained.

This leaves WSN-specific multicast protocols. A good theoretical analysis of the problem is presented in [8], however, the evaluation is purely theoretical. One of the earliest implementations of multicast in WSNs we found was presented in [19]. This approach bypasses the memory problem by only supporting base to node multicast (not peer to peer) with 8-bit node identifiers. This limits

978-1-4577-0351-5/11/$26.00

the maximum multicast state, so that even in the worse case, it would fit into a node's constrained memory. In addition, the protocol itself is a standard multicast approach, requiring periodic subscriptions to avoid expiring nodes.

Branch aggregation multicast (BAM) [15] is the only prior work that also uses the protocol independent approach in WSNs. BAM relies on metrics from the routing layer to choose next hop destinations. The significant difference between BAM and PIM-WSN is that BAM appends multicast subscription information to every data packet. This avoids the overhead of sending join messages and storing forwarding information on each node, while increasing the overhead of sending each data packet. This works with short network addressing schemes and a few subscribers, but will not scale well to IPv6 address and more than a handful of subscribers.

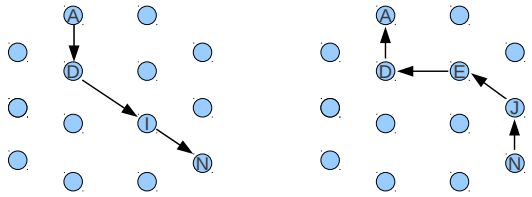
III. Design of PIM-WSN

Our goal is to keep the protocol as lightweight as possible while balancing memory efficiency, energy efficiency, reliable packet delivery, and low latency. We will first give an overview of PIM-WSN and then describe in more detail how we address the problems that are specific to tailoring PIM-SSM for WSNs.

In PIM-WSN an interested node initiates the multicast data transfer the same way as PIM-SSM, i.e., by sending a unicast *join* message to the source node of the multicast containing the source-group pair (S, G). This message is sent using any available unicast routing protocol. When a source node receives a valid *join* message it responds with a *join acknowledgment* unicast back to the subscriber. The destination node continues sending *join* messages until it receives a *join acknowledgment* or the subscription fails after making a maximum number of attempts. After the node receives a *join acknowledgment* it is now a subscriber to the specified multicast and multicast data will be delivered. The subscriber does not need to send any more *join* messages for this multicast (S, G) unless a failure is detected. The subscription process also serves to inform nodes along the path from the source to subscriber that they must forward packets. This is done by nodes overhearing the *join acknowledgment* message. The multicast source and every node on the path to the unicast destination of the *join acknowledgment* store the address of the next hop towards the destination. These nodes are considered subscribers to the multicast (S, G). This information is later used to forward multicast data packets sent from that (S, G).

Figure 1 illustrates node A joining node N's multicast. Arrows indicate the unicast join and join acknowledgement messages. Each node along the path of the join acknowledgement (D, E, and J) updates their local subscription list. For example, node E stores that it must forward multicast packets from N to D. Node D then forwards N's multicast packets to node A. If multiple nodes subscribe to N's multicast, paths to each subscriber are created using the same process. Only one packet is transfered along paths shared by multiple subscribers.

PIM-SSM was designed with wired networks in mind and customizing it for WSN presents several interesting challenges. Three main problems arise with a naive PIM-SSM implementation for WSN. 1) memory usage: storing all the subscription data for each multicast is not possible for a resource constrained devices. 2) reliability: wireless communication is notoriously unreliable; "best effort" delivery works well in wired networks, however, it does not perform well on wireless links. 3) periodic signaling: PIM-SSM keeps forwarding route up to date by each node repeating the subscription process periodically. However, many WSN applications have very low data rates where periodic signaling could dominate energy consumption. Next we illustrate how each of these three issues are addressed in PIM-WSN.



(a) A joins N's multicast (b) N acknowledges A's join message

Figure 1. PIM-WSN subscription process. Join and join acknowledgement messages may take different paths (as shown). Data flows along the path of the join acknowledgement.

