FAST AND ENERGY-EFFICIENT TECHNIQUE FOR JAMMED REGION MAPPING IN WIRELESS SENSOR NETWORKS

by

NABILA RAHMAN

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT ARLINGTON
August 2010

To my parents who have supported me and inspired me all through my life.

ACKNOWLEDGEMENTS

There are so many people I'm grateful to, I'm almost at a loss on how to include all these names within this limited scope. But, first and foremost I would like to express my deepest gratitude to my supervising professor Dr. Matthew Wright not only for supporting me and guiding me throughout the course of my graduate studies, but also for being a very good teacher and showing appreciation in my works. I can definitely say and surely all my lab-mates will agree that he is one of the best advisors in UTA.

I would also like to thank Dr. Dongangg Liu and Dr. Chengkai Li for taking time to be on my thesis committee. Their valuable ideas, suggestions and constructive criticism of my work served to give my thesis work overall improvement and clarity.

I have no words to express my gratitude towards my parents, Dr. Md. Mizanur Rahman and Dr. Lutfun Nahar for all their love, guidance and care since the day I was born so that I can get the very best and have a fulfilling life. Also to my dearest grandparents who are more like second parents to me. I also wish to thank my loving husband Quazi Mainul Hasan, for showing extreme endurance and support for almost eight years and for taking this commitment for life. I would like to thank my brothers Ridwan Rahman and Rakeen Rahman for all the distractions they created for me.

I'm grateful to my country Bangladesh, which occupies a small area in the world map, still providing us with the opportunity to reach our dreams. I would also like to thank all my Bangladeshi friends here in Arlington, who made it possible for me to make Arlington my second home. I would also like to thank all my non-Bangladeshi friends here for making me forget that I am in fact living in a foreign country.

I'm deeply grateful to all my lab-mates and friends in ISEC, UTA, for their friendly support whenever I needed them. Special thanks to Qi Dong, Titus Abraham, Kush Kothari, Pranav Krishnamoorthy, Kartik Shiddhabathula, Vritant Jain, Prateek Gianchandi, Apurv Dhadphale, Gauri Vakde, Nayantara Mallesh, Dazhi Zhang, Na Li, Mayank Raj, Nishant Raithatha, Kiran Bhatta, Jun-Won Ho, Fei Xu, Austin Miller, Ot'lantis Adedeji, for making the lab such a nice place to be.

I would like to express my gratitude to the Dean of Graduate Studies and Department of Computer Science and Engineering for the funding they provided to me. Working with the department as a teaching assistant and research assistant have given me valuable experience. I would like to thank the University of Texas at Arlington for the academically stimulating environment.

Above all, I would like to say my prayer and humble gratitude to Allah for all the blessings I have received so far in my life.

June 15, 2010

ABSTRACT

FAST AND ENERGY-EFFICIENT TECHNIQUE
FOR JAMMED REGION MAPPING IN
WIRELESS SENSOR NETWORKS

Nabila Rahman, M.S.
The University of Texas at Arlington, 2010

Supervising Professor: Matthew Wright

Wireless sensor networks (WSNs) have great practical importance for surveillance systems to perform monitoring by acquiring and sending information about intrusions into a secured area. Very little human intervention is required for WSNs, which is one of the most desirable features of them and thus making them a cheaper and safer alternative for securing large area such as international borders. Jamming attacks in WSNs can be applied to disrupt communications among the sensor nodes in the network by barring the reception of wireless signals of the sensor nodes' transceivers. Since it is difficult to prevent jamming attacks, detection and mapping out the jammed regions is critical to overcome this problem. In a security monitoring scenario, the network operator will be able to take proper measures against jamming once the jammed regions in the network are known to them. It is also desirable to keep the interactions of the sensor nodes in the network minimal, as they are low-powered devices and need to conserve their resources. In this paper, we propose a light-weight technique for faster mapping of the jammed regions. We minimize the

load on the sensors by removing the actual responsibility of mapping from the network to a central base station (BS). After a few nodes report to the BS, it carries out the task of mapping of the jammed regions in the network. We use our simulation results to compare our proposed system with the existing techniques and also to measure the performance of our system. Our results show that the jammed regions in a network can be mapped from just a few nodes reporting to the base station. We used our results to measure the speed, overhead and accuracy of mapping by varying different parameters of the network. Also, we tested with real sensor nodes in the presence of a jamming sensor to experiment with jammed regions and map them using our proposed system.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Wireless sensor networks (WSNs) are making their way from research to real-world applications due to recent advances in electronics and wireless communication technologies. Since cheap commodity devices can be used as sensor nodes, it is possible to do large scale deployment of sensor networks [1, 2]. They are ideal for monitoring to detect intruders physically entering a secured important region [3]. Thus, sensor networks have a wide variety of applications in security monitoring, such as protecting water supplies, chemical plants and nuclear power plants and in border security and battle field surveillance.

A wireless sensor network (WSN) typically consists of a large number of autonomous devices with limited battery power and memory. Although a great amount of research has been devoted to WSNs, they will not be successfully applied if security, dependability, and privacy issues are not adequately addressed. These issues arise from the fact that WSNs are usually used for critical applications and are left unattended for efficient, low-cost monitoring. Thus, they are deployed without any physical protection. Also, the nodes in a WSN communicate with each other through a shared wireless medium. These two features make WSNs particularly vulnerable to a variety of attacks [4, 5, 6]. Jamming [7, 8] is a particularly effective attack against WSNs. An intruder can easily place jamming devices in different parts of the network to cause radio interference and thus disrupt communications among the sensor nodes that are in close proximity to the jamming devices.

1.1 Motivation

A jamming attack effectively creates a denial of service condition in the network. This a major problem in security monitoring applications, in which the lack of sensor communication means that an intruder can physically enter jammed regions without the threat of being detected. For example, in a border security setting, a path may be constructed with jamming devices that allows the intruder to cross back and forth across the border, completely bypassing the security perimeter. In this case, denial of service in the network leads to a major breach of physical security. It is critical in these monitoring applications for the base station to learn about and map out the jammed regions quickly and accurately so as to know where physical security may be threatened and where it may be necessary to increase other security measures like guard patrols or surveillance flights.

Wood et al. propose JAM [9], a jammed area mapping technique, that relies on the ability of the nodes to perform a detailed mapping of the jammed region locally. JAM is very effective at mapping out the jammed region. However, it is also a very complex protocol with high message and storage overheads at the nodes, due to its fully decentralized nature. It requires a lot of interaction among the nodes surrounding the jammed regions to estimate the region and correctly put the jammed nodes into groups. In settings such as the border security scenario, it is vital to protect the sensor network and to detect the intruder as early as possible while keeping communication overhead low to save sensors' battery lifetime.

1.2 Contributions of Thesis

In this thesis we present a method for quickly mapping out the jammed regions in a WSN. We are the first to point out that mapping jammed nodes should be done

quickly and *efficiently*, not just accurately. This observation suggests a different set of design choices than those made in JAM. In particular, instead of mapping being performed locally by the sensor nodes, we leverage the powerful base station to gather information from the network and calculate where the jammed regions are.

In our protocol, rather than an exact mapping, we only aim to get an approximation of the jammed area computed by the central base station. We apply k-means clustering to accurately separate out multiple jammed regions. We then invoke a method based on convex-hull-finding algorithms to find the centers for these regions and then apply iterative adjustment to accurately locate and size the region. Since the main task of mapping is carried our by the base station, this approach relieves the low-powered sensor nodes surrounding a jammed area from the communication overhead and power consumption of calculating the jammed regions.

In this thesis, we present our complete protocol for jammed region mapping for WSNs. We developed a simulator to evaluate our system and compare it with JAM in terms of effective mapping and communication overhead. Our results demonstrate that the proposed protocol performs faster mapping, thereby saving substantial message overhead compared with JAM, and that it provides reasonable mapping accuracy. We also experiment with the tradeoff between the communication overhead of the sensor nodes and the mapping performance of the system. Finally, we describe the results of a set of experiments we conducted on real sensor motes to see the performance of our jammed region mapping technique.

1.3 Organization of this Thesis

The thesis is organized as follows. Chapter 2 gives a detailed background on WSNs and jamming attacks, and it covers the recent literature in existing defense, detection of jamming, and jammed region mapping techniques. Chapter 3 presents

our proposed mapping technique in detail, including a brief overview of the network and intruder models used for this system. Chapter 4 describes the simulations we run to evaluate our proposed system with corresponding results and analysis. Chapter 5 explains our experiments on real sensor motes for jammed region mapping and the results of these experiments. Conclusions from this work are drawn and the scope for future work is discussed in Chapter 6.

