

11-1-2006

SLAM: Sleep-Wake Aware Local Monitoring in Sensor Networks

Issa Khalil

Purdue University

Saurabh Bagchi

Purdue University

Ness B. Shroff

Purdue University

Follow this and additional works at: <http://docs.lib.purdue.edu/ecetr>

Khalil, Issa; Bagchi, Saurabh; and Shroff, Ness B., "SLAM: Sleep-Wake Aware Local Monitoring in Sensor Networks" (2006). *ECE Technical Reports*. Paper 339.

<http://docs.lib.purdue.edu/ecetr/339>

This document has been made available through Purdue e-Pubs, a service of the Purdue University Libraries. Please contact epubs@purdue.edu for additional information.

SLAM: Sleep-Wake Aware Local Monitoring in Sensor Networks

Abstract

Sleep-wake protocols are critical in sensor networks to ensure long-lived operation. However, an open problem is how to develop efficient mechanisms that can be incorporated with sleep-wake protocols to ensure both long-lived operation and a high degree of security. Our contribution in this paper is to address this problem by using local monitoring, a powerful technique for detecting and mitigating control/data attacks in sensor networks. In local monitoring, a node oversees part of the traffic going in and out of its neighbors to determine if the behavior is suspicious, such as, delay in forwarding a packet. However, a direct application of local monitoring will interfere with sleep-wake protocols by having nodes stay awake for the purpose of monitoring. In this work, we present a protocol called SLAM to make local monitoring parsimonious in its energy consumption and integrate it with any extant sleep-wake protocol in the network. The challenge is to enable this in a secure manner and this is addressed in the face of nodes that may be adversarial and not wake up nodes responsible for monitoring its traffic. We prove analytically that security coverage is not weakened by the sleeping. We perform simulation experiments to demonstrate that the performance of local monitoring is practically unchanged while listening energy savings of 30 to 129 times, depending on the network load, is achieved.

1 Introduction

Sensor networks are typically comprised of large numbers of sensor nodes placed in the environment to be monitored. These nodes cooperate among themselves for information gathering and analysis, and are becoming an important platform in several domains, including military warfare, civilian emergency operations, scientific explorations, and monitoring of climate and biological habitats. The open nature of the wireless communication channels, the lack of infrastructure, the fast deployment practices, and the hostile environments where they may be deployed, make them vulnerable to a wide range of security attacks. The attacks could involve eavesdropping, message tampering, or identity spoofing, all of which have been addressed by customized cryptographic primitives for sensor networks. Alternately, the attacks may be targeted to the control or the data traffic in sensor networks. Typical examples of control traffic are routing, monitoring the liveness of a node, topology discovery, and distributed location determination. Different kinds of control traffic attacks have been proposed in the literature, such as, the wormhole attack, the rushing attack, the Sybil attack, the sinkhole attack, and the HELLO flood attack. Control traffic attacks are especially destructive since they can be launched even without having access to any cryptographic keys or compromising any legitimate node in the network and they can be used to subvert the functionality of the network by disrupting data flow.

To mitigate such attacks, many researchers have used the concept of cooperative local monitoring within a node's neighborhood ([30]-[37]). In *neighborhood monitoring* (or *local monitoring*), nodes oversee part of the traffic going in and out of their neighbors. Many protocols have been built on top of neighbor monitoring for intrusion detection (e.g., [29] [30]), building trust and reputation among nodes (e.g. [33] [34]), protecting against control and data traffic attacks (e.g. [35]-[38]) and in building secure routing protocols (e.g. [31] [32] [36]). Specifically, in [36] and [37] the authors have presented a technique for detection of control and data attacks in ad-hoc networks using local monitoring. Control attacks are launched by delaying, dropping, modifying, or fabricating forwarded traffic and can be detected by a group of neighbors, called *guard nodes*, performing local monitoring. The guard nodes are normal nodes in the network and perform the basic operations of sensing, in addition to monitoring. The same attack primitives applied to data traffic can also be detected by local monitoring. However, local monitoring could come at a high cost for energy constrained sensor networks, since it requires the guard nodes to be awake all the time to oversee network behavior. To the best of our knowledge, no one has studied sleeping protocols for optimizing the energy overhead of monitoring while maintaining the quality of the monitoring service. This is the problem we address in this paper. The main challenge lies in providing a *secure* sleeping technique that is not vulnerable to security attacks and does not add to the vulnerability of the network.

In this paper we propose a set of mechanisms (SLAM) that adapt the existing local monitoring technique to significantly reduce the time a node needs to be awake for the purpose of monitoring. The proposed mechanism adapts itself depending on the kind of sleeping protocol used in the network, henceforth referred to as the *baseline sleeping protocol* (BSP). For networks that use synchronized sleeping algorithms (e.g., [4] [18]-[22]), i.e., nodes wakeup and go to sleep in a synchronized manner, SLAM does not need to do anything. There exist several application-specific sleeping algorithms, for example, to maintain a given sensing coverage (each point should be sensed by at least k nodes) or a given network connectivity level (each pair of nodes should have k disjoint paths). For these protocols (e.g., [6]-[12] [16]), SLAM can serve neighbor monitoring as well by modifying an input parameter to the existing sleeping algorithm. The exact modification depends on the BSP itself and we provide in the paper an example of adapting a coverage protocol. Finally, for those networks that have no existing BSP or have on-demand sleep-wake, i.e., nodes are woken up at arbitrary times determined by the communication, SLAM provides a generic on-demand sleeping algorithm, called On-Demand SLAM. This algorithm assumes that in addition to the normal antenna, each node has a passive or a low-power wake-up antenna. A node that is not involved in network activities, such as, data forwarding is ordinarily sleeping according to the BSP. However, for monitoring purposes, it is woken up on demand by a neighboring node using the wake-up antenna.

On- demand SLAM has to account for the fact that wake-up antennas have a delay in waking up nodes on receiving the control signal. By a suitable design, we prevent the additional delay due to sleep-wake from becoming cumulative with the number of hops between the communicating pair of nodes. Instead, a pipelined effect is achieved and the additional delay becomes constant independent of the number of hops. We provide theoretical analysis for energy saving using On-Demand SLAM compared to a baseline monitoring protocol [37]. We build a simulation model for SLAM using *ns-2* and perform a comparative evaluation of local monitoring with and without SLAM. The results show that the performance of local monitoring in terms of false and missed alarms is very close in both cases while the overhead of SLAM in terms of listening energy is between 30 to 129 times lower, depending on the network traffic. The results show the effect of the number of malicious nodes, the traffic load, and the fraction of data being monitored on the overhead of local monitoring.

We summarize our contributions in this paper as follows:

1. We provide a technique for conserving energy while performing local monitoring without significantly degrading its security performance. This we believe is fundamental to deploying local monitoring in any energy conscious network.
2. We propose a generic on-demand sleep-wake algorithm for network monitoring in scenarios where either no application-specific sleeping algorithm exists or the sleep-wake is based on arbitrary communication pattern.
3. We analytically prove that SLAM does not add any vulnerability to the existing local monitoring technique.
4. We conduct extensive simulation experiments on an existing local monitoring technique with and without SLAM and show a significant reduction in monitoring cost with negligible degradation in the monitoring quality of service.

The rest of the paper is organized as follows. Section 2 presents related work in the field of sleep-wake protocols. Section 3 describes SLAM. Section 4 presents mathematical analysis of the energy overhead and security of SLAM. Section 5 presents the simulation experiments and results. Section 6 concludes the paper.

2 Related Work

Node sleeping is an important mechanism to prolong the life time of sensor networks. This topic has been discussed extensively in the literature and many protocols have been proposed for various types of applications such as object tracking [1]-[2]. It has been realized that under current hardware designs, the maximum energy savings can be achieved through putting nodes to sleep—three orders of magnitude less current draw than in an idle node for the popular Mica mote platform for sensor nodes.

Primarily three different mechanisms are used to put nodes to sleep. The *first* is called *synchronized wakeup-sleep scheduling* in which the nodes in the network are put to sleep and woken up at the same time in a centralized (e.g., [21] [22]) or a distributed manner (e.g. [4] [18]-[20]). A disadvantage of such protocols is that the duty cycle is application dependent and not known *a priori*. Most importantly, they require the network to have an accurate time synchronization service. Furthermore, in scenarios with rare event detection, no event happens and the nodes enter sleep mode again in most of the wakeup periods. This means that nodes wake up too often resulting in

wastage of energy. The *second* mechanism is based on selecting a subset of nodes to be woken up to *maintain some properties in the network*, such as sensing coverage (e.g., [6]-[12]), network connectivity (e.g., [4] [5] [13]-[15]), or both coverage and connectivity (e.g. [16]).

The *third* mechanism is based on *on-demand sleep-wake* protocols. These on-demand sleep-wake protocols use either special purpose low-power wake-up antennas (e.g., [23]-[26]) or passive wake-up antennas [27]. These antennas are responsible for receiving an appropriate beacon from a neighbor node and waking up the node for its full operation. Thus, for environments where events of interest are relatively rare, the time for the low power operation with the wake-up antennas being on, dominates. Further details about the operation of the antennas are mentioned in Section 3.4 where SLAM uses these antennas for waking up guard nodes.

Many sensor applications require security and reliability; therefore, researchers consider designing dependable sensor networks that behave reliably and securely. Neighbor monitoring is a well-known technique that is used for securing sensor network protocols. However, to the best of our knowledge none of the previous local monitoring protocols consider operating in a network where nodes may need to be put to sleep for energy conservation. Therefore, we are the first to address this issue.

3 SLAM Protocol Description

The primary goal of SLAM is to minimize the time a node has to be awake to perform local monitoring. Local monitoring is used to make sure that packets are not dropped, delayed, modified, or forged along the path from source to destination [36]. SLAM adds one more task to the list of events that a guard node needs to monitor—verifying whether the node being monitored wakes up the requisite guards or fails to do so due to malicious motivations. Depending on the BSP used in the network, SLAM has three different mechanisms for proposing sleeping for networks with local monitoring—The *No-Action-Required* SLAM protocol, the *Adapted* SLAM protocol, and the *On-Demand* SLAM protocol.

