IPP Hurray!

www.hurray.isep.ipp.pt

Technical Report

Improving Quality-of-Service in Wireless Sensor Networks by Mitigating "Hidden-Node Collisions"

Anis Koubâa, Ricardo Severino, Mário Alves, Eduardo Tovar

HURRAY-TR-090803

Version: 0

Date: 08-29-2009

Improving Quality-of-Service in Wireless Sensor Networks by Mitigating "Hidden-Node Collisions"

Anis Koubâa, Ricardo Severino, Mário Alves, Eduardo Tovar

IPP-HURRAY!

Polytechnic Institute of Porto (ISEP-IPP)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8340509

E-mail:

http://www.hurray.isep.ipp.pt

Abstract

Wireless Sensor Networks (WSNs) emerge as underlying infrastructures for new classes of large-scale networked embedded systems. However, WSNs system designers must fulfill the Quality-of-Service (QoS) requirements imposed by the applications (and users). Very harsh and dynamic physical environments and extremely limited energy/computing/memory/communication node resources are major obstacles for satisfying QoS metrics such as reliability, timeliness and system lifetime. The limited communication range of WSN nodes, link asymmetry and the characteristics of the physical environment lead to a major source of QoS degradation in WSNs – the "hidden node problem". In wireless contention-based Medium Access Control (MAC) protocols, when two nodes that are not visible to each other transmit to a third node that is visible to the formers, there will be a collision – called hidden-node or blind collision. This problem greatly impacts network throughput, energy-efficiency and message transfer delays, and the problem dramatically increases with the number of nodes. This paper proposes H-NAMe, a very simple yet extremely efficient Hidden-Node Avoidance Mechanism for WSNs. H-NAMe relies on a grouping strategy that splits each cluster of a WSN into disjoint groups of non-hidden nodes that scales to multiple clusters via a cluster grouping strategy that guarantees no interference between overlapping clusters. Importantly, H-NAMe is instantiated in IEEE 802.15.4/ZigBee, which currently are the most widespread communication technologies for WSNs, with only minor add-ons and ensuring backward compatibility with their protocols standards. H-NAMe was implemented and exhaustively tested using an experimental test-bed based on "off-the-shelf" technology, showing that it increases network throughput and transmission success probability up to twice the values obtained without H-NAMe. H-NAMe effectiveness was also demonstrated in a target tracking application with mobile robots over a WSN deployment.

Improving Quality-of-Service in Wireless Sensor Networks by Mitigating "Hidden-Node Collisions"

Anis Koubâa, Ricardo Severino, Mário Alves, and Eduardo Tovar

***Abstract*—Wireless sensor networks (WSNs) emerge as underlying infrastructures for new classes of large-scale networked embedded systems. However, WSNs system designers must fulfill the quality-of-service (QoS) requirements imposed by the applications (and users). Very harsh and dynamic physical environments and extremely limited energy/computing/memory/communication node resources are major obstacles for satisfying QoS metrics such as reliability, timeliness, and system lifetime. The limited communication range of WSN nodes, link asymmetry, and the characteristics of the physical environment lead to a major source of QoS degradation in WSNs—the "hidden node problem." In wireless contention-based medium access control (MAC) protocols, when two nodes that are not visible to each other transmit to a third node that is visible to the former, there will be a collision—called hidden-node or blind collision. This problem greatly impacts network throughput, energy-efficiency and message transfer delays, and the problem dramatically increases with the number of nodes. This paper proposes H-NAMe, a very simple yet extremely efficient hidden-node avoidance mechanism for WSNs. H-NAMe relies on a grouping strategy that splits each cluster of a WSN into disjoint groups of non-hidden nodes that scales to multiple clusters via a cluster grouping strategy that guarantees no interference between overlapping clusters. Importantly, H-NAMe is instantiated in IEEE 802.15.4/ZigBee, which currently are the most widespread communication technologies for WSNs, with only minor add-ons and ensuring backward compatibility with their protocols standards. H-NAMe was implemented and exhaustively tested using an experimental test-bed based on "off-the-shelf" technology, showing that it increases network throughput and transmission success probability up to twice the values obtained without H-NAMe. H-NAMe effectiveness was also demonstrated in a target tracking application with mobile robots over a WSN deployment.**

***Index Terms*—Energy-efficiency, hidden-node problem, networked embedded systems, quality-of-service (QoS), timing performance, wireless sensor networks (WSNs).**

I. Introduction

A. Research context

INDUSTRIAL applications such as factory automation, process control, quality control, or smart energy can greatly benefit from or even impose the use of wireless/mobile communication capabilities. Due to the growing tendency for continuously monitoring/controlling everything, everywhere, computing systems tend to be ubiquitous, largely distributed and tightly embedded in their physical environments [1]. To be cost-effective, these systems must be mainly composed of tiny resource-constrained embedded devices with wireless communication capabilities, forming Wireless Sensor/Actuator Networks, usually simply referred as Wireless Sensor Networks (WSNs).

WSN applications can be of many different types and can impose different quality-of-service (QoS) requirements [2], e.g., an air quality monitoring application gathering air parameters measurements has less stringent timing requirements than a mobile robot navigation application. However, all WSN applications benefit from higher network throughput, lower message delay, and longer system lifetime.

The provision of QoS in WSNs is very challenging due to two main problems, though: 1) the usually severe limitations of WSN nodes, such as the ones related to their energy, computational and communication capabilities, in addition to the large-scale nature of WSNs and 2) most QoS properties are interdependent, in a way that improving one of them may degrade others, e.g., increasing throughput (by increasing WSN nodes duty-cycle or increasing bit rate) will decrease system lifetime or providing time-bounded (real-time) communications may imply worst-case resource reservation, leading to lower network throughput and lifetime. These negative facts force system designers to try to achieve the best tradeoffs between QoS metrics. In this paper, a mechanism that enables to improve several QoS properties of a WSN system at the same time is proposed, as it will be presented hereafter.

B. Problem Statement

Most WSNs rely on contention-based medium access control (MAC) protocols such as the carrier sense multiple access (CSMA) family. The problem with this type of MACs is that network performance degrades drastically with the number of nodes and with the traffic load, due to the increasing number of message collisions. This performance degradation is even more acute due to the impact of the hidden-node problem, which is caused by hidden-node collisions. A hidden-node (or "blind") collision occurs when two wireless nodes (e.g., nodes A and B, in Fig. 1) that cannot hear each other (due to limited transmission range, asymmetric links, presence of obstacles, etc.), communicate with a commonly visible node (the node between A and B, in Fig. 1) during a given time interval.

Hidden-node collisions affect four QoS metrics.

1) *Throughput*, which denotes the amount of traffic successfully received by a destination node and that decreases due to additional blind collisions.

Manuscript received December 05, 2008; revised April 23, 2009 and June 17, 2009. Current version published August 07, 2009. Paper no. TII-08-12-0196.

A. Koubâa is with the CISTER Research Unit, Polytechnic Institute of Porto (ISEP-IPP), 431 Porto, Portugal. He is also with the Department of Computer Science, Al-Imam Muhammad Ibn Saud University, 11681 Riyadh, Saudi Arabia (e-mail: aska@isep.ipp.pt).

R. Severino, M. Alves, and E. Tovar are with the CISTER Research Unit, Polytechnic Institute of Porto (ISEP-IPP), 431 Porto, Portugal (e-mail: rars@isep.ipp.pt; mjf@isep.ipp.pt; emt@isep.ipp.pt).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2009.2026643

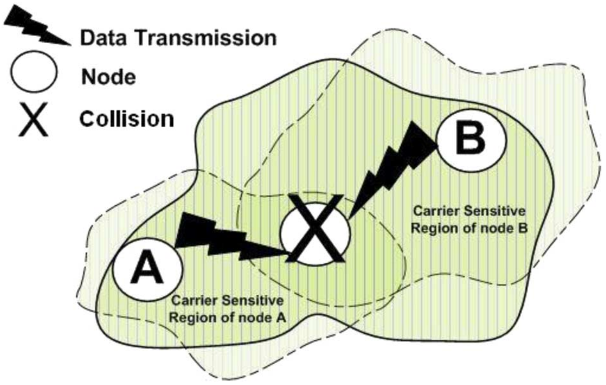

Data Transmission
Node
Collision
B
A
Carrier Sensitive Region of node B
Carrier Sensitive Region of node A

Fig. 1. A hidden-node collision.

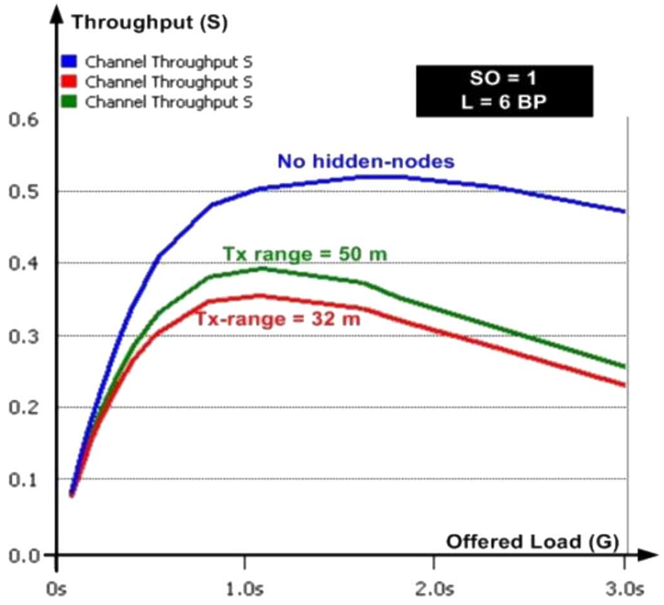


Fig. 2. Hidden-node impact on network throughput.

2) *Transfer delay*, which represents the time duration from the generation of a message until its correct reception by the destination node, and increases due to message retransmissions due to collisions.
3) *Energy-efficiency* that decreases since each collision causes a new retransmission.
4) *Reliability*, since applications may abort message transmission after a number of retransmissions.

Fig. 2 presents an illustrative example of the negative impact of the hidden-node problem, based on our OPNET [3] simulation model [4] for the IEEE 802.15.4 protocol [5]. The simulation scenario encompasses a IEEE 802.15.4 star network spanning over a square surface $(100 \times 100 \text{ m}^2)$ with 100 nodes and where traffic generation followed a Poisson distribution. The throughput performance is shown for different transmission ranges, obtained by setting different receiver sensitivity levels at the nodes. Throughput degradation results from higher hidden-node collision probability when decreasing the transmission range.

In the literature, several mechanisms (outlined in Section II) have been proposed to mitigate the impact of the hidden-node problem in wireless networks. However, to the best of our knowledge, no effective solution to this problem was proposed so far for WSNs. In this context, this paper proposes an efficient solution to the hidden-node problem in synchronized cluster-based WSNs. Our approach is called H-NAMe and is based on a grouping strategy that splits each cluster of a WSN into disjoint groups of non-hidden nodes. It then scales to multiple clusters via a cluster grouping strategy that guarantees no transmission interference between overlapping clusters.

Importantly, neither IEEE 802.15.4 [5] nor ZigBee [28], two of the most prominent communication technologies for WSNs available today [6], support a hidden-node avoidance mechanism. This leads to a significant QoS degradation, as already referred and can be intuitively inferred from Fig. 2. In this line, H-NAMe was applied to the IEEE 802.15.4/ZigBee protocols, requiring only minor add-ons and ensuring backward compatibility. We devised a test-bed based on Commercial-Off-The-Shelf (COTS) technologies and performed an extensive set of experiments that enabled to prove that H-NAMe significantly increases QoS. Notably, network throughput and transmission success probability can reach 100% increase, against the native IEEE 802.15.4 protocol. The integration of the H-NAMe mechanism in IEEE 802.15.4/ZigBee may thus be relevant for leveraging the use of these protocols in WSNs for applications with more stringent QoS requirements, such as in industrial environments.

