# ANTI-SENSOR NETWORK: DISTORTION-BASED DISTRIBUTED ATTACK IN WIRELESS SENSOR NETWORKS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

İBRAHİM KARAASLAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

FEBRUARY 2008

Approval of the thesis:

**ANTI-SENSOR NETWORK: DISTORTION-BASED DISTRIBUTED ATTACK IN WIRELESS SENSOR NETWORKS**

submitted by **İBRAHİM KARAASLAN** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. İsmet Erkmen _____
Head of Department, **Electrical and Electronics Engineering**

Assoc. Prof. Dr. Özgür Barış Akan
Supervisor, **Electrical and Electronics Engineering Dept., METU** _____

**Examining Committee Members:**

Prof. Dr. Semih Bilgen _____
Electrical and Electronics Engineering Dept., METU

Assoc. Prof. Dr. Özgür Barış Akan _____
Electrical and Electronics Engineering Dept., METU

Assoc. Prof. Dr. Elif Uysal Bıyıkoğlu _____
Electrical and Electronics Engineering Dept., METU

Assist. Prof. Dr. Ali Özgür Yılmaz _____
Electrical and Electronics Engineering Dept., METU

Metin Aktaş _____
Engineer, M.Sc., ASELSAN

**Date:** **08.02.2008**

**I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.**

Name, Last name : İbrahim Karaaslan

Signature :

## ABSTRACT

ANTI-SENSOR NETWORK: DISTORTION-BASED DISTRIBUTED ATTACK IN WIRELESS SENSOR NETWORKS

Karaaslan, İbrahim

M.Sc., Department of Electrical and Electronics Engineering

Supervisor: Assoc. Prof. Dr. Özgür Barış Akan

February 2008, 52 pages

In this thesis, a novel anti-sensor network paradigm is introduced against wireless sensor networks (WSN). Anti-sensor network (ASN) aims to destroy application reliability by adaptively and anonymously introducing adequate level of artificial distortion into the communication of the event features transported from the sensor nodes (SN) to the sink. ASN is composed of anti-sensor nodes (aSN) randomly distributed over the sensor network field. aSNs pretend to be SNs to maintain anonymity and so improve resiliency against attack detection and prevention mechanisms. Performance evaluations via mathematical analysis and simulation experiments show that ASN can effectively reduce the application reliability of WSN.

Keywords: Wireless Sensor Networks, Wireless Sensor Network Attack, Distributed Jamming, Event-to-Sink Distortion, Denial of Service (DoS).

# ÖZ

ALGILAMA-ÖNLEYİCİ AĞ: TELSİZ ALGILAYICI AĞLARDA BOZULMA TABANLI DAĞITIK SALDIRI

Karaaslan, İbrahim

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü

Tez Yöneticisi: Doç. Dr. Özgür Barış Akan

Şubat 2008, 52 sayfa

Bu tezde, telsiz algılayıcı ağlara (TAA) karşı yeni bir algılama-önleyici ağ modeli tanıtılmıştır. Algılama-önleyici ağ (AÖA), algılayıcı düğümlerden (AD) toplama noktasına taşınan olay niteliklerinin haberleşmesine, yeterli derecede yapay bozulmayı, uyarlamalı ve kimliğini belirtmeksizin ekleyerek uygulama güvenilirliğini yok etmeyi amaçlamaktadır. AÖA, algılayıcı ağ alanına rastgele dağıtılan algılama önleyici düğümlerden (AÖD) oluşur. AÖD'ler, kimlik belirsizliğini sürdürmek için algılayıcı düğümlermiş gibi davranırlar ve böylece saldırı tespit ve engelleme yöntemlerine karşı esneklik kazanırlar. Matematiksel çözümlemeler ve benzetim deneyleri ile yapılan başarım değerlendirmeleri, AÖA'nın TAA uygulama güvenilirliğini etkin olarak azaltabildiğini göstermektedir.

Anahtar Kelimeler: Telsiz Algılayıcı Ağlar, Telsiz Algılayıcı Ağ Saldırısı, Dağıtık Karıştırma, Olaydan Alıcıya Bozulma, Hizmet Engelleme

To my lovely wife,
Öznur

## ACKNOWLEDGMENTS

I wish to express my deepest gratitude to my supervisor Assoc. Prof. Dr. Özgür B. Akan for his guidance, advice, criticism and insight throughout the research.

I am indebted to my wife for her patience and encouragement throughout this study.

I would like to thank my parents for everything they do for me.

## TABLE OF CONTENTS

## LIST OF TABLES

## LIST OF FIGURES

## LIST OF SYMBOLS

| | |
|---|---|
| ACK | Acknowledgment |
| AODV | Ad Hoc On-Demand Distance Vector |
| ASN | Anti-Sensor Network |
| aSN | Anti-Sensor Node |
| $\text{aSN}^m$ | Master Anti-Sensor Node |
| $\text{aSN}^s$ | Slave Anti-Sensor Node |
| AWGN | Additive White Gaussian Noise |
| BPSK | Binary Phase Shift Keying |
| BSC | Binary Symmetric Channel |
| CSMA/CA | Carrier Sense Multiple Access/Collision Avoidance |
| CTS | Clear-to-Send |
| DoS | Denial of Service |
| DSSS | Direct Sequence Spread Spectrum |
| MAC | Medium Access Control |
| RTS | Request-to-Send |
| SN | Sensor Node |
| TDMA | Time Division Multiple Access |
| UDP | User Datagram Protocol |
| UWB | Ultra Wide Band |
| WSN | Wireless Sensor Network |

# CHAPTER 1

# INTRODUCTION

Wireless sensor networks (WSN) are event-based systems that are designed to carry the information about a phenomenon to a destination node which is called sink. WSN is composed of densely deployed sensor nodes (SNs) which are distributed near or inside of a phenomenon.

A sensor node is mainly composed of sensing, processing, transceiver and power units. Whenever a specific event occurs, sensing unit samples the analog event signal and after converting it into a digital form, the bit sequence is passed to the processing unit. The processing unit handles the Medium Access Control (MAC) and above layer protocols and constructs sensor packets which are fed into the transceiver unit to be transmitted. Transceiver unit is the physical layer of the sensor node and thus, it is responsible for frequency selection, signal detection and modulation. All these units are fed by a power unit equipped with limited power source. Therefore, it is essential to choose power aware protocols and algorithms when designing SNs.

SNs are low power, inexpensive devices with limited radio range, battery resource and computational capacity. Because of the limitations, they are densely deployed over a phenomenon field. This deployment improves wide spatial coverage and fault tolerance of the sensor system. Besides that, SNs form an ad hoc network and hence, they may be left after deployment.

Sensor networks are surveyed in [3] with various aspects such as communication architecture, standards, open research issues, new interests and developments. In this work, it is pointed that, rapid deployment, self-organization and fault tolerance characteristics of sensor networks make them a very useful sensing technique for military, national defense, command, control, communications, computing, intelligence, surveillance, reconnaissance and targeting systems.

Sensor networks may be used in military applications to track enemy movements, gather information about battlefield conditions and detect any chemical or biological threat. Sensors may be deployed at hospitals and even at homes to remotely monitor health status of patients. Additionally, they may also be deployed at appropriate places to monitor weather conditions or activity of a volcano or a glacier. Because of their low cost and low overheads, sensor networks may also be used in residential or civic places to identify and track movements for security or control purposes.

Standardized communication protocols are generally developed without considering protocol attacks. Therefore, WSNs that directly use present communication protocols are vulnerable to the attacks in various communication layer aspects. Besides that, popularity of WSNs in various application areas especially in military and security applications makes them very attractive to researchers to find out any flaws in WSNs. Networks that aim to constitute a Denial of Service (DoS) for sensor networks are called anti-sensor networks.

In [35], any event that diminishes or eliminates a network's capacity to perform its expected function is defined as DoS attack although it is generally used as a term to refer an attempt of an adversary to disrupt, subvert or destroy a network. In this work, it is also mentioned that, hardware failures, software bugs, resource exhaustion, environmental conditions or any complicated interaction between these factors can also cause a DoS. DoS attacks in sensor networks is surveyed in [35] with various respects of communication layers.

Therefore, sensor networks must be developed by taking security considerations into account. In order to do this, some needs may arise at design time to detect

and prevent any hostile manner or out of order condition by some modifications in protocol layers or development of new algorithms. For these reasons, some techniques are proposed to detect and defense against anti-sensor networks.

On the other hand, especially in military systems, it is necessary to use anti-sensor network against any threat that arises from a possible usage of a sensor network by an enemy. Therefore, it is worth to research and develop new techniques for anti-sensor networks. Various WSN attack and counter attack techniques are explained in Chapter 2 in detail.

The most trivial way of attacking a wireless network is physical layer jamming which is based on generating a random signal at the operating frequency of the network which is to be jammed. However, this method is not energy efficient and highly detectable. Other techniques mainly focus on some specific layer attacks such as MAC layer [12], [18], [1], [33] and network layer [35] or attacks against time synchronization protocols [25], [31], [28]. On the other hand, some jamming detection and defense strategies are studied in various works such as [31], [9], [36], [30], [5], [13], [6], [16] and [28]. Details of these techniques are presented in Chapter 2.

Collective information received from various SNs is used to observe the phenomenon reliably. If collective information at the sink node has some acceptable distortion level, then it is assumed that event-to-sink communication is reliable. Distortion is a measure of error between realized and received event features. The upper limit of acceptable distortion is an application requirement. As a result, it is essential for a WSN to have a reliable event-to-sink transport mechanism.

However, none of the proposed anti-sensor techniques deal with attacking WSNs from the event reliability aspect. Event-to-sink reliability is investigated and a transport algorithm is proposed in [2]. This algorithm uses a measure of event reliability which is based on the number of sensor packets received by the sink. It manages to capture sufficient number of samples from sensors. In order to do this, sampling frequency of sensor nodes is adjusted adaptively. However, there is no known reliability mechanism which compares samples generated by the sensors with the samples received by the sink and operates accordingly.

Therefore, it is not sufficient to receive the required number of samples from sensors to detect the event reliably if generated samples differ from the received samples such that distortion at the sink passes beyond the acceptable limit. This may be possible by the modification of the samples along the path toward the sink. If sufficient number of samples are modified, then sensor application becomes unreliable. In order to attack WSNs from this point of view, a new anti-sensor network (ASN) framework is presented in this thesis. ASN is a novel approach that aims to construct an event-to-sink distortion which is higher than what the application can tolerate. ASN is composed of densely and randomly deployed anti-sensor nodes (aSN) over the sensor network field as seen in figure 1.1.

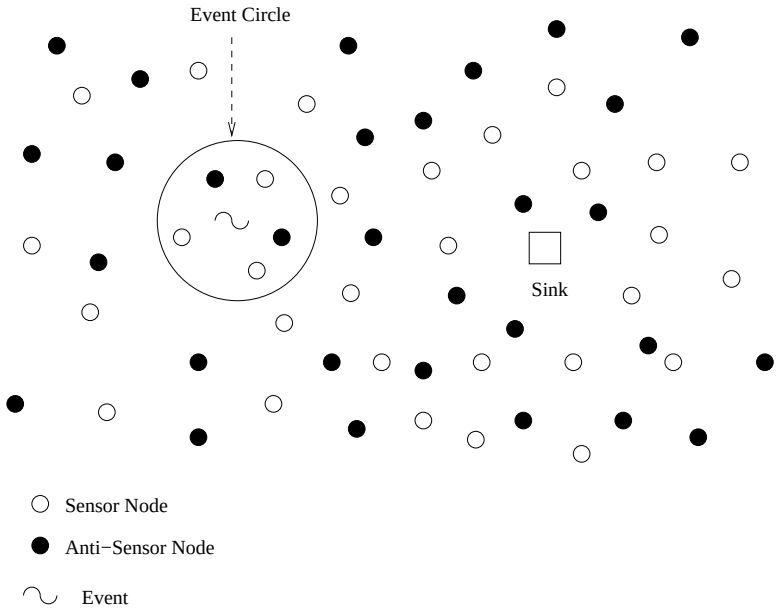


Figure 1.1: Typical sensor network topology with aSNs distributed around the SNs.

ASN is based on the modification of sufficient number of samples encapsulated in sensor packets. Modification is done simply by adding errors to the signal samples carried in the packets. Mathematical analysis done in Chapter 3, guides the ASN such that distortion error added to a sample value must be maximum and as a result,

target distortion is achieved by distorting minimum number of samples and hence, with minimum energy expenditure.