3.1 System Model and Assumptions

SLAM assumes that the network is static and the links are bi-directional. SLAM requires a pre-distribution pairwise key management protocol (e.g. [40], [41]) such that any two nodes can acquire a key for encryption and authentication. In On-Demand SLAM, each node is equipped with either a passive [27] or a low-power wakeup antenna [24]. Any two nodes that need to communicate, establish a route between them using an underlying routing protocol. We assume that the source node is honest. No assumption is made about the adversary nodes following the sleep-wake protocol, only the honest nodes follow it. Each node knows its first-hop neighbors and the neighbors of each neighbor, e.g., using a technique as in [36]. Malicious behavior is manifested through delaying, dropping, fabricating, or modifying packets. The malicious behavior of fruitlessly sending a wake-up signal to a node is not addressed since this potential exists in any on-demand wake-up protocol and SLAM neither exacerbates nor solves this problem.

3.2 The No-Action-Required SLAM Protocol

This scheme is used in a network that has a sleeping algorithm which is completely compatible with local monitoring. Such sleep algorithms fall in a class of protocols in which the network (or the communicating parts of the network) is synchronized in its sleep-wake schedule and all the nodes wake up and go to sleep in distributed or centralized synchrony. Examples of such protocols include Span [4], S-MAC [19], habitat monitoring [22], and those used in applications of sensor networks in [18][20][21]. In this BSP, the guards for the communication would also be woken up since, by definition, the guards are one-hop neighbors of the two nodes that form the link on which the communication is taking place. Thus, for this class of protocols, no modification is necessary to support sleeping and waking up of guards for local monitoring purposes. Local monitoring in such scenarios does not incur any additional overhead on the network aside from the computational overhead.

3.3 The Adapted SLAM Protocol

This scheme is used for the class comprising coverage and/or connectivity preserving sleep-wake protocols. Examples of such kinds of sleeping algorithms are [6]-[12][16]. However, since these algorithms may be application-specific, each one of them may need to be adapted differently to support sleeping of guards as well. Here we will consider a representative sub-class of BSP from this class.

Consider for example the class of protocols that seeks to preserve K_s -coverage or K_c -connectivity in a network and puts nodes off to sleep without violating these properties. The property of K_s -coverage (s for sensing) denotes that every point in the field is sensed by at least K_s nodes. The property of K_c -connectivity (c for coverage) denotes that for critical communication, such as, between a node and the base station, at least K_c routes exist. The fundamental technique for adapting such sleeping algorithms to support sleeping of guards is to modify the value of K_s or K_c and invoke the original BSP.

Consider a protocol that preserves coverage at K_s ([6]-[12]). Assume that the sensing range is R_s , the communication range is R_c , the detection confidence is γ . In local monitoring, γ is defined as the minimum number of neighbors of a node, X , to convince another neighbor of X , say Y , that X is malicious if Y does not directly detect X as malicious [37]. Assume the requisite number of guards needed for detection with sufficiently low missed and false alarm rates is Γ ($\Gamma \geq \gamma$).

We shall try to find a relationship between K_s , R_s , R_c , and γ with the help of Figure 1. What is the value of K_s to guarantee the number of guards is Γ ? Let the density of the nodes in the network be ρ and the density of awake (or alive) nodes be $\rho_l = K_s/\pi R_s^2$. Let the common communication area between X and Y be A_c . Assume uniform distribution of the awake nodes.

The number of nodes that are awake in A_c (N_w) is given by

$$N_w = A_c \cdot \rho_l = A_c \frac{K_s}{\pi R_s^2} \quad [1]$$

Therefore, the required value of K_s to get Γ guards is given by,

$$K_s = \frac{N_w}{A_c} \cdot \pi R_s^2 = \frac{\Gamma}{A_c} \cdot \pi R_s^2 \quad [2]$$

The common communication area A_c for two nodes separated by a distance x is given by

$$A_c = 2R_c^2 \cos^{-1}(x/2R_c) - x\sqrt{R_c^2 - x^2/4} \quad [3]$$

The minimum value of this is achieved when $x = R_c$ and the value is given by $A_{c,min} = 1.23R_c^2$. Thus, the protocol needs to be invoked with a value of

$$K_s = \frac{\Gamma}{A_{c,min}} \cdot \pi R_s^2 = \frac{\Gamma}{1.23R_c^2} \cdot \pi R_s^2 = 2.55 \cdot \Gamma \cdot \left(\frac{R_s}{R_c}\right)^2 \quad [4]$$

This will guarantee that the requisite number of guard nodes is awake to provide detection through local monitoring. Thus, Adapted SLAM invokes the BSP with the increased value of the parameter K_s .

3.4 The On-Demand SLAM Protocol

This protocol is used in a network that either has no BSP in operation or employs on-demand sleep-wake protocols. Therefore, we build a new sleep-wake protocol, called On-Demand SLAM that enables the guards to go sleep when not required for monitoring. The high level approach we take is on-demand sleep-wake of the guards rather than scheduling the sleep-wake periods. The defining characteristic of on-demand sleep-wake protocols is that any node in the network may, at random, initiate communication with any other node in the network. A sleep-wake protocol does not impose any fixed communication pattern in the network. On-Demand SLAM uses either low-power wake-up antennas (e.g., [23]-[26]) or passive antennas with circuitry that can harvest signal energy to trigger a node to wake up [27], as has been described in Section 2 (“Related Work”). These kinds of antennas are commercially available (e.g. [26]) as well as in research labs (e.g., [27]). For example Austriamicrosystems provides a low-power wake-up receiver (AS3931) with data rate of 2.731 KB/s and current consumption in standby mode of 6.6uA [26]. Data transmission and reception require good channel quality, high speed, and thus complex and power consuming hardware, while channel monitoring has the sole purpose of getting binary

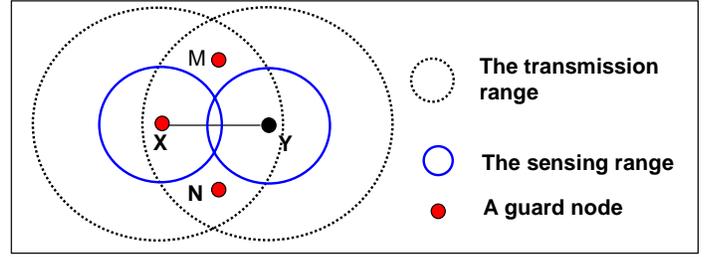


Figure 1: Relationship between communication and sensing ranges

information whether a packet targeted at this node is coming. In the rest of the paper, for ease of exposition we use the term “low-power wake-up radio” to mean either the low-power wake-up hardware or the passive wake-up hardware which consumes no power at all.

In On-Demand SLAM, the low-power wake-up radio remains awake all the time while the normal radio is put to sleep when it is not sending or receiving data or is not required for monitoring. If a node is to send a packet out, it simply wakes up by itself; if a neighbor node is to send a packet to this node, the sender will send a short wake-up beacon using the wake-up radio channel, and on receiving this beacon the wake-up radio triggers the normal radio to be ready for the reception. The main disadvantage of the mechanism is that it still consumes extra energy. Even though the power consumed is small compared to the normal antenna (1uW compared to 10mW in [23]), the energy is non-negligible due to long time of operation.

Hence this mechanism has been modified to use passive wake-up antennas, known as radio-triggered power management mechanisms [27]. In this mechanism a special hardware component – a radio-triggered circuit – is connected to one of the interrupt inputs of the processor. The circuit itself does not draw any current and is thus passive. The node can enter sleep mode without periodic wake-up. The wake-up mode is the usual working mode with all the functional units ready to work, and the average wake-up mode current is 20mA [27]. In sleep mode, a node shuts down all its components except the memory, interrupt handler, and the timer and the sleep mode current is 100uA [27]. When a network node changes from sleep mode to wake-up mode, there is a surge current of 30mA for a maximum of 5ms [27]. When a power management message is sent by another node within a certain distance, the radio-triggered circuit collects enough energy to trigger the interrupt to wake up the node. Except for activating the wake-up interrupt, the radio-triggered circuit is independent of any other components on the node. If supported by hardware, the wake-up packet is sent at a special radio frequency. Other types of radio communication, at a different radio frequency, do not wake up the nodes even if the nodes are within the radio communication range. Note that hardware cost for adding multiple-frequency support is usually fairly low. Many recent low-end radio transceivers support multiple frequency operations [28].

The basic idea in designing On-Demand SLAM is for a node to wake up the requisite guard nodes to perform local monitoring on the communication it is going to send or forward on its outgoing link. The challenge in the design comes from the fact that any of the nodes (except the source) may be malicious and therefore may not faithfully wake up the guards. *Local monitoring* [36] [37] is used to mitigate malicious activities manifested through dropping, delaying, modifying, or forging of packets along the path from source to destination. In local monitoring, the sensor node is called a *guard* when performing traffic overhearing and monitoring of neighbors. The guards of a node A over the incoming packets from a transmitter X are the common neighbors of X and A.

In Figure 2, α_1 and β_1 are the guards of H_1 over the link $S \rightarrow H_1$. Information for each packet sent from X to A is saved in a *watch buffer* at each guard for a time T_w . The information maintained depends on the particular attack primitive to be detected (i.e., drop, delay, modify, or forge). A malicious counter ($MalC(i,j)$) is maintained at each guard node, i , for every node, j , which i is monitoring. $MalC(i,j)$ is incremented for any suspect malicious activity of j that is detected by i .

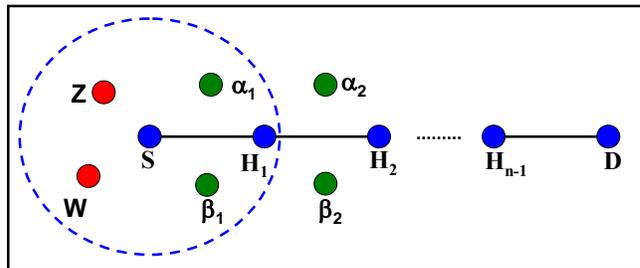


Figure 2: n -hop route between S and D, neighbors of S, and guards of H_1 and H_2

To account for intermittent natural failures that can occur at legitimate nodes, a node is determined to be misbehaving, only if the $MalC$ goes above a threshold. When $MalC(i,j)$ crosses the threshold, node i isolates node j refraining from sending or receiving any packet from node j . Node j is said to be isolated from the network when all its neighbors isolate it. We use the scenario depicted in Figure 2 to explain On-Demand SLAM. A source node S is sending data to a destination node D through an n -hop route $S \rightarrow H_1 \rightarrow H_2 \rightarrow \dots \rightarrow H_{n-1} \rightarrow D$. In a network where all the nodes are honest, S will wake up the next hop H_1 and the guard nodes (α_1 and β_1) before sending the packet to H_1 . In turn H_1 will wake up H_2 and guard nodes α_2 and β_2 before sending the packet on the next hop and so on, till the packet reaches D. Formally, according to [37], the responsibility of a guard node α of H_{i+1} over a link $H_i \rightarrow H_{i+1}$ is to verify that:

1. H_{i+1} forwards the packet within time T_w

2. H_{i+1} does not modify the packet it is forwarding
3. H_{i+1} only forwards a packet if a packet is sent on the $H_i \rightarrow H_{i+1}$ link

SLAM introduces a fourth responsibility.