CHAPTER 2

BACKGROUND

In this chapter, we provide a general overview of wireless sensor networks (WSNs) and jamming attacks in WSNs while discussing the relevant works in this area.

2.1 WSN

Wireless communications and digital electronics have enabled the development of low-cost, low-power, multi-functional sensor nodes that are small in size and communicate with each over short distances. A wireless sensor network (WSN) usually is formed by collaborative effort of these sensor nodes each containing separate components for sensing, data processing, and communicating with other nodes. Since these nodes are generally inexpensive and small in size, it is possible to deploy them as a large scale WSN.

The sensor nodes are typically resource constrained with limited energy lifetime, low-power micro-sensors and actuators, slow embedded processors, limited memory, and low-bandwidth radios. A sensor node generally consists of five basic parts [1, 10]:

- Sensing unit
- ADC (Analog to Digital converter)
- Micro-controller for processing
- Transceiver unit for communication
- Storage unit
- Power unit

A basic diagram for sensor node architecture is shown in Fig. 2.1. It may have additional components attached such as GPS (to find location) or a power generator. Their sensing units usually possess the ability to sense light, sound, temperature, humidity etc.

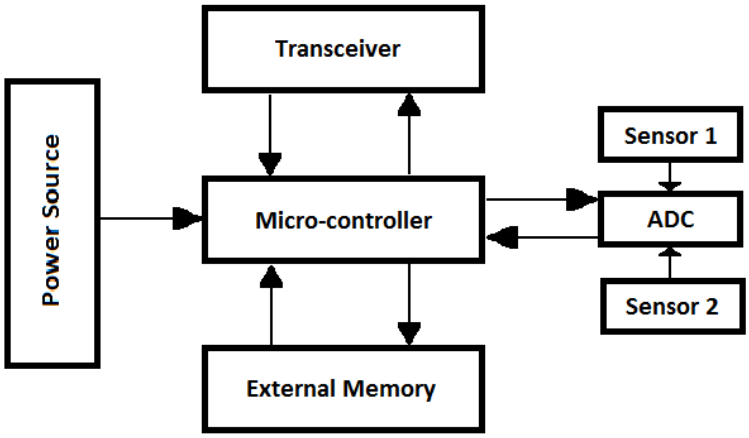


Figure 2.1. Basic Sensor Node Architecture.

The communication protocol stack used in sensor nodes is defined with the corresponding responsibilities as follows [1]:

1. **Physical layer:** frequency selection, carrier frequency generation, signal deflection and modulation
2. **Data link layer:** medium access control (MAC), multiplexing of data streams, data frame detection, data encryption, and error control
3. **Network layer:** assignment of addresses and forwarding of packets
4. **Transport layer:** reliable transport of packets and data encryption
5. **Application layer:** specifying how data is requested

The sensor nodes are typically self-organized to form an ad hoc, multi-hop wireless network that collects and forwards sensor data to an information sink, usually

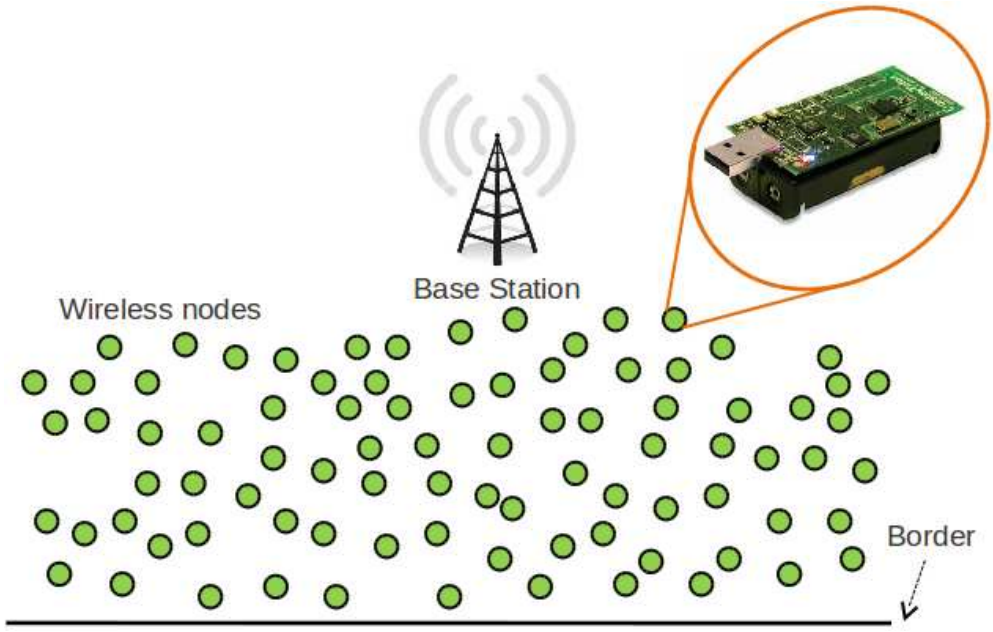


Figure 2.2. Wireless sensor network for border monitoring.

a base station acting as a gateway to the wired Internet [1]. The structure of a typical wireless sensor network is illustrated in Figure 2.2. In general, the computing resources of the base station are much greater than those of the sensor nodes.

Applications of WSNs are rapidly emerging and have become increasingly diverse, ranging from environment or habitat monitoring to industrial monitoring and so on. The wide variety of WSN applications in monitoring includes area monitoring, e.g. in a battlefield or for border security (Fig. 2.2), where a large number of sensors are deployed to cover a large area to perform security monitoring and take reactive measures in case of any intrusion in that area.

2.2 Jamming attacks

One major concern for WSNs arise from the fact that they use wireless communications, which is susceptible to radio interference, both accidental and malicious.

Jamming, one of the most serious security threats against WSNs, involves transmission of powerful interference that overpowers the receiver and drowns out the original transmission. It is defined as the act of intentionally directing electromagnetic energy towards a communication system to disrupt or prevent signal reception [11]. By applying radio interference, jamming makes it impossible to interpret radio signals, effectively creating a denial-of-service scenario at the physical communication layer.

Jamming is a well-known attack on WSNs that interferes with the radio frequencies a network's nodes are using. An adversary can disrupt the entire network with k randomly distributed jamming nodes, putting N nodes out of service, where k is much less than N. For single frequency networks, this attack is simple and effective [5].

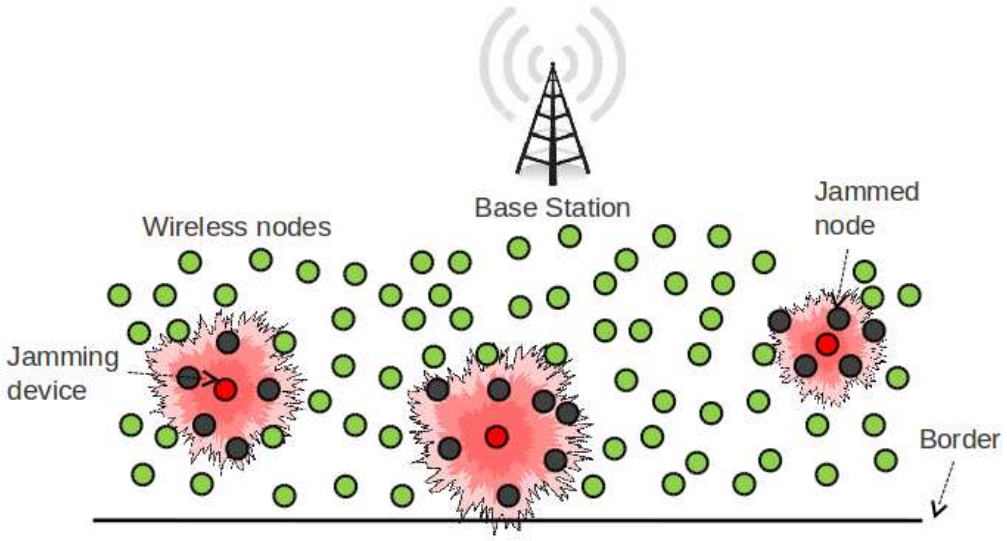


Figure 2.3. Jamming in a WSN.