C. Contributions

The fundamental problem of hidden-nodes has been addressed in some previous works and several techniques have been proposed to overcome it, as presented in Section II. Our objective in this paper is not to find a new theoretical solution to the hidden-node problem. The main objective is to devise a mechanism that uses an existing paradigm, that is the grouping paradigm in a way that: 1) it resolves the hidden-node problem in IEEE 802.15.4/ZigBee-like multiple-cluster networks; 2) it can be implemented and integrated into the IEEE 802.15.4/ZigBee standard protocol stack; and 3) it maintains backward compatibility with these protocol standards, i.e., a fully transparent interoperability between devices that do not implement H-NAME and devices that do.

To the best of our knowledge, our paper is the first work that addresses these challenges and provides an effective solution to all of them. We also prove the validity of our protocol through extensive experimentation.

The main contributions of this paper are the following.

1) We propose H-NAMe, a simple and efficient mechanism for solving the hidden-node problem in synchronized single or multiple cluster WSNs based on the node grouping approach (Section III). We show that H-NAMe is very easy to implement, in contrast to the grouping mechanism proposed in [28].
2) We show how to incorporate H-NAMe in the IEEE 802.15.4/ZigBee protocols with minor add-ons and ensuring backward compatibility with the default specifications (Section IV).
3) We evaluate the performance of the H-NAMe mechanism through an experimental test-bed, showing significant QoS improvements (Section V).

4) We assess the impact of the hidden-node problem in a target tracking application (Section VI) and demonstrate the effectiveness of the H-NAMe mechanism.

II. Related Work

The hidden-node problem is known to be a serious source of performance degradation in wireless communication networks. In [7] and [8], the authors derived a mathematical analysis based on queueing theory and quantified the impact of the hidden-node problem on the performance of small-scale linear wireless networks. Many research works have addressed solutions for eliminating or reducing the impact of the hidden-node problem in wireless networks, roughly categorized as: 1) busy tone mechanisms; 2) Request-To-Send/Clear-To-Send (RTS/CTS) mechanisms; 3) carrier-sense tuning mechanisms; 4) interference cancellation mechanisms; and 5) node grouping mechanisms. These are briefly described next.

A. Busy Tone Mechanisms

In this approach, a node that is currently hearing an ongoing transmission sends a busy tone to its neighbors (on a narrow-band radio channel) for preventing them from transmitting during channel use. This mechanism was early introduced in [9], providing a solution, called the busy tone multiple access (BTMA), for a star network with a base station. Collisions are avoided by inhibiting all nodes within a $2R$ radius (R is the range of the transmitted signal) from the source node (SN), with an out of band tone. An extension of this mechanism for a distributed peer-to-peer network was proposed in [10] known as receiver-initiated busy tone multiple access (RI-BTMA) and in [11] as dual busy tone multiple access (DBTMA). RI-BTMA, though initially proposed as a modification to BTMA to improve efficiency, was probably the first protocol to use the fact that the destination node (DN) is the only node that can identify if a collision is occurring (or not). An improvement to this mechanism was proposed in [12]—wireless collision detect (WCD), also based on a slotted operation mode. Recently, asynchronous wireless collision detection with acknowledgement (AWCD/ACK) was proposed in [13].

The limitation of this kind of mechanisms is the need of a separate radio channel, leading to additional hardware complexity and cost and eventually to additional energy consumption (more hardware must be powered), thus reducing the cost-effectiveness and energy-efficiency of WSNs.

B. Request-To-Send/Clear-To-Send (RTS/CTS) Mechanisms

The idea of making a radio channel reservation around the sender and the receiver through a control-signal handshake mechanism was first proposed in [14]—*split-channel reservation multiple access* (SRMA). The RTS/CTS approach builds on this concept and was introduced in the MACA protocol [15]. The channel reservation is initiated by the sender, which sends an RTS frame and waits for a CTS frame from the destination, before starting the effective transmission. Several refinements were proposed, including MACAW [16], the IEEE 802.11 (DCF) [17], and FAMA [18]. Recently, the double sense multiple access (DSMA) mechanism was proposed in [19], joining the busy tone approach with the RTS/CTS mechanism, using two time-slotted channels.

RTS/CTS-based methods are particularly unsuitable for WSNs (as stated in [20]), mainly due to the following reasons: 1) data frames in WSNs are typically as small as RTS/CTS frames, leading to the same collision probability; 2) the RTS/CTS message exchanges are energy consuming for both sender and receiver; 3) the use of RTS/CTS is only limited to unicast transmissions and does not extend to broadcasts; and 4) it may lead to extra throughput degradation due to the *exposed-node* problem [15].

C. Carrier-Sense Tuning Mechanisms

The idea consists in tuning the receiver sensitivity threshold of the transceiver, which represents the minimum energy level that indicates channel activity, to have extended radio coverage. Higher receiver sensitivities enable a node to detect the transmissions of nodes farther away, thus allowing it to defer its transmission (to avoid overlapping). Many works analyzed the impact of carrier sensing on system performance. This technique was analyzed in [21] to study the effects of carrier sensing range on the performance of the IEEE 802.11 MAC protocol. A similar study was conducted in [22]. More recently, in [23], the authors carried out a thorough study to find an optimal carrier sensing threshold, given multiple network topologies. Finally, in [24], the authors proposed two distributed adaptive power control algorithms that aim at minimizing mutual interferences among links, while avoiding hidden nodes and ensuring a good tradeoff between network capacity and fairness.

One of the limitations of carrier sense tuning mechanisms is that it assumes homogenous radio channels, whereas in reality, hidden-node situations can arise from obstacles and asymmetric links, which may be typical for most WSN applications, particularly in industrial environments. Importantly, increasing receiver sensitivity directly leads to more energy consumption, which might not be acceptable for most WSN applications. Even in situations where energy consumption may not be a major concern, it is not possible to indefinitely increase the carrier sense range due to hardware/physical limitations.

D. Interference Cancellation

The idea of interference cancellation is related to information theory and consists in decoding collisions. Several previous works have investigated the use of interference cancellation in IEEE 802.11 networks [25], [26], and [27]. In [25] and [26], the authors have built a ZigBee prototype of successive interference cancellation, which is only effective when the colliding senders transmit at a bit rate significantly lower than allowed by their respective signal-to-noise ratios (SNRs) and code redundancy. In [27], the authors have overcome this problem and proposed ZigZag, a mechanism implemented at an IEEE 802.11 receiver that increases resilience to hidden-node collisions. The advantage of this mechanism is that it does not impose significant changes to the IEEE 802.11 protocol and is backward compatible with the standard. The main idea is based on decoding interference-free chunks of packets assuming that two consecutive collisions have two different time offsets. More specifically, the objective of the decoding algorithm is to find a collision free chunk, which is used to start the decoding process and extract the information from subsequent collided chunks. The process is iterative and at each stage it produces a new interference-free chunk, decodable using standard decoders.

E. Node Grouping Mechanisms

Node grouping consists in grouping nodes according to their hidden-node relationship, such that each group contains nodes that are "visible" (bidirectional connectivity) to each other. Then, these groups are scheduled to communicate in nonoverlapping time periods to avoid hidden-node collisions. Such a grouping strategy is particularly suitable for star-based topologies with one base station. In that direction, a grouping strategy was introduced in [28] to solve the hidden-node problem in IEEE 802.15.4/ZigBee star networks (formed by the ZigBee Coordinator—ZC—and several nodes within its radio coverage). In [28], the grouping strategy assumes that the ZC can distinguish a hidden-node collision from a normal collision based on the time when the collision occurs. Thus, when the ZC detects a hidden-node collision, it starts the hidden-node information collection process, by triggering a polling mechanism. At the end of the polling process, all nodes report their hidden-node information to the ZC, which executes a group assignment algorithm based on the hidden-node relationship reported by the nodes. The algorithm is shown to have a complexity of $O(N^2)$, where N is the number of nodes. After assigning each node to a group, the ZC allocates to each group a certain time window inside the superframe (slotted CSMA/CA is used). The grouping process is then repeated each time the ZC detects a hidden-node collision.

Our paper proposes an efficient, practical and scalable approach for synchronized cluster-based WSNs—H-NAMe. Importantly, we show how to integrate our approach in the IEEE 802.15.4/ZigBee protocols with only minor add-ons and fully respecting backward compatibility. Our work differs from [28] in many aspects. First, H-NAMe requires no hidden-node detection since it relies on a *proactive* approach (grouping strategy is node-initiated) rather than a *reactive* approach to the hidden-node problem. Second, the complexity of the group join process was drastically reduced, to $O(N)$. The grouping process in [28] is based on polling all the nodes in the coverage of ZC each time a hidden-node collision occurs, resulting in a group assignment complexity of $O(N^2)$ in each grouping process, where N is the number of nodes. This results in significant network inaccessibility times and energy consumption during the polling process. In our approach, for each group assignment, only the requesting node and its neighbors will be subject to the *group join* procedure and not all cluster nodes, resulting in a simpler, more energy-efficient and scalable ($\sim O(N)$) mechanism, especially appealing for more densely deployed clusters. Third, it is shown how H-NAMe can scale to multiple cluster WSNs. Finally, the feasibility of our proposal is demonstrated through an experimental test-bed, whereas the one in [28] relies only on simulation. This is relevant, because we believe an eventual implementation of [28] would not be straightforward, since it requires a mechanism for detecting and interpreting collisions, which might be very difficult to achieve, and implies a non-negligible change to the IEEE 802.15.4 physical layer.

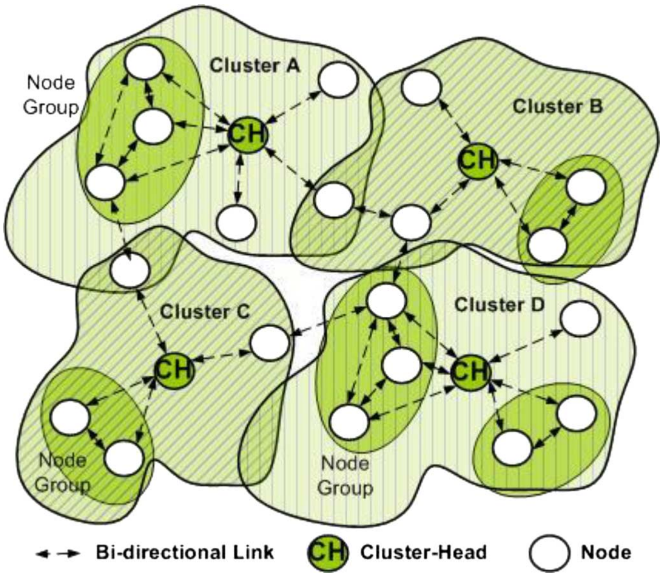


Fig. 3. Network model.

III. The H-NAMe Mechanism

A. System Model

A multiple cluster wireless network where in each cluster there is at least one node with bidirectional radio connectivity with all the other cluster nodes (Fig. 3) is considered. This node is denoted as cluster-head (CH). At least the CH must support routing capabilities, for guaranteeing total interconnectivity between cluster nodes.

Nodes are assumed to contend for medium access during a contention access period (CAP), using a contention-based MAC (e.g., CSMA family). A synchronization service must exist to assure synchronization services to all network nodes, either in a centralized (e.g., GPS, RF pulse) or distributed fashion (e.g., IEEE 802.11 TSF, ZigBee). We also assume that there is interconnectivity between all network clusters (e.g., mesh or tree-like topology). Note that although our current aim is to use the H-NAMe mechanism in the IEEE 802.15.4/ZigBee protocols, the system model is generic enough to enable its application to other wireless communication protocols (e.g., IEEE 802.11).