A. Limiting Memory Usage

We must store subscription data at each forwarding node. Specifically, the triple (D_n, S, G) should be stored at each forwarding node n, where D_n is the next-hop destination of the *join acknowledgment* message along the path from the source to subscriber (S) of mulitcast (group) (G). If we store all the subscription data, the memory usage will quickly exceed the limited storage capacity of resource constrained sensor nodes. On the TelosB storing 213 uncompressed IPv6-style tuples would consume all of the TelosB's memory. This is clearly not an effective approach. Our solution is to use a fixed-sized counting Bloom filter as an improvement to the standard Bloom filters used in [17] to store subscription information.

Because Bloom filters are a probabilistic data structure false-positive results are possible (but not false-negative). As the number of subscribers increases, the false-positive rate also increases. In PIM-WSN, a Bloom filter false positive could result in forwarding multicast data to a node that is not actually subscribed to the multicast. As a result, the efficiency of the protocol is affected. The functionality is still correct because all nodes that are subscribed still receive data. Nodes incorrectly receiving multicast packets can simply drop them. We believe this is a much better solution than dropping a node from the multicast when memory is exceeded as in other multicast implementations (e.g., [3]).

B. Improving Reliability

PIM-WSN uses one-hop broadcast messages to transfer data to all neighboring nodes at the same time. Common link-layer reliability mechanisms (acknowledgments) cannot be applied to broadcast traffic. Branch aggregation multicast (BAM) [15] also uses one-hop broadcast messages where all neighbor nodes acknowledge the packet. This provides good reliability at the expense of increased energy consumption and congestion. Congestion can be reduced by using delayed acknowledgments, however, this increases latency. Robust broadcast propagation (RBP) [20] proposes a probabilistic method to achieve reliable broadcasts where a fraction of neighboring nodes is dynamically computed and if

at least these many acknowledgments are received the packet is considered to be delivered to all nodes. This improves performance, but still requires every node to transmit an acknowledgement to every packet, resulting in high energy usage and congestion. Our alternative approach is to have the sender designate a single node to acknowledge the one-hop broadcast packet. The address of the designated node is included in the packet header. The result is improved reliability while reducing energy consumption, congestion, and latency.

When a node receives a multicast data packet, it must check if it is designated to acknowledge the transmission. There are two well known ways for a node to acknowledge a data packet: implicitly and explicitly. For improved efficiency, PIM-WSN supports both methods of acknowledgment. The implicit mode is used when the designated node must itself forward the packet; the act of forwarding a multicast data packet acts as an acknowledgment. If a forwarder that is waiting for an acknowledgment receives the same data packet from the node designated to provide the acknowledgment, it knows the packet has been received and may stop retransmitting. There is of course no guarantee the original sender will hear the message being forwarded. So, the original forwarder will retransmit the packet with the same designated node. Whenever a node receives a duplicate data packet and it is the designated acknowledgment node, it will send an explicit acknowledgment to the sender.

C. Eliminating Periodic Messaging

Proactive route maintenance is used in PIM-SSM to keep forwarding routes up to date. However, if data is sent infrequently, maintaining the forwarding state can dominate energy consumption. Because low data-rate applications are common in WSNs, we instead reactively update multicast routes. If a multicast delivery failure is detected (which is possible because there is a reliability mechanism), we then notify subscribing nodes that a rejoin is necessary. A rejoin is accomplished by the subscribed nodes on failed forwarding paths rejoining the multicast. This process creates new forwarding paths using current routing information.

A rejoin is triggered after a packet is retransmitted a maximum number of times. To signal this event, the multicast data message is encapsulated in a unicast packed and then sequentially unicast to each subscribed neighbor node. This relies on the unicast routing protocol to route the message (possibly over multiple hops) to the destination node(s). The counting Bloom filter is then decremented on the forwarding node for each (D_n, S, G) triple so that after several failures the node will be removed from the filter. If a node receives a unicast encapsulated multicast packet it assumes there was a delivery failure and repeats the subscription process. If a node receives a unicast data message and it is also a forwarding node it resumes multicast forwarding and follows the usual procedure using one-hop broadcast messages. This localizes the failure to only the effected link, improving efficiency.