4. H_{i+1} should wake up the guards for the communication on the $H_{i+1} \rightarrow H_{i+2}$ link *before* forwarding the packet on that link

If a rule 1-3 is violated then the *MalC* value is incremented by appropriate amount; if rule 4 is violated, the *MalC* value increment is the maximum of the other MalC values because this rule violation may be used to mask violations of any of the rules 1-3.

In general for any multi-hop route connecting a source node S to a destination node D, S is responsible for waking up the correct guards for H_1 , and H_i is responsible for waking up the correct guards of H_{i+1} ($1 \leq i \leq n-2$). The correct guards for H_1 are guaranteed to be woken up by the assumption of honest source S and whether H_i honestly wakes up the next hop guards is monitored by the guards of H_i according to rule 4 above.

In the following we present two variations of On-Demand SLAM depending on the wake-up mechanism a node follows to wake up the guards of next-hop.

3.4.1 Guards-Only On-Demand SLAM (G-SLAM)

The high level design goal in G-SLAM is to minimize the energy wasted in waking up nodes that can not serve as guards. On average half of the nodes within a single transmission range are not guards over a certain link (according to Equation I in [37]). In Figure 2, α_1 and β_1 are valid guards of H_1 over the link from S to H_1 , while Z and W are not. Also, note that the energy spent in warm up (transition between sleep mode and wakeup mode) is relatively high (almost 3 times as much as the energy spent in listening for the antennas described in [27]). Therefore, waking up the appropriate nodes saves considerable amount of energy.

For a guard node to verify honest wake-up, G-SLAM requires each node in the network to know, in addition to the identities of its first-hop and second-hop neighbors that are required by local monitoring, the location of each node within twice of its transmission range. In Figure 2, a guard of H_1 , say α_1 , knows the location of its neighbor H_1 and the location of all the neighbors of H_1 , S, β_1 , β_2 , α_2 , and H_2 . Using this information, α_1 knows the common neighbors of H_1 and H_2 , α_2 and β_2 , which can act as the guards of H_2 over the link $H_1 \rightarrow H_2$. Therefore, α_1 can not be deceived by H_1 waking up its neighbors that can not be guards for H_2 (S and β_1). A disadvantage of G-SLAM is that it requires sophisticated wakeup hardware for a node to wake up a subset of nodes within the communication range using an id-attached beacon [27].

We explain G-SLAM algorithm with the help of Figure 2. Assume that node S has some data to be sent for the destination D over the route $S \rightarrow H_1 \rightarrow H_2 \rightarrow \dots \rightarrow H_{n-1} \rightarrow D$ connecting S to D. G-SLAM uses the following steps to wake up the correct guards along the route from S to D:

1. Node S sends a signal to wake up the first-hop node (H_1) and the guards for H_1 (α_1, β_1). This signal could be either unicast to each of $H_1, \alpha_1,$ and β_1 or a multicast signal that contains the identities of $H_1, \alpha_1,$ and β_1 . This signal is guaranteed to wake up the correct guards of H_1 due to the assumption of honest source S.
2. Node S sends the packets it has to H_1 following the timing schedules presented in Section 3.4.3.
3. Nodes $H_1, \alpha_1,$ and β_1 after being woken up continue to remain awake for T_w . T_w is a parameter of local monitoring that captures the maximum time by which an entry in the watch buffer is evicted (beyond that is evidence of malicious action) [37]. Each time a new packet is sent from S to H_1 , T_w is reinitialized. After T_w expires at a node, it goes back to sleep.
4. Node H_1 , after being woken up, uses the timing schedule in Section 3.4.3 to schedule a wake-up signal for H_2 and the guards of H_2 over the link $H_1 \rightarrow H_2$ (α_2, β_2). The guards of H_1 over the link $S \rightarrow H_1$ are responsible for verifying that H_1 fulfills this requirement.
5. The process continues at each step up to the destination.

3.4.2 All-Neighbors On-Demand SLAM (A-SLAM)

The high level design goal of A-SLAM is to relax the assumption that every node knows the location of its first-hop and second-hop neighbors, and to simplify the wakeup signal and the wakeup hardware. Consider a node S that has some data to send for the destination D over the route $S \rightarrow H_1 \rightarrow H_2 \rightarrow \dots \rightarrow H_{n-1} \rightarrow D$, Figure 2. A-SLAM uses the following steps to wakeup the guards along the route from S to D:

1. Node S broadcasts a wake-up signal to all its first-hop neighbors ($Z, W, H_1, \alpha_1, \beta_1$). The wake-up signal includes the identity of both the current sender (S) and the next-hop (H_1).
2. Each neighbor of S, after being woken up, decides whether to stay awake or go back to sleep based on the role that it may play on the ongoing communication. If that neighbor is the next-hop (H_1), it stays awake to forward the data and to monitor the next-hop from it (H_2). If that neighbor is a guard (α_1, β_1) for the next-hop (H_1), it stays awake to monitor the behavior of H_1 . Finally, if that neighbor is neither a guard for H_1 nor a next-hop, it goes back to sleep immediately.
3. Node S sends the data packet it has to H_1 following the timing schedules presented in Section 3.4.3.
4. Nodes H_1 , α_1 , and β_1 after being woken up continue to do so for T_w . Each time a new packet is sent from S to H_1 , T_w is reinitialized. After T_w expires at a node, it goes back to sleep.
5. H_1 does the same steps that S did to wake up the next-hop (H_2) and its guards (α_2, β_2).
6. The process continues at each step to the destination.

This scheme results in an increase in the energy consumption compared to G-SLAM due to the wake-up of the neighbors that are not guards.

3.4.3 Timing of the Wakeup Signal

In this section we generate the timing schedules for signaling the wake-up of nodes using On-Demand SLAM. This is important because the wake-up antennas have a warm-up period and this could increase the end-to-end delay of the communication. We design SLAM to send the wake-up signal at the earliest possibility so that the additional delay due to the sleep-wake protocol does not add up but is instead a constant independent of the number of hops.

Let $T_{control}$ be the time to send the wake-up packet to the radio-triggered antenna, T_{warmup} be the time for a node to be fully awake and functional from the time it receives the wake-up packet (5 ms for Stankovic *et al.*'s antenna [27]), and T_{data} be the time to send a data packet which includes the forwarding time at intermediate nodes, therefore, within T_{data} , an intermediate node completely receives a data packet and it can immediately start sending it. Moreover, let T_w be the maximum time a guard, after being woken up, waits for the packet to be forwarded. If the packet is not forwarded in this time, malicious action is suspected. Finally, let T_{wake} be the time a node continues to be awake after being woken up.

Let us consider an *isolated* (no other flows interfere with it) flow between S and D , separated by h hops. The intermediate nodes are n_1, n_2, \dots, n_{h-1} . Let g_i represents the guards of node n_i over the link $n_{i-1} \rightarrow n_i$. Let v_i represents the neighbors of n_i that are not guards of n_{i+1} over the link $n_i \rightarrow n_{i+1}$. Consider the following two disjoint cases based on the relation between $(T_{control} + T_{warmup})$ and T_{data} . The analysis assumes a node is sleeping when it receives the wake-up signal. If not, the node just prolongs, if necessary, its wake-up time to meet the requirement imposed by the new wake-up signal. For example, assume a guard node G is currently awake till $\text{Current_time} + \delta$ due to some other activity (forward data, guard for another link, etc.). Assume that G receives at Current_time a wakeup signal that require G to stay awake till $\text{Current_time} + \Delta$. Then, if $\delta \geq \Delta$, G does not need to do anything, otherwise G prolongs its wake-up time by $\Delta - \delta$ and goes to sleep at $\text{Current_time} + \Delta$ instead of $\text{Current_time} + \delta$.

Case I: $(T_{control} + T_{warmup}) > T_{data}$ with $\tau = (T_{control} + T_{warmup}) - T_{data}$

Figure 3 shows the timing schedule for this case. Figure 3 (a) shows the timing schedule for a node in the route between the source and the destination. The node, n_1 , wakes up at T_3 and goes to sleep at T_8 , where $T_8 - T_3 = T_{data}$ (to receive data) + τ (wait for the next-hop to be ready to receive the data) + T_{data} (send the data to the next-hop) + $\{\tau + T_{data}\}$ (as a guard for n_2) = $3T_{data} + 2\tau$. Figure 3 (b) shows the timing schedule for a guard node. The guard, g_1 , wakes up at T_3 and goes to sleep at T_6 , where $T_6 - T_3 = T_{data}$ (to overhear incoming data to the node being monitored, n_1) + τ (wait for the next-hop to be ready to receive the data) + T_{data} (to overhear outgoing data from

the node being monitored, $n_1) = 2T_{data} + \tau$. Figure 3 (c), only meaningful for A-SLAM, shows the schedule for a node that is a neighbor to a node in the route from the source to the destination but is not a guard node. The node, v_1 , wakes up at T_3 , determines that it can not be a guard, and thus go back to sleep immediately.

