May 2006

A new technique for reducing MAC address overheads in sensor networks

Kwan-Wu Chin
University of Wollongong, kwanwu@uow.edu.au

Darryn Lowe
University of Wollongong, darrynl@uow.edu.au

Ricardo Gandia Sanchez
University of Wollongong, rgs653@uow.edu.au

A new technique for reducing MAC address overheads in sensor networks

Abstract

In sensor networks, the size of the medium access control (MAC) address is prohibitively expensive when one considers the small payload and the cost of transmitting one bit. We address this problem by proposing a technique that encodes L-bits of the payload with a key derived from MAC addresses. We show how the decoding process removes the need for MAC addresses in packets.

Publication Details

This article was originally published as: Chin, KW, Lowe, D & Sanchez, RG, A new technique for reducing MAC address overheads in sensor networks, IEEE Communications Letters, May 2006, 10(5), 338-340.

A New Technique for Reducing MAC Address Overheads in Sensor Networks

Kwan-Wu Chin, Darryn Lowe, and Ricardo Gandia Sánchez

***Abstract*— In sensor networks, the size of the medium access control (MAC) address is prohibitively expensive when one considers the small payload and the cost of transmitting one bit. We address this problem by proposing a technique that encodes L-bits of the payload with a key derived from MAC addresses. We show how the decoding process removes the need for MAC addresses in packets.**

***Index Terms*— Sensor networks, MAC address coding.**

I. Introduction

SENSOR networks consist of nodes that are equipped with one or more sensors, e.g., humidity or motion, suited to monitoring a given environment. In general, these nodes are resource constrained where they have very low data rate and limited battery life. These constraints mean that sensor based communication protocols have to be particularly energy efficient, lest a sensor network quickly becomes non-functional.

The authors of [1][2] are the first to document the significant header overhead due to MAC addresses. For example, the payload of a data packet may be only a few bytes in length, but the source and destination MAC addresses take up six bytes each on the Berkeley Motes [3]. A more efficient scheme may consider sizing MAC addresses according to the number of sensor nodes, in which case the number of bits required scales according to $log_2 N$, where N is the number of sensor nodes in a network. However, the size of source and destination addresses remain significant, thus a compromise must be made between system scalability and the size of the address field.

A key observation is that MAC addresses do not need to be unique as long as a node is able to identify the immediate recipient or sender of a packet, thus facilitating the reuse of MAC addresses. Based on this observation, the authors of [1] propose a protocol that coordinates the reuse of MAC addresses. Then, in [2] the same authors propose a protocol that assigns frequently used links with a short label. Packets are then identified using a link's label rather than the sender and receiver MAC addresses, thus providing significant bit savings.

Both of the above schemes rely on control messages to carry MAC addresses or labels, referred subsequently as identifiers. From these control messages, a node is able to learn assigned identifiers in its two hops range, which a node then uses to select a free identifier at random [1] or negotiate one with its neighbors [2]. To reduce the identifier's size, [1] and [2] bias the identifier selection process and apply the Huffman algorithm. The resulting variable codewords are then used as identifiers.

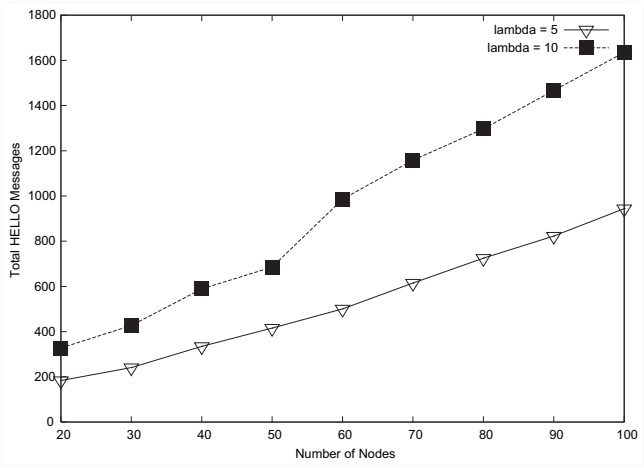

Fig. 1. Total number of HELLO messages transmitted versus network size.

