On the Lifetime Analysis of Always-On Wireless Sensor Network Applications

Santosh Kumar, Anish Arora, and Ten H. Lai
Department of Computer Science and Engineering
The Ohio State University
{kumars, anish, lai}@cse.ohio-state.edu

***Abstract*— Majority of papers in the area of wireless sensor networks (WSNs) have an element of energy-efficiency and associated with it an analysis of network lifetime. Yet, there is no common agreement on how to analyze the lifetime of a WSN. As a result, errors are made on both sides. Some underestimate the network lifetime by an order of magnitude, while others end up overestimating the lifetime by a significant factor. This paper presents a first step towards standardizing the lifetime analysis of WSNs. We focus on WSNs deployed for *always-on* applications, where the problem of power management is most severe because the environment needs to be monitored continuously. Underestimation of network lifetime is common when proposing sleep-wakeup schemes, where it is frequently assumed that in the absence of a sleep-wakeup scheme, a sensor node from the Mica family lasts 3-5 days on a pair of AA batteries. We show that the same sensor node can be made to last more than 36 days, even if it is continuously monitoring the environment. Overestimation typically occurs when proposing non-sleep-wakeup power management schemes such as in-network data aggregation. Overestimation occurs because several network activities (e.g periodic routing messages) are assumed to have negligible effect on the network lifetime and therefore are ignored in the lifetime analysis. We use our recent experience in deploying ExScal (a large-scale WSN for intrusion detection) to identify major components in the network lifetime analysis. We then present a careful lifetime analysis of ExScal and show how to analyze the effects of using various non-sleep-wakeup power management schemes such as hierarchical sensing, low-power listening, and in-network data aggregation on the network lifetime. Our lifetime analysis will be useful as a template in analyzing the lifetime of other WSNs deployed for always-on applications.**

I. Introduction

The problem of power management is a major impediment in achieving long-term unattended operation from large-scale wireless sensor networks (WSNs). Therefore, most proposals for new protocols and algorithms for WSNs are energy-aware. To analyze the effect of energy-efficiency present in a new proposal, an analysis of network lifetime is frequently performed. Since there is no common agreement on how to analyze the lifetime of a WSN, some underestimate the network lifetime by an order of magnitude, while others end up overestimating the network lifetime by a significant factor.

Underestimation is common when proposing sleep-wakeup schemes for WSNs. It is frequently assumed that a sensor node from the Mica family [6] lasts 3-5 days on a pair of AA batteries, if it needs to monitor the environment continuously [5], [9]. This is an underestimation by factor of more than 12.

Overestimation typically occurs when proposing non-sleep-wakeup schemes to save energy. An example is [3]. Because this work focused on presenting the power-saving features of XSM (a Mica family sensor node designed for ExScal) to illustrate the potential of the XSM platform, it ignored the effect of several energy-consuming network activities such as periodic control messages on the network lifetime. Taking those factors into consideration will reduce the network lifetime estimate by a factor of more than 2. The story is similar with papers proposing energy-efficient MAC schemes and in-network data aggregation. Our analysis reveals that the maximum lifetime extension achievable in ExScal by data aggregation is less than 9%, which may come as a surprise to several readers. Although these numbers will change as the application parameters change and as the hardware properties will change, the methods to analyze the lifetime will likely remain the same (at least for a given family of platforms and a given class of applications). Therefore, it is important to arrive at common agreement on how to analyze the lifetime of a WSN.

Although simulation tools (e.g. PowerTOSSIM [14]) exist today to obtain an accurate lifetime estimate of an application before it is deployed, they are not a substitute for analysis. Ideally, we should have a standard method for analyzing the network lifetime, whose results should match the lifetime estimate obtained from PowerTOSSIM, which, in turn, should match the actual lifetime observed in the field. To the best of our knowledge, however, there does not exist any work that focuses on presenting a method for the lifetime analysis of a WSN.

In this paper, we take a first step in this direction by identifying major factors in the lifetime analysis of a WSN. We focus on WSNs deployed for *always-on* applications (e.g. intrusion detection [1], shooter localization [15]), where the problem of power management is most severe because the environment needs to be monitored continu-

ously[1].

We perform a careful analysis of ExScal [1], a recently fielded large-scale WSN ($\sim$ 1000 sensor nodes) to detect and classify intruders of different kinds. Our analysis shows that if no power saving techniques are used, each sensor node in ExScal will last 3 days. We further show that each node can be made to last more than 36 days, even if each node is continuously monitoring the environment, by making use of two non-sleep-wakeup power-saving techniques that are already feasible today — Low Power Listening [11] and Hierarchical Sensing. We then show how to analyze the effects of other non-sleep-wakeup power saving techniques such as reducing the frequency of periodic messages and in-network data aggregation, on the network lifetime.

Our lifetime analysis of ExScal will be useful as a template in analyzing the network lifetime of other always-on applications of WSNs. In several papers proposing non-sleep-wakeup power saving techniques, there is a need to analyze a network lifetime. Since the data we report here are from a real deployed project, these data (together with the method) can be used in the lifetime analysis in those works.

We would like to note that the difficulty of underlying mathematics is not the challenge in network lifetime analysis. Rather, it is the incorporation of major factors, which comes from experience with real projects. We are able to identify several major factors in the lifetime analysis missed previously because of our experience with ExScal, the largest WSN deployed on ground till 2004. As the experience of research community with real WSN projects grows, the lifetime analysis will likely become more and more accurate.

Organization of the Paper. In Section II, we discuss several non-sleep wakeup power management schemes that can be used in a WSN deployed for an always-on application. In Section III, we provide an overview of the ExScal application, the sensor platform used in ExScal, and major requirements and features of the ExScal application that affect the network lifetime. In Section IV, we analyzing the lifetime of ExScal, illustrating the lifetime extensions achievable by using various non-sleep-wakeup power management schemes. Section V concludes the paper.