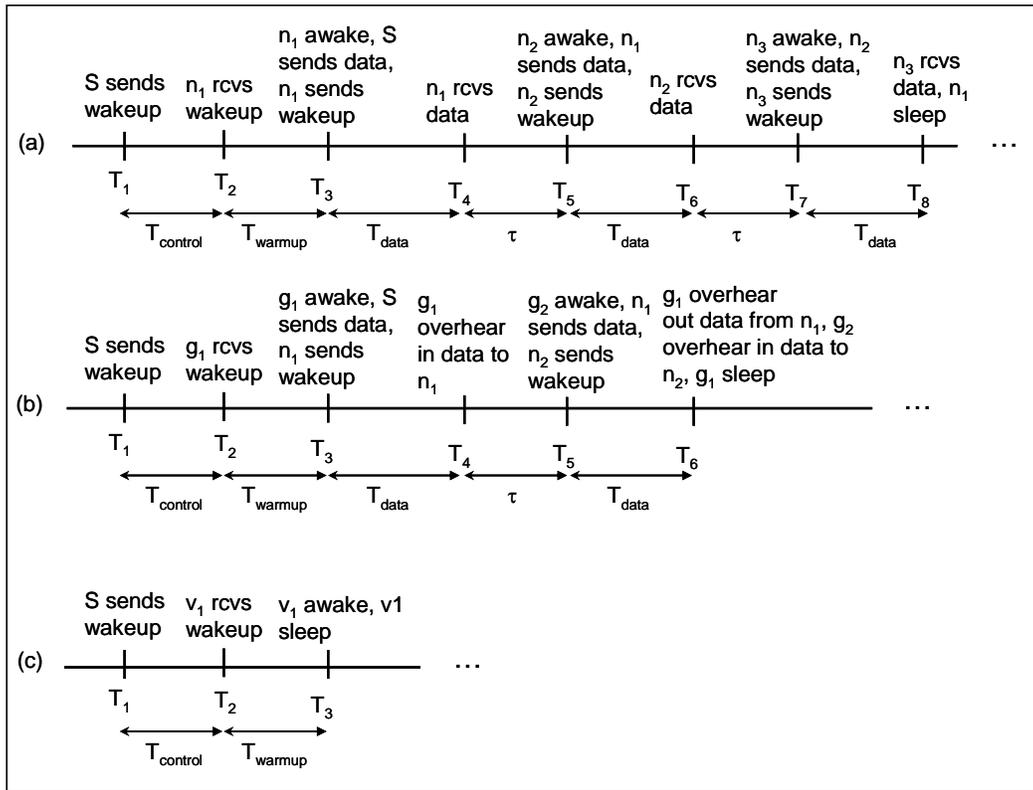


Figure 3: Case I wakeup-sleep timing schedule for (a) a node in the data route; (b) a guard node; (c) a neighbor to a node in the data route that is not valid guard (for A-SLAM only)

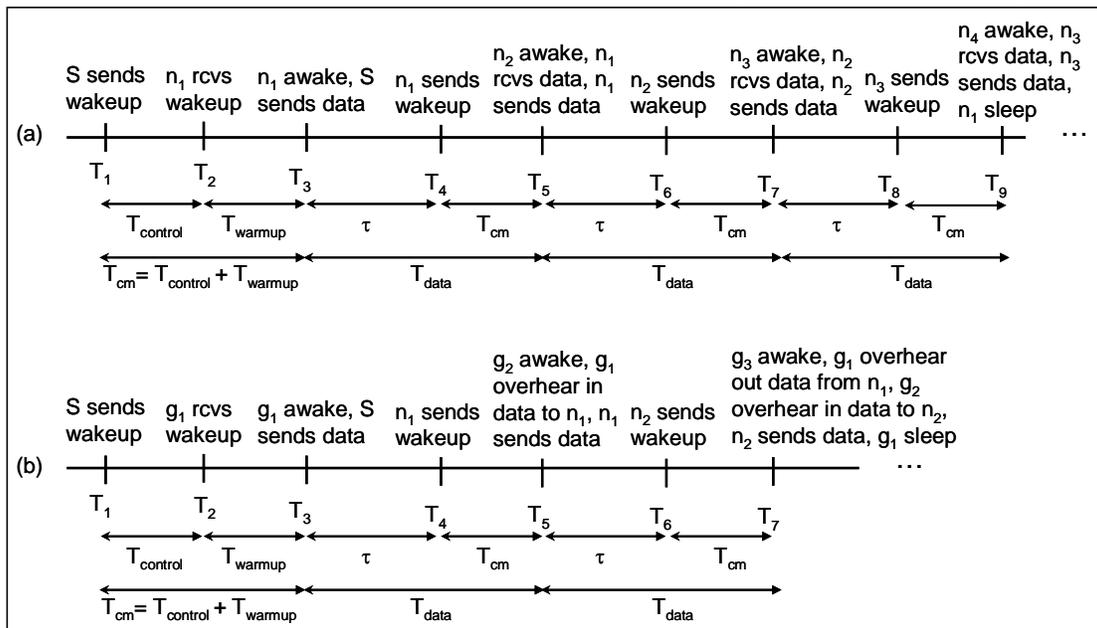


Figure 4: Case II wakeup-sleep timing schedule for (a) a node in the data route; (b) a guard node

According to SLAM, each node sends a wake-up signal at the earliest possible opportunity (as soon as it is awake) to minimize the delay. From Figure 3, it can be seen that per hop, the delay incurred is $T_{control} + T_{warmup}$ and at the last hop, the delay due to data (T_{data}) gets exposed.

Case II: ($T_{control} + T_{warmup}$) $\leq T_{data}$ with $\tau = T_{data} - (T_{control} + T_{warmup})$

Figure 4 shows the timing schedule for this case. Figure 4 (a) shows the schedule for a node in the route between the source and the destination. The node, n_1 , wakes up at T_3 and goes to sleep at T_9 , where $T_9 - T_3 = T_{data}$ (to receive data) + T_{data} (send the data to the next-hop) + T_{data} (as a guard for n_2) = $3T_{data}$. Figure 4 (b) shows the schedule for a guard node. The guard, g_1 , wakes up at T_3 and goes to sleep at T_7 , where $T_7 - T_3 = T_{data}$ (to overhear incoming data to the node being monitored, n_1) + T_{data} (to overhear outgoing data from the node being monitored, n_1) = $2T_{data}$. The timing schedule for a node that is a neighbor to a node in the route from the source to the destination but is not a guard node is the same as its peer in Case I.

4 Mathematical Analysis of On-Demand SLAM

4.1 Security Analysis

Here we prove that On-Demand SLAM does not degrade the security performance of local monitoring [37]. Specifically, we will prove the following premise.

Premise: Due to the sleep-wake mechanism for guards in SLAM, no loss in detection coverage occurs.

For this we prove that for any node H_i in the path $S \rightarrow D$ ($i = 1, \dots, n-1$),

- i. Either, the guards for H_{i+1} on the link $H_i \rightarrow H_{i+1}$ are awake (and monitoring) at the time communication takes place on the link, or
- ii. H_i is suspected of malicious action

We prove this using the first principle of mathematical induction.

Let the guards of H_1 over the link $S \rightarrow H_1$ form the set G_1 , $H_{n-1} \rightarrow D$ the set G_n , and $H_{i-1} \rightarrow H_i$ the set G_i .

Base case: The source S is honest and therefore it wakes up the guard nodes in G_1 .

Inductive hypothesis: All nodes H_1, \dots, H_i ($i \geq 1$) are honest and have woken up the appropriate guards, or have been suspected of malicious action.

To prove: Node H_{i+1} is either honest and wakes up the guard nodes in H_{i+2} or will be suspected of malicious action.

Case 1: One or more of the nodes H_1, \dots, H_i have been suspected of malicious action.

Case 2: All the nodes H_1, \dots, H_i have woken up the appropriate guards.

Proof Case 1: In this case, the malicious action(s) could be detected by rules 1-3 or rule 4 of Section 3.4. If the former, then it does not affect a guard being woken up and all guards in sets G_2, \dots, G_i have been woken up. If the latter, then one or more of the guard nodes in the sets G_2, \dots, G_i have not been woken up. If the node, say H_k , does not wake up the requisite guards, then it will be suspected by rule 4 and its MalC counter value will be incremented.

Proof Case 2: All the nodes in H_1, \dots, H_i have woken up the guards in the sets G_2, \dots, G_{i+1} .

Now G_{i+1} is monitoring if H_{i+1} is sending a wake-up signal to the guards of H_{i+2} over the link $H_{i+1} \rightarrow H_{i+2}$ i.e., G_{i+2} . If H_{i+1} is honest and performs this action, rule 4 is not triggered. But if H_{i+1} does not perform this action, then rule 4 is triggered and H_{i+1} is suspected of malicious action.

Therefore, by the principle of mathematical induction, it is proved that either all guards are woken up at the time of monitoring a communication or the malicious nodes are suspected. Since the detection of the guards according to rules 1-3 is not changed from baseline local monitoring, this proves that no loss of detection coverage happens due to SLAM.

4.2 Energy and End-to-End Delay Analysis

Here we calculate the worst case end-to-end delay of communication with local monitoring without sleep-wake (Baseline-LM) and with On-Demand SLAM. Moreover, an upper bound in the consumed energy is computed for SLAM and for the case with on-demand sleep-wake and no monitoring (Baseline-OD). For SLAM, the energy is calculated separately for a node which is forwarding packets (and, by definition, acting as a guard node), a node which is acting just as a guard, and a node that is in the vicinity of the path but is neither a forwarder nor a guard.

In addition to the notations defined in Section 3.4.3, let $A_{transmit}$ be the current to transmit (at the middle of the transmit range), which is 27mA for Mica2 motes [43]. Let A_{warmup} be the current consumed during the transition

from sleep to wakeup (warm up), which is 30mA for Mica2 motes [43]. Finally, let A_{active} be the current in the computationally active mode = the current in the idle listening mode = the current in receive mode, which is 8mA for Mica2 motes [43].

Let us consider a flow between S and D , separated by h hops. The intermediate nodes are n_1, n_2, \dots, n_{h-1} . The bounding box around S and D covers all possible nodes, including forwarding nodes and guard nodes that may be involved in the communication between S and D . The size of the bounding box is $2r(h+1)r = 2r^2(h+1)$, where r is the transmission range, Figure 5. For On-Demand SLAM, consider the two wakeup-sleep scheduling cases of Section 3.4.3.