Even in a WSN that utilizes strong network and application layer security mechanisms such as encryption for confidentiality and authentication to ensure data integrity, jamming can still be effective because of following reasons:

1. As wireless signals travel over an inherently **shared medium**, an adversary can inject false messages or emit radio frequency (RF) signals to block the wireless medium and thus prevent other wireless devices from communicating.
2. Most of the nodes currently used in WSNs follow the IEEE 802.15.4 protocol that specifies the lower protocol layers – the PHY-layer and the MAC portion of DLL. The protocol is designed for a type of wireless personal area network (WPAN) which focuses on low-cost, low-speed, ubiquitous communication between nodes [12]. It allows nodes to connect to the wireless network via a **limited number of frequencies**, that makes them susceptible to jamming attacks.
3. Most wireless networks consist of commodity devices that are cheap and have low output power (maximum output power is generally 0dBm [13]). Thus, an attacker can easily jam small region in a WSN using a low-power radio and disrupt communications. For this reason jamming a WSN can be considered as *low-cost attack*. A regular sensor node can be programmed as **jamming device** to emit "useless" information that floods the network and can either be white noise or a signal resembling network traffic. This either blocks the communication channel preventing nodes from being able to access the channel to send messages or introduces packet collisions that force repeated back-off.

Jamming can easily lead to denial-of-service (DoS) attacks. A DoS attack is an event that diminishes or eliminates a network's capacity to perform its expected functions [5]. DoS attacks are a major security problem in security monitoring applications, in which the lack of sensor communication means that an intruder can

physically enter jammed regions without being detected. For example, in a border security setting, a path may be constructed with jamming devices that allows the intruder to cross back and forth across the border, completely bypassing the security perimeter. In this way, DoS in the network leads to a major breach of physical security. Fig. 2.3 shows the border security deployment scenario of Fig. 2.2 in which the gray sensor nodes have been jammed by the jamming devices present in the network.

2.3 Jamming techniques and types

Xu et al. [7] define a jammer as an entity who purposefully tries to interfere with the physical transmission and reception of wireless communications. Based on attack strategies, they propose four generic jammer models:

1. A **constant jammer** aims at keeping the channel busy and disrupting nodes communication by continuously emitting random radio signals (noise) on a channel that would normally be used for regular transmissions. The constant jammer sends these continuous messages without following any underlying MAC protocol. Thus, the nodes in the network that correctly follow the MAC protocol are forced to be in a continual wait state until the channel clears, therefore blocking valid reception of the nodes in the network.
2. A **deceptive jammer** emits signals resembling network traffic, so that he can deceive network's defensive mechanisms. This causes the jammed node stay continuously in receiving mode, regardless of whether it has other messages to send or receive.
3. A **random jammer** sleeps for a random period of time and then wakes up and jams the network for a random period of time to trade off between jamming effectiveness and power usage.

4. A **reactive jammer** listens for any activity on the channel and sends out random signals to collide with incoming signals in the case that activity is observed. This allows jammer to both conserve energy and also help him to be less noticeable.

All four of these methods serve the same purpose: to cause collisions with existing signals on the medium and thus make this instance of communication worthless. Constant, deceptive, and reactive jammers are all very effective and, if placed a suitable distance from the victim nodes, can cause the packet delivery ratio (PDR) to fall almost to zero. PDR values can be measured by both sender and receiver node. At the sender side, it is calculated by the number of acknowledgements being received and on the receiver side it is determined by applying a CRC (cyclic redundancy check) on the received packets.

The main drawback of these jammers is that they are not energy-efficient and would exhaust their energy sooner than their victims. Random jammers save energy by sleeping, but they are less effective, as they can not block the signals all the time.

When the nodes in a WSN use multiple channels to communicate, a jammer can apply any of the following different jamming techniques [14]:

1. In the **spot jamming** technique, the jammer directs all its transmission power on a single frequency. This is a powerful technique for jamming attack, because if enough power is applied it can override the original signal. But the nodes in the network may be able to avoid it by switching to another frequency.
2. In **sweep jamming**, the jammer shifts rapidly from one frequency to another and thus is able to jam multiple frequencies in quick succession.
3. In **barrage jamming**, the jammer jams a range of frequencies at a time. This attack is also effective but requires higher output power for the attackers to jam multiple frequencies at a time.

Table 2.1. Jamming in context of communication protocol stack

Layer	Jamming attacks	Jammer Type
Physical	Radio jamming attacks	Constant, random, reactive
Data link	Energy-efficient link-layer jamming attacks	Deceptive

Table. 2.1 shows different jamming attacks in the lower level communication layers of the WSN (briefly discussed in section 2.1).

2.4 Defenses against jamming

Defending against jamming attacks is critical to the security of WSNs, since jamming can greatly constrict the potential of the network by disturbing or completely stopping the normal flow of information.

The traditional approach to coping with radio interference is to employ sophisticated physical-layer technologies, such as spread spectrum. For this reason, countermeasures against jamming are usually addressed through spread spectrum techniques, such as frequency hopping spread spectrum (FHSS) and direct sequence spread spectrum (DSSS) [8, 15, 16]. To attack frequency hoppers, jammers must be able either to follow the precise hopping sequence or to jam a wide section of the wireless band. Mobile phone networks commonly use code spreading as a defense against jamming. We now briefly describe some of these countermeasures:

Frequency Hopping Spread Spectrum (FHSS) transmits radio signals by rapidly switching a carrier among many frequency channels based on a shared sequence known both to the transmitter and the receiver. Both the nodes sharing this sequence, change the frequencies of the transceivers several times within a certain time period. Since this frequency sequence is unknown to the jammer, it is difficult

for the him to jam the entire signal, and may only jam the communication for only a fraction of that time period. In this way, FHSS minimizes the effect of jamming in the network.

In **Direct Sequence Spread Spectrum** (DSSS), before transmission, the data is multiplied with a pseudo-noise (PN) signal. This PN signal is a digital signal composed of a pseudo-random sequence of values 1 and -1 with a frequency much higher than the original signal. By this process, the RF signals are replaced with a very wide bandwidth signal and, at the receiving end, the received signal is multiplied again with the PN signal to recover the original data. Ultra-wide band technology (UBW) [17] is another spread spectrum technique that applies a modulation technique based on transmitting very short pulses on a large spectrum of a frequency band simultaneously, thereby making the transmitted signal very hard to be jammed.

These spread spectrum techniques have significant advantage over jamming because the intruder needs to employ a much higher amount of energy to jam such a wide range of frequencies. Unfortunately, these spread spectrum abilities require greater design complexity, as the sensor nodes will require advanced transceivers. Also, cryptographically secure spread spectrum radios are currently not readily available for very small constrained devices. So it is more likely that the application of low-cost, low-power sensor devices will be limited to single frequency use.

Other countermeasures include antenna polarization and directional transmission (instead of typical omni-directional antennas) that can be useful for reducing interference from unwanted sources [18].

Since it is extremely difficult to prevent jamming attacks in WSNs, various security schemes have been proposed in the WSN literature to address this issue. These security measures can be classified as *proactive*, *reactive*, and *mobile-agent-based* techniques.

Proactive countermeasures are performed in the background and they provide instant response against jamming at the expense of increased computation and energy cost. Some of the proposed proactive countermeasures include detection of jamming [7, 9, 19] and MAC-level countermeasures based on different MAC protocols (S-MAC, L-MAC and B-MAC) for an energy-efficient jammer [20, 21].

Reactive countermeasures require reduced computational and energy cost compared to proactive countermeasures but, in case of deceptive jamming there is a great possibility of delayed sensing. Some of the reactive measures are – mapping of jammed regions [9], channel surfing for wireless devices to change their working channel and spatial retreat [16], wormhole-based anti-jamming techniques. Timing channels [22, 23] have been proposed as a means to allow the delivery of important alarm messages.

Mobile-agent-based solutions employ mobile agents (MAs) [24] to increase the survivability of WSNs without using any specialized hardware [25].

2.5 Detection of jamming in WSNs

Detection of jamming can be challenging because of the difficulty of differentiating a jamming scenario from a legitimate one. For example, if a jammer imitates regular network traffic, it can be difficult for a regular sensor node to detect that it has been jammed. Also, it is not feasible for the nodes to detect jamming continuously, since they need to minimize their power consumption.

MAC-layer and physical-layer jamming detection techniques have been rarely addressed. When an intruder places jamming nodes in a WSN, the nodes surrounding the jamming device are jammed (Fig. 2.3). To the nodes outside this region of jammed nodes, the whole region appears to suffer complete or intermittent failure and they may be unable to determine that this behavior results from a DoS attack. Xu et al. [7]

addresses the detection of jamming by combining several statistics such as packet delivery ratio (PDR), received signal strength indication (RSSI), and carrier sensing time of the sensor nodes to distinguish if the network is jammed or simply congested due to high amounts of traffic. By combining PDR values with corresponding RSSI values under various scenarios and setting appropriate threshold levels [7], a node can be determined to be jammed or unjammed. Also, comparison of carrier sensing time before sending out packets at a node can be used as a measure for determining whether there is a constant or deceptive jammer in the vicinity.