II. Non-Sleep-Wakeup Power Management Schemes

In this section, we discuss some non-sleep-wakeup power management techniques that can be applied to extend the lifetime of a WSN deployed for an always-on application. All of these schemes exploit some redundancy already existing in the network to extend network lifetime without affecting the quality of service offered by the network.

[1]This in contrast to *almost always-off* applications (habitat monitoring [12], subgalacial bed formation [10]), where it is not necessary to monitor the environment continuously because the environment does not change very abruptly.

A. Low Power Listening (LPL)

In an always-on application such as ExScal, most of the time there is no communication in the network. However, the radio can not be turned off on all sensor nodes, because as soon as an event occurs, the event notification message should be quickly propagated to a base station using radios on other sensor nodes that sit in the path of the sensor detecting an event and the base station. Therefore, ideally, we would like to have a radio that can wake up from the sleep mode by hearing a transmission from a neighbor so that it can be put to sleep when there is no communication in its neighborhood. Such a radio, called radio-triggered wakeup radio, was proposed in [4]. However, it is not yet available on current sensor platforms.

Low Power Listening (LPL) proposed in [11] is an approximation to the radio-triggered wakeup. It allows a sensor node to put its radio (and the processor) to sleep mode for a certain interval and wake it up periodically to sample the channel. If the radio detects preamble bytes, it stays awake and extracts the entire packet. Otherwise, it returns to sleep. This feature was implemented on the sensor platform used in ExScal.

One downside of using the low power listening feature is that the sender has to send a preamble at least as long as the sleep period of the radio. So, there is a trade-off in choosing the sleep period, as was pointed out in [11]. We will analyze the effect of this trade-off in Section IV. Despite this trade-off, low power listening feature can significantly extend the lifetime of WSNs deployed for always-on applications because in most such WSNs the communication is rare (less than 10 packets every minute). In Section IV-D, we show that by using LPL we can extend the lifetime of ExScal by 2.6 times.

B. Hierarchical Sensing

The concept of hierarchical sensing was originally introduced in [3] under the name of energy-quality hierarchy. Our contribution here is to provide a more general definition and characteristic of hierarchical sensing so that it can be used in other always-on applications.

In most always-on applications, the environment needs to be monitored continuously. However, keeping all the sensors and the processor continuously active consumes significant energy. For example, in ExScal, if all the sensors and the processor were left continuously active, a sensor node would have lasted less than 5 days, even if the radio was always turned off. If a sensor can sense the environment without having the processor active, then we can significantly extend the network lifetime by putting the processor to sleep until an event is detected. Further lifetime extensions are possible if the following holds for an always-on application:

- The sensing platform used is intended to detect multiple types of events. (In ExScal, the network is required to detect persons on foot and vehicles.)

- All event types are accompanied by a common simple event. (In ExScal, every intruder is a moving object so that the simple event is the movement.)
- A subset of sensors (called wakeup sensors) can detect the simple event. (In ExScal, PIR sensor detects all moving objects.)

A sensor (or a set of sensors) qualifies as a *wakeup sensor*, if it has the following features:

1) It does not need the processor to be active to perform an event detection. Sampling of the environment, signal processing of the sampled data, and thresholding (for detecting an abrupt change in the environment due to an event) can be done in the sensor hardware without involving the processor.
2) It has the hardware circuitry to raise an interrupt that can wakeup a sleeping processor.
3) It can detect the common simple event. (This feature is necessary because otherwise some events can be missed by the network.)
4) It has the longest sensing range of all the sensors mounted on a sensor node. (Again, this feature is necessary because otherwise some events can be missed by the network.)

A sensor platform is said to have the *Hierarchical Sensing* feature if it has at least one wakeup sensor. If a platform has the hierarchical sensing feature, it can just keep the wakeup sensor active continuously and put the processor and all the other sensors to sleep. In case an event is detected by the wakeup sensor, it will wake up the processor, which will further process the sensor data to determine if a real event has occurred, and if so, it will wake up all the sleeping sensors to detect other properties of the event using different sensing modalities. For example, in ExScal, the wakeup sensor can detect the presence of an intruder and the sleeping sensors can help classify the type of the intruder.

For the hierarchical sensor scheme to work, the choice of wakeup sensor is critical. The wakeup sensor should not wakeup the processor very frequently due to false alarms. The best wakeup sensor is the one that draws a small current. For the sleeping sensors, it is important to have a low startup time so that when they are woken up, they do not miss an event. In Section IV-E, we show that by using hierarchical sensing (with PIR sensor as the wakeup sensor) together with LPL we can extend the lifetime of ExScal by 12 times.

C. Other Non-Sleep-Wakeup Schemes

There are several other non-sleep-wakeup schemes that can be used in an always-on application to extend the network lifetime. Some of these are:

1) **Reducing Periodic Messages:** Reducing periodic control messages can result in significant lifetime extensions. The lifetime of ExScal can be increased by 31.6% if there were no periodic messages.
2) **In-Network Data Aggregation:** Aggregating the event detection messages as it flows towards the base station can also extend network lifetime. In ExScal, using data aggregation can result in a lifetime extension of upto 8.91%.
3) **Reduced Control Operations:** Reducing the number of control operations such as wireless reprogramming results in further lifetime extensions.
4) **Reduced Actuations:** Actuations such as blinking LEDs and sounding buzzers can consume significant energy. For example, keeping even one LED continuously active will reduce ExScal's lifetime by more than half.

We provide the details of calculation on how to analyze the effect of the above schemes on the lifetime of a WSN in Section IV.

III. The ExScal Application and the XSM Platform

In this section, we provide an overview of the ExScal application, an overview of the sensor platform used in ExScal (called XSM) and the major requirements (or features) of ExScal that have a significant impact on its lifetime.