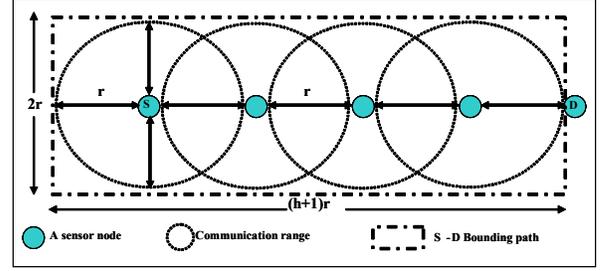


Figure 5: A bounding box over the path $S \rightarrow D$

Case I: $(T_{control} + T_{warmup}) > T_{data}$ with $\tau = (T_{control} + T_{warmup}) - T_{data}$

From Figure 3 it can be seen that delay at the first link ($S \rightarrow n_1$) is $T_{control} + T_{warmup} + T_{data}$. Over each of the succeeding links, the delay is $T_{control} + T_{warmup}$ since the delay due to data (T_{data}) gets exposed. This is due the sleep-wake schedule process that SLAM uses where the wake-up signal is sent at the earliest opportunity. Therefore, the end-to-end delay in SLAM, $\Omega_{SLAM}(h)$, for the communication from S to D is

$$\Omega_{SLAM}(h) = T_{control} + T_{warmup} + T_{data} + (h-1)(T_{control} + T_{warmup}) = h \cdot (T_{control} + T_{warmup}) + T_{data} \quad (5)$$

The end-to-end delay in Baseline-LM is

$$\Omega_{Base-LM}(h) = h \cdot T_{data} \quad (6)$$

In this case, the additional end-to-end delay imposed by SLAM depends on the number of hops between S and D

$$\Omega_{SLAM-Add}(h) = \Omega_{SLAM}(h) - \Omega_{Base-LM}(h) = h \cdot \tau + T_{data} \quad (7)$$

Next, we compute the consumed energy for both Baseline-OD and On-Demand SLAM.

Baseline-OD: here only the forwarding nodes are involved in the sleep-wake protocol. Using Figure 3 (a), a forwarding node n_i ($i = 1, \dots, h-1$) spends $T_{warmup} = T_3 - T_2$ warming up with current consumption of A_{warmup} , $T_{data} = T_4 - T_3$ receiving data with current consumption of A_{active} , $\tau = T_5 - T_4$ idle waiting for the next-hop to be ready with current consumption of A_{active} , and $T_{data} = T_6 - T_5$ sending data with current consumption of $A_{transmit}$. Therefore, the energy expended by a forwarding node n_i ($i = 1, \dots, h-1$) is

$$\mathcal{E}_{f,base} = T_{warmup} \cdot A_{warmup} + (T_{control} + T_{warmup}) \cdot A_{active} + T_{data} \cdot A_{transmit} \quad (8)$$

S spends $T_{control} + T_{warmup} = T_3 - T_1$ idle waiting for n_1 to wake up with A_{active} and $T_{data} = T_4 - T_3$ transmitting data with $A_{transmit}$. Therefore, the energy expended by S is

$$\mathcal{E}_{S,base} = (T_{control} + T_{warmup}) \cdot A_{active} + T_{data} \cdot A_{transmit} \quad (9)$$

D spends T_{warmup} warming up with A_{warmup} and T_{data} receiving data with A_{active} . Therefore, the energy expended by D is

$$\mathcal{E}_{D,base} = T_{warmup} \cdot A_{warmup} + T_{data} \cdot A_{active} \quad (10)$$

On-Demand SLAM: here the sleep-wake protocol involves, in addition to S and D , the forwarding nodes, the guard nodes, the neighbors of the forwarding nodes that are not guards. We will compute separately for the three kinds of nodes (i) forwarding nodes; (ii) guard nodes that do not act as forwarders; (iii) remaining nodes. The energy of S and D is the same as that of Baseline-OD.

- (i) Energy expended by a forwarding node n_i ($i = 1, \dots, h-1$) $\mathcal{E}_{f,SLAM} \leq \mathcal{E}_{f,base} + T_w \cdot A_{active}$. The additional energy is consumed because n_i has to look to see if n_{i+1} forwards the packet that it was just handed by n_i . The inequality comes in because T_w is the worst case time in case n_{i+1} is malicious.
- (ii) Energy expended by a guard node that is not a forwarding node $\mathcal{E}_{g,SLAM} \leq T_{warmup} \cdot A_{warmup} + T_{data} \cdot A_{active} + T_w \cdot A_{active}$. Consider for example the guard g_1 of n_1 over the link $S \rightarrow n_1$. g_1 has to listen to the communication between S to n_1 and then has to stay listening for a maximum of T_w to see that n_1 forwarded the packet.

(iii) Energy expended by a node in the bounding box around S and D that is neither a forwarding node nor a guard node (the “other node”, hence the notation “o” in the subscript). For G-SLAM where the wake-up signal is directed to the relevant guard nodes $\varepsilon_{o,G-SLAM} = 0$. For A-SLAM where the wake-up signal is broadcast in a one-hop neighborhood $\varepsilon_{o,A-SLAM} = T_{warmup} \cdot A_{warmup}$.

Case II: $(T_{control} + T_{warmup}) \leq T_{data}$ with $\tau = T_{data} - (T_{control} + T_{warmup})$

The end-to-end delay for SLAM in this case is exactly the same as that of Case I (Equation (5)) after exchanging T_{data} with $(T_{warmup} + T_{control})$.

$$\Omega_{SLAM}(h) = T_{control} + T_{warmup} + T_{data} + (h-1)(T_{data}) = h \cdot T_{data} + (T_{control} + T_{warmup}) \quad (11)$$

The end-to-end delay for Baseline-LM is exactly the same as that of Case I, (6). In this case, the additional end-to-end delay imposed by SLAM is fixed and does not depend on the number of hops between S and D

$$\Omega_{SLAM-Add}(h) = \Omega_{SLAM}(h) - \Omega_{Base-LM}(h) = T_{control} + T_{warmup} \quad (12)$$

For the energy, again we consider both Baseline-OD and On-Demand SLAM.

Baseline-OD: the energy for S and D are exactly the same as that of Case I (Equations (9) and (10)). The energy of the forwarding nodes is the same as that of Case I after replacing $(T_{warmup} + T_{control})$ with T_{data} .

$$\varepsilon_{f,base} = T_{warmup} \cdot A_{warmup} + T_{data} \cdot (A_{active} + A_{transmit}) \quad (13)$$

On-Demand SLAM: All energy computations are the same as in Case I.

Now consider that there are η concurrent flows going on in the network. The total energy consumed by all the nodes is maximized when there is no spatial and temporal overlap between the multiple flows. In this case the total number of nodes involved is the sum of the number of nodes involved in each flow. (This arises from the

fact that $\left| \bigcup_{i=1}^{\eta} A_i \right| \leq \sum_{i=1}^{\eta} |A_i|$.)

The area of the bounding box, Figure 5, as a function of the number of hops between S and D , h , is $A(h) = 2r^2(h+1)$. The total number of nodes in the bounding box $N(h) = A(h)\rho$, where ρ is the density. The number of forwarding nodes $F(h) = h-1$. The number of guard nodes $G(h) = 0.55N(h) - F(h)$ [37]. The number of other nodes $O(h) = N(h) - (F(h) + G(h))$. Next we compute the total expected energy over all the flows for both Baseline-OD and On-Demand SLAM, ignoring the energy of S and D .

Baseline-OD: The expected energy expended by the entire set of forwarding nodes for a single flow is

$$E[\varepsilon_{\{f\},1,base}] = \varepsilon_{f,base} \cdot E[F(h)] \quad (14)$$

On-Demand SLAM: The expected energy expended by the entire set of nodes in the bounding box for a single flow is

$$E[\varepsilon_{\{N\},1,SLAM}] \leq \varepsilon_{f,SLAM} \cdot E[F(h)] + \varepsilon_{g,SLAM} \cdot E[G(h)] + \varepsilon_{o,SLAM} \cdot E[O(h)] \quad (15)$$

These computations depend on the value of $E[h]$. To compute $E[h]$, consider the source S at the center of a set of concentric circles – the first one of radius r (the transmission range), the second of radius $2r$, and so on. The nodes in the second ring are two hops away from S , those in the third ring are three hops away, and so on. Let the number of nodes in ring i be m_i . Assuming a Poisson process for distribution of the nodes with rate ρ . $m_i = \pi r^2 \rho$ when $i=1$ and $m_i = \pi[(i+1)r]^2 - (ir)^2$ when $i>1$. In general, through simplification, $m_i = \pi r^2 \rho(2i-1)$

$$E[h] = \sum_{i=1}^{\Gamma} i \cdot \frac{m_i}{n}, \text{ where } \Gamma \text{ is such that } \sum_{j=1}^{\Gamma} m_j = n \quad (16)$$

In Figure 6, we plot the extra delay of SLAM over Baseline-LM for cases I (Equation (7)) and II (Equation (12)) above with $T_{data} = 7\text{ms}$ and $\tau = 1\text{ms}$. The figure shows that the additional delay due to SLAM increases linearly with the number of hops for Case I while it remains constant for Case 2.

The expected value of the total energy expended (for all the η concurrent flows) is upper-bounded by η times the energy for a single flow.

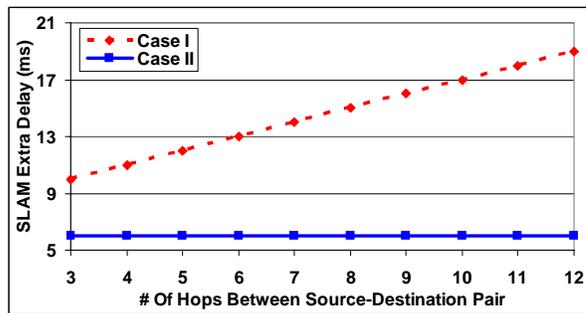


Figure 6: Extra delay due to SLAM over Baseline-LM

5 Simulation Results

We use the *ns-2* simulator [42] to simulate a data exchange protocol over a network with local monitoring enabled according to the protocol in [37]. We simulate two scenarios individually without A-SLAM (the *baseline*) and with A-SLAM. The baseline is an implementation of a state-of-the-art local monitoring protocol presented in [37]. A-SLAM scenario is built on top of the baseline scenario to provide sleep-wake service for the guards. Nodes are distributed randomly over a square area with a fixed average node density, 100 nodes over $204\text{m} \times 204\text{m}$. Each node acts as a source and generates data according to a Poisson process with rate μ . The destination is chosen at random and is changed using an exponential random distribution with rate λ . A route is evicted if unused for $T_{\text{Out}_{\text{Route}}}$ time. The experimental parameters are given in Table 1. The results are averages over 30 runs. The malicious nodes are chosen at random so that they are more than 2 hops away from each other.