In what follows, we start by proposing the H-NAMe intracluster node grouping strategy (Section III-B) and then, in Section III-C, a strategy to ensure the scalability to multiple cluster networks.

B. Intracluster Grouping

Initially, all nodes in each cluster share the same CAP, thus are prone to hidden-node collisions. The H-NAMe mechanism subdivides each cluster into node groups (where all nodes have bidirectional connectivity) and assigns a different time window to each group, during the CAP. The set of time windows assigned to node groups' transmissions is defined as group access period (GAP), and must be smaller or equal to the CAP. In this way, nodes belonging to groups can transmit without the risk of causing hidden-node collisions. The H-NAMe intracluster grouping strategy comprises four steps, presented hereafter and illustrated in Figs. 4 and 5. We start by assuming that there is

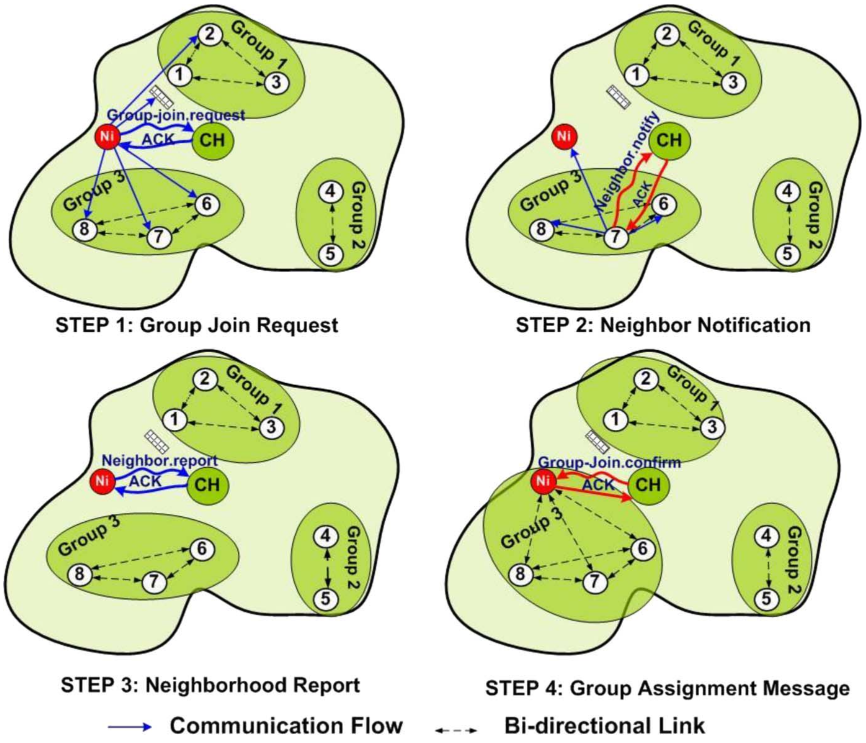


Fig. 4. Intracluster grouping mechanism.

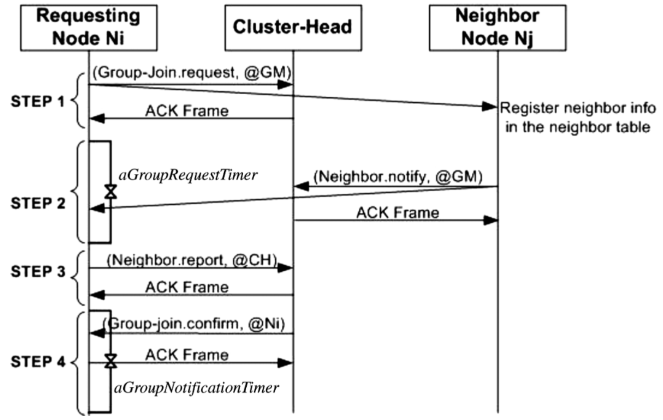


Fig. 5. Intracluster grouping message sequence chart.

no interference with adjacent clusters, since that might also instigate hidden-node collisions.

Step 1—Group Join Request: Let us consider a node N_i that wants to avoid hidden-node collisions. Node N_i sends a *Group-join.request* message to its CH, using a specific broadcast address referred to as *group management address* $@_{\mathrm{GM}}$ in the destination address field. $@_{\mathrm{GM}}$ is defined as an *intracluster broadcast address*, which must be acknowledged by the CH (in contrast to the typical broadcast address). Obviously, the acknowledgment message (ACK) will be received by all cluster nodes, since the CH is assumed to have bidirectional links with all of them.

Such an acknowledged broadcast transmission ensures that the broadcasted message is correctly received by all the neighbors of the broadcasting node (recalling that no intercluster interference is assumed). In fact, if any collision occurs inside the cluster during the transmission of the broadcast message, then the CH will certainly be affected by this collision since it is in direct visibility with all nodes in its cluster. If no collision occurs, then the broadcast message will be correctly received by all nodes and acknowledged by the CH.

Hence, since the *Group-join.request* message is sent using the group management address $@_{\mathrm{GM}}$, CH sends back an ACK frame to N_i notifying it of the correct reception of the group join request.

On the other side, all cluster nodes in the transmission range of N_i (thus received the *Group-join.request* message) and that already belong to a group, check if they have N_i already registered as a neighbor node in their *Neighbor Table*. We assume that the Neighbor Table is created and updated by each node during network setup and runtime phases. The Neighbor Table stores the addresses of neighbor nodes and the link symmetry information, which specifies if the link with a corresponding neighbor is bidirectional or not. If a node hears the *Group-join.request* message and does not belong to any group (it is transmitting in the CAP, thus not in the GAP), then it simply ignores the message. On the other hand, if a node N_j is already in a group and hears the join message, then it records the information about N_i in its Neighbor Table, if it is not registered yet, and will update the link symmetry with direction $N_i \rightarrow N_j$.

Step status: At the end of this step, each node in the transmission range of N_i knows that node N_i is asking for joining a group and registers the neighborhood information of N_i. This only ensures a link direction from N_i to this set of nodes. The link symmetry verification is the purpose of the next step.

Step 2—Neighbor Notification: After receiving the ACK frame of its *Group-join.request* message, node N_i triggers the *aGroupRequestTimer* timer, during which it waits for neighbor notification messages from its neighbors that heard its request to join a group and that already belong to a group. Choosing the optimal duration of this timer is out of the scope of this paper, but it must be large enough to permit all neighbors to send their notification.

During that time period, all nodes that have heard the join request and that already belong to a group must initiate a *Neighbor.notify* message to inform node N_i that they have heard its request. One option is that a node N_j directly sends the *Neighbor.notify* message to node N_i with an ACK request. The drawback of this alternative is that node N_j cannot know when its *Neighbor.notify* message fails to reach N_i (i.e., ACK frame not received), whether the lost message is due a collision or to the nonvisibility of N_i. No clear decision can be taken in that case. A better alternative is that node N_j sends the *Neighbor.notify* message using the group management address $@_{\mathrm{GM}}$ in the destination address field. As previously mentioned, the correct reception of the *Neighbor.notify* message by the CH followed by an ACK frame means that this message is not corrupted by any collision and is correctly received by all nodes in the transmission range of N_j. Particularly, node N_i will correctly receive the neighbor notification message if it is reachable from node N_j; otherwise, the link between N_i and N_j is unidirectional (direction $N_i \rightarrow N_j$). If N_i receives the *Neighbor.notify* message from N_j, then it updates its Neighbor Table by adding as a new entry the information on N_j with *Link Symmetry* set to bidirectional ($N_i \leftrightarrow N_j$), if this information has not been recorded yet. If N_j has already been registered as a neighbor node, N_i must be sure to set the *Link Symmetry* property to bidirectional. This procedure is executed by all nodes responding to the *Group-join.request* message during the timer period *aGroupRequestTimer*.

Step status: At the end of this step, the requesting node N_i will have the information on all bidirectional neighbors that have already been assigned to groups. Since N_i does not know the number of nodes in each group, it cannot decide alone which group it will join. The group assignment is the purpose of the next steps.

Step 3—Neighbor Information Report: The CH is assumed to be the central node that manages all the groups in its cluster. Thus, CH has a full knowledge of the groups and their organization. For that reason, after the expiration of the *aGroupRequestTimer* timer, node N_i sends the *Neighbor.report* message, which contains the list of its neighbor nodes (that have been collected during the previous step), to its CH (using the CH address $@_{\mathrm{CH}}$ as a destination address). The CH must send back an ACK frame to confirm the reception. Then, node N_i waits for a notification from CH that decides whether N_i will be assigned to a group or not. CH must send the group assignment notification before the expiration of a time period equal to *aGroupNotificationTimer*. If the timer expires, node N_i concludes that its group join request has failed and may retry to join a group later.

Step status: At the end of this step, N_i will be waiting for the group assignment confirmation message from CH, which tries to assign N_i to a group based on its neighbor information report and the organization of the groups in its cluster. The group assignment procedure and notification is presented in the next step.

Step 4—Group Assignment Procedure: The CH maintains the list of existing groups. After receiving from node N_i the *Neighbor.report* message containing the list of its bidirectional neighbors, CH starts the group assignment procedure to potentially assign N_i to a given group, according to its neighborhood list and available resources. In each cluster, the number of groups must be kept as low as possible in order to reduce the number of state information that needs to be managed by the CH.

```
Group Assignment Algorithm
int aMaxGroupNumber;  // maximum number of groups
                         in a cluster
Type Group;
Group G;                        // list of all groups
G[1]..G[aMaxGroupNumber]
|G[i]| = number of elements in group G[i]
Type Neighbor_List;       // {Np .. Nq}= Neighbor List of
                             the requesting Node N
int Count [|G[i]|] = {0, 0, .., 0};  // Number of nodes in
Neighbor List that belongs to the group G[i]

int grp_nbr;   // the current number of groups managed
by CH
// group_index function returns the group index of the
node NL[i]
function int group_index(Neighbor_List NL, int i)
//the group assignment function.
int group_assign (Neighbor_List NL, Group G, int
grp_nbr) {
   int res = 0;
   int index = 0;
   while ((res = = 0) and (index < |NL|)) {
        if (++Count[group_index (NL, index)] = =
                             |G[group_index          (NL,
index++])|)
               res = group_index (NL, --index); break;
   }
   if (res = = 0) {   //that means that no group is found
        if (grp_nbr = = aMaxGroupNumber) return (res)
           else return (++grp_nbr);
   }
  else return (res);
}
```

Fig. 6. Group assignment algorithm.

We impose that the number of groups inside each cluster must not exceed *aMaxGroupNumber*, which should be equal to six, by default (the reader is referred to [40] for further intuition). The group assignment algorithm is presented in Fig. 6.

Upon reception of the *Neighbor.report* message, the CH checks the neighbor list of the requesting node N_i. If there is a group whose (all) nodes are neighbors of node N_i, then N_i will be associated to that group. The CH runs the following algorithm (as in Fig. 6). For each neighbor node N_j in the list, the CH increments $Count\ [group\_index\ (N_j)]$, which denotes the number of neighbor nodes of N_i that belong to the group of the currently selected neighbor N_j. Note that $group\_index\ (N_j)$ denotes the index of the group of node N_j. If this number is equal to the actual number of nodes of the latter group, it results that all nodes in this group are neighbors of node N_i. Thus, N_i can be assigned to this group since it is visible to all its nodes.