In our evaluation we do not send periodic join messages and rely entirely on fault detection. However, it is possible that the unicast data messages may also be lost. For example, if a node became disconnected from the network for an extended period of time. To ensure the correct operation of PIM-WSN it is necessary to detect network disconnect/reconnect events and resubscribe to every multicast.

IV. Performance Evaluation via Simulation

Our goal is to evaluate PIM-WSN under real world conditions, which are hard to replicate using traditional network simulators like TOSSIM, OmNet++, and ns2. Therefore, we use connectivity traces from real wireless sensor network deployments. Because real network traces are used they inherently contain real-world packet loss and noise. Our simulator, WsnSimPy, is a trace-based simulation tool built on the discrete event simulator SimPy [1]. For validation of the simulation tool, we direct readers to our previous work [12].

The connectivity traces are from the "soda" dataset, collected by J. Ortiz and D. Culler in a UC Berkeley office space[2]. They were obtained as follows. 46 IEEE802.15.4-compliant TelosB motes are deployed in a 50m x 50m indoor environment, and are constantly listening for packets. One after the other, each mote transmits a burst of 100 packets with a 20ms inter-packet time and a transmission power of 0dBm on each of the 16 frequency channels. Timers are used to ensure that all nodes switch channels simultaneously.

A. PIM-WSN Setup

There are several questions about PIM-WSN that we must now answer. They are: 1) How is the designated acknowledgment node selected? 2) When is a link failure assumed? 3) How is the bloom filter configured? First, we use the weakest link acknowledgment policy. The subscribing node with the worst routing metric (but not worse than a preset threshold) is designated to provide the acknowledgment. Second, we assume a link failure after a multicast packet is transmitted four times without acknowledgment. The rationale to use a relatively low value here is to avoid multicast forwarding paths with poor links. This is useful in our simulation because the network is well connected; i.e. removing links with $ETX > 4$ still results in a well connected network. In sparse deployments, this might generate excessive subscription traffic if the only available links have $ETX > 4$. In general we could dynamically assign this threshold based on local network conditions.

Third, we must define the parameters for the Bloom filter used to hold subscription information. Selection of filter size is application specific because it should be large enough to achieve low false-positives with the expected number of multicast subscribers. For our evaluation we set the filter size to achieve good performance with each node subscribing to one multicast. Our simulation has 43 nodes; therefore, we want a low false positive rate after 43 insertions. Solving for the Bloom filter false positive rate, P_{fp} = 0.1 with n=43, yields m=206 and k=3 (rounded to the nearest integer). Each counter is 4-bits, so the total memory requirement is 103 octets. This is an awkward size to use in practice so we round down to the nearest word-aligned value and use a fixed-sized Bloom filter of 96 octets.

B. Simulation Results

For each simulation run we fix the number of source nodes and number of subscribers per source. Each run is assigned a unique value to seed the simulator's random number generator (Mersenne

[1] http://simpy.sourceforge.net/

[2] http://wsn.eecs.berkeley.edu/connectivity/

twister [13]). Source nodes are selected randomly without replacement. Subscriber nodes are selected independently for each source without replacement. This does not prohibit a node from being a subscriber to multiple (different) multicasts, or a source of one multicast being a subscriber to one or more other multicasts.

The performance of PIM-WSN is benchmarked against branch aggregation multicast (BAM), simple flooding, and sequential unicast communication. BAM was selected because it is the only other protocol independent multicast approach implemented in WSNs. In addition BAM uses optimized multicast delivery trees and claims to be "very energy efficient." Flooding has the advantage of no setup costs and optimal routing (all routes are used, therefore the optimal route must be used). Finally, comparing to sequential unicast allows us to explore the minimum number of subscribes needed to make multicast a practical solution.