The goal in the ExScal application is to deploy a wireless sensor network over a large region to monitor intrusion activities. The network is required to detect different types of intruders breaching the perimeter of the protected region, classify them into some predetermined category (e.g. person, soldier, car, tank), and track their trajectory of intrusion. The network is also required to notify the nearest base station of an intrusion event in less than 2 seconds.

The key issues in ExScal are to minimize the cost of coverage, minimize the power consumption to maximize the network lifetime, provide accurate (i.e. low false alarm rate) and timely (i.e. less than 2 seconds from the occurrence of the event) detection of intrusion events in the face of unavoidable hardware and software failures, and do all of this with low human involvement.

To demonstrate the concept, approximately 1000 XSMs were deployed in a 1,200m×288m rectangular region [8] and intruders such as persons and Sport Utility Vehicles (SUVs) were shown to be detected and classified by the sensor network. At the end of year 2004, this was the largest wireless sensor network in the world deployed on the ground.

A. The XSM Platform

The XSM (Extreme Scale Mote) [3] is a sensor platform developed for the ExScal project. It is a refinement of the Mica 2 platform [2]. Its design was optimized for use in intrusion detection applications. It had three sensors — a 2-axis Magnetometer to detect ferrous materials, a Passive Infrared (PIR) to detect motion, and an Acoustic sensor to detect objects making sounds (e.g. vehicles). The sensing ranges for these sensors for various types of objects appear in Table I.

The current consumption of the major components of XSM appears in Table II. We use mA (milliamperes) for

Sensor	Sensed Object	Sensing Range
Magnetometer	SUV	7 m
PIR	SUV	30 m
	Person	12 m
Acoustic	SUV	30 m

TABLE I

SENSING RANGES (IN METERS) OF THE THREE SENSORS USED IN THE EXSCAL PROJECT

the unit of current consumption. The amount of energy consumed by a component will depend on the amount of time that it is used.

Component	State	Current (in mA)
Processor	active	8
	sleep	0.01
Radio	active	8
	transmit	16
	sleep	0.001
PIR	active	0.292
	sleep	0.001
Acoustic	active	0.575
	sleep	0.001
Magnetometer	active	6.48
	sleep	0.001
One LED	active	2.2
	sleep	0.001
Flash	Read	6.2
	Write	18.4
	Sleep	0.002
Buzzer	active	15
	sleep	0.001

TABLE II

CURRENT CONSUMPTION OF MAJOR COMPONENTS IN THE XSM PLATFORM

B. Factors Affecting ExScal's Lifetime

The major factors affecting the network lifetime of the ExScal are as follows:

1) **Continuous Monitoring:** The region should be continuously monitored so that intruders can be detected instantly. This may require keeping at least one sensor continuously active, consuming significant energy.
2) **Event Notification Requirement:** Intrusion detection events should be communicated to a base station quickly. In the ExScal application, the requirement is to receive event detection notification at the nearest base station within 2 seconds. In order to communicate event-notification messages quickly over a multi-hop wireless sensor network, several, if not all, sensors need to keep their radio in the receive mode either continuously or frequently enough so that they can route an urgent event-detection message towards the base station. This again consumes significant energy.
3) **Periodic Control Messages:** Two middleware services namely, *routing* and *time synchronization* require every XSM to transmit periodic messages. As we will see in Section IV-F, sending periodic control messages consumes significant energy.
4) **One Time Control Operations:** There are several one time activities performed in the ExScal application. The major ones among them are wireless reprogramming and localization. These operation require the sensor nodes to be active for a long duration (on the order of tens of minutes), send a large number of messages (in reprogramming), and perform actuation activities (e.g. sounding buzzers). All of these consume significant energy.
5) **Frequency of Events:** Every event requires the sensors near the event to not only stay awake for few seconds to detect the event but also to transmit messages in a multi-hop sensor network, and potentially route other XSM's messages. Staying awake with the processor and all the sensors active consumes significant energy.

Each of the above factors dictate which non-sleep-wakeup power management schemes can be used in ExScal and which ones are not usable. For example, LPL can be used in ExScal but the periodic sleeping time of the radio should be low enough (less than 400 ms) to satisfy the 2 second event notification latency.

IV. ANALYZING THE LIFETIME OF EXSCAL

In this section, we develop a template to analyze the lifetime of an always-on application and apply it to analyze the lifetime of ExScal. We first discuss some key assumptions needed in the lifetime analysis, then we derive the parameters needed in the lifetime analysis. Next, we use these parameters to derive the network lifetime in the fully active mode, when using the LPL feature, and when using the hierarchical sensing feature. Finally, we use our framework to analyze the effects of other non-sleep-wakeup schemes such as varying the frequency of periodic control messages, performing in-network data aggregation, tuning the number of wireless reprogrammings, and controlling the amount of actuations performed in the network, on the lifetime of a WSN.

We define the lifetime of a WSN to be the time period during which the network continuously satisfies the application requirement. The application requirement can be stated in various forms. One simple way to express the requirement of an always-on application is in terms of the degree of coverage and the notification latency. For example, in ExScal, all intruders were required to be detected by the network at least 5 times in their trajectory through the network (in order to perform detection with a high probability) and the event notification was required to reach the closest base station in at most 2 seconds. In this paper, we derive a lower bound on the lifetime of a WSN. The purpose of doing so is to allow some buffer so that even if some factors are missed in the analysis (which almost always are), the network has a high likelihood of lasting at least as long as predicted by the analysis.