If the list of neighbors is run through without satisfying such a condition, the CH will create a new group for N_i if the number of groups is lower than *aMaxGroupNumber*; otherwise, the *Group-join.request* message of N_i will be considered as failed. So it must transmit during the CAP (not in the GAP), and may retry a new group join request later.

At the end of the group assignment process, CH sends a *Group-join.notify* message to node N_i to notify it about the result of its group join request. If the requesting node is assigned a group, then it will be allowed to contend for medium access during the time period reserved for the group, which is called GAP. This information on the time period allocated to the group is retrieved in the subsequent frames sent by the CH.

Importantly, the complexity of the algorithm (Fig. 6) for assigning a group to a node depends on the number of neighbors of this node. In any case, it is smaller than $O(N)$, where N is the number of nodes in the cluster, thus has significantly lower complexity than the $O(N^2)$ complexity of the algorithm for group assignment proposed in [28]. Moreover, in that proposal each new node that enters the network is unaware of the existing groups and will cause a hidden-node collision, after which the groups are reconstructed. In our mechanism, a node is not allowed to transmit during the time period allocated to groups (only being able to communicate during the CAP) until it is assigned to a given group.

Group load-balancing: Note that the algorithm presented in Fig. 6 stops when a first group of non-hidden nodes is found for the requesting node. However, a requesting node can be in the range of two different groups, i.e., all nodes in two separate groups are visible to the requesting node. In this case, one possible criterion is to insert the requesting node into the group with the smallest number of nodes, for maintaining load-balancing between the different groups. For that purpose, the algorithm should go through all the elements of the neighbor list and determine the list of groups that satisfy the condition in lines 18 and 19 of the algorithm (Fig. 6). In this case, if more than one group satisfies this condition, N_i will be inserted in the group with the smallest number of nodes.

Bandwidth allocation: The time-duration of each group in the GAP can be tuned by the CH to improve the mechanism efficiency. This can be done via different strategies, e.g.: i) evenly for all the node groups; ii) proportionally to the number of nodes in each group; iii) proportionally to each group's traffic requirements. How to perform this assignment is not tackled in this paper.

One interesting feature of the H-NAMe mechanism is that it is intrinsically resilient to node failures. If a group-join request fails, the requesting node will not be assigned to any group or may retry to join a group later. For instance, the node can keep retrying to join a group for a predetermined number of attempts until the group-join request succeeds and then competes for medium access within its assigned contention-access group (CAP). If the group-join failure persists, the requesting node withdraws from the group-join process and limits its communication to the CAP only, and thus will not affect the groups already formed.

C. Scaling H-NAMe to Multiple-Cluster Networks

Solving the hidden-node problem in multiple-cluster networks involves greater complexity due to intercluster interference. The assumption that there is no interference from other clusters made before is no longer valid. Hence, even if non-hidden node groups are formed inside all clusters, there is no guarantee that hidden-node collisions will not occur, since groups in one cluster are unaware of groups in adjacent clusters.

The most straightforward strategy for completely avoiding the intercluster hidden-node problem is to reserve an exclusive time window for each cluster. However, this strategy is definitely not adequate for large-scale WSNs, where the number of clusters may be significantly high and most of them nonoverlapping (in terms of radio interference range).

Our approach consists in defining another level of grouping by creating distinct groups of clusters whose nodes are allowed to communicate during the same time window. Therefore, each cluster group will be assigned a time window, during which each cluster in the cluster group will manage its own group access period (GAP), according to the intracluster mechanism presented in Section III-B.

The cluster grouping concept is illustrated in Fig. 3. Clusters A and B have overlapping radio coverage, which can lead to inter-cluster interference and thus to hidden-node collisions. Thus, they will be assigned to different cluster groups that are active in different time windows. The same applies for cluster pairs (C, D), (A, C) and (B, D). Therefore, our cluster grouping mechanism forms two cluster groups: Group 1, which comprises clusters A and D, and Group 2 containing clusters B and C.

The challenge is to find the optimal cluster grouping strategy that ensures the minimum number of cluster groups. We define a cluster group as a set of clusters whose nodes are allowed to transmit at the same time without interference.

Cluster grouping and time window scheduling strategies were proposed and effectively implemented and validated in [29], for engineering ZigBee cluster-tree WSNs. A more detailed description of the cluster grouping mechanism can be found in [40]. A grouping criterion and a graph coloring algorithm for an efficient scheduling of the cluster groups activity are proposed.

IV. Instantiating H-NAMe in IEEE 802.15.4/ZigBee

This section elaborates on how to instantiate the H-NAMe mechanism in the IEEE 802.15.4/ZigBee protocols, namely addressing synchronized (beacon-enabled) cluster-tree WSNs. This network model is scalable, enables energy-efficient and real-time communications and fits into the H-NAMe network model. Importantly, the H-NAMe mechanism is implemented in a "backward compatible" way, i.e., such that "traditional" (not implementing H-NAME) and "new" (implementing H-NAMe) WSN nodes can coexist and intercommunicate in the same WSN.

A. IEEE 8021.5.4/ZigBee Overview

IEEE 802.15.4 [31] and ZigBee [32], particularly the synchronized cluster-tree network model, emerge as potential solutions for industrial WSNs, since they enable to fulfill QoS requirements such as energy-efficiency (dynamically adjustable duty-cycle in a per-cluster basis) and timeliness (best effort/guaranteed traffic differentiation and deterministic tree-routing) [33].

The IEEE 802.15.4 MAC protocol supports two operational modes that may be selected by the ZigBee Coordinator (ZC), which identifies and manages the whole WSN: i) the non

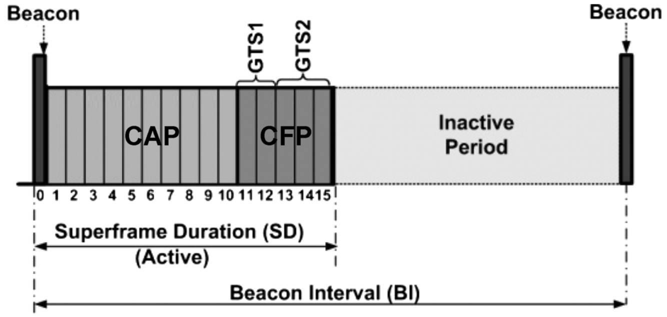


Fig. 7. IEEE 802.15.4 superframe structure.

beacon-enabled mode, in which the MAC is simply ruled by nonslotted carrier sense multiple access with collision avoidance (CSMA/CA); and ii) the beacon-enabled mode, in which beacons are periodically sent by the ZC for synchronization and network management purposes.

In the beacon-enabled mode, the ZC defines a superframe structure (Fig. 7), which is constructed based on the Beacon Interval (BI), which defines the time between two consecutive beacon frames, and on the Superframe Duration (SD), which defines the active portion in the BI, and is divided into 16 equally-sized time slots, during which frame transmissions are allowed. Optionally, an inactive period is defined if $BI > SD$. During the inactive period (if it exists), all nodes may enter in a sleep mode (to save energy). BI and SD are determined by two parameters, the Beacon Order (BO) and the Superframe Order (SO), respectively, as follows:

$$\left.\begin{array}{l} BI = aBaseSuperframeDuration \cdot 2^{BO} \\ SD = aBaseSuperframeDuration \cdot 2^{SO} \end{array}\right\} \quad for\ 0 \leq SO \leq BO \leq 14 \quad (1)$$

where $aBaseSuperframeDuration = 15.36$ ms (assuming 250 kb/s in the 2.4 GHz frequency band) denotes the minimum superframe duration, corresponding to $SO = 0$.

During the SD, nodes compete for medium access using slotted CSMA/CA in the CAP. For time-sensitive applications, IEEE 802.15.4 enables the definition of a contention-free period (CFP) within the SD, by the allocation of guaranteed time slots (GTSs).

As can be observed in Fig. 7, low duty-cycles are achieved by setting small values of the superframe order (SO) as compared to the beacon order (BO), leading to longer sleeping (inactive) periods.

ZigBee defines network and application layer services on top of the IEEE 802.15.4 protocol. In the cluster-tree model, all nodes are organized in a parent-child relationship, network synchronization is achieved through a distributed beacon transmission mechanism and a deterministic tree routing mechanism is used.

A ZigBee network is composed of three device types: (i) the ZigBee Coordinator (ZC), which identifies the network and provides synchronization services through the transmission of beacon frames containing the identification of the PAN and other relevant information; ii) the ZigBee Router (ZR), which has the same functionalities as the ZC with the exception that it does not create its own PAN—a ZR must be associated to the ZC or to another ZR, providing local synchronization to its cluster (child) nodes via beacon frame transmissions; and (iii) the ZigBee End-Device (ZED), which neither has coordination nor routing functionalities and is associated to the ZC or to a ZR.

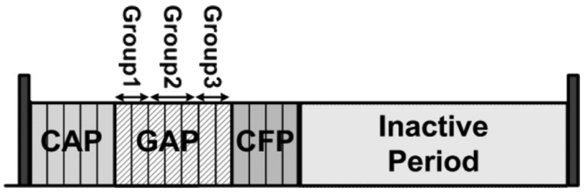


Fig. 8. CAP, GAP, and CFP in the superframe.

B. Integrating H-NAMe in IEEE 802.15.4

Basically, the idea is that each node group (resulting from the H-NAMe mechanism) will be allocated a time window in each superframe duration. The idea is to use part of the CAP for the GAP, as illustrated in Fig. 8. Note that a minimum duration of 440 symbols must be guaranteed for the CAP in each superframe [5].

In our intracluster grouping strategy, a node that has been assigned a group will track the beacon frame for information related to the time window allocated to its group, and will contend for medium access during that period with the other nodes of the same group. We propose the *GAP Specification* field illustrated in Fig. 9 to be embedded in the beacon frame (such a specification is missing in [28]).

The GAP is specified by the *Group ID* field that identifies the node group (up to eight groups per cluster can be defined). The time window in the superframe is specified by a given number of backoff periods (BPs). A practical problem is that the number of a BP in a superframe may be quite large for high superframe orders (up to 16 time slots $*2^{16}$ BP/time slot), which requires a huge amount of bits in the field to express the starting BP and the final BP for each group. The objective is to maintain as low overhead as possible for the specification of a given group. For that purpose, a group is characterized by its *start time slot* and *end time slot* (between 0 and 15) and the corresponding *BP offsets*. The start and end offsets for the time duration of a group is computed as follows:

$$Relative\ Offset = (\text{Start/End})\text{Backoff Period Offset} * 2^{SO}.$$

The choice of a *BP Offset* subfield encoded in two bits is argued by the fact that the minimum number of BPs in a time slot is equal to 3 (for $SO = 0$). For $SO > 0$, each time slot is divided in three parts to which the start/end instant of a given GAP must be synchronized.

This GAP implementation approach only requires two bytes of overhead per group. The maximum number of groups depends on the SO values, since lower superframe orders cannot support much overhead in the beacon frame due to short superframe durations. Also, it allows a flexible and dynamic allocation of the groups, since all nodes continuously update their information about their group start and end times when receiving a beacon frame, at the beginning of each superframe.

bits	0-2	3-6	7-8	9-12	13-14	15	
	Group ID	Group Start Slot	Start Backoff Period Offset	Group End Slot	End Backoff Period Offset	Reserved	
Length	3	4	2	4	2	5	= 16 bits

Fig. 9. GAP specification field of a beacon frame.

V. Experimental Evaluation

A. Implementation Approach

We have implemented the H-NAMe mechanism in nesC/TinyOS [34] over our open-source implementation of the IEEE 802.15.4/ZigBee protocol stack (open-ZB) [35] to evaluate its performance, and to demonstrate its feasibility through real experimentation.