Each protocol was implemented on the same simulated IPv6 stack (similar to the one provided by Blip). BAM had to be modified to work with the routing protocol (RPL) because each node only maintains one route (to the root) in RPL while BAM requires each node to maintain routes to every other node in the network. We solve this problem by initially unicasting the multicast packet from the source to the root. The root maintains a complete routing table and therefore can then implement the BAM algorithm. The computed multicast tree is encoded in the packet header and sent down the tree to the subscribers following the original BAM protocol. Simple flooding is implemented by modifying PIM-WSN to include a multicast group where every node is implicitly subscribed. This retains the one-hop reliability mechanism of PIM-WSN, where every node will forward each packet once, while removing the subscription overhead. Finally, for our unicast implementation, each source node maintains a list of subscribers that is initialized at the start of the simulation. When transmitting a multicast packet, it is unicast sequentially to each subscriber with a 25 millisecond delay between transmissions. We experimentally found that a 25 millisecond delay was sufficient to avoid collisions due to congestion at the source.

For each simulation we collect packet delivery ratio (PDR), latency, average number of hops, and radio utilization. PDR is computed as ratio of the number of received data packets to how many should have been received (which is 100). In the case of flooding, where every node should receive the packet, we only count receptions by a node if it was selected as a multicast subscriber. Latency is computed by time stamping the multicast packet when it is first transmitted and comparing this to the current time when it is received. In the case of unicast, where packets are transmitted sequentially, the transmission time is the time of the transmission to the first subscribing node. Average hop count is computed by storing the hop count contained in the first reception of each packet if duplicate packets were received. Because latency is strongly dependent on hop count (which may vary from simulation to simulation) we divide the latency by the hop count for comparison. Average radio utilization is computed as the average percentage of the time that the radio was transmitting or receiving. The radio utilization relates the efficiency of each protocol and represents a lower bound on the radio duty cycle using an ideal low-power listening strategy. However, this is not necessarily a good indicator of node energy consumption, because it does not take into account timing or CPU energy consumption.

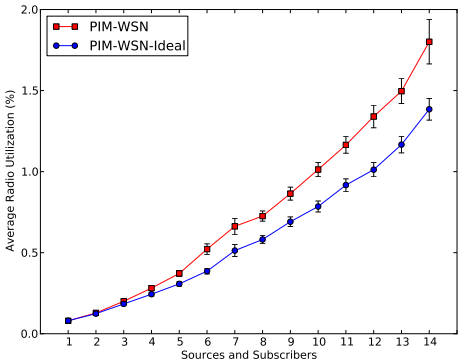


Figure 2. The effect of bloom filter errors.

Radio utilization includes every node in the network while PDR, latency, and hop counts are computed only on subscribing nodes.

We repeat each trial 20 times with 100 data packet transmissions from each source sent at a rate of 32 packets per minute (PPM) unless otherwise noted. The results are averaged and 95% confidence intervals computed and plotted when visible.

1) The effect of Bloom filter errors: Figure 2 shows the effect of bloom filter false positives on PIM-WSN. For this experiment we compare PIM-WSN to PIM-WSN-Ideal. PIM-WSN-Ideal is implemented with an unlimited buffer for storing exact subscription information on each node. As a result there are no false positive results, which result in the unnecessary forwarding of packets. In these experiments we increase both the number of sources and subscribers per source; always keeping the number of sources and subscribers per source equal. For example, if there were 5 source nodes each source also has 5 subscribers. This is necessary to increase the load on the bloom filter and therefore create more false positive results. PDR and latency is not shown because there was no significant effect.

Because false positive Bloom filter results cause PIM-WSN to forward packets unnecessarily, we expect to see an increase in average radio utilization as the number of sources and subscribers increases. Both algorithms had nearly identical radio utilization when there were few sources and subscribers (due to little unnecessary forwarding). When the number of sources and subscribers increases, the number of insertions into the bloom filter also increases, resulting in a higher rate of false positive results. These false positive results cause unnecessary forwarding, which increases the radio duty cycle. In the case of 14 sources and subscribers there are 196 (source, subscriber) pairs that could be inserted into the bloom filter of each node. With this number of insertions into the Bloom filter the false positive rate could be as high as 86.7% (if every multicast went through a single node). The result of these false positives resulted in an increase in radio utilization by 29%. This demonstrates the trade-off PIM-WSN makes to achieve constant memory usage.