We argue that the aforementioned identifier negotiation process becomes expensive as node densities rise; a critical performance issue in large scale high density sensor networks. To motivate the problem space and also provide comparisons to our analysis in Section IV, we carried out the following simulation. We implemented [2] in the *ns-2* simulator and recorded the total number of HELLO messages required for every node in a sensor network to obtain an identifier. We place N nodes randomly with transmission range R (set to 250m) in a grid of size $G \times G\ m^2$. The average connectivity can then be denoted by parameter λ:

$$\lambda = \frac{N}{G^2}\pi R^2 \tag{1}$$

Fig. 1 shows the total number of HELLO messages, averaged over 10 simulation runs, required before every node has an identifier, and Fig. 2 shows the corresponding energy expenditure; see Section IV for transmission and reception cost. Here, we assume that HELLO messages contain source and destination MAC addresses, which occupy 96 bits in total, in addition to labels a node knows of within its two hops range. From these results, it is clear that as the number of neighbors increase, more energy is required to resolve a non-conflicting address as more HELLO messages need to be exchanged. Later, we show this energy cost is removed using

Manuscript received December 7, 2005. The associate editor coordinating the review of this letter and approving it for publication was Prof. Carla-Fabiana Chiasserini.

The authors are with the Wireless Technologies Laboratory, University of Wollongong, Northfields Avenue, Australia 2522 (e-mail: {kwanwu, darrynl, rgs653}@uow.edu.au).

Digital Object Identifier 10.1109/LCOMM.2006.05005.

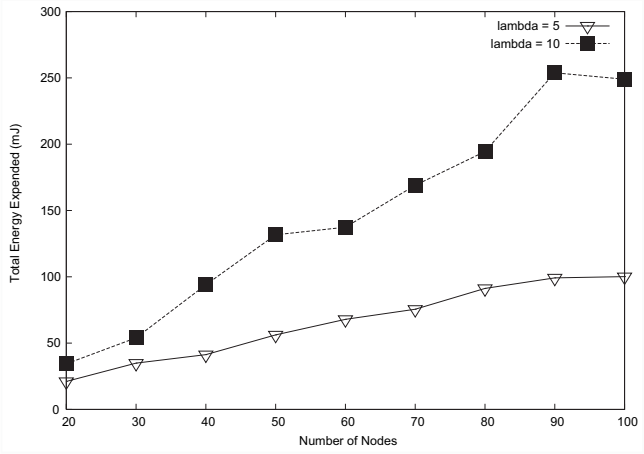

Fig. 2. Total energy expended due to HELLO messages.

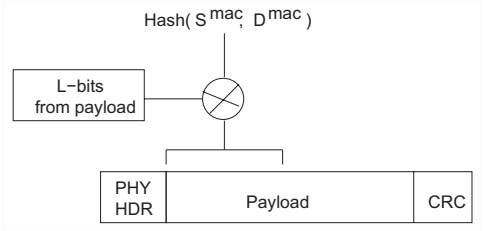

Fig. 3. Overloading the payload.

our approach.

The above results only consider static nodes and assume HELLO messages are sent periodically. However, the delays and overheads associated with setting up identifiers will undoubtedly increase when we consider nodes' sleep cycles and mobility. Apart from that, existing schemes are either sensitive to traffic load or require node density knowledge to maximize outputs from the Huffman algorithm. In the following, we outline an approach that solves the MAC address overheads by trading communication for computation cost.

II. The Approach

We propose to encode L-bits of the payload using a key derived from the packet's source and destination MAC addresses, see Fig. 3. Upon receiving a packet, a node decodes or performs an XOR operation on the first L-bits of the payload with each neighbor's key and determines whether the payload's checksum passes. If it does, the node accepts the packet.