A. Key Assumptions Needed in Lifetime Analysis

In analyzing the lifetime of any WSN, some key assumptions need to be made based on the expected use of the network. We make the following assumptions for ExScal, with the goal of deriving a conservative estimate of its lifetime:

1) The lifetime of the network is determined by the heaviest-loaded XSM. This is an XSM close to the base station. This is conservative because ExScal network will continue to meet its requirements even if all XSMs within one hop of a base station fail.
2) Every hour, 6 intrusion events occur in the vicinity of the heaviest-loaded XSM[2]. With equal probability, the event can be the intrusion of a person or that of an SUV. Further, the intruders are assumed to follow the least-covered path through the network. With this last assumption, the number of sensors detecting an intruder in the ExScal network is given by the values in Table III.
3) Every time an event occurs, the heaviest-loaded XSM keeps its processor as well as all three of its sensors active for an average of 10 seconds. This is because, on average, a slow moving target (a person on foot) will spend an average of 10 seconds in the sensing range of a sensor.
4) One-eighth of the event detection messages generated in the vicinity of the heaviest-loaded XSM are routed by the heaviest-loaded XSM[3].
5) The average number of times a data packet is transmitted for reliable delivery across a single hop is $1/0.7 = 1.43$. This is because the per-hop reliability of data packets was 0.7 in ExScal[4]. If we assume the probability of losing a packet is independent and identically distributed, the expected number of times a packet needs to be transmitted for successful delivery is $1/p$ (expectation of geometric distribution [13]), where p is the probability of success in each transmission[5].
6) Control messages are not retransmitted.
7) We assume a fixed packet length of 36 bytes, which is the maximum length of a packet in B-MAC [11] (the MAC protocol used in ExScal), and that it takes 20 ms to transmit a packet. In practice, event detection messages are very short, and therefore the packet length will be smaller and so will the transmission time.

The assumptions for any other always-on application will mostly be along the lines of the above assumptions with some changes specific to that application.

B. Parameters for Lifetime Analysis

In this section, we derive all the parameters used in analyzing the lifetime of a WSN. We also derive the values of these parameters for the ExScal application. This derivation of values will act as an example when deriving the values of these parameters for any other always-on application.

The notations for the parameters used in lifetime analysis appears in Table IV. In the following, we derive the values or expressions for those parameters whose values are not straightforward to calculate or whose values are variable (in the context of the ExScal application).

t_{slp}^{lpl}: Sleep period of the radio in the LPL mode. It can take a value of 0.01, 0.02, 0.05, 0.1, 0.2, 0.4, 0.8, or 1.6 seconds [11]. We do not use sleep periods of 0.8 seconds and 1.6 seconds in our analysis because the notification latency requirement of 2 seconds can not be satisfied with these sleep periods.

i_{lpl}: Amortized current drawn by an XSM in the LPL mode. Its value depends on the following four factors:

1) t_{slp}^{lpl} - The sleep period used for the radio (listed above).
2) i_{slp}^{lpl} - The average current consumed by an XSM when the processor and radio are sleeping (excluding the current consumed by any active sensor). By measurement on an XSM, we found that i_{slp}^{lpl} =0.1974 mA[6].
3) i_{sample}^{lpl} - The average current consumed by an XSM in sampling the channel every time the radio wakes up. By measurement on an XSM, we found that $i_{sample}^{lpl} = 9.6571$ mA[7].
4) t_{sample}^{lpl} - Time taken to sample the channel every time the radio wakes up to sample the channel. By measurement on an XSM, we found that $t_{sample}^{lpl} = 0.014$ seconds.

Using the above values, we obtain

$$\begin{aligned} i_{lpl} &= \frac{i_{slp}^{lpl} * t_{slp}^{lpl} + i_{sample}^{lpl} * t_{sample}^{lpl}}{t_{slp}^{lpl} + t_{sample}^{lpl}} \\ &= \frac{0.1974 * t_{slp}^{lpl} + (9.6571 * 0.014)}{t_{slp}^{lpl} + 0.014}. \end{aligned} \quad (1)$$

[2]This event rate is higher than what ExScal was required to handle.

[3]This is because grid routing [16] is used in ExScal, which balances the routing load on multiple routes and because of the topology used in ExScal [8].

[4]The per-hop reliability was close to 1 in the absence of interference but was 0.7 in the presence of interference (as is the case in an operating sensor network).

[5]The value of 1.43 is an approximation. The actual value will be lower than 1.43 because the maximum times a packet was transmitted was 3, whereas in geometric distribution the maximum number of trials is assumed to be infinite (trials are made until there is a success).

[6]This is a significantly higher value than expected in sleep mode. We expected it to be close to 0.035 mA. The higher value of 0.1974 mA is because in the LPL mode, the XSM wakes up every 27.6 ms. Every time it wakes up, it follows the following pattern: it consumes 0.035 mA during the 27.6 ms that the XSM is sleeping, it consumes 0.4 mA for the next 0.8 ms, 0.75 mA for the next 2.25 ms, and 6.35 mA for the next 0.5 ms, after which the 27.6 ms of sleep period follows. With careful configuration and some hardware improvements, the value of i_{slp}^{lpl} may be brought down to 0.035 mA, which will significantly increase an XSM's lifetime.

[7]Similar values for i_{sample}^{lpl} were reported in [3].

Intruder	# of PIRs	# of Acoustic	# of Magnetometers
SUV	30	30	5
Person	10	None	None

TABLE III

NUMBER OF SENSORS DETECTING AN INTRUDER TYPE IN THE EXSCAL APPLICATION

Identifier	Meaning	Value
$e_{battery}$	Usable capacity of the batteries	1800 mA-hr
i_{act}^{proc}	Current drawn by the processor in active mode	8 mA
i_{act}^{rad}	Current drawn by the radio in active mode	8 mA
i_{tx}^{rad}	Current drawn by the radio in transmit mode	16 mA
t_{tx}^{pkt}	Time to transmit a packet	0.02 s
t_{slp}^{lpl}	Sleep period of the radio in the LPL mode	variable
i_{lpl}	Amortized current drawn by an XSM in the LPL mode	variable
m_{pr}	# control messages transmitted every hour	240
m_e	# event-detection messages generated by the heaviest-loaded XSM every hour	variable
m_{pp}	# event-detection messages routed by the heaviest-loaded XSM every hour	variable
i_{act}^{msg}	Current drawn in transmitting 1 packet when the radio is in active mode (no LPL)	variable
i_{lpl}^{msg}	Current drawn in transmitting 1 packet when the radio is in LPL mode	variable
i_{awk}^{event}	Amortized current drawn in staying awake due to events	variable
i_{sensor}	Current drawn by continuously active sensors	variable
f_{repr}	# reprogramming is performed wirelessly	6
e_{repr}	Energy spent in 1 wireless reprogramming	18.22 mA-hr
$e_{localization}$	Energy spent in 1 localization	4 mA-hr