2) The effect of increasing subscribers: Figure 3 shows the result of one source node sending data while increasing the number of subscribers from 1 to 14. All methods achieve good PDR; over 95% in all cases. The average PDR over all experiments were 100%, 99.62%, 98.67%, and 97.67% for flooding, BAM, unicast, and PIM-WSN respectively. Flooding is able to do well because

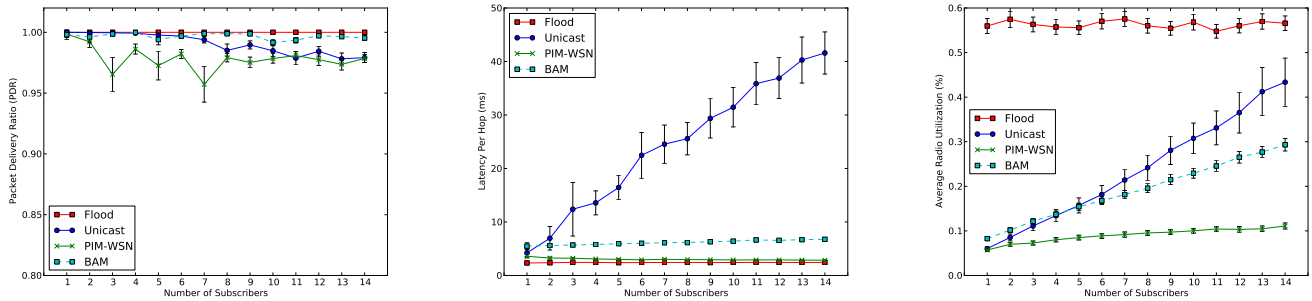


Figure 3. The effect of increasing the number of subscribers.

our network is rather dense, so there are multiple opportunities for a node to receive each packet. Although PIM-WSN had the worst PDR, we believe 97.67% is good enough for many applications and is very energy efficient. Losses in PIM-WSN arise when a node is forwarding to multiple one-hop neighbors. Because PIM-WSN designates only one node to acknowledge each data packet, it has no way to detect if a non-designated node fails to receive the packet. We could improve the PDR of PIM-WSN by requiring every subscribed neighbor node acknowledge each multicast packet as in BAM. However, BAM achieves only a 2% improvement in PDR compared to PIM-WSN while radio utilization increases by 100%. The best solution might be to dynamically select a variable number of subscribed neighbor nodes to acknowledge each packet based on the local network conditions; [20] explores this idea for broadcast traffic.

Radio utilization in PIM-WSN was lower than all other protocols, even in the case of just one subscriber. On average it was 52.66% lower than BAM, 61.11% lower than unicast, and 83.99% lower than flooding. Although PIM-WSN has the additional overhead of joining the multicast, it reduces packet transmissions by using a single designated acknowledgement node and by using overhearing for implicit acknowledgments whenever possible. The setup costs happen once while the savings on data transmissions occur every time a packet is sent. The scaling of PIM-WSN is also better than either unicast or BAM. This is interesting because PIM-WSN does not attempt to use optimized routes while BAM does. We see higher radio utilization in BAM because of two reasons. 1) it does not have a join phase where routes are setup, rather the route is encoded in every multicast packet and as the number of subscribers increases the amount of routing overhead also increases. 2) BAM requires every neighbor node acknowledge each packet. The net effect is that BAM had surprisingly high radio utilization. In fact, unicast was better than BAM until there were 6 subscribers. Flooding had the highest radio utilization and did not depend on the number of subscribing nodes, because every node receives every packet regardless of whether it subscribed or not. If we extrapolate these trends linearly, unicast will surpass flooding at 21 subscribers and BAM at 31; PIM-WSN remains more efficient than flooding.

Flooding and PIM-WSN both achieve very low per hop latency (even better than unicast). Because every node forwards every packet in flooding, it must therefore be forwarded along the optimal path for each packet. PIM-WSN does well because it uses implicit acknowledgments to reduce the overhead of forwarding a packet. The latency of PIM-WSN is the only one to decrease as the number of subscribers increases. This is due to the relative number of implicit acknowledgments increasing; PIM-WSN effectively approaches the case of no explicit acknowledgments, as the number of subscribers increases. BAM has higher latency because of two reasons: 1) in our implementation the packet must first be unicast to the root and then multicast to the subscribers; and 2) BAM uses delayed acknowledgments because every subscribed neighbor node must send an acknowledgement.