Our approach starts as follows. Once a node detects a new neighbor and learns its MAC address, e.g., via HELLO messages, a node calculates a key, called Key_i, by hashing the neighbor's MAC address with its own. Note, the size of Key_i is at byte boundary and matches the number of bits to be encoded, i.e., the parameter L. To reduce computation, the keys are only computed once and each node stores the tuple $< Key_i, MACaddr_i >$ for each neighbor i. Note that the required memory scales according to the number of neighbors. For example if a node has 1000 neighbors, keys only occupy 1 KB of memory space, assuming 8-bit key size.

A sender encodes a packet going to node-r as follows. First, it computes the payload's CRC. The sender then looks up Key_r, which it then uses to encode the most significant L-bits of the payload. In other words, the sender does an XOR operation on the L-bits using Key_r. The value of L is either predefined or set dynamically via the physical layer header. Finally, the packet is transmitted to the receiver.

At the receiver, since the packet does not contain a MAC address, it needs to find the key and hence the neighbor that can decode the first L-bits of the packet correctly. I.e., a key that produces a payload that passes the CRC. To do this, the receiver iterates through the keys in the tuple $< Key_i, MACaddr_i >$ until the CRC passes. When this happens, the receiver accepts the packet, otherwise the packet is discarded.

An optimization to our approach is to simply use Key_i to seed the CRC computation process. For example, instead of the standardized seed value 0xFFF in CCITT-CRC16, we use Key_i. This means, at the receiver, the process becomes finding the key used to seed the CRC such that the packet's CRC passes. The main advantage of this optimization is that packets no longer has L-bits encoded with Key_i.

III. Implementation Considerations

To speed up packet reception, we recommend each node sorting its keys according to neighbors' traffic load or according to communication time. Thus, two nodes that communicate frequently can decode each other's packets quickly. Another possibility is to sort keys based on routing information. For example, sorting based on neighbors with a path to a sink node.

Key size, L, affects the amount of computation and also the number of receivers that accepts a packet due to key collision. An 8-bit key allows up to 256 nodes within a two hops radius. Therefore, as node density grows the key size must be expanded to avoid key collisions. An application designer has the option of hard-coding the key size during deployment or have the MAC re-adjust the key size after determining that the number of neighbors via HELLO messages in its two hops range exceeds 2^L. Note that, if using the said optimization, this is a matter of adjusting the CRC, e.g., CRC-8 to CRC-16, and using the corresponding key or seed size.

It must be noted that recovering the address hash with the CRC degrades the ability of the CRC to detect errors. When there are one or more errors in the L-bits block, a key belonging to a different neighbor may result in a correct checksum. In normal circumstances, when the number of neighbors, bounded by L, is much less than the CRC length, and the packet error rate (PER) is small, the likelihood of this occurring will be quite small. In the extreme case, when the length of the CRC field is equivalent to $log_2 2^L$, it can be noted that the CRC field has effectively become an address field. In other words, the CRC field is no longer able to detect errors. In this way, it can be said that, for a given CRC length, the reliability of the network has been made proportional to the local node density. Therefore, care must be taken in the application design to ensure that the communication process does not critically fail if an invalid packet is received.

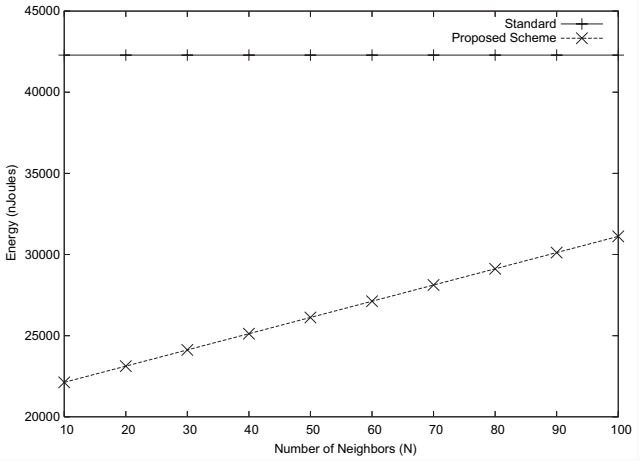

Fig. 4. Energy Consumption

IV. Analysis

In our analysis, we use the same radio and processor configurations as [2]. There are, a RFM radio operating at 2.4 kbps that consumes 0.18 μJ/bit and 0.94 μJ/bit during transmission and reception (R=20m) respectively, and a Strong ARM SA-1100 processor, which consumes 1.5 nJ/instr at 150 MIPS.