TABLE IV

NOTATION FOR PARAMETERS USED IN THE LIFETIME ANALYSIS AND THEIR VALUES IN EXSCAL. THE VALUES OF THE PARAMETERS THAT ARE VARIABLE ARE DERIVED IN SECTION IV-B.

m_{pr}: Number of control messages transmitted every hour. With the configuration used in the ExScal application, every XSM sends 3 routing messages and 1 time synchronization messages every minute, for a total of 240 messages every hour. Therefore, in ExScal, $m_{pr} = 240$.

m_e: Number of event-detection messages generated by the heaviest-loaded XSM every hour. Two messages are generated for every sensor for every event. It means an XSM detecting an event sends 2 messages per sensor per event, for a total of 6 messages. With our assumption of 6 events per hour and a per-hop retransmission factor of 1.43 per event-detection message, $m_e = 52$.

m_{pp}: Number of event-detection messages routed by the heaviest-loaded XSM every hour.The heaviest-loaded XSM is the one close to the base station. Every event generates 2 messages per sensor from every XSM that detects this event. The number of XSMs that detect an intruder type appears in Table III. These numbers depend on the topology of the sensor network deployed [8] and the assumption that intruders always cross through the network via least-covered paths. Assuming both the intruder types are equally likely to occur, every event results in the generation of $2 * ((30 + 30 + 5) + (10 + 0 + 0))/2 = 75$ messages (using the values from Table III). The routing used in the ExScal application [16] balances the routing load on 8 XSMs that are within one hop of the base station. Therefore, the most energy constrained XSM routes an average of 10 messages for every event. Assuming a retransmission factor of 1.43, it will transmit an average of 15 messages for every event. Since 6 events are assumed to occur every hour, 90 event detection messages are routed by the most energy constrained XSM every hour. Therefore, in ExScal, $m_{pp} = 90$.

i_{act}^{msg}: Current drawn in transmitting 1 packet when the radio is in active mode (no LPL). Its value is given

by the following expression:

$$\begin{aligned} i_{act}^{msg} &= \frac{(m_{pr}+m_e+m_{pp})*t_{tx}^{pkt}*(i_{tx}^{rad}-t_{act}^{rad})}{3600} \\ &= \frac{(240+52+90)*0.02*(16-8)}{3600} \\ &= 0.02\text{mA}. \end{aligned} \quad (2)$$

i_{lpl}^{msg}: Current drawn in transmitting 1 packet when the radio is in LPL mode. Its value depends on the sleep period, t_{slp}^{lpl} used by the radio. More specifically,

$$\begin{aligned} i_{lpl}^{msg} &= \\ &\frac{(m_{pr}+m_e+m_{pp})*(t_{tx}^{pkt}+t_{slp}^{lpl})*(i_{act}^{proc}+i_{tx}^{rad})}{3600} \\ &= \frac{(240+52+90)*(0.02+t_{slp}^{lpl})*(8+16)}{3600}. \end{aligned} \quad (3)$$

i_{awk}^{event}: Amortized current drawn in staying awake due to events. Its value depends on two factors:

1) t_{awk} - Time (in seconds) that the heaviest-loaded XSM is awake every hour due to events. We assume that an XSM close to the base station stays awake for 10 s after the occurrence of an event. This may be for routing all event detection messages. With the assumption of 6 events every hour, $t_{awk} = 60$ seconds in ExScal.
2) $i_{sensors}^{slp}$ - The current consumed in the active mode by all the non-wakeup sensors. From Table II, we obtain, $i_{senors}^{slp} = 0.595+6.48 = 7.075$ mA.

With the above values and the fact that the processor is also awake when the XSM is awake due to events, we obtain,

$$\begin{aligned} i_{awk}^{event} &= \frac{t^{awk}*(i_{act}^{proc}+i_{sensors}^{slp})}{3600} \\ &= \frac{60*(8+7.075)}{3600} = 0.2512\text{mA}. \end{aligned} \quad (4)$$

i_{sensor}: Current drawn by continuously active sensors. Its value depends on which sensors are kept continuously active. The current consumption for the sensors used on the XSM appear in Table II.

e_{repr}: Energy spent in 1 wireless reprogramming. One reprogramming of 55 kByte program (the size of ExScal program) requires every XSM to stay awake for approximately 45 minutes. Assuming the XSMs are not in the LPL mode during reprogramming, just staying awake for 45 minutes consumes 18 mA-hr of energy[8]. The reprogramming of 55 kByte requires the most constrained XSM to send out 1,942 packets, where each packet has 29 bytes of data. Assuming a retransmission factor of 1.43, approximately 2,771 packets are sent by the most energy constrained XSM. One packet transmission takes 20 ms and consumes an extra current of 8 mA. Therefore, the additional energy spent in transmission of 2771 packets is 0.12 mA-hr (8*2771*0.02/3600.). Writing 55 kByte to flash takes less than 4 seconds. Each second, it consumes 18.4 mA of current. So, total energy consumed in flash writing is less than 0.02 mA-hr. Adding up the energy consumed in staying awake for 45 minutes (18 mA-hr), in transmissions (0.12 mA-hr), and in flash writing (0.02 mA-hr), we get a total of approximately 18.14 mA-hr of energy consumed in 1 reprogramming. Therefore, in ExScal, $e_{repr} =$ 18.14 mA-hr.