Table 1: Default simulation parameters

Parameter	Value	Parameter	Value	Parameter	Value
Tx Range (r)	30 m	Destination change rate (λ)	0.02 per sec	Simulation time	1500 sec
Number of neighbors (N_B)	8	Number of malicious nodes (M)	4	Fraction of data monitored (f_{dat})	0.6
$T_{\text{Out}_{\text{Route}}}$	50 sec	Packet generation rate (μ)	0.1 per sec	Watch time (T_w)	30ms
Channel BW	40 kbps	Warm up time (T_{warmup})	5ms	Number of nodes (N)	100

Adversary model: We are simulating a selective forwarding attack launched by a group of malicious nodes that collude and establish wormholes in the network [38], [44]-[46]. During the wormhole attack, a malicious node captures packets from one location in the network, and “tunnels” them to another malicious node at a distant point, which replays them locally. This makes the tunneled packet arrive either sooner or with a lesser number of hops compared to the packets transmitted over normal multihop routes. This creates the illusion that the two end points of the tunnel are very close to each other. The two malicious end points of the tunnel may use it to pass routing traffic to attract routes through them and then launch a variety of attacks against the data traffic flowing on the wormhole, such as selectively dropping the data packets. Unless otherwise mentioned, each node selectively drops a packet passing through it with uniform probability of 0.6.

Variable Input metrics: (i) *Fraction of data monitored (f_{dat})* – each guard node randomly monitors a given fraction of the data packets. At other times, it can be asleep from the point of view of a guard’s responsibility. (ii) *Data traffic load (μ)*. (iii) *Number of malicious nodes (M)* – the number of malicious nodes that collude to establish wormholes and afterwards selectively drop the data.

Output metrics: *Delivery ratio* – the ratio of the number of packets delivered to the destination to the number of packets sent out by a node averaged over all the nodes in the network. *Watch buffer size* – the runtime count of the maximum size of the watch buffer being maintained at a guard, measured in number of entries. The maximum is taken over all the guards. *% Average monitor wakeup time* – the time a node has to wakeup specifically to do monitoring averaged over all the nodes as a percentage of the simulation time. *Average end-to-end delay* – the time it takes a data packet to reach the final destination averaged over all successfully received data packets. *% True isolation* – the percentage of the total number of malicious nodes that is isolated. *% False isolation* – the

percentage of the total number of nodes that is isolated due to natural collisions on the wireless channel. *Isolation latency* – the time between when the node performs its first malicious action to the time by which *all* the neighbors of the node have isolated it averaged over all isolated malicious nodes.

Note that our goal is not to show the variation of the output metrics with the input parameters for local monitoring, since that has been amply covered in [36][37]. Our goal here is to study the relative effect on local monitoring with ASLAM and without.

5.1 Effect of fraction of data monitored

The amount of data traffic is typically several orders of magnitude larger than the amount of control traffic. It may not be reasonable for a guard node to monitor all the data traffic in its monitored links. Therefore a reasonable optimization is to monitor only a fraction of the data traffic. In this set of experiments, our goal is to investigate the effect of this optimization quantified by the fraction f_{dat} on the output metrics.

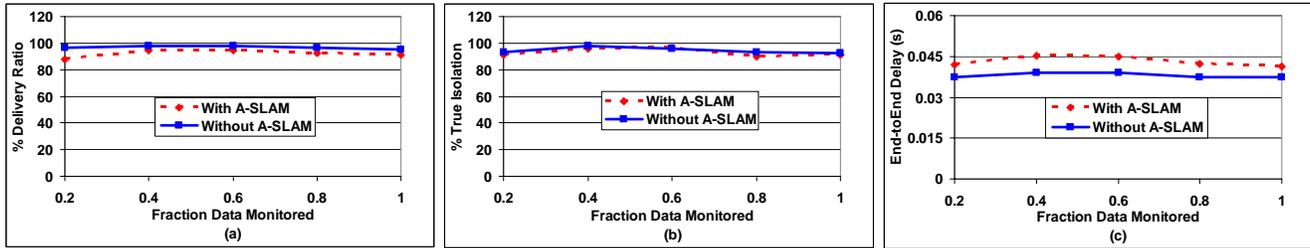


Figure 7: Effect of fraction of data monitored on (a) delivery ratio, (b) % true isolation, and (c) end-to-end delay for both SLAM and the baseline

Figure 7 shows the variations of delivery ratio, % true isolation, and end-to-end delay as we vary f_{dat} . Figure 7 (a) shows that the % delivery ratio is almost stable above 90% irrespective of the value of f_{dat} . This desirable effect is achieved by proper selection of the *MalC* increment for each value of f_{dat} . The *MalC* increment is designed with an inverse relation to the f_{dat} . Figure 7 (b) shows that the % of true isolation is almost stable as we vary f_{dat} due to the same reasoning as for Figure 7 (a). Importantly, the delivery ratio and the % true isolation in A-SLAM are close to the baseline for all values of f_{dat} . However, the results in A-SLAM are slightly worse than those of the baseline. This is because some of the data packets are additionally dropped in A-SLAM by forwarding, destination, or guard nodes that happen to be asleep when the data packet arrives. This unwanted sleep may occur due to collision in the sleep-wake control channel which prevents the respective nodes from waking up. Although the control channel is a separate channel contention still occurs, where a guard of two consecutive links are sent separate wake-up signals concurrently. Figure 7 (c) shows that the end-to-end delay is slightly higher for A-SLAM due to the additional warm up time required when the source sends a packet to the first hop.

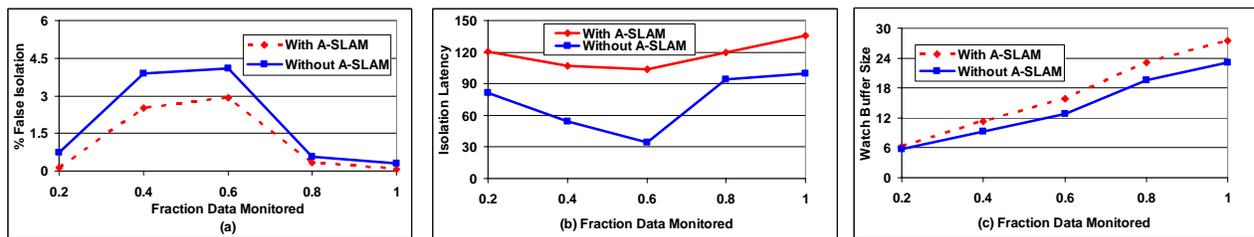


Figure 8: Effect of f_{dat} on (a) % false isolation; (b) isolation latency; (c) watch buffer size for local monitoring with and without SLAM

Figure 8 shows the variations of % false isolation (Figure 8 (a)), isolation latency (Figure 8 (b)), and watch buffer size (Figure 8 (a)) as we vary f_{dat} . Figure 8 (a) clearly shows that the maximum watch buffer size in ASLAM is close to the base line. Figure 8 (a) shows that the % false isolation is slightly lower for ASLAM than the baseline since some of the packets that may falsely identify a node as malicious may get lost in ASLAM due to unwanted sleep, which lowers the % false isolation. Figure 8 (b) shows that the isolation latency in ASLAM is higher than that of the baseline due to data packets that may be missed while a node is in unwanted sleep. This causes the malicious counters at the guards to reach the threshold value of detection slower, and consequently increases the time of detection and isolation. Figure 8 (c) shows that as the fraction of data monitored increases, the watch buffer size increases due the increase in the number of packets monitored. This shows the benefit in terms of

overhead by monitoring only a small fraction of packets while maintaining almost the same detection coverage (Figure 7). However, note that even though the watch buffer sizes in A-SLAM and the baseline are close, that in A-SLAM is slightly higher. This is due to the extra delay in packet forwarding in A-SLAM due to warm up of the nodes before sending the data. This delay causes the monitored packets to stay longer in the watch buffer thereby increasing its size.

5.2 Effect of number of malicious nodes

In this section we study the relative effect of varying the number of malicious nodes on the output metrics.

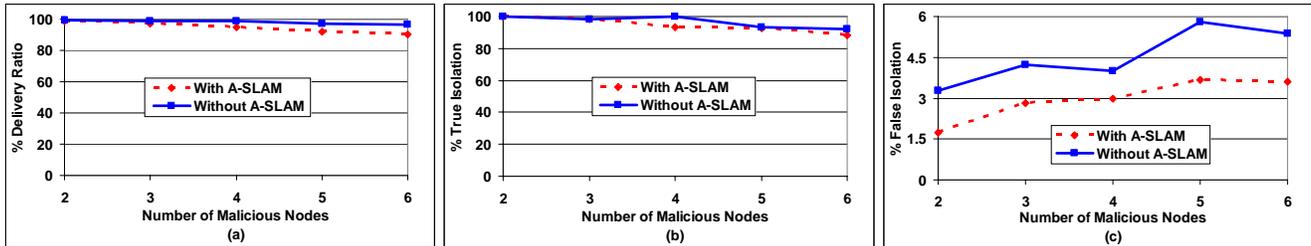


Figure 9: Effect of number of malicious nodes on (a) % delivery ratio, (b) % true isolation, and (c) % false isolation for local monitoring with and without ASLAM

Figure 9 shows the variations of % delivery ratio, % true isolation, and % false isolation as we vary the number of malicious nodes (M). Figure 9 (a) shows that the % delivery ratio slightly decreases as M increases. This is due to the packets dropped before the malicious nodes are detected and isolated. As the number of malicious nodes increases, this initial drop increases and thus the delivery ratio decreases. Figure 9 (b) shows that the % true isolation also slightly decreases as we increase M . This is because the number of available guards in the network decreases as more and more nodes get compromised. These two metrics in A-SLAM are slightly lower than those of the base line due to the unwanted sleep described in the explanation of Figure 7. Figure 9 (c) shows that the % false isolation increases as we increase M . This is because not all guard nodes come to the decision to isolate a malicious node at the same time. Therefore, a given guard node may suspect another guard node when the latter isolates a malicious node but the former still has not. The occurrence of this situation increases with M and hence the % of false isolation increases with M . For example, a guard node G_i detects a malicious node M earlier than the other guard nodes for the link to M . G_i subsequently drops all the traffic forwarded to M and is therefore suspected by other guard nodes for M . This problem can be solved by having an authenticated one-hop broadcast whenever a guard node performs a local detection. The % false isolation in A-SLAM is lower than that of the baseline. Again, this is because some of the packets that may falsely identify a node as malicious may get lost in ASLAM due to unwanted sleep.