For that purpose, we have carried out a thorough experimental analysis to understand the impact of the H-NAMe mechanism on network performance, namely in terms of *network throughput* (S) and *probability of successful transmissions* (Ps), for different *offered loads* (G), in one cluster with a star-based topology. These metrics have also been used to evaluate the performance of the slotted CSMA/CA MAC protocol in [36]. The network throughput (S) represents the fraction of traffic correctly received normalized to the overall capacity of the network (250 kb/s). The success probability (Ps) reflects the degree of reliability achieved by the network for successful transmissions. This metric represents the throughput S divided by G, which refers to the amount of traffic sent from the application layer to the MAC sublayer, also normalized to the overall network capacity.

To guarantee a reliable measurement process, we have ensured that the IEEE 802.15.4 physical channel was free from interference with IEEE 802.11 networks operating at the same frequency range, by selecting Channel 26 for the IEEE 802.15.4 network and by using a spectrum analyzer for checking channel integrity.

In addition, to have an “unbiased” idea on the impact of the hidden-node phenomenon independently from other parameters, we have configured the superframe order to a sufficiently high value $(SO = 8)$ to avoid the collisions related to the CCA deference for low SO, in the slotted CSMA/CA mechanism (refer to [36]). Note that CCA deference occurs when the remaining time of a superframe is not sufficient to completely send a frame which imposes the deference of the transmission to the next superframe. For low SO and due to the lower superframe duration, it is more probable that this deference occurs (in more nodes), resulting in multiple collisions at the beginning of the next superframe.

B. Test-Bed Scenario

The experimental test-bed consisted of 18 MICAz motes [37] (featuring an Atmel ATmega128L 8-bit microcontroller with 128 kB of in-system programmable memory) scattered in three groups hidden from each other, a ZC and a Chipcon CC2420 protocol analyzer [38], capturing the traffic for processing and analysis (Fig. 10).

The 18 nodes have been programmed to generate traffic at the Application Layer with preset interarrival times. The three node groups were placed at the ground level near walls in order to ensure that groups were hidden from each other (Fig. 10). For that purpose, we carried the following simple test. We have programmed a MICAz mote to continuously perform clear channel assessment, toggling a led when energy was detected on the channel. By placing this mote at different spots while a group of nodes was transmitting, it was possible to identify an area to place a new node so that it would be hidden from the nodes in the other groups. This procedure was repeated until we got three groups of six nodes each.

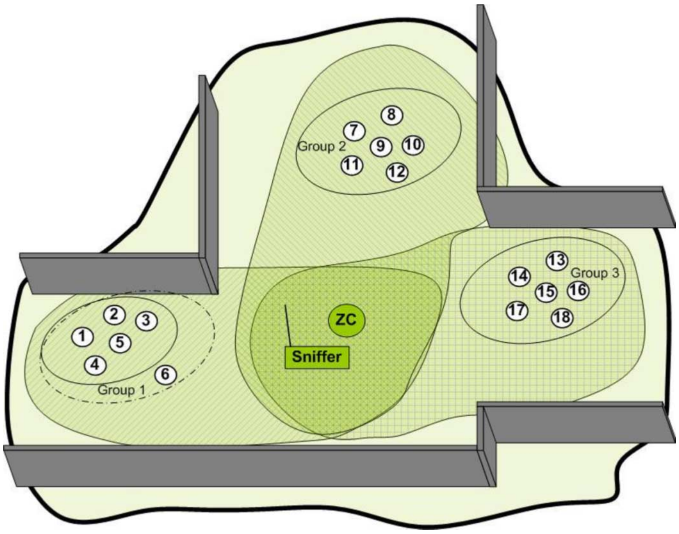


Fig. 10. Experimental test-bed.

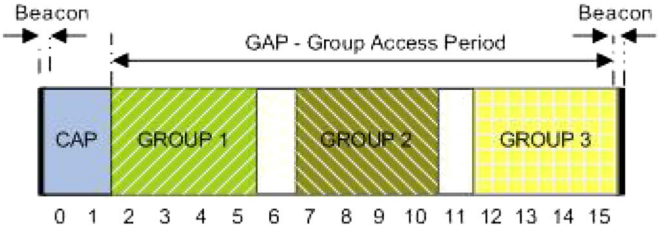


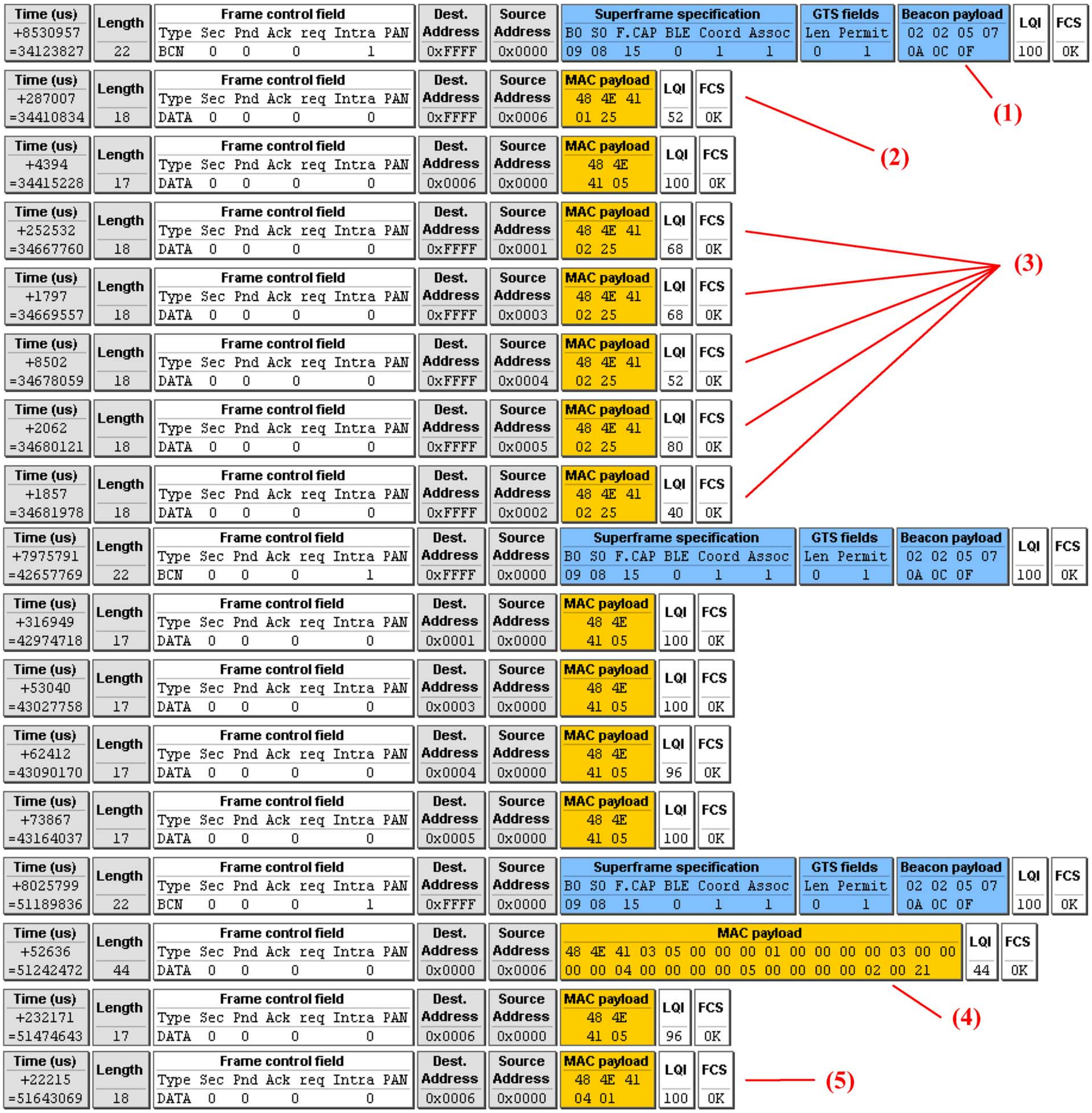
Fig. 11. Groups allocation in the superframe.

C. Experimental Results

Fig. 11 presents the GAP created by the H-NAMe mechanism, for the test-bed scenario just described.

The H-NAMe algorithm has assigned four time slots to each group, which represents a theoretical duration of 983.04 ms per group for a superframe order $SO = 8$, and assuming equal group access duration for an equal number of nodes per group).