However, in order to attack a WSN, ASN needs some assumptions such as awareness of the operating frequency, MAC mechanism and packet structure of the WSN. Additionally, it is assumed that control and data packets are not encrypted by the SNs.

The remainder of this thesis is organized as follows. In Chapter 2, various related work on sensor network attack and defense strategies is summarized. Theoretical modeling and analysis of distortion based attack are derived in Chapter 3. In Chapter 4, the proposed anti-sensor network (ASN) is explained with detailed operational principles. Simulation results and discussions are presented in Chapter 5. Finally, thesis is concluded in Chapter 6.

# CHAPTER 2

# OVERVIEW OF EXISTING WIRELESS SENSOR NETWORK ATTACKS

Recent work related to the Denial of Service (DoS) against WSNs mainly focuses on layer based attacks. Various types of sensor network attack techniques are classified and solutions are offered in the literature. Layer based attack types overviewed in this chapter are summarized in Table 2.1.

Table 2.1: DoS attacks and defenses in WSNs [35, 8]

| Layer | Attack | Defense |
|---|---|---|
| Physical | Jamming | DSSS,FHSS |
| Link | Collision | Error Correcting Codes |
| Network and Routing | Selective Forwarding<br>Sinkhole<br>HELLO Flood<br>Sybil<br>Routing Cycles | Redundancy<br>Authorization<br>Link Verification<br>Authentication<br>Routing Checks |
| Transport | Flooding<br>Desynchronization | Client Puzzles<br>Authentication |

## 2.1 Physical Layer Attacks

Physical layer jamming is the most trivial attack technique and realized by emitting a continuous random signal at the frequency that is used by the WSN. This type of jamming is also called blind jamming and it is a well known attack technique

for all wireless networks. If the operating frequency of the WSN is not known by the attacker, a broad bandwidth must be jammed by the jammer. However, this needs a large average power which necessitates a transmitter with large size, weight and power supply.

Some anti-jamming techniques are developed to avoid physical layer jamming such as Direct Sequence Spread Spectrum (DSSS) and Frequency Hopping Spread Spectrum (FHSS). These are very well known techniques applied by various wireless networks not just by WSNs. In DSSS technique, transmitted signal is spread away a very wide bandwidth and such that a very small amount of energy is allocated for each unit of frequency band. Therefore, jammers may not receive enough energy to determine a wireless network activity. In FHSS technique, wireless stations use an array of prearranged frequency channels such that jammers can not follow the precise hopping sequence if the array can not be discovered.

Another defense method against physical layer jamming is to use Ultra Wide Band (UWB) technology which is well suited to sensor network applications. UWB systems have a very wide band, noise-like signals up to several gigahertz so that they are resistant to severe multi path and jamming. [27] presents the architecture of UWB sensor systems based on low power and low complexity UWB transceivers.

## 2.2 Link Layer Attacks

Another type of attack is called link layer jamming which is more attractive than physical layer jamming because of a very low energy requirement and a low probability of detection. In [12], [18] and [19], it is argued that among various DoS attacks, link layer jamming is more advantageous option for attackers than physical layer jamming. The attacker saves significant amount of energy by taking the advantage of the operating MAC protocol. Such jammers are also called protocol aware jammers.

In [1] and [33], some intelligent jamming mechanisms are proposed for 802.11b wireless networks. [19] proposes various link layer jamming attacks on some specific MAC protocols such as S-MAC, L-MAC and B-MAC. Furthermore, [18] in-

vestigates some jamming attacks on S-MAC protocol, the level of effectiveness and a countermeasure that can be implemented against one of these attacks.

[26] analyzes a jamming attack on the CSMA/CA based MAC protocol. In this work, jammer quickly detects every RTS packet and jams that packet by sending its own RTS packet simultaneously. Therefore, network nodes can not have a chance to access the medium to transmit packets. This is a typical collision based link layer jamming attack.

However, in all of these link layer attack cases, sensor nodes can detect the attack and malicious activity after sometime and may react accordingly.

## 2.3 Network Layer Attacks

Attacks proposed for the network and above layers assume that attackers are aware of the protocol which is used by the sensor network. Most common network layer attacks are selective forwarding, sinkhole and hello flood attacks [15].

In selective forwarding attack, attacker refuses forwarding certain messages while forwarding most of them. Sinkhole attack is a special case of selective forwarding. In this case, attacker pretends to be attractive to the surrounding nodes with respect to routing quality metrics and tries to draw all the traffic. SNs may send duplicate messages along the same path or different paths by using the redundancy defense mechanism against these types of attacks.

In hello flood attack, attacker broadcasts very powerful hello routing packets. As a result, receiving SNs suppose that attacker is their neighbor node and start routing their messages toward the attacker which is probably out of range.

In sybil attack, attacker forges multiple identities. Therefore, memories of neighbor SNs keep useless and wrong information which causes them run out of resources sooner. Besides that, wrong routing information causes sensor packets to go wrong directions. This may also cause packets to go around in circles which is called routing cycles. Various defense mechanisms are developed to avoid these types of attacks such as authorization, link verification, authentication and routing checks. Detailed information about these mentioned attacks may be found in [8].

In [13], a network layer based wormhole attack is studied. In this work, an attacker tunnels the received packets from a source node in the network to a random distant node by replaying the packets. If the tunneled packets are routing packets, then it causes routing protocol to fail to find routes when they are not actually neighbors. If the tunneled packets are authentication exchange messages, an attacker could gain unauthorized access. [13] proposes TIK (TESLA with Instant Key Disclosure) protocol to defend such network layer attacks by restricting the maximum transmission distance of a packet.

In [35], various layer based attack types and defense mechanisms are explained including transport layer attacks such as flooding and desynchronization. In flooding attack, attackers send many connection requests to SNs to make them run out of resources. As a defense mechanism, number of connections may be limited or called node may ask calling node to solve a puzzle before accepting the connection request. This method is appropriate but requires more computational capacity.

In desynchronization attack, attacker transmits packets with wrong sequence numbers and causes end node to request retransmission of missed packets. Authentication of all packets including transport protocol header is a defense mechanism against this type of attack.

## 2.4 Other Sensor Network Attacks

As previously mentioned in Chapter 1, one of the primary goals of the attackers is to cause power consumption of SNs to deplete more energy. [28] presents two types of attacks in sensor networks: the barrage attack and the sleep deprivation attack. The barrage attack bombards SNs with legitimate requests and the sleep deprivation attack sends requests to SNs as often as necessary to keep the SNs awake. As a result, these attacks cause SNs to consume their battery sooner. However, unusual behaviour of anti-sensor nodes causes detection of such attack types.

Time synchronization protocols provide local and remote clock synchronization of wireless SNs. Therefore, it is a major building block for WSNs because of distributed nature of nodes. In addition, some communication protocols in WSNs

rely on time synchronization for scheduling node services, such as TDMA (Time Division Multiple Access) based channel access protocols.

In [25], the most common time synchronization protocols are surveyed and a set of attacks for each protocol are outlined. In this work, attackers distribute erroneous time synchronization information to the network to disturb synchronization. In [31], various attacks are considered that are effective to several representative time synchronization schemes. Additionally, some solutions are also proposed to detect and accommodate these types of attacks.

In [5], wormhole-based anti-jamming techniques are explained. This work shows that SNs can exploit channel diversity in order to create wormholes that lead out of the jammed region. By means of this, an alarm can be transmitted to the sink when an adversary masks the events that the sensor network should detect.

In [30], a spam attack type is studied. In this technique, anti-sensor nodes generate dummy data packets that make the nodes relaying them and deplete more energy. Furthermore, a detection and defense mechanism called DADS is presented against spam attacks in this work.

Another type of sensor attack called bootloader attack is studied in [34]. In this work, attackers try to take the control of the nodes by downloading the malicious code into the nodes. On the other hand, intrusion detection and failure recovery methods against this attack are also presented in this study.

## 2.5 Attack Detection and Defense Mechanisms

An adversary can analyze the traffic patterns and deduce the location of the base station or sink within the WSN topology. [9] investigates basic decorrelation of WSN traffic to inhibit traffic analysis attacks. A similar anti-traffic and intrusion tolerance strategies for WSNs are analyzed in [10]. In this work, two secure strategies are proposed to defend against such attacks. First one is secure multi-path routing which is designed to send sensor packets to multiple base stations. Second one is anti-traffic analysis strategy which is proposed against eavesdroppers.

On the other hand, various defense mechanisms are developed to defend against attacks. [24] develops a random channel selection protocol to facilitate communication among nodes in the presence of physical and MAC layer jamming. [4] proposes on-the-fly statistical technique that can detect and distinguish faulty data from malicious data of adversary nodes to initiate a correct recovery action in a distributed sensor network. This technique classifies faults versus attacks by learning correct system behaviour dynamically using Hidden Markov Models. In [36], a mapping protocol is described to detect and map the jammed region. As a result, network traffic may be routed around the mapped region without entering.

In [21] and [41], MAC layer anomalies and intrusion detection are studied by proposing methods to determine the misbehaving nodes in ad hoc networks. These solutions may also be applied to WSNs to detect some kind of attacks.

As a summary, various observation statistics which enable detection of jamming are presented in [36] and [37] as follows:

- Repeated inability to access wireless channel
- Bad framing
- Checksum failures
- Illegal values for address or other fields
- Protocol violations (e.g., missing ACKs)
- Excessive received signal level
- Low signal-to-noise ratio
- Repeated collisions
- Signal strength
- Carrier sensing time
- Consistency checks for sensor location and received signal strength

On the other hand, none of the proposed anti-sensor networks take application reliability requirements of WSN into account. Additionally, they do not depend on any adaptive mechanism that monitors the performance and sufficiency of the attack operation and controls accordingly. Hence, we propose a novel energy efficient anti-sensor mechanism that aims to inject distortion error to the minimum number of samples to construct a sufficient application level distortion with an adaptive mechanism that depends on feedbacks.

# CHAPTER 3

# EVENT-TO-SINK DISTORTION ANALYSIS

As will be explained in detail in Chapter 4, ASN is based on dynamically injecting artificial distortion error ($d_e$) into the signal samples carried in sensor packets in order to create event-to-sink distortion. In this chapter, we analyze the theoretical computation of event-to-sink distortion for single-hop and multi-hop cases. As a result, we aim to observe how distortion errors injected to individual samples affect the overall event-to-sink distortion.

## 3.1 Single-hop Distortion in AWGN Channel

Let us assume a sensor system with a single SN and a sink. SN quantizes its analog measurements and sends them over the Binary Symmetric Channel (BSC) as depicted in Figure 3.1. Single-hop distortion computations are already done in [23].

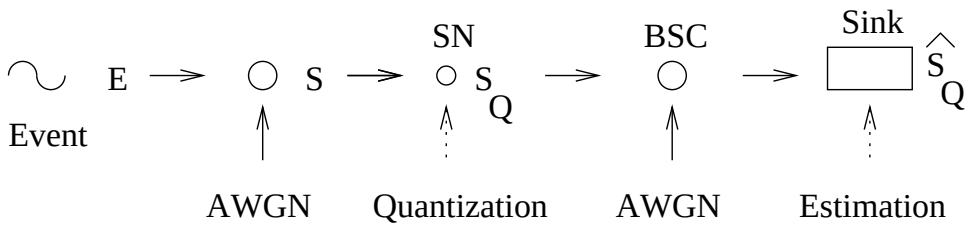


Figure 3.1: Event-to-sink single-hop sample transmission.

Let us consider an observation as $S = E + n$ where $E$ and $n$ are event signal and AWGN, respectively. It is quantized as $S_Q$ by the SN and estimated at the sink node as $\hat{S}_Q$. If we let $S = \sum_{i=1}^{\infty} b_i 2^{-i}$, we consider $N$-bit uniform quantization as follows:

$$S_Q = \sum_{i=1}^{N} b_i 2^{-i} \tag{3.1}$$

The quantized N bits ($b_i$) are transmitted through wireless BSC channel and they are estimated at the sink as $\hat{b}_i$. Sink reconstructs the observation using a single estimation as follows:

$$\hat{S}_Q = \sum_{i=1}^{N} \hat{b}_i 2^{-i} \tag{3.2}$$

Various channel noise models can be used to compute estimated bit errors at the sink. For the sake of simplicity we consider only AWGN channel for the BSC model.