Detection of run-time radio interference among sensor nodes in WSNs is addressed in [19] and the interference detection results from adjacent nodes are used for designing real time collision-free TDMA (Time Division Multiple Access) protocols.

Heuristics are applied to detect jamming in [9], where channel utilization rate is used to determine if there is any DoS condition occurring in the network. A sensor node can decide that it has been jammed when its channel utility is below a certain threshold. The utility metric includes a variety of factors: repeated inability to access wireless channel, corrupted packets such as bad framing, checksum failures, protocol violations such as missing ACKs, and repeated collisions. By using these metrics in a sufficiently dense network, some nodes located close to the edge of the jamming signals will be able to determine if jamming is taking place. Also, a node may distinguish jamming from the failure of its neighbors by determining that constant energy, not lack of response, is hampering communication. These nodes can report the jamming to unaffected nodes outside the region, even if reception errors prevent the reporting nodes from receiving reliable acknowledgments. This is the technique proposed by Wood et al., in which they override the MAC-layer protocol to send a high priority, unacknowledged, broadcast message [9]. In this way, nodes may be able to send a few high-power, high-priority messages back to a base station to report the attack.

To report jamming, nodes need to cooperate to maximize the probability of successfully delivering such messages, which could mean switching to a prioritized transmission scheme that minimizes collisions. Nodes can also buffer high-priority messages indefinitely, hoping to relay them when a gap in the jamming occurs. In a large-scale deployment, an adversary is less likely to succeed at jamming the entire network, especially if sensor devices similar to the nodes in the network are used for jamming. As a scenario in Fig. 2.3, an appropriate response would be to call on the nodes surrounding the affected region to cooperatively map and report the DOS attack boundary to a base station [9].

2.6 Mapping

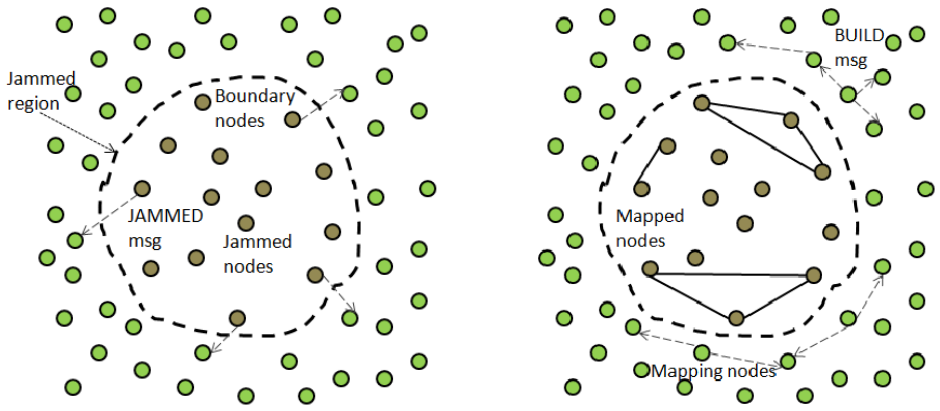


Figure 2.4. Nodes mapping jammed region by JAM protocol.

Wood et al. propose JAM as, a complete protocol for mapping the jammed area in a WSN after detection [9]. Mapping is mainly performed to increase network efficiency, because after the mapping is done, data is routed around the jammed regions to conserve the nodes' energy. The proposed mapping service can provide benefits such as:

- Creating a higher level abstraction than local congestion
- Feedback to routing and directory services
- Aid to power management strategies for nodes inside and around jammed regions
- Support for avoiding the region by network controlled vehicles, military assets, emergency personnel
- Reports to base-station for further jamming localization

According to this protocol, in the jammed nodes, the MAC protocol must provide a way to override carrier sense to indicate a clear channel and allow the nodes to broadcast a brief, high priority, unacknowledged message. By this technique, to initiate mapping, a node that discovers that it has been jammed (detection techniques discussed in Section 2.5) broadcasts **JAMMED** messages to its neighbors informing them that it has been jammed. Some of the neighboring nodes that are further away from the jamming effect but located near the boundary of the jammed region will be able to receive these messages and initiate the JAM protocol (Fig. 2.4). By this protocol, the *boundary* nodes map out the jammed area by exchanging messages among themselves regarding the jammed nodes.

JAM uses a decentralized method of mapping, in which the boundary nodes outside of a jammed area map the area by exchanging messages and building lists of jammed nodes. The boundary nodes receiving **JAMMED** messages from a jammed neighbor are called *mapping* members or nodes.

Mapping members start the mapping protocol by forming groups of jammed nodes. The jammed nodes near the boundary area are the *mapped* nodes and they are formed into groups by the mapping nodes based on their locations with respect to the mapping nodes. These groups are formed based on the compatibility of the jammed nodes measured by their orientation from the mapping node. There can be

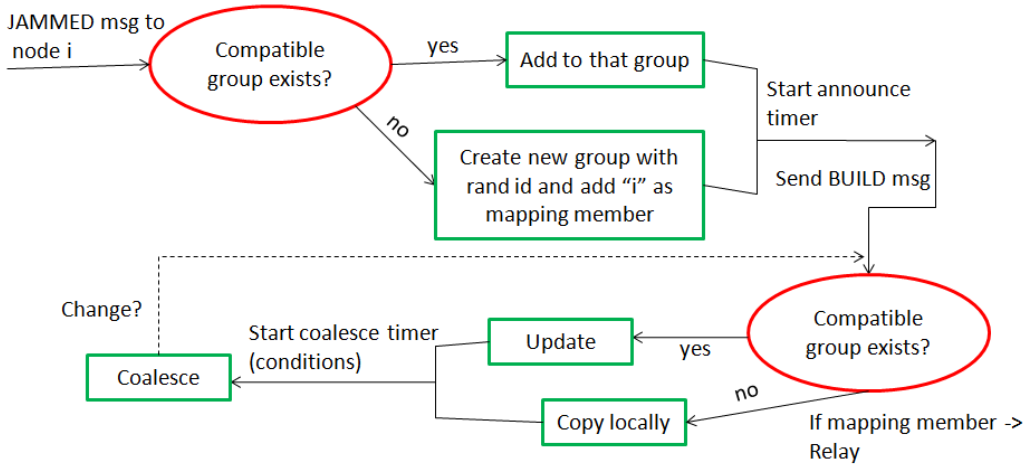


Figure 2.5. Mapping process in JAM.

multiple groups under one mapping node, and the mapping node will try to merge them into larger groups based on these relative locations. Once the groups are formed, each mapping member sets a timer and, after its expiration, the mapping member sends **BUILD** messages containing information of mapped groups to its neighbors. Thus, the group information is being exchanged by the mapping nodes, helping them to form a larger group of jammed nodes. If the neighboring node receiving build messages is also an active mapping node, then it will first try to merge the received groups with its local groups. If there are any changes, it will relay the new group information after waiting for a timer to expire.

Ideally, the mapping members will eventually synchronize their information with each other. Finally, when all the mapping nodes are done with sending build and replay messages, all the groups must have been coalesced into a single dominant group consisting of information of all the mapping nodes and mapped nodes. By this continual process of exchanging messages regarding jammed nodes, the mapping members will eventually map out the boundary of the jammed area. The flow of the mapping process is shown in Fig. 2.5

Drawback to JAM: In JAM all the mapping nodes in the boundary of the jammed region participate in mapping, therefore only those nodes know about the region that is being jammed. Thus, JAM applies a decentralized technique for mapping and does not take any reactive measures against jamming attacks. Also, for all the mapping nodes to have updated information on jammed regions, a high number of messages are being exchanged among the mapping nodes causing a very high message overhead to determine this perfect mapping. High message overhead creates higher power consumptions for the low-powered sensor nodes. Also, the time overhead is very high, since it takes longer time to achieve the accurate mapping of the jammed regions.

CHAPTER 3

FAST MAPPING OF JAMMED REGIONS

In this chapter, we present our protocol for mapping the jammed regions in a WSN. We propose a complete mapping service based on the idea that low-powered nodes should not be using excessive energy to perform the mapping and also the mapping needs to be performed fast. The key points of this design–

1. Application of central instead of local mapping of jammed regions, i.e. the responsibility of mapping the jammed regions is removed from the nodes in the network and performed by the BS.
2. A node reports to the BS as soon as it detects jamming among its neighbors.
3. Use of probabilistic selection of reporter nodes to achieve energy efficiency and to minimize overall message overhead.
4. We aim for a fast, approximate mapping instead of a slow, accurate one.