$e_{localization}$: Energy spent in 1 localization. As seen in the calculation done above for reprogramming, the energy consumed in staying awake is the dominant part of energy consumption. In ExScal, the XSMs were awake for 10 minutes during localization, consuming 4 mA-hr. Therefore, $e_{localization} =$ 4mA-hr.

[8]Keeping XSMs in the LPL mode during reprogramming will cause a higher energy consumption because the XSMs will need to send long preambles, consuming significant energy. Also, it will take much longer than 45 minutes to reprogram the network if the XSMs are in the LPL mode, because of reduced channel capacity in the LPL mode.

C. Lifetime in the Fully Active Mode

If an sensor node is always in the active mode, its lifetime, ℓ_{hr} (in hours) will be given by:

$$\ell_{hr} = \frac{e_{battery} - f_{repr}*e_{repr} - e_{localization}}{i_{act}^{proc}+i_{act}^{rad}+i_{act}^{msg}+i^{sensor}}. \quad (5)$$

Substituting the following values

$$\begin{aligned} i_{act}^{msg} &= 0.02\text{mA}, \quad \text{(from (2))} \\ i^{sensor} &= 0.292+0.575+6.48 = 7.347\text{mA}, \quad \text{(from Table II)} \end{aligned}$$

and those from Table IV, we obtain $\ell_{hr} = 72.2$ hours (or 3 days) for the lifetime of ExScal.

D. Lifetime When Using Low Power Listening

If we use the low power listening mode (see Section II-A), then the network lifetime, ℓ_{hr} (in hours) is given by

$$\ell_{hr} = \frac{e_{battery} - f_{repr}*e_{repr} - e_{localization}}{i_{slp}^{lpl}+i_{lpl}^{msg}+i_{sensor}+i_{awk}^{event}}. \quad (6)$$

Substituting

$$\begin{aligned} i_{slp}^{lpl} &= \frac{(0.1974*t_{slp}^{lpl})+(9.6571*0.014)}{t_{slp}^{lpl}+0.014}, \quad \text{(from (1))} \\ i_{lpl}^{msg} &= \frac{(382)*(0.02+t_{slp}^{lpl})*(8+16)}{3600}, \quad \text{(from (3))} \\ i^{sensor} &= 0.292+0.575+6.48 = 7.347, \quad \text{(from Table II)} \\ i_{awk}^{event} &= 0.2512\text{mA}, \quad \text{(from (4))} \end{aligned}$$

and those from Table IV, we obtain a graph of the ℓ_{hr} as a function of t_{slp}^{lpl} shown in Figure 1 for the lifetime of ExScal. The lifetime curve is concave because there is trade-off in choosing the wakeup period for the radio in the LPL mode. The higher the wakeup interval, the lower the energy consumption when an XSM is sleeping, but higher the energy consumed in sending a longer preamble. The optimal value of lifetime occurs at 187.99 hours or 7.83 days. This represents an increase by a factor of 2.6 over that in the fully active mode.

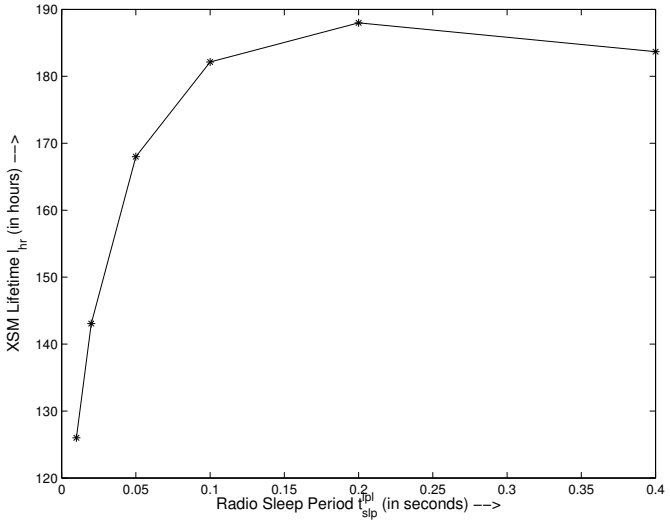

Fig. 1. ExScal network lifetime in the low power listening mode as a function of radio sleep period, t^{lpl}_{slp}.

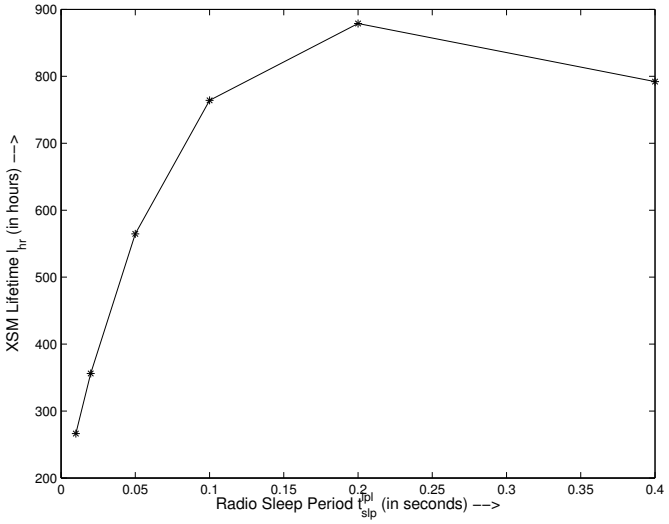

Fig. 2. ExScal network lifetime in the low power listening mode and hierarchical sensing with PIR as the wakeup sensor, as a function of radio sleep period, t^{lpl}_{slp}.

E. Lifetime When Using Hierarchical Sensing With LPL

When using a hierarchical sensor with the LPL mode, the network lifetime is still given by (6), except that now the value of i_{sensor} becomes lower because some other sensors are put to sleep.