1) The Group-Join Procedure: Fig. 12 illustrates a snapshot from the Chipcon CC2420 protocol analyzer showing the group-join procedure. In this example, a node with short address



```
Time (us) | Length | Frame control field                | Dest.   | Source  | Superframe specification       | GTS fields | Beacon payload | LQI | FCS
+8530957  |        | Type Sec Pnd Ack req Intra PAN     | Address | Address | BO SO F.CAP BLE Coord Assoc    | Len Permit | 02 02 05 07    |     |
=34123827 | 22     | BCN  0   0   0       1             | 0xFFFF  | 0x0000  | 09 08 15    0   1     1        | 0   1      | 0A 0C 0F       | 100 | OK

Time (us) | Length | Frame control field                | Dest.   | Source  | MAC payload | LQI | FCS
+287007   |        | Type Sec Pnd Ack req Intra PAN     | Address | Address | 48 4E 41    |     |
=34410834 | 18     | DATA 0   0   0       0             | 0xFFFF  | 0x0006  | 01 25       | 52  | OK

Time (us) | Length | Frame control field                | Dest.   | Source  | MAC payload | LQI | FCS
+4394     |        | Type Sec Pnd Ack req Intra PAN     | Address | Address | 48 4E       |     |
=34415228 | 17     | DATA 0   0   0       0             | 0x0006  | 0x0000  | 41 05       | 100 | OK

Time (us) | Length | Frame control field                | Dest.   | Source  | MAC payload | LQI | FCS
+252532   |        | Type Sec Pnd Ack req Intra PAN     | Address | Address | 48 4E 41    |     |
=34667760 | 18     | DATA 0   0   0       0             | 0xFFFF  | 0x0001  | 02 25       | 68  | OK

Time (us) | Length | Frame control field                | Dest.   | Source  | MAC payload | LQI | FCS
+1797     |        | Type Sec Pnd Ack req Intra PAN     | Address | Address | 48 4E 41    |     |
=34669557 | 18     | DATA 0   0   0       0             | 0xFFFF  | 0x0003  | 02 25       | 68  | OK

Time (us) | Length | Frame control field                | Dest.   | Source  | MAC payload | LQI | FCS
+8502     |        | Type Sec Pnd Ack req Intra PAN     | Address | Address | 48 4E 41    |     |
=34678059 | 18     | DATA 0   0   0       0             | 0xFFFF  | 0x0004  | 02 25       | 52  | OK

Time (us) | Length | Frame control field                | Dest.   | Source  | MAC payload | LQI | FCS
+2062     |        | Type Sec Pnd Ack req Intra PAN     | Address | Address | 48 4E 41    |     |
=34680121 | 18     | DATA 0   0   0       0             | 0xFFFF  | 0x0005  | 02 25       | 80  | OK

Time (us) | Length | Frame control field                | Dest.   | Source  | MAC payload | LQI | FCS
+1857     |        | Type Sec Pnd Ack req Intra PAN     | Address | Address | 48 4E 41    |     |
=34681978 | 18     | DATA 0   0   0       0             | 0xFFFF  | 0x0002  | 02 25       | 40  | OK

Time (us) | Length | Frame control field                | Dest.   | Source  | Superframe specification       | GTS fields | Beacon payload | LQI | FCS
+7975791  |        | Type Sec Pnd Ack req Intra PAN     | Address | Address | BO SO F.CAP BLE Coord Assoc    | Len Permit | 02 02 05 07    |     |
=42657769 | 22     | BCN  0   0   0       1             | 0xFFFF  | 0x0000  | 09 08 15    0   1     1        | 0   1      | 0A 0C 0F       | 100 | OK

Time (us) | Length | Frame control field                | Dest.   | Source  | MAC payload | LQI | FCS
+316949   |        | Type Sec Pnd Ack req Intra PAN     | Address | Address | 48 4E       |     |
=42974718 | 17     | DATA 0   0   0       0             | 0x0001  | 0x0000  | 41 05       | 100 | OK

Time (us) | Length | Frame control field                | Dest.   | Source  | MAC payload | LQI | FCS
+53040    |        | Type Sec Pnd Ack req Intra PAN     | Address | Address | 48 4E       |     |
=43027758 | 17     | DATA 0   0   0       0             | 0x0003  | 0x0000  | 41 05       | 100 | OK

Time (us) | Length | Frame control field                | Dest.   | Source  | MAC payload | LQI | FCS
+62412    |        | Type Sec Pnd Ack req Intra PAN     | Address | Address | 48 4E       |     |
=43090170 | 17     | DATA 0   0   0       0             | 0x0004  | 0x0000  | 41 05       | 96  | OK

Time (us) | Length | Frame control field                | Dest.   | Source  | MAC payload | LQI | FCS
+73867    |        | Type Sec Pnd Ack req Intra PAN     | Address | Address | 48 4E       |     |
=43164037 | 17     | DATA 0   0   0       0             | 0x0005  | 0x0000  | 41 05       | 100 | OK

Time (us) | Length | Frame control field                | Dest.   | Source  | Superframe specification       | GTS fields | Beacon payload | LQI | FCS
+8025799  |        | Type Sec Pnd Ack req Intra PAN     | Address | Address | BO SO F.CAP BLE Coord Assoc    | Len Permit | 02 02 05 07    |     |
=51189836 | 22     | BCN  0   0   0       1             | 0xFFFF  | 0x0000  | 09 08 15    0   1     1        | 0   1      | 0A 0C 0F       | 100 | OK

Time (us) | Length | Frame control field                | Dest.   | Source  | MAC payload                                     | LQI | FCS
+52636    |        | Type Sec Pnd Ack req Intra PAN     | Address | Address | 48 4E 41 03 05 00 00 00 01 00 00 00 00 03 00 00 |     |
=51242472 | 44     | DATA 0   0   0       0             | 0x0000  | 0x0006  | 00 00 04 00 00 00 00 05 00 00 00 00 02 00 21    | 44  | OK

Time (us) | Length | Frame control field                | Dest.   | Source  | MAC payload | LQI | FCS
+232171   |        | Type Sec Pnd Ack req Intra PAN     | Address | Address | 48 4E       |     |
=51474643 | 17     | DATA 0   0   0       0             | 0x0006  | 0x0000  | 41 05       | 96  | OK

Time (us) | Length | Frame control field                | Dest.   | Source  | MAC payload | LQI | FCS
+22215    |        | Type Sec Pnd Ack req Intra PAN     | Address | Address | 48 4E 41    |     |
=51643069 | 18     | DATA 0   0   0       0             | 0x0006  | 0x0000  | 04 01       | 100 | OK
```

Fig. 12. Packet analyzer capture of a group join.

0×0006 (see Fig. 10) requests to join a group. Notice that the beacon payload includes the GAP specification of the groups already formed (labeled (1) in Fig. 12).

The requesting node initiated the process by sending a *Group-join.request* message to the ZC [label (2)] and receiving an ACK. Then, all the other nodes in its transmission range replied with a *Neighbor.notify* message [label (3)]. When the requesting node receives these messages, it knows that it shares a bidirectional link with its neighbors. As soon as the timer for receiving *Neighbor.notify* messages expires, the requesting node sends a *Neighbor.report* message to the ZC identifying its neighbors (label (4)). The ZC runs the H-NAMe intra-cluster grouping algorithm to assign a group to that node and sends a *Group-join.confirm* message, notifying the node of which group to join (label (5)). The node (assigned to Group 1) can transmit during the GAP part reserved for Group 1 (Fig. 11).

2) H-NAMe Performance Evaluation: The performance evaluation of the H-NAMe mechanism was carried out using $BO = SO = 8$ (100% duty cycle), with a constant frame size of 904 bits. We repeated the experiment several times (one for each packet interarrival time), to evaluate the network performance at different offered loads (G).

Fig. 13 presents the throughput (S) and the success probability (Ps) obtained from three experimental scenarios: 1) a network with hidden-nodes that does not use the H-NAMe mechanism (triangle-marker curve); 2) a network with hidden-nodes using the H-NAMe mechanism (circle markers curve), and 3) a network without hidden-nodes (square markers curve), which means that all nodes are in a single broadcast domain, i.e., all nodes hear each other. The average values of the throughput and probability of success were computed with a 95% confidence interval for a sample size of 3000 packets at each offered load.

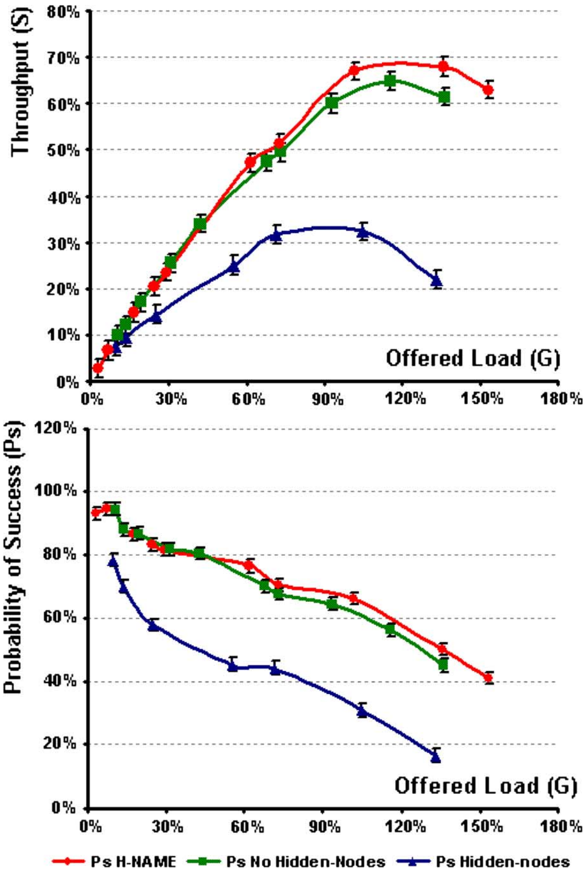


Fig. 13. Experimental performance results for (S) and (Ps).

The confidence interval is displayed at each sample point by a vertical black bar.

From these results, we can observe that, even at low offered loads, H-NAMe leads to a significant performance improvement. For instance, for an offered load (G) of 30%, the success probability (Ps) using H-NAMe is roughly 50% greater than without using H-NAMe. For higher loads, H-NAMe doubles the throughput of the conventional network with hidden-nodes. At 90% of offered load (G), the throughput of the network using H-NAMe reaches 67% and is still increasing; however, without using H-NAMe a saturation throughput of 32% is achieved, which represents an improvement of more than 100%.

It can also be observed that for high offered loads the H-NAMe mechanism has actually up to 5% better throughput performance than that of a network without hidden-nodes. It is not unrealistic to have H-NAMe outperforming a little bit the non-hidden node scenario since when H-NAMe is used, the number of nodes in each group is smaller than in the entire non-hidden node network, thus the number of nodes competing for the medium over time is lower, which reduces collisions. In fact, using H-NAME, at most 6 nodes (one group) contend for the medium at a time (GAP) instead of 18 nodes when grouping is not used.

Due to the extremely low probability of success of the scenario with hidden-nodes, the number of necessary transmission attempts to successfully send a packet increases with the offered load, leading to a higher number of transmissions to send the same amount of packets. This obviously impacts nodes’ energy consumption. Fig. 14 presents the energy consumption of the radio transceiver of one node for the three experimental scenarios. These results were computed according to the current draw values listed in the MICAz [37] and CC2420 [42] datasheets.

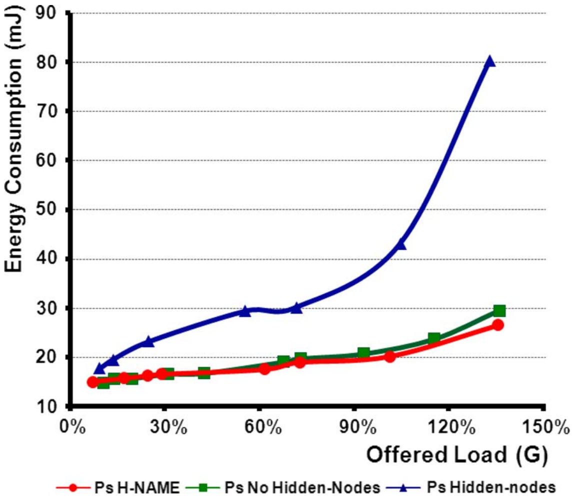


Fig. 14. Experimental performance results for energy consumption.

Notably, the energy consumption in the scenario without H-NAMe is already 50% higher when it reaches 60% of offered load. The increased energy consumption is even more significant at higher loads, where it is approximately two and a half times higher than the energy consumption with H-NAMe.

In this experimental scenario, there were no retransmitted packets (due to collisions). However, if we consider one retransmission for each lost packet, the increase in the number of transmissions would be significant in the case of the network without H-NAMe, leading to a much higher energy loss, even at low offered loads. Notice that for $G = 30\%$, Ps is around 50% when H-NAMe is not used, meaning that half of the packets transmitted do not reach their destination.

In conclusion, it has been experimentally shown that the H-NAMe mechanism significantly improves the network performance in terms of throughput and success probability, at the small cost of some additional overhead to setup the different groups in the clusters.

VI. Performance Evaluation in a Target Tracking Application

This section aims at demonstrating the impact of the hidden-node problem and of the use of the H-NAMe mechanism in a target tracking application in a WSN deployment.

A. Experimental Test-Bed Description

The objective of this application is to detect, localize and rescue a target entity, within a certain region covered by a WSN deployment. We have used two mobile robots acting as target and rescuer/pursuer entities (Fig. 15). The navigation of the target robot, that is supposed to be in distress (search and rescue context) or to be an intruder (pursuit-evasion context), is remotely controlled by an operator. A WSN node mounted on top

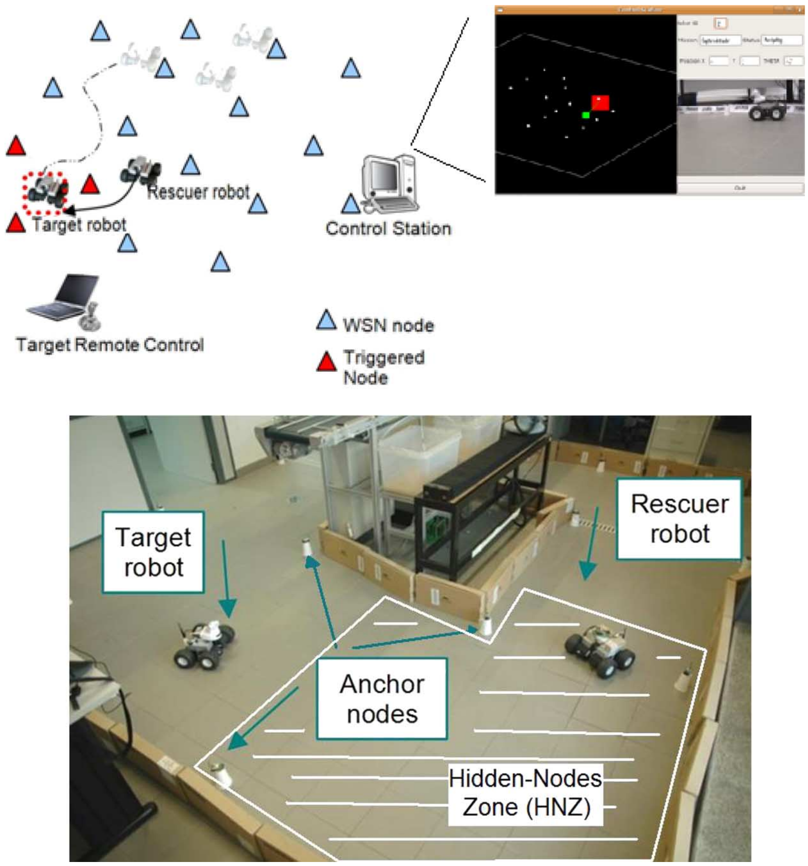


Fig. 15. Snapshot of the ART-WiSe search and rescue test-bed application.