Our approach trades transmission for computation cost; a reasonable trade given the low computation cost. This trade equates to a saving of 17.28 μJ in transmission cost per packet; due to the removal of MAC addresses (96 bits in total).

However, this trade means that low computation cost is crucial since each packet needs to be parsed multiple times. For each packet, a node needs to perform an XOR operation on L-bits, compute the CRC and check its result. Clearly, the most costly operation is the CRC computation. Fortunately, fast CRC computation solutions exist; both hardware and software. For example, Williams [4] presented a CRC implementation that uses only a shift, OR and XOR operations plus a table lookup per-byte.

Fig. 4 shows the energy saving obtained using our scheme compared to the standard packet with source and destination MAC addresses. To receive a packet, we assume that the CRC computation takes on average 60 instructions and other computations such as decoding takes another 40 instructions, for a total cost of 15 nJ; packets are 96 bits in size.

As the packet size grows, the relative cost, 107.52μ Joule per packet, of MAC addresses compared to data bits becomes smaller. However, due to the small computation cost of 1.5 nJ/instr, for each additional byte, the cost only increases by 7.5 nJ, assuming five additional instructions per byte. This means, in our scheme, the packet needs to grow an additional 14.4 kB before computation cost is on par with the cost of transmitting MAC addresses. To put this in perspective, given the payload size of 16 bytes [1][2], using our approach means a 900 times increase in payload size.

Another consideration is packet reception time, i.e., time needed to process a packet before the next packet arrives. Ideally, we want this time to be shorter than the transmission time, thereby allowing ample time to receive the next incoming packet. At 2.4 kbps, a packet of size 184 bits takes 80ms to transmit, ignoring propagation time. In our scheme, if a receiver has 100 neighbors the reception time is 66.68 μs; ample time before the next packet arrives.

Compared to [2][1], our scheme does not incur the expensive process of learning and selecting identifiers. This process alone incurs $N^2 + N + 1$ control message exchanges per node; each message includes approximately N^2 to $\frac{N^2(N-1)}{2}$ identifiers, which are needed by neighboring nodes to determine a non-conflicting identifier. Moreover, when we consider mobility or adapting to changes in neighbors, the exchange of control messages becomes very costly.

We like to point out that at low node density, the setup cost incurred by [2] and [1] is minimal. However, as network density increases, the setup cost becomes expensive, as illustrated in Figs. 1 and 2. On the other hand, our approach is capable of working in both low and high densities due to it having better scalability.

V. Conclusion

Our scheme is energy efficient, adaptable to scenarios such as mobility and does not incur energy cost associated with signaling, especially as node density increases. Currently, we are optimizing the packet reception process on our sensor platform and designing our MAC around the proposed scheme.

VI. Acknowledgment

Thanks to all reviewers for their constructive feedback. This work is funded by the Australian Research Council (ARC), DP0559769.

References

[1] C. Schurgers, G. Kulkarni, and M. B. Srivastava, "Distributed assignment of encoded MAC addresses in sensor networks," in *Proc. MobiHOC 2001*, pp. 295–298.

[2] G. Kulkarni, C. Shurgers, and M. B. Srivastava, "Dynamic link labels for energy efficient MAC headers in wireless sensor networks," in *Proc. IEEE International Conference on Sensors (Sensors'02)*.

[3] J. M. Hellerstein, and P. J Haas, and H. Wang, "System architecture directions for networked sensors," in Proc. ASPLOS 2000.

[4] R. N. Williams, "A painless guide to CRC error detection algorithms, Aug. 1993; available online at: http://www.ross.net/crc/download/crc_v3.txt