Point-to-point error can be separated into two expectation factors with and without the quantization error, which are $D_Q$ and $D_C$, respectively. $D_Q$ is the expectation of difference between estimated sample by the sink ($\hat{S}_Q$) and sampled signal by the sensor ($S$) and can be written as $D_Q = E[|S - \hat{S}_Q|]$. This difference includes quantization, channel and estimation errors. By using (3.1) and (3.2), $S - \hat{S}_Q$ can be expressed as follows:

$$\begin{aligned} S - \hat{S}_Q &= S - S_Q + S_Q - \hat{S}_Q \\ &= \sum_{i=N+1}^{\infty} b_i 2^{-i} + \sum_{i=1}^{N} (b_i - \hat{b}_i) 2^{-i} \end{aligned} \tag{3.3}$$

Then, by using the triangle equation, the absolute value of (3.6) can be bounded as follows:

$$|S - \hat{S}_Q| \leq \sum_{i=N+1}^{\infty} 2^{-i} + \sum_{i=1}^{N} |b_i - \hat{b}_i| 2^{-i} \tag{3.4}$$

Taking the expectation of (3.4) requires computation of bit error probability. Let us assume that $E_s$ is the symbol transmission energy so that transmission energy per bit is $E_b = E_s/N$. Probability of bit error with binary phase shift keying (BPSK) modulation over AWGN channel with zero mean and unit variance can be written as $Q(\gamma)$ where $\gamma$ is $\sqrt{2E_b/N_0}$, $N_0$ is channel noise level and $Q()$ is given by

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-\frac{t^2}{2}} dt = \frac{1}{2} erfc(\frac{x}{\sqrt{2}}) \tag{3.5}$$

where $erfc()$ is well known error function in mathematics. Then, $D_Q$ can be expressed by taking the expectation of both sides of (3.4) as follows:

$$\begin{aligned} D_Q &= E[|S - \hat{S}_Q|] \\ &\leq 2^{-N} + \sum_{i=1}^{N} E[|(b_i - \hat{b}_i)2^{-i}|] \\ &= 2^{-N} + Q(\gamma) \sum_{i=1}^{N} 2^{-i} \\ &= 2^{-N} + Q(\gamma)(1 - 2^{-N}) \end{aligned} \tag{3.6}$$

$D_C$ is the expectation of difference between estimated sample by the sink ($\hat{S}_Q$) and quantized signal by the sensor ($S_Q$) and hence, it does not include quantization error and can be written as follows:

$$\begin{aligned} D_C &= E[|S_Q - \hat{S}_Q|] \\ &= \sum_{i=1}^{N} E[|(b_i - \hat{b}_i)2^{-i}|] \\ &= Q(\gamma)(1 - 2^{-N}) \end{aligned} \tag{3.7}$$

As will be explained in Section 3.2, total exposed event-to-sink distortion ($D_{J^2}$) in the presence of aSN for the multi-hop operation case, can be represented as $E[|E - \hat{S}_{SQ}|^2]$ where $S_{SQ}$ is the estimation of the sample by the sink node. $D_{J^2}$ is an expectation expression of absolute square error and hence, we need to compute

the second moments of errors whose first moments are given in (3.6) and (3.7), so they can be expressed as follows:

$$\begin{aligned} D_{Q^2} &= E[|S-\hat{S}_Q|^2] \\ &= E[|\sum_{i=N+1}^{\infty}\sum_{m=N+1}^{\infty} b_i b_m 2^{-i-m} \\ &\quad +2\sum_{i=N+1}^{\infty}\sum_{m=1}^{N} b_i(b_m-\hat{b}_m)2^{-i-m} \\ &\quad +\sum_{i=1}^{N}\sum_{m=1}^{N}(b_i-\hat{b}_i)(b_m-\hat{b}_m)2^{-i-m}|] \\ &\leq 2^{-2N}+Q(\gamma)2^{-N}(1-2^{-N}) \\ &\quad +Q^2(\gamma)(1-2^{-N})^2-Q^2(\gamma)(1-2^{-2N})/3 \\ &\quad +Q(\gamma)(1-2^{-2N})/3 \end{aligned} \tag{3.8}$$

Then, the expectation of the absolute square of $S_Q-\hat{S}_Q$, can be obtained by

$$\begin{aligned} D_{C^2} &= E[|S_Q-\hat{S}_Q|^2] \\ &= \sum_{i=1}^{N}\sum_{m=1}^{N}(b_i-\hat{b}_i)(b_m-\hat{b}_m)2^{-i-m} \\ &= Q^2(\gamma)(1-2^{-N})^2-Q^2(\gamma)(1-2^{-2N})/3 \\ &\quad +Q(\gamma)(1-2^{-2N})/3 \end{aligned} \tag{3.9}$$

$D_{C^2}$ represents single-hop distortion from sensor to sink with channel and estimation errors whereas $D_{Q^2}$ is the distortion which additionally consists of quantization error as well. Therefore, as seen in (3.8) and (3.9), if there is no channel error ($Q(\gamma)$=0), $D_{C^2}$ is zero as expected whereas $D_{Q^2}$ is bounded with $2^{-2N}$ which is introduced by the quantization operation. These derivations are for single-hop cases and we use them for the computation of multi-hop event-to-sink distortion in Section 3.2.

### 3.2 Multi-hop Distortion in AWGN Channel in the Presence of a Single aSN

Let us first consider a sensor system with L number of SNs and a sink as seen in Figure 3.2. Assume that the first SN quantizes observed samples and sends them to the sink node which is located L-1 number of hops away. Assume an aSN is located anywhere in between the first SN and the sink. This aSN captures the sensor sample and sends it to its next destination after injecting distortion error ($d_e$) to the sample as shown in Figure 3.2. As will be explained in detail in Chapter 4, only one aSN must add error to the samples in the packet all along the path until the sink.

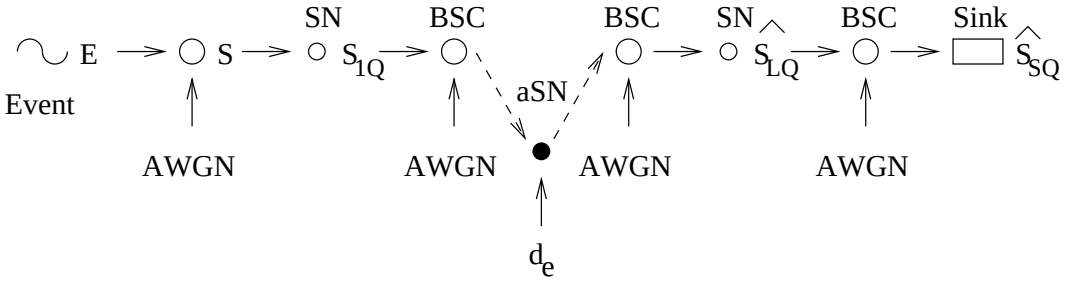


Figure 3.2: Event-to-sink multi-hop sample transmission in the presence of aSN.

Assume that first SN samples $S$ and obtains $S_{1Q}$. Distortion error added by aSN and reconstructed signal at the sink, are represented as $d_e$ and $\hat{S}_{SQ}$, respectively. Let $\hat{S}_{XQ}$ is the estimation of $X^{th}$ SN. Then, absolute multi-hop event-to-sink distortion $D_{J^2}$ in the presence of an aSN can be written as:

$$\begin{aligned} D_{J^2} &= E[|E - \hat{S}_{SQ}|^2] \\ &= E[|E - \hat{S}_{SQ} + \hat{S}_{LQ} - \hat{S}_{LQ}|^2] \\ &= E[|\hat{S}_{LQ} - \hat{S}_{SQ} + E - \hat{S}_{LQ}|^2] \end{aligned} \tag{3.10}$$

Let $\hat{S}_{LQ} - \hat{S}_{SQ} = \hat{S}_{(L-1)Q} - \hat{S}_{LQ} = ... = C$ and from (3.7) and (3.9), we can write $E[|C|] = D_C, E[|C|^2] = D_{C^2}$, so we rewrite (3.10) as follows:

$$\begin{aligned} D_{J^2} &= E[|C + E - \hat{S}_{LQ}|^2] \\ &= E[|C + E - \hat{S}_{LQ} + \hat{S}_{(L-1)Q} - \hat{S}_{(L-1)Q}|^2] \\ &= E[|2C + E - \hat{S}_{(L-1)Q}|^2] \\ &= E[|2C + E - \hat{S}_{(L-1)Q} + \hat{S}_{(L-2)Q} - \hat{S}_{(L-2)Q}|^2] \\ &= E[|3C + E - \hat{S}_{(L-2)Q}|^2] \\ &\vdots \end{aligned} \tag{3.11}$$

If we continue to rewrite (3.11) until we reach node number two by considering $d_e$ as seen in Figure 3.2 and inserting $E = S - n$ in (3.11) we obtain

$$D_{J^2} = E[|LC + S - n - \hat{S}_{2Q} - d_e|^2] \tag{3.12}$$

Let $S - \hat{S}_{2Q} = Q$ and from (3.6) and (3.8) we can write $E[|Q|] = D_Q$, $E[|Q|^2] = D_{Q^2}$ and then we rewrite (3.12) as follows:

$$\begin{aligned} D_{J^2} &= E[|LC - n + Q - d_e|^2] \\ &= E[|(LC - n + Q - d_e)(LC - n + Q - d_e)|] \\ &\leq L^2 D_{C^2} + L(2D_Q D_C - 2D_C E[d_e]) \\ &\quad + E[{d_e}^2] - 2E[d_e]D_Q + D_{Q^2} + E[n^2] \end{aligned} \tag{3.13}$$

where $E[n^2] = var(n)$ which is the variance of AWGN. On the other hand, total multi-hop distortion without aSN can be found easily by neglecting the terms with $d_e$ in (3.13). Because, $d_e$ is the injected distortion error by the aSN. Additionally, if

there is no aSN, then total number of hops from sensor to sink will decrease to L-1. Hence, multi-hop total distortion without aSN can be written as follows:

$$D_{N^2} \leq (L-1)^2 D_{C^2} + 2(L-1) D_Q D_C + D_{Q^2} + E[n^2] \tag{3.14}$$

(3.13) and (3.14) represent only one packet transition that contains a single sample. To compute total event-to-sink distortion at the sink, we must take the average of all the sensor sample distortions.

Therefore, total event-to-sink distortion in the presence of multiple aSNs and SNs can be expressed as follows:

$$\begin{aligned} D_T(M) &= E[|E - \hat{S}_{SQ}|^2] \\ &= E[(\frac{1}{\tau f K})(\sum_{i=1}^{\tau f}\sum_{k=1}^{K} E_{ki} - \hat{S}_{SQki})^2] \\ &\geq (\frac{1}{\tau f K})\sum_{i=1}^{\tau f}\sum_{k=1}^{K} E[|E_{ki} - \hat{S}_{SQki}|^2] \\ &= (\frac{1}{\tau f K})[\sum_{i=1,k=1}^{\tau f K - M} D_{N^2 ki} + \sum_{i=1,k=1}^{M} D_{J^2 ki}] \end{aligned} \tag{3.15}$$

where $D_{J^2 ki}$ and $D_{N^2 ki}$ are the total event $(E_{ki})$ to sink $(\hat{S}_{SQki})$ distortion of the $i^{th}$ sample of the $k^{th}$ SN with and without an aSN, respectively. $\tau$, $f$ and $K$ are observation time, sampling frequency and number of SNs, respectively. Obviously $\tau f K$ is the total received samples by the sink during the observation time. Number of captured and distorted samples by aSNs before reaching the sink is M. Therefore, $\tau f K - M$ is the number of samples that directly reach the sink without being captured and distorted.

In Figure 3.3, the total event-to-sink distortion derived in (3.15) is plotted with respect to the number of distorted samples (M) for various values of $d_e$. From this figure, it can be easily deduced that the total event-to-sink distortion ($D_T$) increases with the increment of M and $d_e$. Values of some parameters used in (3.15) are also

given in the figure. Wireless channels are prone to errors and hence, value of $E_b/N_0$ is chosen as 21.63 such that bit error rate (BER) for the BSC channel is $10^{-5}$.

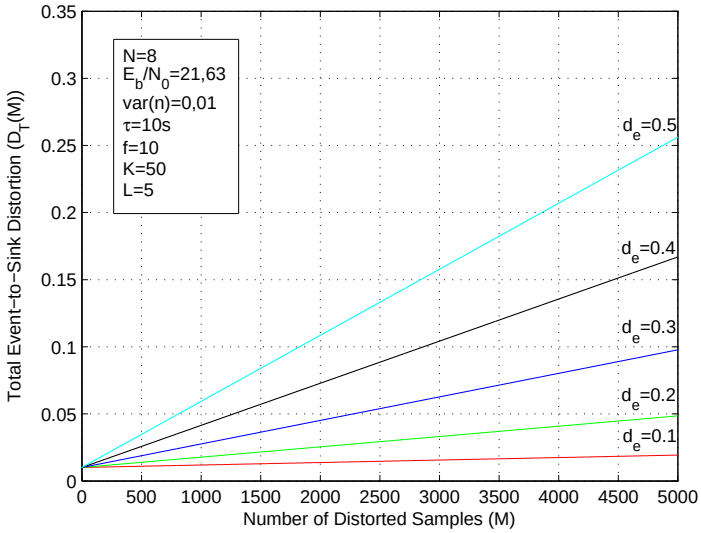


Figure 3.3: Total event-to-sink distortion wrt. variation of $d_e$.