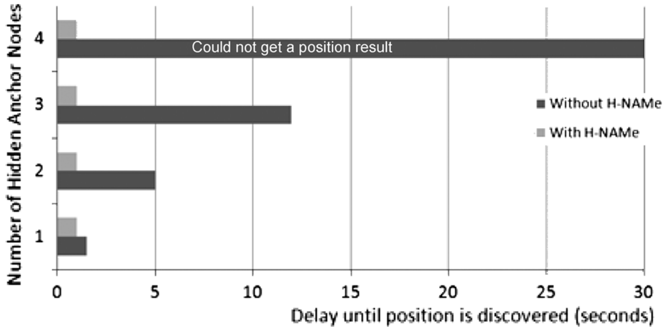


Fig. 16. Localization delay for test 1.

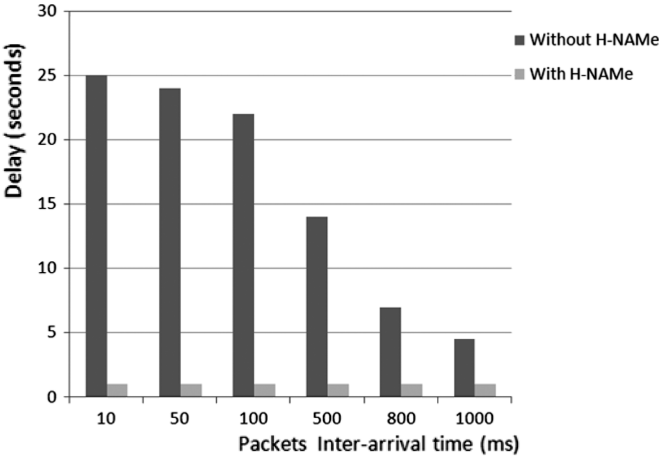


Fig. 17. Localization delay for test 2.

of it sends periodic messages to signal its presence, which are relayed by the WSN to the Control Station with the necessary data to trigger localization. Then, the Control Station computes the target robot location, displays it in a virtual scenario and informs the rescuer/pursuer robot that immediately initiates its mission by moving towards the last known position of the target robot. The process is repeated until the rescuer/pursuer robot is close enough to the target robot (Fig. 14).

Localization is based on radio signal strength (RSS) readings from the CC2420 transceiver to derive the relative distance between the sender and the receiver and on a trilateration algorithm for determining the position of a mobile robot, with the knowledge of the positions of three anchor nodes. For more details about the localization mechanism, please refer to [41].

In order to assess the impact of the presence of the hidden-nodes on the behavior of the test-bed, we have created a hidden-node zone (HNZ, refer to Fig. 15). Within this area, some nodes were programmed as hidden-terminals, by changing the clear channel assessment (CCA) threshold value of the nodes' transceivers to a maximum value, so that they would always consider the channel as idle.

B. Impact of Hidden-Node Problem on the Localization Mechanism

In order to measure the impact of the hidden-node problem on the application, namely on the localization and target tracking mechanisms, we performed two different sets of experiments and compared the delay in computing the position of the target robot when inside the HNZ, with the delay in the case of no hidden-nodes.

In the *first set of experiments* (reported in Section VI-B1), the WSN nodes triggered in the localization mechanism (the anchor nodes) were set as hidden and we measured the delay to get the position of the target robot. As for the *second set of experiments* (reported in Section VI-B2), we placed one hidden-node inside the HNZ generating traffic at preset rates. This node could not sense the four anchor nodes responsible of localization. However, this time the anchor nodes were able to sense each other and the extra traffic generating node, thus resulting in a unidirectional link between those and the hidden-node. We performed ten measurements for each traffic value.

In both tests, one set of experiments was performed using the H-NAMe mechanism to demonstrate the feasibility and effectiveness of this mechanism in a real application scenario.

1) Results of the First Set of Experiments: For the first test, only one hidden anchor node was used, then two, three, and finally all four anchor nodes as hidden-nodes. Fig. 16 presents the measured delay for getting the localization of the target, in each case. With all the four anchor nodes programmed as hidden nodes, the time spent to get a correct location output was higher than 30 s. On the other hand, without any hidden-node the time to get the position of the target was smaller than 1 s (approximately 400 ms).

It was noticed that with only one of the four anchor nodes in the HNZ acting as hidden-node, there was little impact on the delay. This was due to the fact that there were always three anchor nodes with full connectivity and distance information available (the minimum to run the localization algorithm). In fact, when one of those three anchor nodes was disconnected, the delay increased to 5 s, since there were only two nodes with full connectivity available to perform localization.

With the H-NAMe mechanism, one group was assigned to each hidden-node. The performance improvement was immediately noticed, since it allowed localization in approximately one second, even when all of the four anchors used for localization were hidden.

2) Results of the Second Set of Experiments: A hidden-node was programmed to generate traffic with preprogramed interarrival times. This node was then placed inside the Hidden Node Zone. The target was also placed inside the Hidden Node Zone and the localization mechanism was enabled. We performed several sets of experiments for different traffic generation rates (ten for each inter-arrival time). This test is different from the previous one in the sense that now there is a unidirectional link between the anchor nodes and the hidden-node (the anchor nodes can sense the hidden-node but the hidden-node cannot sense the anchors). Interference was not expected to be very high since the anchor nodes could use the IEEE 802.15.4 slotted CSMA/CA for performing collision avoidance, thus avoiding collisions with the hidden-node. Nevertheless, we still observed some delay, as showed in Fig. 17.

For low interarrival times (around 1 s) there was little impact on the delay, since the probability of collisions was not very high. Nevertheless, collisions still occurred, leading to a delay of around 4 s. However, as the interarrival time decreases (lower than 100 ms), the impact is much higher, taking over 20 s to get the position of the target. This renders the localization mechanism useless and the tracking application fails, since it takes too long to output a target position. On the other hand, when H-NAMe is used, the delay remains approximately the same (around 1 s), as it is completely independent from the hidden-node traffic rate.

This test was repeated with the target robot moving (remotely controlled) at a constant speed. As expected, we observed that for interarrival values lower than 800 ms in the traffic generating node, as the robot was going through the HNZ, the Control Station failed to output the robot's current position. As the robot left that zone, the Control Station was able to correctly inform the position of the target once again. With H-NAMe, the localization delay was constant, both inside and outside the HNZ zone.

VII. Concluding Remarks

This paper proposes a simple but effective solution to the hidden-node problem, which is a fundamental impairment to QoS in wireless communication networks and particularly for WSNs. Our solution is very attractive for WSN applications with more stringent QoS requirements, as the hidden-node problem represents one of the major causes of QoS degradation, particularly in what concerns network throughput, message delay, energy-consumption, and reliability.

The proposed mechanism—H-NAMe—eliminates hidden-node collisions in synchronized single or multiple cluster WSNs using contention-based MAC protocols. It follows a proactive approach, since it avoids hidden-node collisions before occurring, through the creation of hidden-node interference-free node groups and node cluster groups.

One of the most important contributions of our work is the integration of H-NAMe in the IEEE 802.15.4/ZigBee protocols, which currently are the dominant communication technologies for WSNs. This integration is shown to be very simple and in a way that WSN nodes implementing H-NAMe are fully and transparently interoperable with the default WSN nodes (not implementing H-NAMe). Also important, the implementation of H-NAMe will be available as an open-source, within our open-ZB tool suite [35].

Finally, the feasibility and effectiveness of the H-NAMe mechanism was implemented, tested, validated and demonstrated both in a dedicated test-bed and in a real application scenario, leading to significant performance improvements.

References

[1] J. Stankovic, I. Lee, A. Mok, and R. Rajkumar, "Opportunities and obligations for physical computing systems," *IEEE Computer*, vol. 38, no. 11, pp. 25–33, 2005.

[2] B. Raman and K. Chebrolu, "Censor networks: A critique of "sensor networks" from a systems perspective," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 3, pp. 75–78, 2008.

[3] OPNET Tech. [Online]. Available: http://www.opnet.com, 2006

[4] P. Jurcčk and A. Koubâa, The IEEE 802.15.4 OPNET Simulation Model: Reference Guide v2.0 IPP-HURRAY Tech. Rep., HURRAY-TR-070509, May 2007.

[5] *Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)*, IEEE-TG15.4, IEEE Standard for Information Technology, 2003.

[6] A. Cunha, “On the use of IEEE 802.15.4/ZigBee as federating communication protocols for wireless sensor networks,” M.Sc. thesis, , 2007, HURRAY-TR-070902.

[7] S. Ray, D. Starobinski, and J. B. Carruthers, “Performance of wireless networks with hidden nodes: A queuing-theoretic analysis,” *Comput. Commun.*, vol. 28, pp. 1179–1192, 2005.

[8] S. Ray, J. Carruthers, and D. Starobinski, “Evaluation of the masked node problem in ad-hoc wireless LANs,” *IEEE Trans. Mobile Comput.*, vol. 4, pp. 430–442, 2005.

[9] F. A. Tobagi and L. Kleinrock, “Packet switching in radio channels: Part II—The hidden terminal problem in carrier sense multiple-access and the busy-tone solution,” *IEEE Trans. Commun.*, vol. 23, pp. 1417–1433, 1975.

[10] C. S. Wu and V. O. K. Li, “Receiver-initiated busy-tone multiple access in packet radio networks,” in *Proc. ACM Workshop on Frontiers in Comput. Commun. Technol.*, Stowe, Vermont, 1987, pp. 336–342.

[11] Z. J. Haas and J. Deng, “Dual busy tone multiple access (DBTMA)—A multiple access control scheme for ad hoc networks,” *IEEE Trans. Commun.*, vol. 50, pp. 975–985, 2002.

[12] A. Chandra, V. Gummalla, and J. O. Limb, “Wireless collision detect (WCD): Multiple access with receiver initiated feedback and carrier detect signal,” in *Proc. IEEE ICC*, 2000, pp. 397–401.

[13] B. Ji, “Asynchronous wireless collision detection with acknowledgement for wireless mesh networks,” in *Proc. IEEE Veh. Technol. Conf.*, Sep. 2005, vol. 2, pp. 700–704.

[14] F. A. Tobagi and L. Kleinrock, “Packet switching in radio channels: Part III—Polling and (dynamic) split channel reservation multiple access,” *IEEE Trans. Comput.*, vol. 24, no. 7, pp. 832–845, Aug. 1976.