In ExScal, the PIR sensor qualifies as a wakeup sensor. Fortunately, this is also the lowest energy consuming sensor, drawing 20 times less current than magnetometer. Figure 2 shows ℓ_{hr} for ExScal as a function of t^{lpl}_{slp}. Here ℓ_{hr} is given by (6) with $i_{sensor} = 0.292$ mA. We observe that the maximum lifetime achievable increases to 878.85 hours (or 36.62 days). This represents an increase by a factor of 4.67 over that achieved by just using the LPL mode and a factor of 12 over that achieved in the fully active mode.

F. Effect of Periodic Control Messages on the Lifetime

In this section, we analyze the effect of varying the frequency of periodic control messages on the network lifetime. To study the effect of periodic messages on the lifetime, we vary the value of m_{pr}. The effect of varying m_{pr} on the lifetime in the fully active mode is straightforward (derive a new value for i^{msg}_{act} in (5)). Analyzing the effect of varying m_{pr} when the network is using the LPL mode is, however, nontrivial, because the optimal lifetime now occurs at different values of t^{lpl}_{slp} depending on the range of values of m_{pr}.

The optimal value of ℓ_{hr} (given by (6)) occurs at $t^{lpl}_{slp} = 0.4$ seconds when $m_{pr} \leq 82$, at $t^{lpl}_{slp} = 0.2$ seconds when $83 \leq m_{pr} \leq 672$, and at $t^{lpl}_{slp} = 0.1$ seconds when $673 \leq m_{pr} \leq 2580$. We plot the optimal values of lifetime when m_{pr} is varied from 0 to 2580 to analyze the effect of periodic messages on the lifetime of ExScal in Figure 3 when hierarchical sensing and LPL both are used. We notice that if there were no periodic messages, ExScal's life increases to 1157.1 hours (or 48.2 days). This represents an improvement of 31.67%.

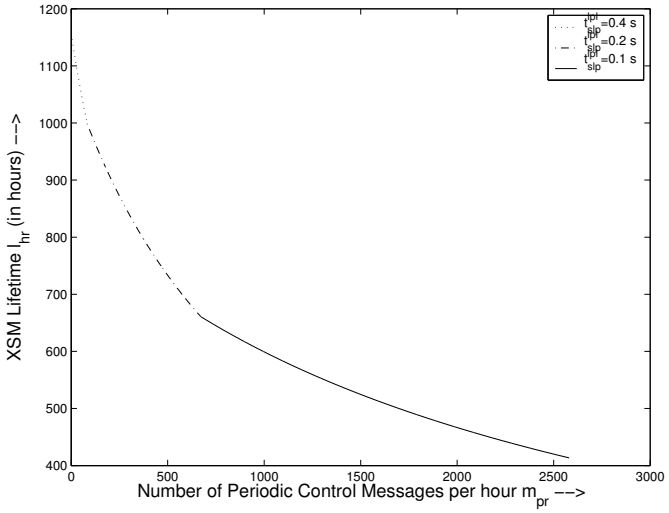

Fig. 3. Optimal ExScal network lifetime in the low power listening mode and hierarchical sensing when the number of periodic control messages (m_{pr}) is varied from 0 to 2580 messages per hour.

G. Effect of In-Network Data Aggregation on the Lifetime

In this section, we analyze the effect of in-network data aggregation on the lifetime of a WSN. The lifetime of a WSN with data aggregation is still given by (5) and (6), but with new values for m_e and m_{pp}. The amount of data aggregation that can be performed in a WSN depends on the application traffic generated and on the topology as well as the routing protocol used. We can perform the following data aggregation in ExScal, assuming the most optimistic scenario:

1) We can combine the detection message for all three sensors at an XSM into one message. This will result in reducing m_e by a factor of 1/3 to 17.
2) We can perform aggregation of detection messages flowing upward in a routing tree. Assuming that both intruder types are equally likely to occur, an average of $(30+10)/2 = 20$ XSMs detect an event (from Table III). Since the routing load is distributed on 8 XSMs, each XSM forwards the detection messages from at most 3 other XSMs, all of whose data can be combined into one packet. Since each XSM generates two messages for every event, separated in time, an XSM close to the base station will need to forward 2 packets per event. Assuming a retransmission factor of 1.43, and 6 events per hour, each XSM close to the base station will forward 18 messages. Therefore, $m_{pp} = 18$.

Substituting $m_e = 17$, $m_{pp} = 18$, and $i_{sensor} = 0.292$ mA in (6), we obtain a graph of ℓ_{hr}, shown in Figure 4, for the lifetime of ExScal, if hierarchical sensing and LPL mode continue to be used. The maximum life achievable is now 954.6 hours (or 39.78 days). This represents an increase of 8.91% in the lifetime of ExScal over that achieved by using only the hierarchical sensing and LPL mode. In practice, it may not be feasible to achieve this extent of data aggregation. So, the increase in lifetime that we can achieve using data aggregation will be at most 8.91%.

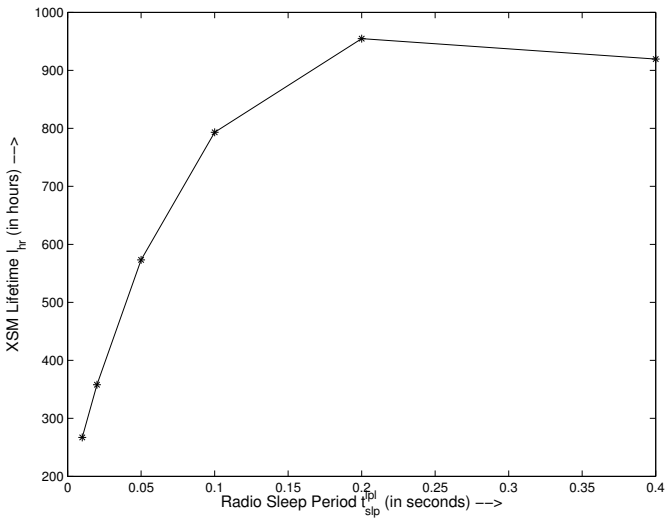

Fig. 4. ExScal network lifetime in the low power listening mode with PIR as the wakeup sensor and in-network data aggregation, as a function of radio sleep period, t_{slp}^{lpl}.