The required target distortion values differ according to the various types of sensor applications. After the determination of this value, the number of samples which must be distorted can be found for a given value of $d_e$ using (3.15). Let us assume for instance a distortion with a value of 0.1 is sufficient to cause an application to become unreliable, then the number of samples that must be distorted are nearly 1800, 2875 and 5000 for $d_e$ values 0.5, 0.4 and 0.3, respectively. Achieving 0.1 distortion with $d_e$ values 0.1 and 0.2 is impossible even though all the samples are distorted, which yields distortions around 0.05 and 0.02, respectively.

Obviously, energy expenditure of ASN increases with the increment of the number of samples that must be distorted. Therefore, achieving a distortion objective by distorting minimum number of samples provides energy efficiency. In order to realize this, $d_e$ must be chosen as large as possible. Maximization of $d_e$ requires shifting samples to their maximum or minimum quantization levels. Therefore, maximum $d_e$ value for signal samples depends on the distribution of the analog

event signal which is to be observed by the WSN. In (3.15), number of hops (L) for each sample is kept same and only one sample is transmitted at each transition.

However, in WSN, a few samples are encapsulated in a sensor packet for energy and traffic efficiency reasons and a sensor packet may reach the sink using independent routes and hence, number of hops may vary for each packet. MAC protocols used in WSN generally employ channel coding mechanisms to detect and correct channel errors, however, (3.15) does not handle these operations. Besides that, BSC channel model is insufficient to represent all wireless channels in practical networks. Because of all these reasons, we can not directly use (3.15) to determine the required number of samples to capture and distort for a given target distortion.

# CHAPTER 4

# ANTI-SENSOR NETWORK

In this chapter, we give a brief information about event-to-sink sensor communication and then explain ASN framework in detail.

In WSNs, assume that whenever a SN senses an event, it continuously samples the event signal with a sampling period ($T_{sa}$). Then, after a predetermined number of samples are accumulated, they are encapsulated in a sensor packet and sent to the sink node. As a result, sensor packet generation period ($T$) is multiple times of the sampling period $T_{sa}$. This accumulation mechanism provides energy efficiency and prevents overloaded traffic in terms of network performance.

## 4.1 Overview of ASN

ASN is composed of aSNs randomly distributed in sensor network field. aSNs have similar physical constraints like SNs such as limited power source and short radio communication range. Therefore, ASN must also operate energy efficiently as WSN.

Besides attacking SNs, aSNs form an ad hoc ASN at a frequency channel that is different from the one used by the WSN. Therefore, an aSN has two transceiver units connected to the separate antennas. One channel is used to communicate with other aSNs and the other one is used to attack the sensor packets. Attack operation is done by capturing the determined sensor packets and then distorting the captured

packets by injecting distortion errors to the samples. In order to capture a sensor packet, ASN must be aware of the MAC protocol that is used by the WSN.

MAC protocol selection is an important factor in terms of energy efficiency. Therefore, various MAC protocols are proposed for WSNs studied in [29], [38], [22], [14], [7] and [32], respectively. These MAC protocols are compared in various aspects such as energy efficiency and reliability in [20], [40], [11], [17] and [39]. Although there are various MAC layer protocols proposed for WSNs, there is no protocol accepted as a standard. Generally, choice of the MAC protocol for WSNs is application dependent. We assume that MAC protocol for ASN is based on the RTS/CTS method with CSMA mechanism such that majority of the WSN MAC protocols generally employ.

Additionally, we assume that control and data packets generated by SNs are not encrypted such that they are fully recognizable by the ASN. Besides that, aSNs are assumed to be aware of the:

- Operating frequency of the WSN
- Sensor packet structure used by the SNs
- Protocol of each communication layer used by SNs and can generate control and data packets

Fundamental corollary from the discussions done in Chapter 3, is to distort minimum number of samples by injecting maximum amount of $d_e$ through shifting samples to their maximum or minimum quantization levels. Hence, ASN achieves a given distortion level energy efficiently by just injecting distortion to the minimum number of sensor packets.

Another corollary is the impossibility to determine the number of samples to distort in advance in order to achieve a given target distortion ($D_{Ta}$) which causes WSN to become unreliable. Therefore, an adaptive distortion mechanism is required based on the comparison of the given $D_{Ta}$ with the total injected distortion ($D_{Ti}$) which is determined by means of the incoming feedback about the distortion levels injected by the aSNs. In order to do this, a central controller aSN is needed to

compare these values and manage the attack operation by sending commands to other aSNs.

Therefore, initially an aSN is chosen as master aSN (aSN$^m$) to act as a central controller point and all the other aSNs are called slave aSNs (aSN$^s$). This selection procedure is called *MasterDetermination* algorithm and detailed in Section 4.2.

In order to start attacking sensor packets, initially some sensor parameters must be discovered by the aSN$^m$ such as initial sensor packet generation time ($T_{st}$) and sensor packet generation period ($T$). By means of these parameters, aSN$^m$ can determine the generation time of the future sensor packets which will be captured and distorted by the aSN$^s$s. Parameter determination is achieved after reception of sufficient number of report messages produced by the aSN$^s$s immediately after an event starts occurring. The sufficient number of reports is two for each aSN$^s$ and it is explained in Section 4.3 while detailing the procedure which is called *SensorParameterDetermination* algorithm.

In order to capture a sensor packet, an aSN$^s$ creates MAC layer collision to the Request-to-Send (RTS) packet and pretend to be the actual receiver SN for this packet. By means of this, ASN obtains resistance against Denial of Service (DoS) detection mechanisms. Then, the aSN$^s$ injects distortion error ($d_e$) to all the samples in the packet and transmits the packet to its destination MAC address. This capturing and injecting distortion procedure is called *DistortionInjection* algorithm and detailed in Section 4.4.

After a successful transmission of the distorted sensor packet, aSN$^s$ informs aSN$^m$ about injected distortion errors as a feedback information. By means of this feedback, aSN$^m$ calculates $D_{Ti}$ and determines the next sensor packet which must be captured and sends a command to all aSN$^s$ including the generation time of the packet. This determination procedure is called *CaptureTimeDetermination* algorithm and detailed in Section 4.5.

In fact, communication from aSN$^m$ to aSN$^s$s includes common information for all aSN$^s$s and hence, flooding mechanism is used by broadcasting the messages until they reach the farthest node in the network. However, communication from an

aSN$^s$ to the aSN$^m$ does not concern other aSN$^s$s and hence, it is point-to-point.

All the above procedures are summarized in Figure 4.1 in a message sequence transition chart assuming that aSN$^m$ determination is already done. All SNs start sending sensor packets periodically at time $T_{st}$ with a period of $T$. aSNs which hear any sensor packet transition in WSN, start transmission of *SensorReport* messages to the aSN$^m$. After determination of necessary parameters to attack a sensor packet, aSN$^m$ sends its initial command message called *AttackRequest* which states attack time $T_a$ and attack duration $T/2$ to all aSN$^s$s. Note that, in this figure, only one aSN is used in these transitions for the sake of simplicity. At time $T_a + T/2$, aSN sends distorted packet and an *AttackConfirmation* message to the receiver SN and the aSN$^m$ as a feedback information, respectively. As a result, aSN$^m$ determines the next sensor packet to attack and hence, operations start again in a cyclic manner beginning from the transmitting an *AttackRequest* message.

At the end of this chapter, whole ASN procedure which is called *ASNOverall-Operation* algorithm is summarized in a pseudo-code in Algorithm 5.

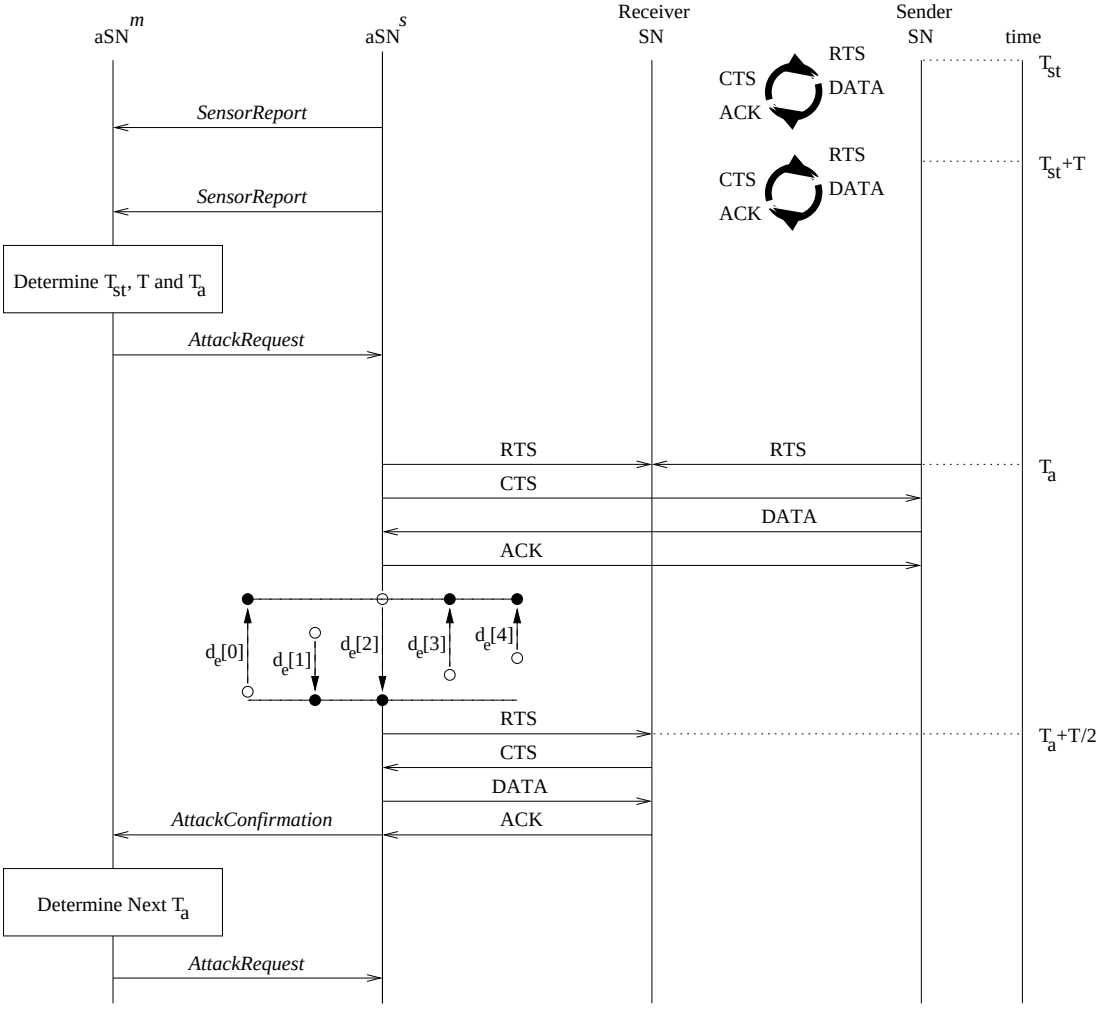


Figure 4.1: Typical message sequence transition in ASN.

## 4.2 Master Anti-Sensor Node (aSN$^m$) Determination

Initially, all aSNs set a random timer. First aSN whose timer expires, broadcasts a *MasterAdvertisement* message and sets its node type as master aSN (aSN$^m$). Whenever an aSN receives a *MasterAdvertisement* message it broadcasts the message in order to inform all its neighbor aSNs and sets its node type as slave aSN (aSN$^s$). This message contains the address of the aSN$^m$. Random timer interval is chosen sufficiently long such that the address of the aSN$^m$ is transmitted to the farthest aSN before another timer expiration. This sufficient duration depends on the point-to-point packet transmission duration and the number of aSNs in the network. As a result, address of the aSN$^m$ is spread to all other aSNs. Message transition of aSN$^m$ determination operation and its pseudo-code are shown in Figure 4.2 and in Algorithm 1, respectively.