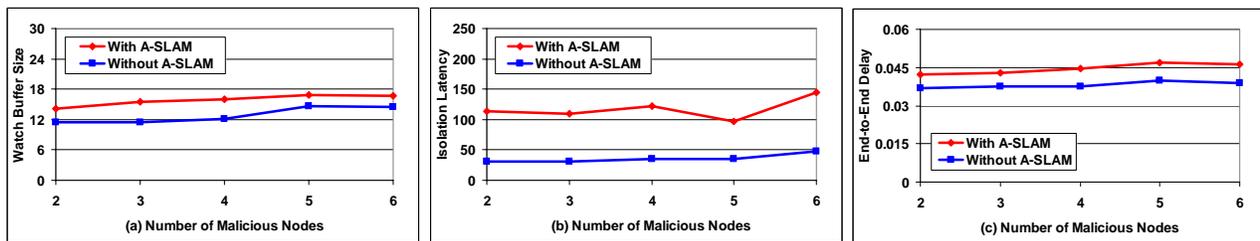


Figure 10: Effect of number of malicious nodes on (a) watch buffer size, (b) isolation latency, and (c) end-to-end delay with and without ASLAM

Again, Figure 10 confirms that ASLAM has a very close behavior on the output metrics as the baseline. The slight change is due to the same effects that we show in the explanation of Figure 7 and Figure 8.

5.3 Effect of data traffic load (μ)

In this section we study the effect of varying the data traffic load on the output metrics.

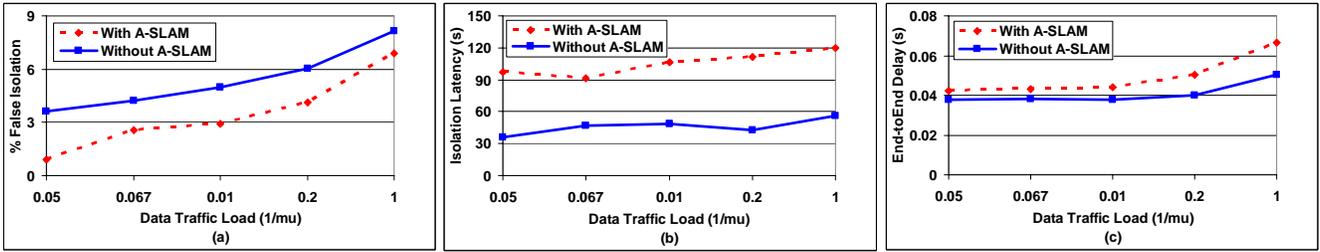


Figure 11: Effect of data traffic load on (a) % false isolation, (b) isolation latency, and (c) end-to-end delay for local monitoring with and without SLAM

Figure 11 shows the variations of % false isolation, isolation latency, and end-to-end delay as we vary the data traffic load ($1/\mu$). Figure 11 (a) shows that the % false isolation increases as the traffic load increases ($1/\mu$ increases). As the traffic load increases, the probability of collision increases. This in turn increases the possibility of false accusation since a guard, say G, may falsely accuse a node, say A, of not forwarding a packet if either G has a collision when A forwards or A has a collision while receiving the packet. The explanation of the relative performance with and without A-SLAM is the same as for Figure 9 (c). Figure 11 (b) shows that the isolation latency increases as the traffic load increases. As the traffic load increases, the *MalC* increment decreases. This causes the *MalC* threshold to be reached slower at a guard node, which results in increasing the isolation latency of the malicious nodes. Also the higher traffic load lays it open to the possibility of some packets being missed due to natural collisions and thereby preventing the increment to the malicious counter and therefore, reaching the threshold faster. Note that the isolation latency in A-SLAM is higher than that of the baseline because of the additional packets missed due to the unwanted sleep. Figure 11 (c) shows that end-to-end delay increase as the traffic load increases due the higher contention for the channel. The relative explanation of end-to-end delay with and without A-SLAM is the same as that of Figure 8.

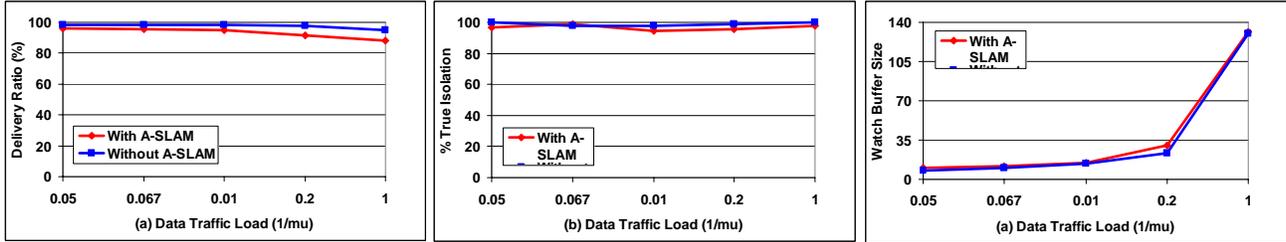


Figure 12: Effect of data traffic load on (a) % delivery ratio, (b) % true isolation, and (c) watch buffer size with and without SLAM

Again, Figure 12 confirms that ASLAM has a very close behavior on the output metrics as the baseline. The slight change is due to the same effects that we show in the explanation of Figure 7 and Figure 8.

5.4 Wakeup time variations

In this section, we study the effect of varying the fraction of data monitored (f_{dat}), the number of malicious nodes (M), and the data traffic load (μ) on the percentage of time that a node needs to stay awake using A-SLAM to fulfill the quality of service measures imposed by the underlying local monitoring scheme.

Figure 13 (a) shows that the percentage of wakeup time required for monitoring increases as the fraction of monitored data increases due to the increase in the number of data packets that a node needs to overhear in its neighborhood. Figure 13 (b) shows that the percentage of wakeup time decreases as we increase the number of malicious nodes. As the number of malicious nodes increases, the number of data packets in the system decreases since the malicious nodes are isolated and disallowed from generating data packets. Therefore, the number of packets that need to be monitored decreases, which results in a decrease in the average percentage of wakeup monitor time. Figure 13 (c) shows that the average percentage of monitoring wakeup time increases as the data traffic load increases due the increase of data packets that need to be monitored.

Overall, compared to the no sleeping case, A-SLAM saves 30%-129% listening energy for different amounts of data traffic load (μ).

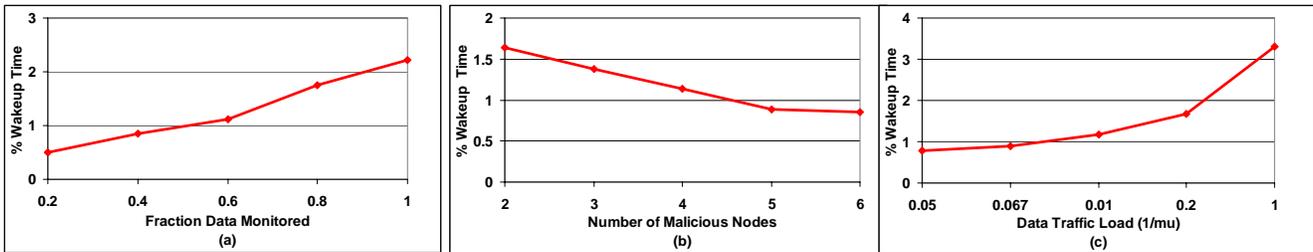


Figure 13: Variations on the percentage of monitoring wakeup time as we vary (a) the fraction of data monitored (f_{dat}); (b) number of malicious nodes (M); and (c) data traffic load (μ)

5.5 Effect of distance on delay

We evaluate the variations of the end-to-end delay with the number of hops between the source and destination pairs. Figure 14 (a) shows that the end-to-end delay in A-SLAM is always higher than that of the baseline due to the warm-up time needed to wake up the nodes before sending the data. However, due to the scheduling strategy in A-SLAM in which each node sends a wake-up signal at the earliest possible opportunity (Section 3.4.3), the warm-up time is only in the critical path at the first hop and therefore, the delay is not cumulative with the number of hops. Figure 14 (b) shows that the difference in the end-to-end delay has a horizontal trend – it fluctuates between 6.5 and 10 ms due to the randomness in the traffic pattern and the location of the source-destination pair. The standard deviation in the difference is only 9.1%, expressed as a percentage of the baseline delay. This horizontal trend of the additional delay due to SLAM follows the trend obtained analytically in Section 4.2 for the case when $(T_{control} + T_{warmup}) < T_{data}$ which is true in these simulation settings.

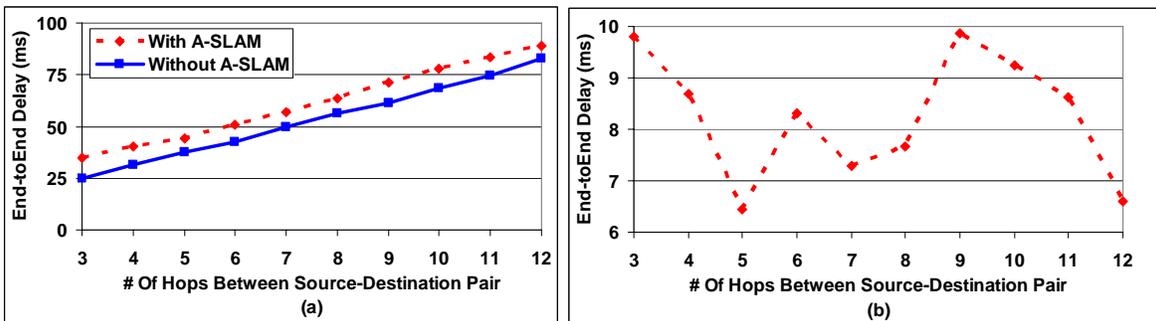


Figure 14: (a) Variation of the end-to-end delay with the hop count for local monitoring with and without A-SLAM; (b) the difference in the end-to-end delay with and without A-SLAM

6 Conclusion

In this paper, we have presented a protocol called SLAM to make local monitoring in sensor networks energy-aware while maintaining the detection coverage. We classify the domain of sleep-wake protocols into three classes and SLAM correspondingly has three manifestations depending on which baseline sleeping protocol (BSP) is used in the network. For the first class (synchronized sleep-wake), local monitoring needs no modification. For the second class (connectivity-coverage preserving sleep-wake), local monitoring can call the BSP with changed parameter values. For the third class (on-demand sleep-wake), adapting local monitoring is the most challenging and requires hardware support as low-power or passive wake-up antennas. We propose a scheme whereby before communicating on a link, a node awakens the guard nodes responsible for local monitoring on its next hop. We design the scheme to work with adversarial node behavior. We prove analytically that On-Demand SLAM does not weaken the security property of local monitoring. Simulation experiments bring out that over a wide range of conditions, the performance of local monitoring with SLAM is comparable to that without SLAM, while listening energy savings of 30-129 times is realized, depending on the network load. Our ongoing work is looking at providing security guarantees in mobile ad hoc networks and building trust framework for such networks.