3) The effect of increasing sources: Figure 4 shows the result of increasing the number of sources while holding the number of subscribers per source constant at 5. This is a direct analog to increasing the number of subscribers. As expected the results follow the same pattern. One notable difference is that the latency of unicast does not depend on the number of sources because each source is independent. The radio utilization of all algorithms increases approximately linearly because no aggregation is performed by any of the algorithms between multicast packets from different sources. This result demonstrates that PIM-WSN can efficiently handle multicast data from many sources.

V. Implementation

To implement PIM-WSN we use TinyOS 2.x with the Blip IPv6 networking stack as base. Because Blip is still a work in progress, PIM-WSN is implemented above the IP layer. This isolates PIM-WSN from changes in the Blip networking stack. In IPv6 multicast is indicated by the destination address of the packet matching the prefix ff00::/8. Instead we use the protocol (next header) field of the IP interface to indicate a multicast packet. PIM-WSN wires to this protocol on the IP interface provided by Blip. PIM-WSN data packets are sent with the IP destination address FF02::1 which is translated by Blip into the IEEE 802.15.4 broadcast address `0xffff`. All PIM-WSN packets are sent with a common 20-byte header which could be reduced to 10-bytes by removing diagnostic fields.

Our implementation of PIM-WSN on the TelosB platform requires 5,978 bytes of ROM and 235 bytes of RAM. In a 10-hop linear network with one source and one subscriber PIM-WSN delivered 96.76% of the packets. Repeating the experiment using UDP yielded a comparable delivery rate of 97.63%. The latency (measured via GPIO pins) averaged 326.8 ms from end-to-end for PIM-WSN while UDP averaged 253.6 ms, or a difference of 7.32 ms per hop. The increased latency is because the reliability mechanism in PIM-WSN is implemented on top of IP. The IP

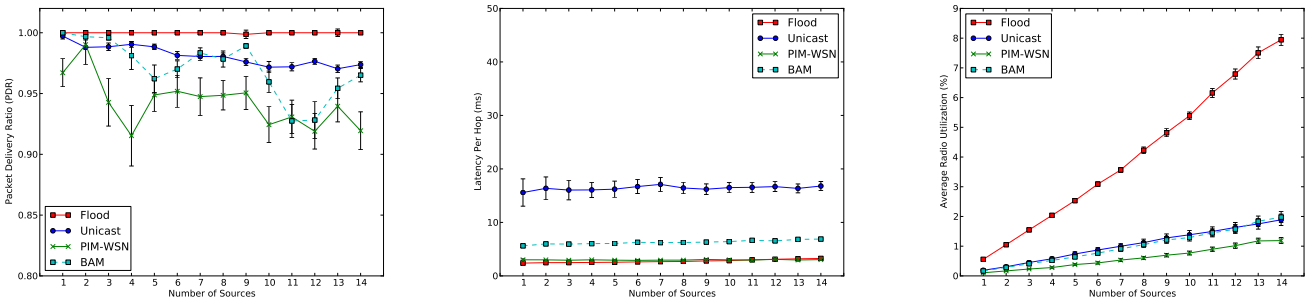


Figure 4. The effect of increasing the number of sources.

interface design forced us to use a longer resend delay because it hides all packet queuing and processing delays at and below the IP layer. Blip implements reliability at layer 2 and is able to use a 15 ms resend delay. Because we do not have access to the internal state of the IP protocol, we were use a 30 ms resend delay to account for the unknown processing and queuing delays. This results in PIM-WSN having slightly higher latency than UDP in practice, however, the primary reason to use PIM-WSN is to enable multicast communication. These experiments are highly favorable to PIM-WSN–they show that even in the worse case of one subscriber there is only a very small performance penalty when compared to native UDP. In our future work, we plan to better integrate PIM-WSN with the IP layer to completely eliminate this problem.