Fast mapping of jammed regions can be critical in a security monitoring scenario in which the BS can learn about and map out the jammed regions quickly and accurately so as to know where physical security is threatened and where it may be necessary to increase other security measures.

Before presenting a complete description of our protocol, we first discuss the network model and the intruder model we use for our proposed system in the following section. Then we give a detailed description our protocol with an example sequence to show how it operates.

3.1 Model

In this section we present the basic characteristics of the network and intruder models that we are going to use for our protocol.

3.1.1 Network model

For our system we consider a homogeneous network model in which all the nodes are stationary, location aware and also have roughly the same sensing capabilities. These sensor nodes possess limited power and utilize wireless channels to communicate with other nodes with in their signal range.

The sensor nodes close to a jamming device are unable to receive any messages from their neighbors since the channel is jammed and therefore unable to communicate with any of their neighbors. But, the nodes that are on the edge of a jammed region are able to send messages to their un-jammed neighbors outside the jamming range and send notification messages to them, once they detect jamming (discussed in section 2.5) among some of their neighboring nodes. This nodes are called the boundary nodes.

The base station (BS) is a distinguished component with much more computational power and communication resources than the sensor nodes in the network. The BS is also aware of network topology and the location of the sensor nodes in the network [26, 27].

During the deployment phase, flooding messages are sent by the BS, to construct a spanning tree rooted at the BS; applying breadth first search. All the nodes within the range of the BS, sets the BS as their parent and rebroadcast this message to their neighboring nodes. By this way each node has information on their predecessor nodes and minimum distance to the BS, which is applied for routing mechanism in the network [28, 29].

An example network is shown in Fig. 3.1(a).

3.1.2 Intruder model

In a traditional wireless communication system, a jammer launches jamming attacks on the physical and data link layers of the WSNs with the goal of preventing reception of communications at the receiver end using as little power as possible (section 2.2). In section 2.3 we discuss four jammer models – constant, deceptive, random and reactive presented by [7].

In this paper, we consider an intruder who can deploy jamming devices that act as constant jammers. The intruder can place these malicious nodes or jamming devices at any random loactions in the network for the purpose of creating a DOS or a path through the network.

The jamming devices have similar hardware capabilities as the sensor nodes in the network. As these devices act as constant jammers, they can disrupt any communication in the surrounding area by emitting continuous radio signals. They can be implemented using regular wireless devices continuously sending out random bits to the channel without following any MAC-layer protocol [7] and thus effectively able to block the legitimate traffic sources from using the channel.

The range of these jamming devices may vary, and their range is not known to the sensor nodes or the BS. A node located inside the range of a jamming device is unable to communicate with its neighbors.

For example, in Fig. 3.1(b), a scenario is presented where there are five jamming sensors in the network. The red nodes are the sensor nodes that are jammed and are unable to communicate with the unjammed nodes marked in green.

3.2 Mapping protocol

In this section, we present a detailed description of our protocol for mapping the jammed regions in a WSN using an example sequence (Fig. 3.1). The mapping is performed in few steps–

These steps are described in following sections.

3.2.1 Jamming detection and notification to the BS

A jamming device present in the network jams all the benign nodes with in its signal range. Since the nodes in the network are able to detect jamming, any node surrounding this jammed region with some of its neighbors being jammed nodes should be able to detect jamming among its neighbors. The nodes that are near the jammed region, as soon as they detects jamming, some of them send reports to the BS. The nodes that report to the BS about jamming are the *reporter* nodes. To keep the overall message overhead to minimal, not all the nodes that detect jamming report to the BS. If a node will be sending report to the BS depends on following two decisions –

- **Decision to become reporter:** When a node detects jamming among its neighbors, it decides with some probability P_{rep}, whether or not to become a reporter. This probability determined according to the density of the network and also depending on the required accuracy of mapping. Once jamming starts and an unjammed node finds jamming among its neighbor nodes, it starts making a list of these jammed nodes. In Fig. 3.1(c), the green nodes are the reporter nodes of the network , with $P_{rep} = 0.5$, and the red ones are the jammed nodes that are reported to the BS.
- **Decision to notify neighbors:** The farther a reporter node is located from the BS, the greater the number of messages required for sending jamming notifications

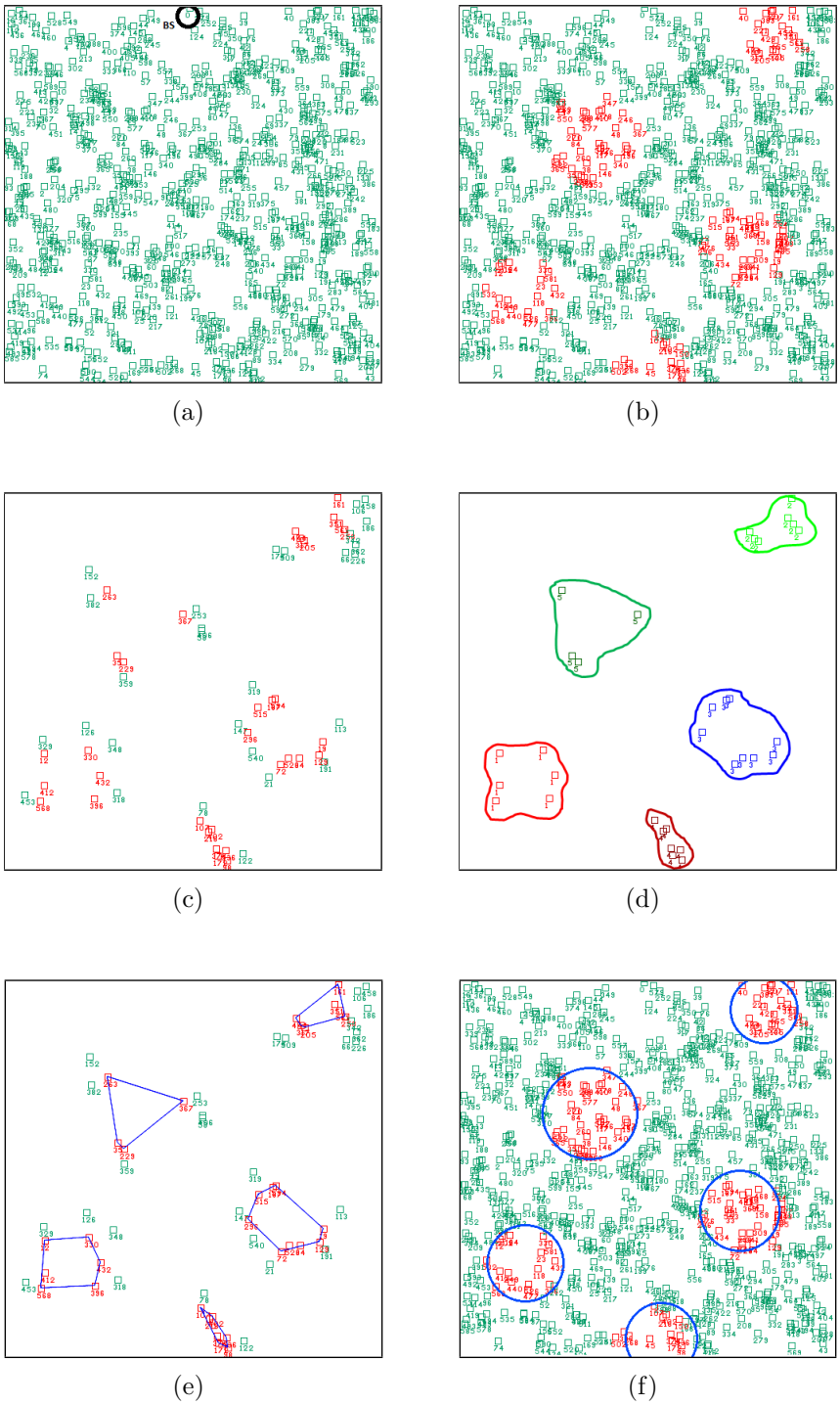


Figure 3.1. Mapping sequence. (a)WSN, (b)Jamming in WSN, (c)Reporter nodes, (d)Finding jammed regions, (e)Convex hull of the regions, (f)Mapped area.

to the BS. To reduce this overhead we use a simple scheme. The reporter node multiplies its distance to the BS with its number of neighbors and with P_{rep}. If this value is greater than a threshold (Th_{rep}) value, the node will send an alert to its neighbors. Otherwise it'll not send a notification message again to the BS as at least another node already sent a message from similar location of the network. For example, a reporter node i will alert its neighbors if Eq. 3.1 is fulfilled.

$$Dist_i \times Neighbors_i \times P_{rep} \geq Th_{rep} \tag{3.1}$$

A neighboring node that gets an alert from a neighboring reporter node will not send any notification messages to the BS, even if it has selected itself as a reporter node.