7 References

- [1] W. Zhang and G. Cao, "DCTC: Dynamic Convoy Tree-Based Collaboration for Target Tracking in Sensor Networks," *IEEE Transactions on Wireless Communication*, vol. 3 (5), 2004, pp. 1689-1701.
- [2] S. Patten, S. Poduri, and B. Krishnamachari, "Energy-quality tradeoffs for target tracking in wireless sensor networks," in *second workshop on Information Processing for Sensor Networks (IPSN)*, 2003.
- [3] C. Gui and P. Mohapatra, "Power conservation and quality of surveillance in target tracking sensor networks," in *Proceedings of the 10th annual international conference on Mobile computing and networking (MOBICOM)*, 2004, pp. 129-143.
- [4] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks," in *MobiCom'01*.
- [5] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed energy conservation for ad hoc routing," in *MobiCom*, 2001, pp. 70-84.
- [6] K. Chakrabarty, S. S. Iyengar, H. Qi, and E. Cho, "Grid coverage for surveillance and target location in distributed sensor networks," in *IEEE Transactions on Computers*, 51(12), pp.1448-1453, 2002.
- [7] D. Tian and N. D. Georganas, "A coverage-preserved node scheduling scheme for large wireless sensor networks," in *Proceedings of First International Workshop on Wireless Sensor Networks and Applications (WSNA)*, pp. 32-41, 2002.
- [8] T. Yan, T. He, and J. A. Stankovic, "Differentiated surveillance for sensor networks," in *The First ACM Conference on Embedded Networked Sensor Systems(Sensys)*, pp. 51-62, 2003.
- [9] F. Ye, G. Zhong, S. Lu, and L. Zhang, "Peas: A robust energy conserving protocol for long-lived sensor networks," in the *23rd International Conference on Distributed Computing Systems (ICDCS)*, pp. 169-177, 2003.
- [10] S. Bhattacharya, G. Xing, C. Lu, G.-C. Roman, O. Chipara, and B. Harris, "Dynamic wake-up and topology maintenance protocols with spatiotemporal guarantees," in *Information Processing in Sensor Networks (IPSN)*, pp. 28-34, 2005.
- [11] G. Xing, C. Lu, R. Pless, and J. A. O'Sullivan, "Co-Grid: an efficient coverage maintenance protocol for distributed sensor networks," in *Proceedings of the third international symposium on Information processing in sensor networks (IPSN)*, pp. 414 - 423 , 2004.
- [12] S. Kumar, T. H. Lai, and J. Balogh, "On k-coverage in a mostly sleeping sensor network" in *International Conference on Mobile Computing and Networking (MOBICOM)*, pp. 144-158, 2004.
- [13] J. V. Greunen, D. Petrovic, A. Bonivento, J. Rabaey, K. Ramchandran, and A.S. Vincentelli, "Adaptive sleep discipline for energy conservation and robustness in dense sensor networks," in *IEEE International Conference on Communications*, Vol. 6, pp. 3657 – 3662, 2004.
- [14] F. Koushanfar, A. Davare, D. Nguyen, M. Potkonjak, A. Sangiovanni-Vincentelli. "Low power coordination in wireless ad-hoc networks" in *International Symposium on Low Power Electronics and Design (ISLPED)*, pp. 475 – 480, 2003.
- [15] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks," in the *Proceedings of the 33rd Hawaii International Conference on System Sciences*, 2004, pp. 3005-3014.
- [16] G. Xing, X. Wang, Y. Zhang, C. Lu, R. Pless, and C. Gill, "Integrated coverage and connectivity configuration for energy conservation in sensor networks," in *ACM Transactions on Sensor Networks (TOSN)*, Vol. 1 , Issue 1, pp. 36-72, 2005.
- [17] E. Riedy and R. Szewczyk. "Power and control in networked sensors," <http://webs.cs.berkeley.edu/tos/papers/cs294-8.pdf>, May 2000.
- [18] J. W. Hui, Z. Ren, , and B. Krogh, "Sentry-based power management in wireless sensor Networks," in the *2nd International Workshop on Information Processing in Sensor Networks (IPSN)*, pp. 458-472, 2003.
- [19] W. Ye, J. Heidemann, and D. Estrin, "An energy efficient MAC protocol for wireless sensor Networks," in *INFOCOM*, pp. 1567- 1576, 2002.

- [20] R. Naik, S. Biswas, and S. Datta, "Distributed Sleep-Scheduling Protocols for Energy Conservation in Wireless Networks," in Proceedings of the 38th Annual Hawaii International Conference on System Sciences (HICSS), pp. 285b - 285b, 2005.
- [21] S. Liu, K. Fan, and P. Sinha, "Dynamic Sleep Scheduling using Online Experimentation for Wireless Sensor Networks," in the Proceedings of the Third International. Workshop on Measurement, Modeling and Performance Analysis of Wireless Sensor Networks (SenMetrics), 2005.
- [22] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, "Wireless sensor networks for habitat monitoring," in ACM International Workshop on Wireless Sensor Networks and Applications, pp. 88-97, 2002.
- [23] C. Guo, L. C. Zhong, and J. M. Rabaey, "Low power distributed MAC for ad hoc sensor radio networks," in IEEE Global Telecommunications Conference (GLOBECOM '01), pp. 2944-2948, vol.5, 2001.
- [24] J. Rabaey, J. Ammer, T. Karalar, S. Li, B. Otis, M. Sheets, and T. Tuan, "Picoradios for wireless sensor networks: The next challenge in ultra-low-power design," in the Proceedings of the International Solid-State Circuits Conference, pp. 200-201, 2002.
- [25] J. Silva, J. Shamberger, M. J. Ammer, C. Guo, S. Li, R. Shah, T. Tuan, M. Sheets, J. M. Rabaey, B. Nikolic, A. Sangiovanni-Vincentelli, and P. Wright, "Design methodology for picoradio networks," in the Proceedings of the Design Automation and Test in Europe, pp. 314-323, 2001.
- [26] http://www.austriamicrosystems.com/03products/data/AS3931Product_brief_0204.pdf.
- [27] L. Gu and J.A Stankovic, "Radio-Triggered Wake-Up Capability for Sensor Networks," in Real-Time and Embedded Technology and Applications Symposium (RTAS), pp. 27-36, 2004.
- [28] Chipcon CC1000 Datasheet, Chipcon Inc. <http://www.chipcon.com/files/CC1000DataSheet21.pdf>.
- [29] Y. Huang and W. Lee, "A cooperative intrusion detection system for ad hoc networks," in Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks, pp. 135-147, 2003.
- [30] A. Silva, M. Martins, B. Rocha, A. Loureiro, L. Ruiz, and H. Wong, "Decentralized intrusion detection in wireless sensor networks," in Proceedings of the 1st ACM international workshop on Quality of service & security in wireless and mobile networks, pp. 16-23, 2005.
- [31] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," MobiCom'00, pp. 255-265, 2000.
- [32] S.J. Lee and M. Gerla, "Split Multipath Routing with Maximally Disjoint Paths in Ad Hoc Networks," in IEEE International Conference on Communications (ICC), pp. 3201-3205, 2001.
- [33] A. A. Pirzada and C. McDonald, "Establishing Trust In Pure Ad-hoc Networks," Proceedings of 27th Australasian Computer Science Conference (ACSC'04), pp. 47-54, 2004.
- [34] S. Buchegger, J.-Y. Le Boudec, "Performance Analysis of the CONFIDANT Protocol: Cooperation Of Nodes - Fairness In Distributed Ad-hoc NeTworks," in MobiHoc'02, pp. 80-91, 2002.
- [35] I. Khalil, S. Bagchi, and N. B. Shroff, "Analysis and Evaluation of SECOS, a protocol for Energy Efficient and Secure Communication in Sensor Networks", accepted for publication in Ad Hoc Networks Journal (ADHOC), number of pages: 32, notification date: Dec. 2005.
- [36] I. Khalil, S. Bagchi, and C. Nina-Rotaru, "Dicas: Detection, Diagnosis and Isolation of Control Attacks in Sensor Networks," IEEE/CreateNet SecureComm, pp. 89-100, 2005.
- [37] I. Khalil, S. Bagchi, and N. Shroff, "LiteWorp: A Lightweight Countermeasure for the Wormhole Attack in Multihop Wireless Networks," International Conference on Dependable Systems and Networks (DSN '05), pp. 612-621, 2005.
- [38] C. Karlof and D. Wagner, "Secure Routing in Sensor Networks: Attacks and Countermeasures," in the 1st IEEE International Workshop on Sensor Network Protocols and Applications, pp. 113-127, 2003.
- [39] S. Zhu, S. Xu, S. Setia, and S. Jajodia, "Establishing Pair-wise Keys For Secure Communication in Ad Hoc Networks: A Probabilistic Approach," in the 11th IEEE International Conference on Network protocols (ICNP'03), Atlanta, Georgia, pp. 326- 335, 2003.

- [40] W. Du, J. Deng, Y. Han, and P. Varshney, “ A Pairwise Key Pre-distribution Scheme for Wireless Sensor Networks,” in Proceedings of the 10th ACM conference on Computer and communication security (CCS), pp. 42-51, 2003.
- [41] D. Liu and P. Ning, “Establishing Pair-wise Keys in Distributed Sensor Networks,” in Proceedings of the 10th ACM conference on Computer and communication security (CCS), pp. 52-61, 2003.
- [42] “The Network Simulator ns-2,” At: www.isi.edu/nsnam/ns/
- [43] http://www.xbow.com/products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf.
- [44] Y. C. Hu, A. Perrig, and D.B. Johnson, “Packet leashes: a defense against wormhole attacks in wireless networks,” in Proceedings of the 22nd INFOCOM, pp. 1976-1986, 2003.
- [45] L. Hu and D. Evans, “Using Directional Antennas to Prevent Wormhole attacks,” in Network and Distributed System Security Symposium, 2004.
- [46] L. Lazos, R. Poovendran, C. Meadows, P. Syverson, and L. W. Chang, “Preventing wormhole attacks on wireless ad hoc networks: a graph theoretic approach,” IEEE Wireless Communications and Networking Conference (WCNC), Vol. 2, pp. 1193 – 1199, 2005.