VI. Conclusion

In this paper we present PIM-WSN, a protocol independent multicast routing protocol tailored for IPv6 wireless sensor networks. The primary challenge to implementing multicast in this domain is efficiency in terms of memory and energy. Our novel design addresses the memory problem by approximating multicast group membership using a fixed sized Bloom filter. Energy efficiency is addressed by modifying the procedure for maintaining the multicast state. Our approach does not remove subscribers due to timeout (thus requiring periodic rejoining) but rather on delivery failure. In the event of delivery failure, PIM-WSN falls back to unicast routing and signals the node to rejoin the multicast. This gives nodes high confidence that they will remain in the multicast even after the failure of multicast forwarding paths. PIM-WSN has been thoroughly evaluated using trace-based simulation and validated on a wireless sensor network test bed of TelosB motes. Testbed results validate the functionality of PIM-WSN and demonstrate that it performs well in practice.

VII. Acknowledgments

This work was supported in part by NSF grant CNS-0855060 and the U.S. Department of Energy through the National Renewable Energy Laboratory under contract number DE-AC36-08GO28308.

References

[1] W. S. A. Adams, J. Nicholas. Protocol independent multicast - dense mode (PIM-DM): Protocol specification (revised). RFC3973, 2005.

[2] B. H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13, 1970.

[3] B.-r. Chen, K.-K. Muniswamy-Reddy, and M. Welsh. Ad-hoc multicast routing on resource-limited sensor nodes. In *REALMAN*, 2006.

[4] S. Dawson-Haggerty and A. Tavakoli. Berkeley ip implementation for low-power networks. http://smote.cs.berkeley.edu:8000/tracenv/wiki/blip, accessed 2009.

[5] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *MobiCom*, 2003.

[6] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C. gung Liu, and L. Wei. The PIM architecture for wide-area multicast routing. *IEEE/ACM Transactions on Networking*, 4:153–162, 1996.

[7] B. Fenner, M. Hadley, H. Holbrook, and I. Kouvelas. Protocol independent multicast - sparse mode (PIM-SM): Protocol specification (revised). RFC4601, 2006.

[8] R. Flury and R. Wattenhofer. Routing, anycast, and multicast for mesh and sensor networks. In *InfoCom*, 2007.

[9] Q. Huang, C. Lu, and G. catalin Roman. Mobicast: Just-in-time multicast for sensor networks under spatiotemporal constraints. In *IPSN*, 2002.

[10] A. Laouti, P. Jacquet, P. Minet, L. Viennot, T. Clausen, and C. Adjih. Multicast Optimized Link State Routing. 2003.

[11] S.-J. Lee, W. Su, J. Hsu, M. Gerla, and R. Bagrodia. A performance comparison study of ad hoc wireless multicast protocols. In *InfoCom*, 2000.

[12] A. Marchiori, L. Guo, J. Thomas, and Q. Han. Realistic performance analysis of WSN protocols through trace based simulation. In *PE-WASUN*, 2010.

[13] M. Matsumoto and T. Nishimura. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *TOMACS*, 8(1), 1998.

[14] J. Moy. Multicast extensions to ospf. RFC1584, 1994.

[15] A. Okura, T. Ihara, and A. Miura. Bam: branch aggregation multicast for wireless sensor networks. In *MASS*, 2005.

[16] S. K. Park and K. W. Miller. Random number generators: good ones are hard to find. *Communications of the ACM*, 31(10), 1988.

[17] S. Ratnasamy, A. Ermolinskiy, and S. Shenker. Revisiting IP multicast. *ACM SIGCOMM Computer Communication Review*, 36(4):15–26, 2006.

[18] J. A. Sanchez, P. M. Ruiz, and I. Stojmenovic. Energy-efficient geographic multicast routing for sensor and actuator networks. *Computer Communications*, 30(13), 2007.

[19] A. Sheth, B. Shucker, and R. Han. VLM2: A very lightweight mobile multicast system for wireless sensor networks. In *WCNC*, 2003.

[20] F. Stann, J. Heidemann, R. Shroff, and M. Z. Murtaza. RBP: robust broadcast propagation in wireless networks. In *SenSys*, 2006.