The value Th_{rep} is determined based on the average distance to the BS and average number of neighbors for the sensor nodes in the network. There is a trade off between the cost of sending notification from a reporter node to the BS and sending notifications to the reporter's neighbors. The choice is made such that it reduces the message overhead.

3.2.2 Locating the jammed regions of the network

Once the BS starts receiving notifications of jamming from the un-jammed nodes located at the various parts of the network, it starts building its own list of known jammed nodes in the network. Since the BS may get notifications of jamming from different parts of the network, it applies a well-known clustering algorithm to decide on the number and location of the jammed regions in the network. For this reason, we used the k-means [30, 31], which is a partition-based clustering algorithm for this purpose. In this clustering method each cluster is represented by the mean value of the objects in the cluster. The reason for choosing k-means are-

1. This algorithm quite intuitive and one of the simplest of the unsupervised learning algorithms,
2. It is very fast and
3. It is suitable for finding spherical shaped clusters in datasets of limited size

A brief description k-means algorithm is presented in following section -

3.2.2.1 K-means clustering

The basic idea of k-Means clustering is [32] – given a database of n data items, k partitions are created where $k \leq n$. Thus the partitioning method classifies the data items into k clusters satisfying the requirements: (1) each group must contain at least one object, and (2) each object must belong to exactly one group.

Once the clusters of items with the same target category are identified, and predictions for new data items are made by assuming they are of the same type as the nearest cluster center.

Two important issues in k-Means clustering are:

1. Determining the optimal number (k) of clusters to create.
2. Determining the initial centers for each cluster.

The basic k-means clustering algorithm is shown in algorithm 1.

To measure the similarity between an object and a centroid of cluster usually some distance function is used, e.g. Euclidean distance measure between the points.

The performance of the algorithm is measured by SSE, that stands for sum of squared error, calculated by the formula -

$$SSE = \sum_{i=1}^{k} \sum_{x \in C_i} dist^2(m_i, x) \tag{3.2}$$

Here, x is a data point in cluster C_i and m_i is the centroid of cluster C_i and $dist()$ function gives the distance measure between any 2 points.

SSE can be used as a parameter for stop condition of the clustering algorithm and also to determine the quality of the result of clustering. For a fixed k, smaller value of SSE generally stands for better clustering result.

3.2.2.2 Locating the jammed regions by BS

In our proposed system, list of jammed nodes built by BS is input to the k-means algorithm. The geographical position of the nodes is used as data points and the euclidean distance among these nodes is used as the distance measure. The output is the clusters which represents the jammed regions in the network.

In section 3.2.2.1 we have addressed the issue that to perform k-means clustering, one needs to know the value of k or the number of possible clusters for the given dataset. Here, k is the number of jammed regions in the network and is not known prior to the BS. If there is no manual assistance available, to determine the probable value of k, the BS needs to decide on the best value of k, based on the information it received from the nodes in the network.

Finding the number of jammed regions by BS:

Since, the quality of the clustering by k-means greatly depends on the selection of initial centroids, the BS runs the algorithm on a maximum value of k for certain number of times, each time using k random points as initial centroids. So, the BS starts from a high value of k and picks k random nodes among its list of jammed nodes to serve as k initial centroids. BS then runs the k-means algorithm a fixed number of times for this value of k. Among all the rounds BS picks the best result with lowest SSE. For the next round, it merges 2 of the closest clusters from the previous best result and generating a new centroid from from the mean of the centroids for these 2

clusters. The BS then uses this new centroid and the other $k-2$ centroids as initial centroids for the next round of k-means for $k-1$ clusters. Except for the first round with the maximum value of k, BS runs the k-means algorithm only once for each decrement of k, until $k = 1$.

The steps of finding the jammed regions are shown in Fig. 3.2. In Fig. 3.2(a) the jammed nodes that are being reported to BS are displayed in red. BS performs its region finding procedure on these nodes. BS starts the process with $k_{max} = 10$ (Fig. 3.2(b)). The maximum number of time $times_{max}$ to run k-means is set to 20, for this value of k_{max} and the best clustering result is shown in the figure (Fig. 3.2(b)). In each of the next rounds, 2 of the closest cluster centroids are merged from the previous round and used as initial centroids along with all the other centroids generated from the previous round. E.g. in Fig. 3.2(d) and Fig. 3.2(e) it can be seen that the centroids for cluster number 1 and 4 of Fig. 3.2(d) are merged and generated a single cluster 1 in Fig. 3.2(e), while other centroids for $k = 8$ are used as initial centroids for $k = 7$. The clustering results for $k = 9, 8, \cdots, 3$ are are shown in the Fig. 3.2.

After generating k_{max} number of clusters by the process described above, the BS decides on the optimal value of k from the results. To do this, BS compares the improvements of SSE for each of the clustering results from $k = 1$ to k_{max} as improvement Imp_k –

$$Imp_k = (SSE_{k-1} - SSE_k)/SSE_{k-1} \tag{3.3}$$

If Imp_k is negative for a certain k, clustering results for that k is discarded. This is because of the intuition that with the increase of k, SSE should always decrease.

From the remaining clustering results, starting from the lower value of k the BS checks the nodes belonging to the cluster corresponding for that value of k and accepts the clustering result if all the nodes are within 2 standard deviations from

the mean of the cluster. If any of the nodes is more than two standard deviations away from the mean, the cluster is discarded. Otherwise, this k is decided to be the number of jammed regions in the network.

E.g. from the Fig 3.2, the BS starts with Imp_2 for k =1 to 2, Imp_3 for k =2 to 3 and so on. From these values the clusters that have improved SSE by decrementing the value of k are selected by the BS. Then starting from $k = 1$ to k_{max}, if the nodes in each of the clusters for that k are within 2 standard deviation from the mean of that cluster, the BS selects that k as the probable number of jammed regions in the network. The clustering result shown in Fig 3.2(g), $k = 5$ is selected as the best clustering result.

When the value of k that is decided by BS is less than or greater than the original number of jammed regions in the network, following situations occurs -

- $k <$ number of jammed regions: If k is less than the number of jammed regions in the network, multiple regions can be grouped to a single region. So, more un-jammed nodes will be inside a mapped region, causing more FN.
- $k >$ number of jammed regions: If k is greater than the number of jammed regions in the network, a single region can be divided in to multiple regions. This will create many small regions in place of a single one, thus leaving out many jammed nodes out side the mapped areas and causing more FP.

3.2.3 Jammed region mapping

BS first decides on the number and location of the jammed regions present in the network. After the locating the jammed regions, BS maps the jammed area for each of the regions. For mapping the regions, BS works as a classifier for two-class prediction problem, where the outcomes are labeled as positive (p) or negative (n) class. The jammed nodes are labeled as positive and the un-jammed nodes are labeled as negative. If a node inside a mapped region is actually jammed then it is labeled as true positive (TP); however if the node inside the mapped region is un-jammed, then the outcome is a false positive (FP). Conversely, a true negative (TN) can occur when an un-jammed node falls outside the mapped regions and false negative (FN) happens when a jammed node falls outside the mapped regions. Also, the *known positive* (TP/FP) nodes to the BS are the jammed nodes that were reported to BS by the reporter nodes and the *known negative* (TN/FN) nodes to the BS are these reporter nodes.

The mapping is performed based on the fact that jamming devices (using an omni directional antenna) usually have circular signal range creating spherical shaped jamming regions in the network. The output of the mapping protocol is shown in Fig. 3.1(f) which shows the original network with the mapping by the BS.

To map the region for each of the jammed areas found by BS, it first determines the center for that particular region, then fits the jammed nodes in a circular region and finally moves and increase the size of this region to increase the number of known TP and while keeping the number of known FP to minimum.

The steps are as follows-

- **Determining the center for the region:** The BS first determines the center (C_j) for a region. To achieve better accuracy in mapping, the center of a cluster is determined by following way–

- If the cluster size is less than 3, the center is the mean of the points in the cluster.
- Else the BS finds the convex hull of the cluster and the mean of the vertices of this hull is used as the center for that region.

The reason for applying convex hull [33] is to minimize the effect of the center (C_j) being biased to the side of the region where the node density is higher. We used Graham scan algorithm [34] to find the convex hull. Fig. 3.3(a) shows the convex hulls for the jammed regions found by the BS for the mapping sequence of Fig. 3.1 with the corresponding centers for each of the regions.