H. Effect of Wireless Reprogramming and Other Operations on the Lifetime

In this section, we analyze the effects of performing frequent wireless reprogramming, blinking LEDs, sounding buzzers, and writing frequently to the flash, on the lifetime of ExScal.

One wireless reprogramming consumes 18.14 mA-hr of energy. From (6), we get that if f_{repr} is reduced from 6 to 5, the lifetime of ExScal will increase from 871.81 hours to 881.2 hours, an increase of approximately 10 hours.

Today's sensors have limited actuation abilities (e.g. bilnking LEDs or sounding a buzzer). In future, sensor nodes are expected to have more actuation abilities. Actuations are often a major source of energy drain. To analyze the effect of actuations on the lifetime, we can use (5) and (6) with a new term in the denominator to represent the average current draw per hour.

For example, an LED draws a current of 2.2 mA on an XSM(see Table II). If one LED is kept active continuously, then the lifetime of ExScal will decrease from 871.81 hours to 412.4 hours (i.e. reduce it by more than half). (Substitute $i_{sensor} = 0.292$ mA in (6) and use a new term with a value of 2.2 in the denominator.) Similarly, to analyze the effect of sounding a buzzer for 1 minute every hour on the lifetime of ExScal, substitute $i_{sensor} = 0.292$ mA in (6) and use a new term with a value of $15/60 = 0.3642$ mA in the denominator (because the buzzer draws 15 mA of current on an XSM). By doing so, we find that an XSM's life will decrease from 871.81 hours to 780.38 hours (a decrease of more than 90 hours of life) as a result of sounding the buzzer for 1 minute every hour by the actuator.

V. Conclusion

Majority of papers in the area of wireless sensor networks (WSNs) have an element of energy-efficiency and associated with it an analysis of network lifetime. But, there is no common agreement on how to analyze the lifetime of a WSN. This paper presents a first step in this direction. Although we have accounted for several major factors in the lifetime analysis by leveraging our experience in deploying a large-scale WSN, we may not have accounted for all major factors. It is a continuous process to keep identifying major factors in network lifetime analysis as the research community experiences with more and more WSN projects. Ideally, we should have a standard energy profile for a given platform and a standard method for network lifetime analysis such that researchers can use it obtain an accurate estimate of network lifetime.

References

[1] *Extreme Scale Wireless Sensor Networking*, http://www.cse.ohio-state.edu/exscal, 2004.
[2] *Mica2 Series Motes*, http://www.xbow.com, 2004.
[3] P. Dutta, M. Grimmer, A. Arora, S. Bibyk, and D. Culler, Design of a Wireless Sensor Network Platform for Detecting Rare, Random, and Ephemeral Events, *ACM IPSN*, Los Angeles, CA, 2005.
[4] L. Gu and J.A. Stankovic, Radio Triggered Wake-Up Capability for Sensor Networks, *In Proceedings of Real Time Applications Symposium,* May 2004.
[5] T. He, S. Krishnamurthy, J.A. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, and L. Hu, Energy-Efficient Surveillance System Using Wireless Sensor Networks, *ACM Mobisys*, Boston, MA, 2004, pp. 270-283.
[6] J. Hill and D. Culler, Mica: A Wireless Platform for Deeply Embedded Networks, *IEEE Micro*, Vol.22 No.6, 2002, pp. 12-24.

[7] J. Hui, Z. Ren, and B. H Krogh, Sentry-Based Power Management in Wireless Sensor Networks, *IPSN*, Palo Alto, CA, 2003.
[8] S. Kumar and A. Arora, Topology and Naming for ExScal Clean Point Demonstration, *ExScal Note Series ExScal-OSU-EN03-2004-12-05*, December 2004.
[9] S. Kumar, T.H. Lai, and J. Balogh, On k-Coverage in a Mostly Sleeping Sensor Network, *ACM MobiCom*, Philadelphia, PA, 2004, pp. 144-158.
[10] K. Martinez, J.K. Hart, and R. Ong, Environmental Sensor Networks *IEEE Computer*, Vol.37, No.8, August 2004, pp. 50-56.
[11] J. Polastre, J. Hill, and D. Culler, Versatile Low Power Media Access for Wireless Sensor Networks, *ACM SenSys*, Baltimore, MD, 2004.
[12] J. Polastre, R. Szewczyk, A. Mainwaring, D. Culler, and J. Anderson, Analysis of Wireless Sensor Networks for Habitat Monitoring, in C.S. Raghavendra, K.M. Sivalingam and T. Znati (eds.) *Wireless Sensor Networks*, Boston, Kluwer Academic Publisher, 2004, pp. 399-423.
[13] S.M. Ross, *Introduction to Probability Models*, Academic Press, 2001.
[14] V. Shnayder, M. Hempstead, B. Chen, G. W. Allen, and M. Welsh, Simulating the Power Consumption of Large Scale Sensor Network Applications, *ACM SenSys*, Baltimore, MD, 2004.
[15] G. Simon, M. Maroti, A. Ledeczi, G. Balogh, B. Kusy, A. Nadas, G. Pap, J. Sallai and K. Frampton, Sensor Network-Based Countersniper System, *ACM SenSys*, Baltimore, MD, 2004.
[16] Y. Choi, M.G. Gouda, H. Zhang and A. Arora, Routing on a Logical Grid in Sensor Networks, *UTCS Technical Report TR-04-09*, 2004.