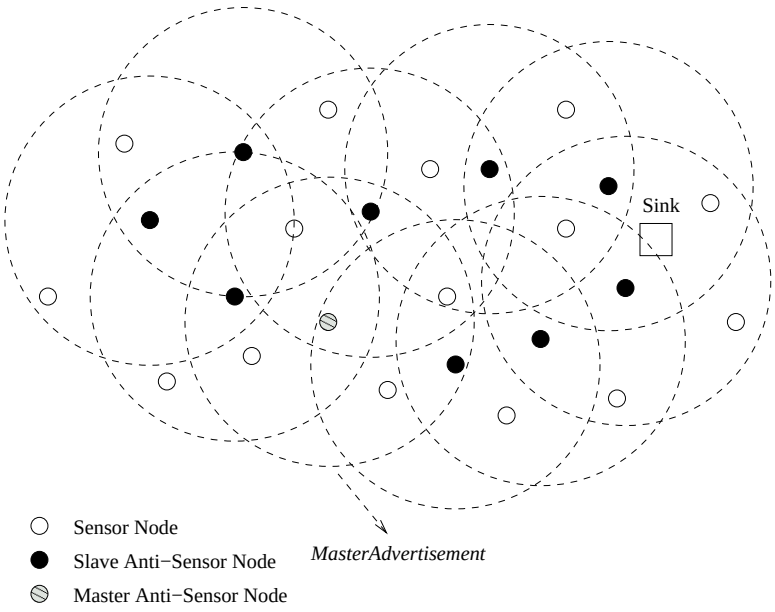


Figure 4.2: Master anti-sensor node (aSN$^m$) determination among aSNs.

**Algorithm 1**: *MasterDetermination*

```
1  start:
2  nodeType = Undetermined
3  Set(randomTimer)
4  if randomTimer expires then
5      SendBcMessage(ChAsn, MasterAdvertisement)
6      nodeType = Master
7  end
8  if rxMessage == MasterAdvertisement then
9      masterAddress = rxMessage.masterAddress
10     if nodeType == Master AND masterAddress != myAddr then
           // More than one node decide to be master
11         goto start:
12     end
13     else if nodeType == Undetermined then
14         SendBcMessage(ChAsn, MasterAdvertisement)
15         nodeType = Slave
16         Reset(randomTimer)
17     end
18 end
```

### 4.3 Eavesdropping and Intelligence Gathering

$T_{st}$ is the transmission time of the initial eavesdropped sensor packet and $T$ is the time difference between the transmissions of two consecutive sensor packets. Hence, in order to determine these values, it is sufficient to eavesdrop only two sensor packets. aSN$^s$s send a *SensorReport* message to the aSN$^m$ for each eavesdropped sensor packet. Each report consists of the following information:

- Sequence number of the eavesdropped sensor packet: $N_{seq}$
- Sampling time of the last sample in the sensor packet: $T_{samp}$

It is assumed that a SN transmits a sensor packet immediately after accumulating all the samples carried in a sensor packet. Therefore, second parameter of the *SensorReport* message gives the start time of transmission of a sensor packet.

Sensor packets are eavesdropped by all aSN$^s$s that hear a packet transition between any two SNs. As a result, aSN$^m$ receives superfluous reports and writes

them to a table and each time it sets a timer which has an interval greater than the maximum possible sensor packet period plus transmission time of a *SensorReport* message to a farthest aSN$^s$. When this timer expires, no more *SensorReport* message is expected and as a result, $T$ can be determined by just subtraction of two $T_{samp}$ values and then dividing the result with the difference of sequence numbers. Apparently, $T_{st}$ is the smallest $T_{samp}$ value among all the reports. Determination of $T_{st}$ and $T$ is given in a pseudo-code in Algorithm 2.

**Algorithm 2**: *SensorParameterDetermination*

```
1 if rxMessage == SensorReport then
2     SensorReportTable = InsertTable(SensorReport)
3     Set(DeterminationTimer)
4 end
5 if DeterminationTimer expires then
      // No more report, determine parameters
6     T=(T_samp[m]-T_samp[n])/(N_seq[m]-N_seq[n])
      // Sensor packet period
7     T_st = FindSmallest(T_samp[], SensorReportTable)
      // Initial generation time of sensor packets
8     ParametersDetermined = TRUE
9 end
```

Therefore, aSN$^m$ can determine the generation times of future sensor packets by just adding multiple times of $T$ to the $T_{st}$. For an instance, generation time of the $(k+1)^{th}$ sensor packet is $T_{st} + kT$. Additionally, aSN$^m$ can determine the total number of generated sensor packets ($N_{cur}$) until the current time as follows:

$$N_{cur} = (CurrentTime - T_{st})/T \tag{4.1}$$

### 4.4 Capturing Packets and Injecting Distortion by aSN$^s$s

As will be explained in detail in Section 4.5, aSN$^m$ sends an *AttackRequest* message which states the attack duration ($T/2$) and attack start time ($T_a$). All aSN$^s$s that receive this message start listening to the sensor channel beginning from the

time $T_a$. After time $T_a$, whenever an aSN$^s$ receives the first bit of an RTS packet, it starts sending an arbitrary RTS packet to collide the actual RTS instead of sending a continuous random jamming signal. aSNs which are out of range of any transmitting SN, do not interfere the attack operation. The reason of using an RTS packet as the jamming signal is to deceive SNs to suppose that it is a collision rather than an attack.

During collision operations, aSN$^s$ uses both of its transceiver units at the WSN frequency. As a result, it receives total of both its RTS jamming signal and collided RTS signal from one antenna while transmitting its RTS jamming signal from the other antenna. Therefore, if the transmitted jamming signal (RTS) is subtracted from the total received signal, aSN$^s$ may obtain actual RTS signal transmitted by the sender SN.

Actually, sender SN can not deduce these events because it can not receive anything while transmitting. Because of the link layer jamming (collision), actual receiver SN can not receive correct RTS packet and continues its operations like after a normal collision occurrence. aSN$^s$ starts transmission of CTS packet pretending to be the actual receiver for the captured RTS packet. After receiving CTS, sender SN sends data packet to the aSN$^s$ which acknowledges it with an ACK packet. Therefore, sender SN assumes that packet transmission is successful and actual receiver SN can never know about this data transmission. As a result, aSN$^s$ captures sensor packet successfully.

On the other hand, note that there may be various aSN$^s$s in the radio coverage area of the sender SN. Therefore, a problem arises about which aSN$^s$ will capture the transmitted RTS packet. This can be solved by setting a random timer after receiving the first bit of the RTS packet from the sender SN. Therefore, aSN$^s$s do not start transmitting jamming signal immediately but wait for a random timer duration. Maximum duration of this timer must be less than the duration of an RTS packet transmission. First aSN$^s$ whose timer expires, starts sending RTS signal and owns the right to capture this sensor packet. Other aSN$^s$s which sense an RSSI (Received Signal Strength Indicator) level jump, stop their timers and start listening to capture another RTS transmission until the end of the attack duration ($T_a + T/2$).

After capturing a sensor data packet, aSN$^s$ injects distortion errors to the samples by shifting them to the farthest quantization level namely the highest or the lowest quantization level.

Distorted packet must be sent to the actual destination before the next periodic sensor packet is generated by the SNs. Therefore, transmission time of distorted packet must be chosen in between $T_a$ and $T_a + T$ and most suitably it is chosen as $T_a + T/2$. As a result, duration of capturing operation is $T/2$ and all aSN$^s$s hold their captured data packets until the end of this time in a cooperative manner. This manner also prevents recapturing of already distorted packet by another aSN$^s$ uselessly. Otherwise, if one packet is captured by multiple aSNs, it will cause waste of energy for the ASN.

Therefore, aSN$^s$ starts sending distorted data packet after time $T_a + T/2$ by transmitting an RTS packet to the actual receiver whose MAC address is parsed from the original RTS packet. After receiving the CTS, it transmits distorted data packet to the receiver which acknowledges the aSN$^s$ with an ACK packet. As a result, capturing packet and injecting distortion operation completes successfully.

However, aSN$^s$ which receives an RTS packet of a sender SN, must be in the radio coverage of both sender and receiver SN. Because, captured packet is transmitted to the destination MAC address of the packet. For this reason, aSN$^s$ which receives the RTS signal above a certain RSSI quality level ($TH_{rssi}$) must attempt to start jamming operation. This level is determined according to the possible maximum communication distance between a sender SN and a receiver SN.

All these sequential message transitions are summarized in the pseudo-code given in Algorithm 3.

**Algorithm 3**: *DistortionInjection*

start:
Set(Antenna2, ChSn)
myOwnAddr = myAddr
**if** *RxFromChSn AND RxRssi* $> TH_{rssi}$ **then**
    Set(randomTimer)
    **if** *RSSI jumps* **then**
        Reset(randomTimer)
    **end**
    **else if** *randomTimer expires* **then**
        StartTx(ChSn, RTS, Antenna2)
        RTS = Decode(SignalOnAntenna1 - SignalOnAntenna2)
        `// Sensor RTS Signal = Total Signal - Jamming Signal`
        (Src, Dest) = Parse(RTS)
        myAddr = Dest
        SendPacket(ChSn, CTS, Src)
        ReceivePacket(ChSn, DATA)
        SendPacket(ChSn, ACK, Src)
        DistortedDATA = InjectDistortion(DATA, $d_e[]$)
        $N_{seq}$ = Parse(DATA)
        *AttackConfirmation* = PrepareMessage($N_{seq}$, $d_e[]$)
        wait($T$/2)
        myAddr = Src
        SendPacket(ChSn, RTS, Dest)
        `// Send actual RTS pretending to be the sender SN`
        ReceivePacket(ChSn, CTS)
        SendPacket(ChSn, DistortedDATA, Dest)
        `// Send distorted DATA to the actual destination`
        ReceivePacket(ChSn, ACK)
        SendPpMessage(ChAsn, *AttackConfirmation*, masterAddress)
        myAddr = myOwnAddr
    **end**
**end**
**if** *periodicFlag == TRUE* **then**
    goto start
**end**

Injected distortion errors are written to an array $d_e[]$ and sent to the aSN$^m$ in the *AttackConfirmation* message as a feedback information. By means of this feedback, aSN$^m$ becomes capable to determine the total injected distortion $D_{Ti}$. *AttackConfirmation* message includes two informations as follows:

- Sequence number of the distorted sensor packet: $N_{seq}$
- Distortion error array added to the samples of the captured sensor packet: $d_e[]$

$N_{seq}$ is necessary to prevent the duplication of distortion errors for a sensor packet with the same $N_{seq}$ captured by different aSN$^s$s. $d_e[]$ is used to determine the current injected distortion ($D_{Ti}$) by the aSN$^m$ to compare it to the target distortion ($D_{Ta}$).

Total distortion for an application can be found as follows:

$$D_T = \sum_{n=1}^{M} (e[n])^2 / M \tag{4.2}$$

where e[n] is the difference between transmitted and received values of the $n^{th}$ sample and M is the total number of samples received by the application.

Therefore, aggregated distortion ($D_{Agg}$) is determined by using the information in the *AttackConfirmation* message as follows:

$$D_{Agg} = D_{Agg} + \sum_{k=1}^{G} (d_e[k])^2 \tag{4.3}$$

where $G$ is the total number of samples in a sensor packet. As a result, $D_{Ti}$ can be easily calculated by dividing $D_{Agg}$ to $N_{cur}$ as follows:

$$D_{Ti} = D_{Agg} / N_{cur} \tag{4.4}$$

## 4.5 Adaptive Distortion Management by aSN$^m$

After determination of sensor parameters such as $T_a$ and $T$, aSN$^m$ broadcasts its initial *AttackRequest* message. This message is spread to all aSN$^s$s with flooding mechanism. $T_a$ for this initial message is the next possible generation time of the

sensor packet to attack and hence, it does not depend on any feedback. However, following *AttackRequest* messages depend on the received feedbacks after successful distortions injected by aSN$^s$s as will be explained in detail in this section.