- **Mapping the region:** After finding the centroid (C_j) of the jammed nodes belonging to a region, BS takes the largest distance between any two jammed nodes of this region as the diameter of a circle centered at C_j and finds the number of known jammed nodes (TP), reporter nodes (known FP) that are inside this circle. It also calculates the center (C_i) of the known FP nodes (Fig. 3.4).
- **Improvement in Mapping:** To improve the mapping, BS moves C_j one step (away from C_i) at a time, alternating between vertical and horizontal axes. It continues to move the circle until either any known jammed node goes out of the circular region or if the number of known FP increases. If new TP nodes are added to the region, the BS increases the diameter of the circle by a factor to consume more jammed nodes. Fig. 3.4 shows the final area mapped by the BS with the new centroid for the jammed nodes located at (C'_j).

  Fig. 3.4 shows the region mapped by the BS based on the information collected on the jammed nodes from the reporter nodes. Here, the orange nodes are true positives (TP), the green nodes are false positives (FP) and the red nodes are false negatives (FN) among the nodes found as jammed by the BS. The TP nodes are the jammed nodes that are among the nodes considered as jammed by the BS. On the

other hand, FP nodes are the unjammed nodes that are considered as jammed by the BS. FN nodes are the jammed nodes that the BS has not mapped as jammed. In case of jamming regions that are asymmetric or non-circular shaped, the BS use the result of previously calculated convex hull to map the region.

Algorithm 1: Basic k-means clustering

Input:

$I = \{i_1, \cdots, i_n\}$, *(Instances to be clustered)*

k *(number of clusters)*

Output:

$C = \{c_1, \cdots, c_k\}$, *(cluster centroids)*

$m : I \to C$ *(cluster membership)*

Set C to initial value *(random points as centroids)*;

foreach $i \in I$ **do**

  $m(i) = \arg\min_{j \in \{1 \cdots k\}} distance(i, c_j)$;

end

while m *has changed* **do**

  foreach $j \in \{1 \cdots k\}$ **do**

    Compute c_j as the centroid for the cluster of $\{i | m(i) = j\}$

  end

  foreach $i \in I$ **do**

    $m(i) = \arg\min_{j \in \{1 \cdots k\}} distance(i, c_j)$;

  end

end

return C;

K-means for localizing the jammed regions at BS

Input:

$I = \{i_1, \cdots, i_n\}$, *(n jammed nodes to be clustered)*

k_{max}, *(maximum number of clusters)*

$times_{max}$ *(maximum number of times to run k-means for* $k = k_{max}$*)*

Output:

$C = \{C_1, \cdots, C_{k_{max}}\}$, where $C_k = \{c_1, \cdots, c_k\}$ and $k \in \{1, \cdots, k_{max}\}$

(set of cluster centroids)

$C_{k_{max}} \leftarrow kmeans(I, k_{max})$;

foreach $i \in \{2, \cdots, times_{max}\}$ **do**

  $C_i \leftarrow kmeans(I, k_{max})$;

  if $SSE_i < SSE_{i-1}$ **then**

    $C_{k_{max}} \leftarrow C_i$;

  end

end

for $i = k_{max} - 1$ **to** 2 **do**

  Compute mean of the 2 closest points in C_{i+1} to generate C'_i;

  $C_i \leftarrow kmeans(I, i)$, with C'_i as initial centroids;

end

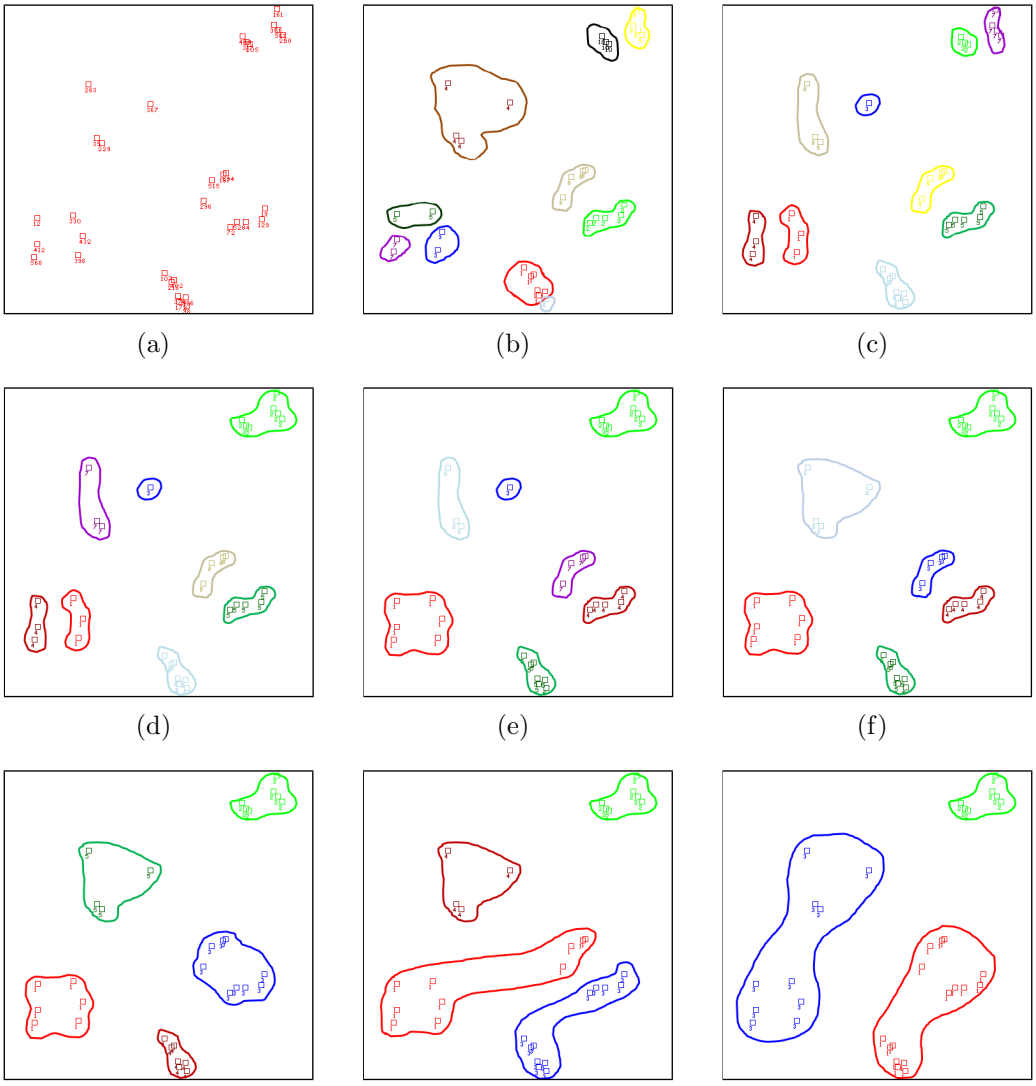
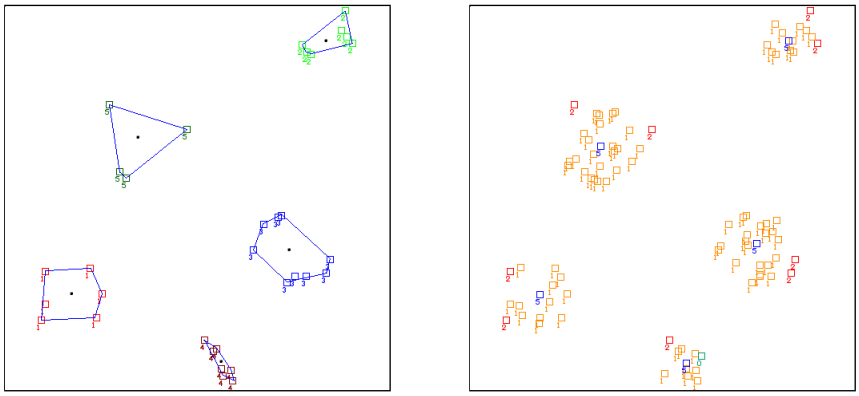
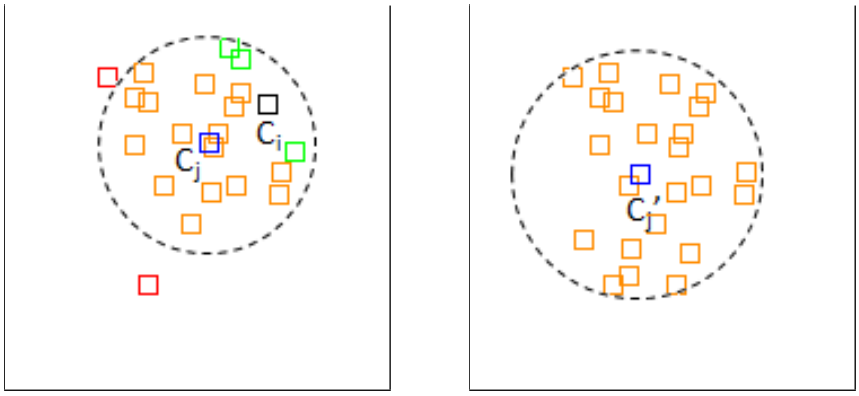


Figure 3.2. Clustering sequence for the mapping of Fig. 3.1. (a)Jammed nodes, (b)$k = 10$ (k_{max}), (c)$k = 9$, (d)$k = 8$, (e)$k = 7$, (f)$k = 6$, (g)$k = 5$ (selected), (h)$k = 4$, (i)$k = 3$.