[15] P. Karn, “MACA—A new channel access method for packet radio,” in *Proc. 9th ARRL/CRRL Amateur Radio Comput. Netw. Conf.*, 1990, pp. 134–140.

[16] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, “MACAW: A media access protocol for wireless LAN’s,” in *Proc. ACM SIGCOMM*, London, U.K., Aug. 1994, pp. 212–225.

[17] *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, ISO/IEC IEEE-802-11, IEEE Standard for Information Technology, 1999.

[18] C. L. Fullmer and J. J. Garcia-Luna-Aceves, “Solutions to hidden terminal problems in wireless networks,” in *Proc. ACM SIGCOMM*, Cannes, France, Sep. 1997, pp. 39–49.

[19] Y. Yang, F. Huang, X. Ge, X. Zhang, X. Gu, M. Guizani, and H. Chen, “Double sense multiple access for wireless ad-hoc networks,” *Int. J. Comput. Telecomm. Netw.*, vol. 51, no. 14, pp. 3978–3988, 2007.

[20] K. Xu, M. Gerla, and S. Bae, “How effective is the IEEE 802.11 RTS/CTS handshake in ad hoc networks?,” in *Proc. Global Telecomm. Conf. (GLOBECOM’02)*, 2002, vol. 1, pp. 72–76.

[21] J. Deng, B. Liang, and P. K. Varshney, “Tuning the carrier sensing range of IEEE 802.11 MAC,” in *Proc. IEEE Global Telecomm. Conf. (GLOBECOM)*, 2004, vol. 5, pp. 2987–2991.

[22] F. Ye, S. Yi, and B. Sikdar, “Improving spatial reuse of IEEE 802.11 based ad hoc networks,” in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM’03)*, San Francisco, CA, Dec. 2003, vol. 2, pp. 1013–1017.

[23] H. Zhai and Y. Fang, “Physical carrier sensing and spatial reuse in multirate and multihop wireless ad hoc networks,” in *Proc. IEEE INFOCOM*, Apr. 2006, pp. 1–12.

[24] I. Ho and S. Liew, “Impact of power control on performance of IEEE 802.11 wireless networks,” *IEEE Trans. Mobile Computing*, vol. 6, no. 11, pp. 1245–1258, Nov. 2007.

[25] D. Halperin, J. Ammer, T. Anderson, and D. Wetherall, “Interference cancellation: Better receivers for a new wireless MAC,” in *Proc. Hotnets*, 2007, pp. 339–350.

[26] D. Halperin, T. Anderson, and D. Wetherall, “Taking the sting out of carrier sense: Interference cancellation for wireless LANs,” in *Proc. ACM MobiCom’08*, Sep. 2008, pp. 339–350.

[27] S. Gollakota and D. Katabi, “Zigzag decoding: Combating hidden terminals in wireless networks,” in *Proc. ACM SIGCOMM 2008*, Seattle, WA, Aug. 17–22, 2008, pp. 159–170.

[28] L. Hwang, “Grouping strategy for solving hidden node problem in IEEE 802.15.4 LR-WPAN,” in *Proc. IEEE 1st Int. Conf. Wireless Internet (WICON’05)*, Budapest, Hungary, 2005, pp. 26–32.

[29] A. Koubâa, A. Cunha, and M. Alves, “A time division beacon scheduling mechanism for IEEE 802.15.4/ZigBee cluster-tree wireless sensor networks,” in *Proc. 19th Euromicro Conf. Real-Time Systems (ECRTS’07)*, Pisa, Italy, Jul. 2007, pp. 125–135.

[30] R. Severino, “On the use of IEEE 802.15.4/ZigBee for Time-Sensitive Wireless Sensor Network Applications,” M.Sc. thesis, School of Engineering, Polytechnic Institute of Porto, Porto, Portugal, Oct. 2008.

[31] IEEE 802.15.4 task group., [Online]. Available: http://www.ieee802.org/15/pub/TG4b.html 2006

[32] ZigBee-Alliance, ZigBee Specification. [Online]. Available: http://www.ZigBee.org/ 2006

[33] A. Koubâa, M. Alves, and E. Tovar, “Modeling and worst-case dimensioning of cluster-tree wireless sensor networks,” in *Proc. 27th IEEE Real-Time Syst. Symp. (RTSS’06)*, Rio de Janeiro, Brazil, Dec. 2006, pp. 412–421.

[34] TinyOS. [Online]. Available: http://www.tinyos.net, 2007

[35] An Open-Source Toolset for the IEEE 802.15.4/ZigBee Protocol Stack. [Online]. Available: http://www.open-zb.net 2006

[36] A. Koubâa, M. Alves, and E. Tovar, “A comprehensive simulation study of slotted CSMA/CA for IEEE 802.15.4 wireless sensor networks,” in *Proc. 6th IEEE Workshop Factory Commun. Syst. (WFCS’06)*, Torino, Italy, Jun. 2006, pp. 183–192.

[37] MICAz/MICA2 Mote Datasheets. [Online]. Available: http://www.xbow.com

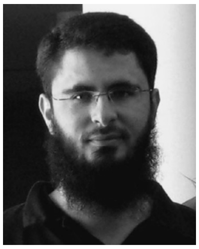
[38] Chipcon, Chipcon Packet Sniffer for IEEE 802.15.4. 2006.

[39] I. Ho and S. Liew, “Impact of power control on performance of IEEE 802.11 wireless networks,” *IEEE Trans. Mobile Comput.*, vol. 6, no. 11, pp. 1245–1258, Nov. 2007.

[40] A. Koubâa, R. Severino, M. Alves, and E. Tovar, H-NAMe: Specifying, implementing and testing a hidden-node avoidance mechanism for wireless sensor networks, Tech, Rep. HURRAY-TR-071113, Nov. 2007.

[41] R. Severino and M. Alves, On a test-bed application for the ART-WiSe framework Tech. Rep. HURRAY-TR-061103, Nov. 2006.

[42] CC2420 Transceiver Datasheet. [Online]. Available: http://www.ti.com



Anis Koubâa was born in 1977. He received the Engineering degree in telecommunications from the Higher School of Telecommunications, Tunis, Tunisia, in 2000, and the M.Sc. and Ph.D. degrees in computer science from the National Polytechnic Institute of Lorraine (INPL), Nancy, France, in 2001 and 2004, respectively.

He is currently an Assistant Professor at the College of Computer Science and Information Systems, Al-Imam Muhammad Ibn Saud University, Riyadh, Saudi Arabia, and a Research Associate at the CISTER/IPP-HURRAY Research Group, Porto, Portugal. He is actively working on the design of large-scale wireless sensor netwoks and cyber-physical systems, while maintaining QoS, security, mobility and reliability. He is particularly interested in the assessment and improvement of the IEEE 802.15.4/ZigBee standard protocol stack for large-scale wireless sensor networks. He has driven the research efforts in the context of the ART-WiSe and open-ZB research frameworks that have contributed to the release of an open-source toolset of the IEEE 802.15.4/ZigBee protocol stack. He is the Chair of the TinyOS ZigBee Working Group, whose purpose is achieving a standard-compliant open-source implementation of ZigBee over TinyOS. In addition, he is involved in the CONET European Network of Excellence, in particular with the research cluster COTS-based architecture for QoS in large-scale distributed embedded systems. He is also actively participating as a reviewer or a program committee member in some reputed international journals, conferences and workshops dealing with real-time networks, quality-of-service, wireless communications, and related issues.

Ricardo Severino was born in 1982. He received the Degree in 2006 and the M.Sc. degree in 2008 in electrical and computer engineering from the Polytechnic Institute of Porto, School of Engineering (ISEP/IPP), Porto, Portugal.

Since 2006, he has been working in the area of wireless sensor networks, namely on improving quality-of-service in WSNs by using standard and commercial-off-the-shelf (COTS) technology, at the CISTER/IPP-HURRAY Research Unit. In this line, he has been actively participating in the ART-WiSe (www.artwise.cister-isep.info) and Open-ZB (www.open-zb.net) research frameworks, as well as in international projects such as ArtistDesign (FP7 NoE on Embedded System Design), CONET (FP7 NoE on Cooperating Objects), and EMMON (FP7 JU on Embedded Monitoring). He is also a founding member and contributor of the 15.4 and ZigBee TinyOS Working Groups.

Mr. Serverino was awarded the EWSN'09 Best M.Sc. Thesis Award at the prestigious European Conference on Wireless Sensor Networks (EWSN'09). He has several publications in reputed conferences (e.g., MASS, RTCSA, ECRTS) and journals (e.g., IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS) and has served as a Reviewer for several conferences (e.g., IEEE ETFA, SUTC, and VTC).

Mário Alves was born in 1968. He received the Degree in 1991, and the M.Sc. and Ph.D. degrees in electrical and computer engineering from the University of Porto, Porto, Portugal, in 1995 and 2003, respectively.

He is a Professor of Electrical and Computer Engineering at the Polytechnic Institute of Porto (ISEP/IPP) and a Research Associate of the CISTER/IPP-HURRAY Research Unit, focusing on real-time, distributed and embedded computing systems. He participated in several international projects related to industrial communication systems (e.g., CCE-CNMA, RFieldbus). He has been serving as a reviewer and has published in top conferences (e.g., RTSS, ECRTS, ICDCS, OPODIS, MASS) and journals (e.g., IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, Elsevier ComNet, Springer RTSJ) in his expertise areas. His current research interests are mainly devoted to improving quality-of-service in wireless sensor networks by using standard and commercial-off-the-shelf (COTS) technology (ART-WiSe, open-ZB). He is currently involved in international projects on networked embedded systems (ArtistDesign), cooperating objects (CONET), large-scale embedded monitoring using wireless sensor networks (EMMON) and cyber-physical systems for monitoring critical physical infrastructures (PT-CMU).

Prof. Alves received Best Paper Awards (e.g., ECRTS'07) and supervised the EWSN'09 Best M.Sc. Thesis Award. He actively participated in the organization and TPC of several international conferences and workshops, e.g., IEEE WFCS'00, ECRTS'03, and EWSN'10.

Eduardo Tovar was born in 1967. He received the Licentiate, M.Sc., and Ph.D. degrees in electrical and computer engineering from the University of Porto, Porto, Portugal, in 1990, 1995 and 1999, respectively.

Currently, he is a Professor of Industrial Computer Engineering in the Computer Engineering Department, Polytechnic Institute of Porto (ISEP-IPP), where he is also engaged in research on real-time distributed systems, wireless sensor networks, multiprocessor systems, cyber-physical systems, and industrial communication systems. He heads the CISTER/IPP-HURRAY Research Unit (UI 608), a top ranked ("Excellent") unit of the FCT Portuguese network of research units. He is team leader within the Seventh Framework ICT Network of Excellence Artist Design, on distributed embedded systems. Since 1991, he authored or coauthored more than 100 scientific and technical papers in the area of real-time computing systems, wireless sensor networks, distributed embedded systems, and industrial computer engineering.

Prof. Tovar has been consistently participating in top-rated scientific events as member of the Program Committee, as Program Chair or as General Chair. Examples are the IEEE RTSS (Real Time Systems Symposium); IEEE RTAS (Real-Time and Embedded Technology and Applications Symposium); IEEE SDRS (Symposium on Distributed Reliable Systems); IEEE ICDCS (International Conference on Distributed Computing Systems); ACM EMSOFT (Annual ACM Conference on Embedded Software); Euromicro ECRTS (Euromicro Conference on Real-Time Systems); IEEE ETFA (Emerging Technologies on Factory Automation) or IEEE WFCS (Workshop on Factory Communication Systems). Notably, he has been Program Chair for ECRTS 2005, WDES 2006, OPODIS 2007, and CPS-CA08.

A