Sensor data packet capturing operation must start at sensor packet transmission times such as $T_{st} + kT$ and hence, attack time ($T_a$) is chosen from these times. Determination of $T_a$ means the determination of the transmission time of the next sensor packet which will be captured by the aSN$^s$s. In ASN, $T_a$ is chosen as the time that $D_{Ti}$ value starts falling below the $D_{Ta}$ value. As a result, it is achieved to keep $D_{Ti}$ value above the $D_{Ta}$ value. Let $N_{next}$ is the sequence number of the next sensor packet which will be distorted next time and so we can obtain it by using (4.3) as follows:

$$N_{next} = D_{Agg}/(D_{Ta} * G) \tag{4.5}$$

where $D_{Ta}$ is the target distortion required by the application which is to be distorted. $D_{Agg}$ is updated each time after an *AttackConfirmation* message is received and as a result, $N_{next}$ is calculated to determine the next $T_a$ time after each reception of the *AttackConfirmation* message. aSN$^m$ compares $N_{next}$ to $N_{cur}$ to determine $T_a$ and then, informs all its aSN$^s$s about new $T_a$ time by broadcasting an *AttackRequest* message. aSN$^s$s rebroadcast the received message to their neighbors until it reaches all aSN$^s$s using flooding mechanism as shown in Figure 4.3.

If $N_{next}$ is smaller than $N_{cur}$, then aSN$^s$s must capture the sensor packets periodically with the period $T$. Because, in that case $D_{Ti}$ value is below the $D_{Ta}$ value and aSN$^s$s must distort all the following sensor packets until $D_{Ti}$ is greater than $D_{Ta}$. In this case, $T_a$ is chosen as the nearest sensor packet transmission time. In order to prevent to transmit *AttackRequest* messages for each following packets, the message includes a flag (*periodicFlag*) which can be set on and off before sending the message. Hence, if periodic capture is sent before, and $N_{next}$ is still smaller than $N_{cur}$, then there is no need to send an *AttackRequest* message again.

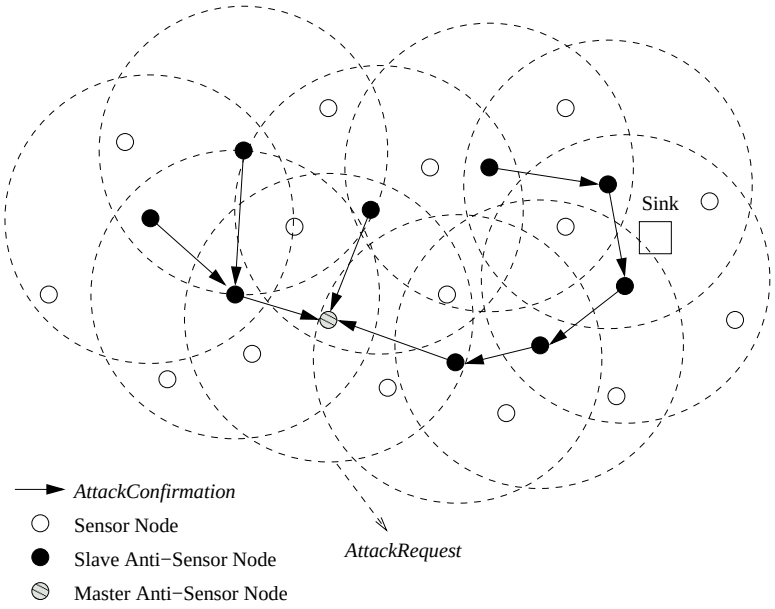


Figure 4.3: Transmission of distortion feedback by aSN$^s$s and attack operation triggered by the aSN$^m$.

Before sending *AttackRequest* message, $T_a$ value is shifted by some multiple of $T$ until it is sufficiently big such that the farthest aSN$^s$ receives this packet before time $T_a$. aSN$^s$ starts capturing simultaneously at time $T_a$ until the end of $T_a + T/2$ and if a sensor data packet is distorted, aSN$^s$ sends an *AttackConfirmation* message and then, waits for another *AttackRequest* message. However, if *periodicFlag* is on, then it continues capturing operation periodically with period $T$ at $T_a + kT$ times. Pseudo-code of $T_a$ determination is given in Algorithm 4.

**Algorithm 4**: *CaptureTimeDetermination*

```
1  if N_next > N_cur then
2      periodicFlag = FALSE
3      T_a = T_st + N_next * T
4      sendFlag = TRUE
5  end
6  else if periodicFlag == TRUE then
7      sendFlag = FALSE
       // Periodic capture is sent before and hence, no need
          to send again
8  end
9  else
10     periodicFlag = TRUE
       // Start periodic capturing
11     sendFlag = TRUE
12     while T_a < CurrentTime do
13         T_a = T_st + T
           // Next packet generation time
14     end
15 end
16 while T_a < (CurrentTime + MinPropToFarthest) do
17     T_a = T_a + T
       // Packet must reach all aSN^s s before time T_a
18 end
19 if sendFlag == TRUE then
20     SendBcMessage(AttackRequest, T, T_a, periodicFlag)
21 end
```

During these operations, if battery level of aSN$^m$ reaches below a certain threshold ($TH_{bat}$), it broadcasts a *MasterReleaseAdvertisement* message and all aSN$^s$s rebroadcast this message using flooding mechanism. Afterward, all aSNs start master determination operation as explained in section 4.2.

Overall ASN operations by the aSNs are summarized in a pseudo-code given in Algorithm 5. This algorithm combines all preceding algorithms explained in this chapter.

**Algorithm 5**: *ASNOverallOperation*

```
1  MasterDetermination
2  while nodeType == Slave do
3      reportCnt = 0
4      if RxChSn AND reportCnt < 2 then
           // Only 2 reports is enough for aSN^m
5          ReceivePpMessage(ChSn, DATA, FromSensors)
           // Listen to sensor nodes
6          SensorReport = PrepareMessage(T_samp,N_seq)
7          SendPpMessage(ChAsn, SensorReport, aSN^m)
           // Send report to aSN^m
8          reportCnt++
9      end
10     if RxChAsn AND AttackRequest then
11         SendBcMessage(ChAsn, AttackRequest)
12         (T, T_a, periodicFlag) = ParseMessage(AttackRequest)
           // Receive distortion parameters
13         if CurrentTime == T_a then
14             DistortionInjection
15         end
16     end
17     if RxChAsn AND MasterReleaseAdvertisement then
18         SendBcMessage(ChAsn, MasterReleaseAdvertisement)
19         MasterDetermination
20     end
21 end
   // End of while Slave
22 while nodeType == Master do
23     if rxMessage == SensorReport then
           // Receive reports from all aSN^s
24         SensorParameterDetermination
25         if ParametersDetermined == TRUE then
26             while T_a < (CurrentTime + MinPropToFarthest) do
27                 T_a = T_st + T
28             end
29             SendBcMessage(AttackRequest, T, T_a, FALSE /*periodicFlag*/)
               // Initial AttackRequest message
30         end
31     end
32     else if rxMessage == AttackConfirmation then
33         CaptureTimeDetermination
           // Determine next T_a and send AttackRequest message
34     end
35     if BatteryLevel < TH_bat then
36         SendBcMessage(ChAsn, MasterReleaseAdvertisement)
37     end
38 end
```

# CHAPTER 5

## SIMULATION RESULTS AND PERFORMANCE EVALUATION

In this chapter, we present simulation results and performance analysis of the ASN. Simulation is implemented using ns-2 tool in linux environment. A sensor network with 200m x 200m grid topology which consists of 50 SNs, 50 aSNs and a sink node is used. The parameters used in our simulation are given in Table 5.1.

Table 5.1: Simulation parameters

| | |
|---|---|
| Area of sensor field | 200×200 $m^2$ |
| Initial energy of nodes | 1.2V×0.5Ah [3] |
| Rx consumption power | 0.395 W |
| Tx consumption power | 0.660 W |
| Radio coverage of nodes | 40 m |
| Number of quantization levels | 256 |
| Lower and upper quantization limits | $0, 1$ |
| Number of events | 1 |
| Number of samples in packets | 5 |
| Number of SNs | 50 |
| Number of aSNs | 50 |
| MAC protocol | 802.11b |
| Transport protocol | UDP |
| Routing protocol | AODV |

First of all, we created two files, one of which consists of uniformly distributed values between 0 and 1 that simulate analog signal ($E$) and the other file is constructed from the first file by just adding normally distributed noise with mean 0 and standard deviation 0.01. It is assumed that total number of encapsulated samples in a sensor data packet is five and a phenom node broadcasts each five value in a phenom packet periodically in periods such as 0.25, 0.5 sec. Surrounding SNs which sense this packet, quantize the analog values in the packet with 256 quantization levels and send it to the sink in UDP packet by giving a sequence number.

When an aSN$^s$ is commanded to distort a sensor packet, it captures the sensor packet as explained in Chapter 4 and shifts the samples before sending them to the destination. Sink node receives all transmitted data packets which are either distorted or not distorted by the aSN$^s$s. Sink node determines the analog signal values by multiplying the quantization level of the samples with the quantization step. Then, it writes the results to an output file in receiving order. Therefore, event-to-sink achieved distortion ($D_{Ac}$) can be determined by adding all square differences between the values written in the output file with the values written in the first created file and then dividing the total to the number of transmitted samples as given in (4.2). In this simulation, sample values are uniformly distributed between 0 and 1 and hence, distortion values ($D_{Ta}, D_{Ac}$) are normalized in between 0 and 1.

## 5.1 Distortion Performance of ASN

In our ASN system, we have two varying parameters. One is sensor packet period ($T$) in seconds and other is $D_{Ta}$ which depends on the requirement of the sensor application which is to be attacked. In order to create a given $D_{Ta}$ at the sink node, aSN$^m$ quickly evaluates distortion feedbacks (*AttackConfirmation*) coming from aSN$^s$s and calculates next attack time. In order to obtain an initial feedback, first distortion command (*AttackRequest*) is sent immediately after the determination of sensor parameters. However, following *AttackRequest* messages are sent according to the incoming feedbacks from aSN$^s$s.

It is shown in Figure 5.1 that sequence number of the first distorted sensor packet is 4. At this time, $D_{Ac}$ jumps with an amount of distortion which does not depend on any feedback. Sequence number of the next distorted sensor packet is 61. At this point $D_{Ac}$ reaches $D_{Ta}$, because time of distortion is determined according to the feedback about the distorted packet with the sequence number of 4.

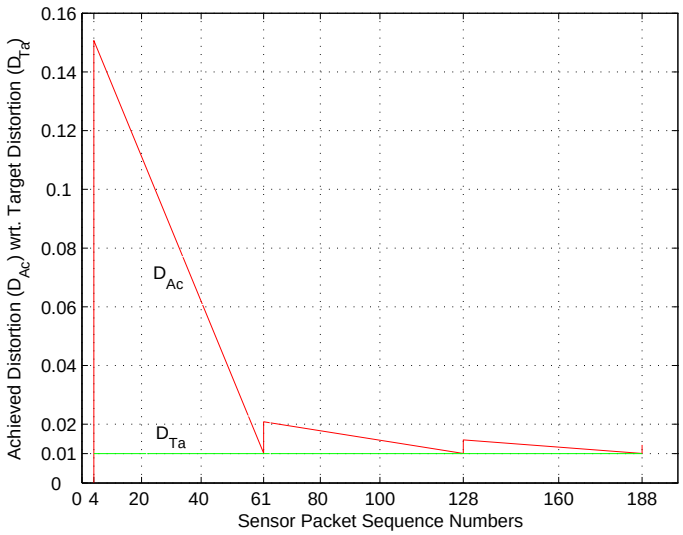


Figure 5.1: Variation of $D_{Ac}$ wrt. sensor packet sequence number $(D_{Ta} = 0.01)$.

Amount of jumps gets smaller for the following distortion points, because at each time, $D_{Ac}$ is calculated over all received packets until that time and hence, effect of a distorted packet to the overall distortion gets smaller as the number of received samples increases. As a result, $D_{Ac}$ approaches $D_{Ta}$ and keeps staying above it.

At these sequence number points (4,61,128,188), $D_{Ac}$ reaches $D_{Ta}$ as expected from the behaviour of the ASN as explained in Section 4.5. Peaks of jumps extend above the $D_{Ta}$ because of distortions at the points that $D_{Ac}$ reaches $D_{Ta}$.

It is shown in Figure 5.2 that $D_{Ac}$ follows $D_{Ta}$ adaptively without depending on $T$. In WSNs, $T$ is chosen as big as possible by encapsulating various samples in a sensor packet. As a result, traffic load in WSNs decreases and lifetime of the sensor

network prolongs. Hence, in our simulation, we choose the periods in between 0.5 seconds and 1.5 seconds.

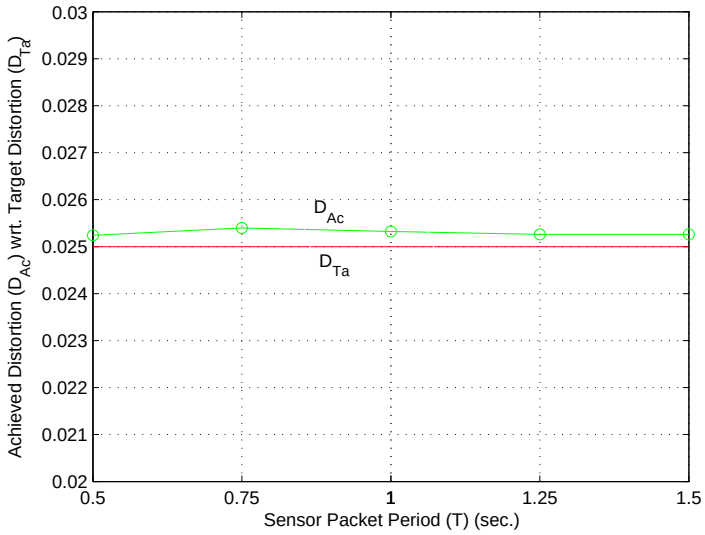


Figure 5.2: Variation of $D_{Ac}$ wrt. $T$ ($D_{Ta} = 0.025$).

Sensor networks are used for various applications which require different $D_{Ta}$ values. Therefore, behaviour of the ASN is shown in Figure 5.3 for various values of $D_{Ta}$. When $D_{Ta}$ equals to zero, all sensor packets reach the sink without being exposed to any distortion operation by aSN$^s$s. We see that ASN can follow $D_{Ta}$ successfully for small $D_{Ta}$ values for instance in between 0.01 and 0.1.

However, when $D_{Ta}$ increases beyond 0.1, $D_{Ac}$ saturates after some value. Sensor samples in our simulation are uniformly distributed between 0 and 1. Therefore, even all samples are distorted, $D_{Ac}$ can not exceed the saturation value which depends on the distribution of the sensor sample values. For instance, distortion value of 1 means that all sample values are 0 or 1 and they are shifted to the farthest quantization level which is 1 and 0, respectively.

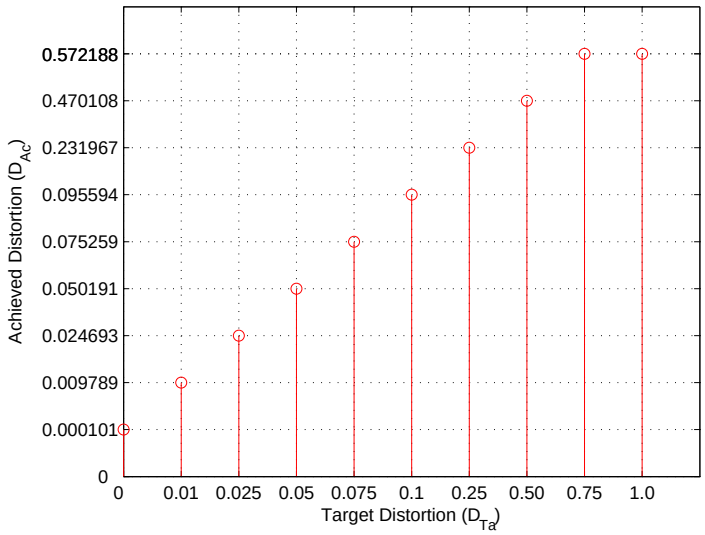


Figure 5.3: Variation of $D_{Ac}$ wrt. $D_{Ta}$ $(T = 0.25sec)$.

## 5.2 Lifetime Performance of ASN

Packet generation rate is an important factor for WSNs in terms of lifetime performance. On the other hand, lifetime of ASN depends mainly on the number of packets which is distorted by the aSN$^s$s. Therefore, ASN lifetime depends on both packet generation rate and $D_{Ta}$. In other words, when either packet generation rate or $D_{Ta}$ increases, ASN capture and distort more sensor packets which causes to decrease the lifetime of ASN.

SNs and aSNs are equipped with a limited power source ($<$0.5 Ah, 1.2 V) as pointed in [3]. Therefore, initial energy is chosen as 2160 J (0.5 $\times$ 1.2 VAh) and the lifetime values are measured in hour.

Variation of average lifetime of both SNs and aSNs for various values of sensor packet periods is shown in Figure 5.4 for a given $D_{Ta}$ value. Lifetime of both SNs and aSNs increases with the sensor packet period. Because when $T$ is small SNs generate more sensor packets and cause more energy consumption. On the other hand, aSNs capture and distort more sensor packets and hence, consume more energy. Additionally, it is shown that lifetime of aSNs are much more longer than lifetime of SNs for all values of $T$. As a result, ASN can successfully distort WSN

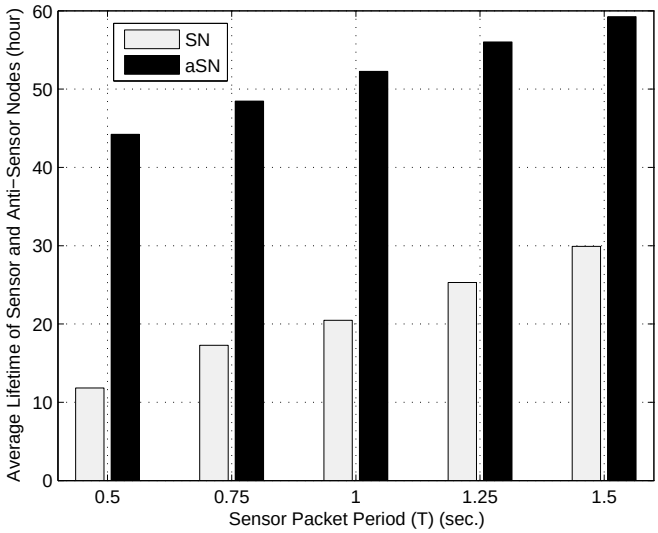


Figure 5.4: Average lifetime comparison of SNs and aSNs wrt. variation of $T$ ($D_{Ta} = 0.025$).

as long as the WSN operates.

Variation of average lifetime of both SNs and aSNs for various values of $D_{Ta}$ is shown in Figure 5.5 for a given value of $T$. Lifetime of SNs does not change so much by the variation of $D_{Ta}$ because $T$ is kept constant and hence, they generate nearly same traffic. However, lifetime of aSNs decreases because they capture and distort more sensor packets as $D_{Ta}$ increases. Moreover, above a certain $D_{Ta}$ value, lifetime of aSNs decreases below the lifetime of SNs. However, it is practically impossible for an application to require a $D_{Ta}$ value around 0.5 which is unreasonably large. It is surprisingly observed in this figure that lifetime of both aSNs and SNs increases for very big values of $D_{Ta}$ such as 0.75 and 1.0. Because, aSN$^m$ sends *AttackRequest* message by setting its *periodicFlag* bit on. If necessity of periodic distortion continues, aSN$^m$ does not uselessly transmit any more *AttackRequest* messages and hence, aSNs save a significant amount of energy. On the other hand, surprising increment of lifetime of SNs is explained in Section 5.3 by means of the reduction of generated traffic by the SNs.

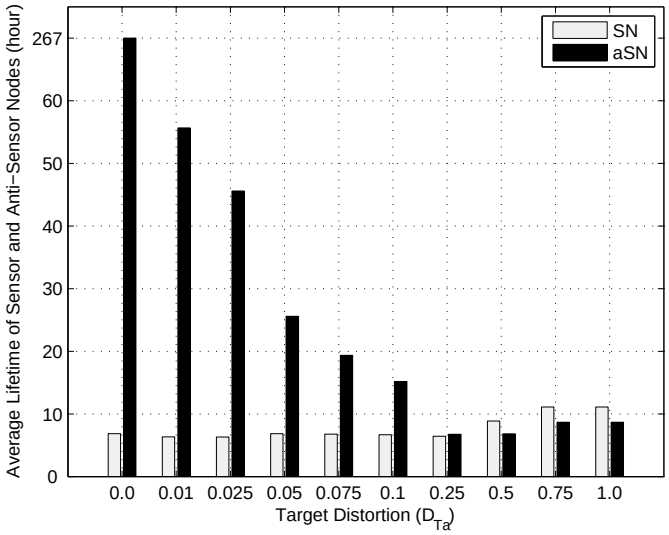


Figure 5.5: Average lifetime comparison of SNs and aSNs wrt. variation of $D_{Ta}$ $(T = 0.25sec)$.

### 5.3 Data Traffic Performance of ASN

Variation of data traffic in terms of packet generation of both SNs and aSNs for various values of $T$ is shown in Figure 5.6 for a given value of $D_{Ta}$. Note that, this traffic measure does not include control packet traffic and hence, it is similar, but not exactly same as lifetime performance. As $T$ increases, number of packets which must be distorted increases and as a result, traffic generation of ASN increases. Note that, at each case, traffic generated by the ASN is smaller than the WSN and hence, ASN operates efficiently in terms of traffic generation.

In addition, data traffic in terms of packet generation of both SNs and aSNs for various values of $D_{Ta}$ is shown in Figure 5.7 for a given value of $T$. As $D_{Ta}$ increases, number of packets which must be distorted increases and as a result, traffic generation of ASN increases. On the other hand, traffic generation of SNs does not change so much because $T$ is kept constant.

However, above a certain $D_{Ta}$ value, data traffic of both SNs and aSNs starts decreasing. This is because of too many collisions created by the aSNs and hence, SNs can not get a chance to transmit their data packets as collisions increase. This

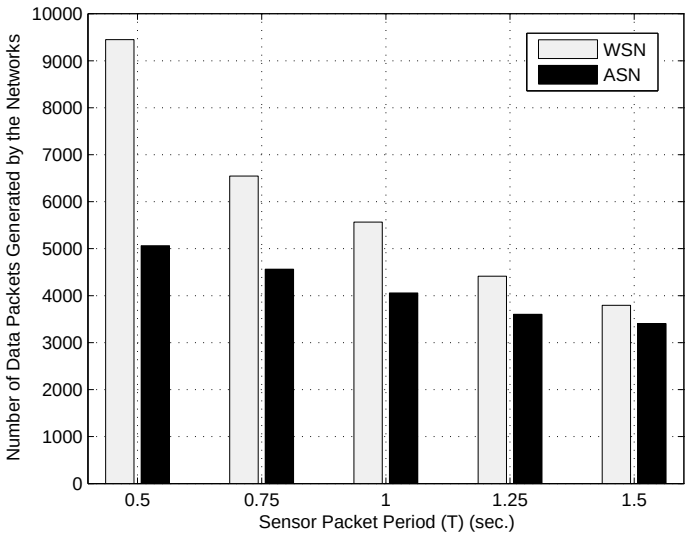


Figure 5.6: Total traffic comparison of WSN and ASN wrt. variation of $T$ ($D_{Ta} = 0.025$).

also explains surprising observation of lifetime increment of SNs as mentioned in Section 5.2 for the values of $D_{Ta}$ such as 0.75 and 1.0 as seen in Figure 5.5. Additionally, for these values, aSN$^m$ does not uselessly transmit any more *AttackRequest* messages by using the *periodicFlag* bit as explained in Section 4.5 and hence, data traffic of aSNs decreases.

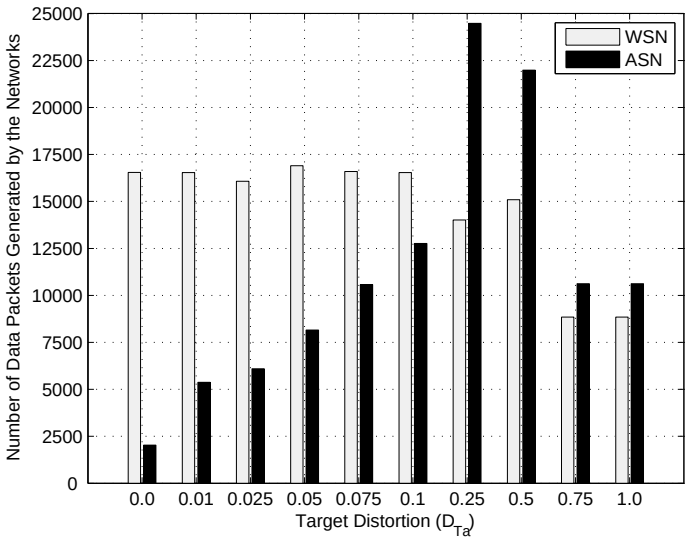


Figure 5.7: Total traffic comparison of WSN and ASN wrt. variation of $D_{Ta}$ $(T = 0.25sec)$.

# CHAPTER 6

# CONCLUSION

Our work is motivated by the fact that sensor applications can not achieve their performance needs if event-to-sink transport is unreliable. Therefore, in this thesis, we proposed a new anti-sensor network (ASN) paradigm which aims to damage event-to-sink reliability by adaptively monitoring the total injected event-to-sink distortion to the sensor application to force WSN malfunction. This notion of ASN distinguishes it from other existing DoS attacks. ASN is the first known study which attacks WSN in order to achieve the target event-to-sink distortion by modifying the sufficient number of sensor samples.

ASN is composed of distributed aSNs which are constructed by SN like hardware structure with low cost and limited capacities in terms of power source, radio coverage and computational capability. Therefore, energy efficiency is an essential requirement for the aSNs.

The key idea in order to provide energy efficiency is to distort minimum number of packets by injecting maximum amount of error to the samples with an adaptive mechanism that depends on feedback about injected distortion errors. Additionally, ASN is cost-effective because of low cost and easy deployment of aSNs.

ASN has cross layer attacking capability. aSNs create Request-to-Send (RTS) packet collision at the MAC layer to capture sensor data packets by pretending to be SNs and hence, they achieve resiliency against detection methods. Captured packets are distorted by injecting error to the samples at the application layer assuming that

sensor packets are not encrypted.

The effectiveness and adaptiveness of the ASN is verified through various simulations using ns-2. In order to do this, ASN is analyzed with respect to achieved distortion ($D_{Ac}$), lifetime and data traffic performance for a variety of network parameters such as sensor packet period ($T$) and target distortion ($D_{Ta}$). Results of the simulations showed that ASN can follow $D_{Ta}$ values successfully for various values of $T$. Additionally, lifetime of ASN is greater than the lifetime of the attacked WSN for various $D_{Ta}$ and $T$ values such that ASN can keep on attacking as long as WSN operates.

As a future work, some defense and attack mechanisms would be considered against the ASN such as encryption of sensor samples or construction of a DoS attack aSN$^m$ after determination of its location. Therefore, it would be possible to construct a self-organized ASN without an aSN$^m$ and without using another frequency channel other than WSN frequency channel, considering undetectability requirement.

In ASN, only a single event is considered with stationary aSNs. Therefore, separate event-to-sink distortions for multiple concurrent events would be studied and as well as mobile SNs and aSNs would be used.

ASN is proposed for WSNs which use CSMA/CA based MAC layer protocols with RTS/CTS mechanism. On the other hand, data capturing operation of ASN would be modified by considering other MAC layer protocols such as TDMA, FDMA or hybrid based schemes.

In ASN, event-to-sink distortion is created by shifting sensor sample values whereas sensor sampling time values written in the packets remain same without any modification by the aSNs. Performance of ASN would be studied by modifying the sampling times written in the sensor data packets.

In our simulations, number of aSNs are kept equal to the number of SNs. However, network size in terms of the number of aSNs would be varied and performance of ASN would be investigated. Therefore, it would be possible to distort event-to-sink reliability of WSN by using less number of aSNs than the number of SNs.

## REFERENCES


[1] M. Acharya, T. Sharma, D. Thuente, and D. Sizemore, "Intelligent Jamming in 802.11b Wireless Networks," *in Proc. of the OPNETWORK 2004 Conference,* Washington DC, USA, August 2004.

[2] Ö. B. Akan, I. F. Akyildiz, "Event-to-Sink Reliable Transport in Wireless Sensor Networks," *IEEE/ACM Transaction on Networking,*vol. 13, no. 5, pp. 1003-1016, October 2005.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, "A Survey on Sensor Networks," *IEEE Communications Magazine,* August 2002.

[4] C. Basile, M. Gupta, Z. Kalbarczyk, R. K. Iyer, "An Approach for Detecting and Distinguishing Errors versus Attacks in Sensor Networks," *International Conference on Dependable Systems and Networks, DSN 2006,* Philadelphia, PA, 2006, pp. 473-484.

[5] M. Cagalj, S. Capkun, and J. Hubaux, "Wormhole-Based Anti-Jamming Techniques in Sensor Networks," *IEEE Transactions on Mobile Computing,* vol. 6, issue 1, pp. 100-114, January 2007.

[6] P. Codenotti, A. Sprintson, and J. Bruck "Anti-Jamming Schedules for Wireless Broadcast Systems," *in Proc. of 2006 IEEE International Symposium on Information Theory (ISIT),* Seattle, Washington, July 2006, pp. 1856-1860.

[7] T. van Dam, and K. Langendoen, "An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks," *Proc. of the 1st International Conference on Embedded Networked Sensor Systems,* session: Energy-efficient MAC, Los Angeles, California, USA, November 2003, pp. 171-180.

[8] S. Datema, "A Case Study of Wireless Sensor Network Attacks," *Delft University of Technology,* Master's Thesis in Computer Science, September 2005.

[9] J. Deng, R. Han, and S. Mishra, "Decorrelating Wireless Sensor Network Traffic To Inhibit Traffic Analysis Attacks," *Elsevier Pervasive and Mobile Computing Journal, Special Issue on Security in Wireless Mobile Computing Systems,* vol. 2, issue 2, pp. 159-186, April 2006.

[10] J. Deng, R. Han, and S. Mishra, "Intrusion Tolerance and Anti-Traffic Analysis Strategies For Wireless Sensor Networks," *in Proc. of the 2004 International Conference on Dependable Systems and Networks (DSN'04),* vol. 0, 28 June - 1 July 2004, pp.637-646.

[11] G. P. Halkes, T. van Dam, and K. G. Langendoen, "Comparing Energy-Saving MAC Protocols for Wireless Sensor Networks," *Mobile Networks and Applications,* volume 10 , issue 5, pp. 783-791, 2005.

[12] L. van Hoesel, Y. W. Law, J. Doumen, P. Hartel, and P. Havinga, "Energy-Efficient Link-Layer Jamming Attacks Against Wireless Sensor Network MAC Protocols," *Proc. of the 3rd ACM Workshop on Security of Ad Hoc and Sensor Networks,* pp. 76-88, November 2005.

[13] Y. Hu, A. Perrig, and D. B. Johnson "Packet Leashes: A Defense Against Wormhole Attacks in Wireless Ad Hoc Networks," *In Proc. of IEEE INFOCOM 2003,* vol. 3, pp. 1976- 1986, April 2003.

[14] R. Kalidindi, L. Ray, R. Kannan, and S. Iyengar, "Distributed Energy Aware MAC Layer Protocol For Wireless Sensor Networks," *International Conference on Wireless Networks,* Las Vegas, Nevada, June 2003.

[15] C. Karlof, and D. Wagner, "Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures," *Elseiver's Ad Hoc Networks Journal, Special Issue on Sensor Network Applications and Protocols,* pp. 293-315, September 2003.

[16] M. Krishnan, "Intrusion Detection in Wireless Sensor Networks," *Department of EECS-University of California at Berkeley,* Project, Spring 2006.

[17] M. Kubish, H. Karl, and A. Wolisz, "A MAC Protocol for Wireless Sensor Networks with Multiple Selectable, Fixed Orientation Antennas," *Telecommunications Networks Group,* Technische Universitat Berlin, Frequenz, Zeitschrift für Schwingungs und Schwachstromtechnik, March 2004.

[18] Y. W. Law, P. Hartel, J. den Hartog, and P. Havinga "Link-Layer Jamming Attacks on S-MAC," *in Proc. of 2nd EWSN,* pp. 217-225, December 2005.

[19] Y. W. Law, "Key Management And Link-Layer Security of Wireless Sensor Networks," *Centre for Telematics and Information Technology (CTIT),* Ph. D. Thesis Series, Chapter 7, December 2005.

[20] R. Lin, Z. Wang, and Y. Sun, "Energy Efficient Medium Access Control Protocols for Wireless Sensor Networks and Its State-of-Art," *2004 IEEE International Symposium on Industrial Electronics,* vol. 1, May 2004, pp. 669-674.

[21] Y. Liu, Y. Li, and H. Man, "MAC Layer Anomaly Detection in Ad hoc Networks," *Proc. from the 6th Annual IEEE SMC Information Assurance Workshop, 2005, IAW '05,* pp. 402-409, June 2005.

[22] G. Lu, B. Krishnamachari, and C. S. Raghavendra, "An Adaptive Energy-Efficient and Low-Latency MAC for Data Gathering in Sensor Networks," *Proc. 18th International Parallel and Distributed Processing Symposium,* April 2004, pp. 224-.

[23] X. Luo, and G. B. Giannakis, "Energy-Constrained Optimal Quantization for Wireless Sensor Networks," *2004 First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, IEEE SECON 2004,* 4-7 October 2004, pp. 272-278.

[24] K. Mahedevan, S. Hong, and J. Dullum, "Anti-Jamming: A Study," *www.users.itlabs.umn.edu/classes/Fall-2006/csci5271/jamming.pdf,* Project Document, December 2005.

[25] M. Manzo, T. Roosta, and S. Sastry, "Time Synchronization Attacks in Sensor Networks," *Proc. of the 3rd ACM Workshop on Security of Ad Hoc and Sensor Networks,* pp. 107-116, November 2005.

[26] R. Negi, and A. Perrig, "Jamming Analysis of MAC Protocols," *Department of Electrical and Computer Engineering/Carnegie Mellon University,* Technical Memo, February 2003.

[27] I. Oppermann, L. Stoica, A. Rabbachin, Z. Shelby, and J. Haapola "UWB Wireless Sensor Networks: UWEN-A Practical Example," *IEEE Communications Magazine,* vol. 42, issue 12, pp. S27-S32, December 2004.

[28] M. Pirretti, S. Zhu, V. Narayanan, and R. Brooks "A Comparison of Mechanisms for Improving TCP Performance over Wireless Links," *IEEE/ACM Transactions on Networking,* vol. 5, pp. 756-769, December 1997.

[29] J. Polastre, J. Hill, and D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks," *Proc. of the 2nd International Conference on Embedded Networked Sensor Systems,* Baltimore, MD, USA, session: Routing and MAC, November 2004, pp. 95-107.

[30] S. Sancak, E. Cayirci, V. Coskun, and A. Levi, "Sensor Wars: Detecting and Defending Against Spam Attacks in Wireless Sensor Networks," *2004 IEEE International Conference on Communications,* vol.6, June 2004, pp. 3668-3672.

[31] H. Song, S. Zhu, and G. Cao, "Attack-Resilient Time Synchronization for Wireless Sensor Networks," *Proc. of IEEE International Conference on Mobile Ad-hoc and Sensor Systems,* November 2005.

[32] C. Suh, Y. Ko, and D. Son, "An Energy Efficient Cross-Layer MAC Protocol for Wireless Sensor Networks," *Proc. of IWSN,* pp. 410-419, 2006.

[33] D. J. Thuente, and M. Acharya, "Intelligent Jamming in Wireless Networks with Applications to 802.11b and Other Networks," *in Proc. of the 25th IEEE Communication Society MILCOM,* October 2006.

[34] H. Vogt, M. Ringwald, and M. Strasser, "Intrusion Detection and Failure Recovery in Sensor Nodes," *Workshop Proc. in Tagungsband INFORMATIK 2005,* LNCS, Heidelberg, Germany, September 2005.

[35] A. D. Wood, and J. A. Stankovic, "Denial of Service in Sensor Networks," *IEEE Computer,* vol. 35, issue 10, pp. 54-62, October 2002.

[36] A. D. Wood, J. A. Stankovic, and S. H. Son, "JAM: A Jammed-Area Mapping Service for Sensor Networks," *Proc. of the 24th IEEE International Real-Time Systems Symposium,* December 2003, pp. 286-297.

[37] W. Xu, W. Trappe, Y. Zhang, and T. Wood, "The Feasibility of Launching and Detecting Jamming Attacks in Wireless Networks," *Proc. of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing,* Urbana-Champaign, IL, USA, session: Security, May 2005, pp. 46-57.

[38] W. Ye, J. Heidemann, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," *IEEE Proc. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies,* vol. 3, pp. 1567-1576, June 2002.

[39] W. Ye, and J. Heidemann, "Medium Access Control in Wireless Sensor Networks," *USC/Information Sciences Institute,* Technical Report ISI-TR-580, October 2003.

[40] M. Younis, and T. Nadeem, "Energy Efficient MAC Protocols For Wireless Sensor Networks," *Book Chapter, Wireless Ad-Hoc and Sensor Networks,* Ed. Ahmed Safwat, Kluwer Academic Publishers.

[41] Y. Zhang, and W. Lee, "Intrusion Detection in Wireless Ad-Hoc Networks," *Proc. of the 6th Annual International Conference on Mobile Computing and Networking,* Boston, Massachusetts, United States, 2000, pp. 275-283.