(a) (b)

Figure 3.3. Finding the center and Area mapping. (a)Convex hull for the regions found in Fig. 3.2(g), (b)Final area mapping.



(a) (b) Final mapping

Figure 3.4. Jammed region mapping by the BS. (a)Initial mapping, (b)Final mapping.

CHAPTER 4

SIMULATION

We built a simulator to evaluate our proposed system. In this chapter we present the experiments we have done by using our simulator. We use the results to evaluate the performance of our system and to compare our proposed system with JAM [9]. Comparison is done based on two metrics: 1) time to perform the mapping of all the jammed regions present in the network; 2) amount of messages needed to be exchanged among the sensor nodes to do the mapping.

Also to evaluate the performance of our system we measure the performance of the mapping done by the BS in terms of true positives, false positives and false negatives (incrementing the node selection probability leads to minimizing the number of false negatives and false positives).

In the following sections we describe the experimental setup, methods for comparison with JAM and then measure the performance of the system.

4.1 Experimental setup

We built the simulator in C++ to simulate a border region of dimension 200×200 units. In this simulator we considered a WSN which is a connected graph of 500 – 1000 nodes. These nodes are randomly deployed in this area with one base node at the upper horizontal border of the network serving as the BS (Fig. 3.1(a)) and to whom all the other nodes report. The neighbor discovery and setting up of the path to the BS for each of the sensor node is done during the deployment phase of the network. The nodes are of fixed signal radii (10 – 20 units) and have 7 – 17 neighbors

Table 4.1. Basic setup for the simulation environment

Sensor node signal range	15 units
Jammer signal range	18 – 26 units
Average number of jammed regions	4.33

Table 4.2. Results from different size of networks

Size of Network	600	800	1000
Average number of neighbors	9.88	13.23	16.53
Total number of jammed nodes	89.48	111.34	135.5

on average. The neighbor discovery and setting up of the path to the BS for each of the sensor node is done during the deployment phase of the network.

During our simulation of events an intruder may place jamming devices randomly at any location of the network, these devices have higher signal radii (17 – 27 units) than the sensor nodes and this range can be different for the individual jamming devices in the network. When any of the node in the network is jammed the neighboring nodes are notified by the detection process in the network. When a working node finds out jamming among its neighbors it makes the decision of becoming a reporter node based on the probability P_{rep}.

The network parameters are shown in Table. 4.1 and Table. 4.2.

4.1.1 Comparison with JAM

We did couple of experiments on our proposed system. The first experiment involves comparison of performance between JAM [9] and our proposed system. To do this we simulated both JAM and our system in the exact same condition of the network.

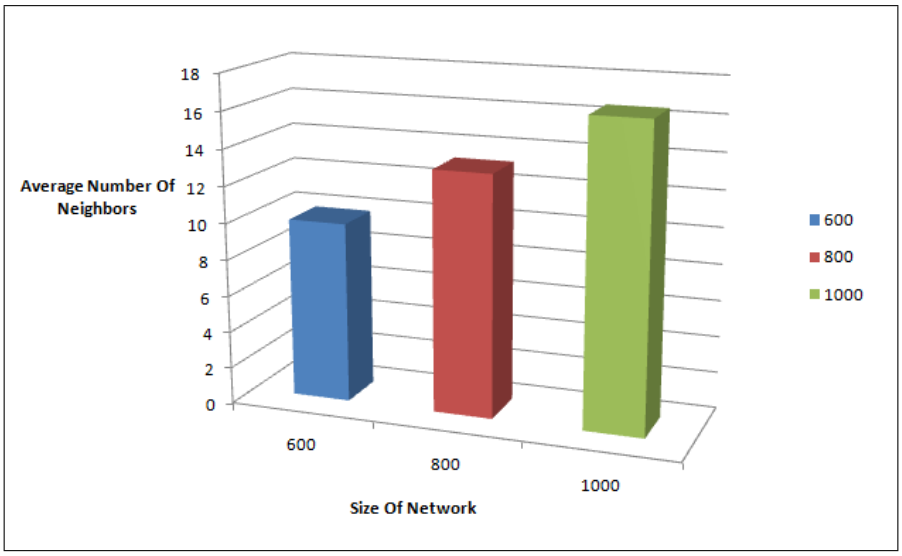


Figure 4.1. Average number of neighbors for different sizes of network.

Time to Map: We measured the time to map all the jammed regions in the network by the BS.

For JAM, the time to get the mapping is measured by the number of coalition of jammed groups that occur at a mapping node. This number determines the number of the rounds of build messages that are going to be sent by the mapping nodes before the final mapping result is being sent to BS. By this protocol after the mapping is done, ideally there should be one dominant group of mapped (jammed) members and only one mapping node send this information to BS.

For our system total time to map is calculated by the number of alert messages a reporter node sends to its neighbors, which determines the time it takes for BS to receive all the notifications of jammed nodes in the network.

Message overhead: To achieve the final goal of having the BS up-to-date with all the jammed regions in the network; during simulation, after mapping is done according to JAM, exactly one node (ideally the creator of the dominant group) from each jammed region sends message to the BS containing information on the related

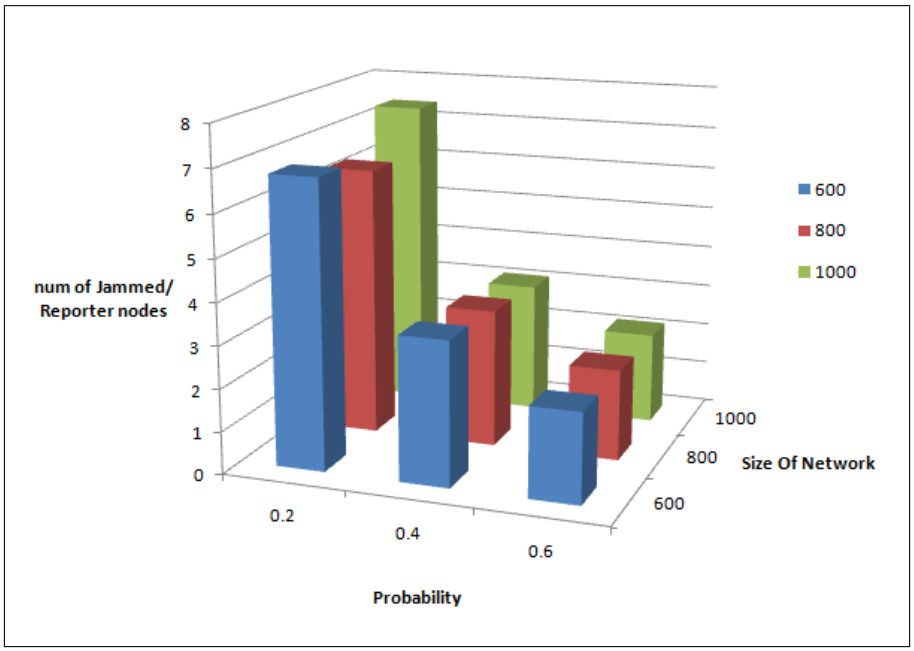


Figure 4.2. Ratio of number of jammed vs. reporter nodes for different sizes of network and probability.

mapped nodes. In our protocol the overhead is the sum of the number of messages required to be sent by all the reporter nodes to notify the BS.

For the first experiment, three different densities of network are considered with 600, 800 and 1000 nodes. The average number of jammed regions and average number of jammed nodes in the network is shown in Table. 4.2. For each of these setups, the simulation results are calculated by running JAM, our system with $P_{rep} = 0.2$, $P_{rep} = 0.4$ and with $P_{rep} = 0.6$. And for each of these cases the simulation has been run 100 times and results are calculated by the arithmetic mean of these simulations.

The box-whisker graph in the Fig. 4.3 shows the time to get the ultimate mapping result for JAM and our system for three different probabilities and for three different densities of the network. In the bar graph in Fig. 4.4, comparison of overhead according to the number of messages are shown for the similar setup.

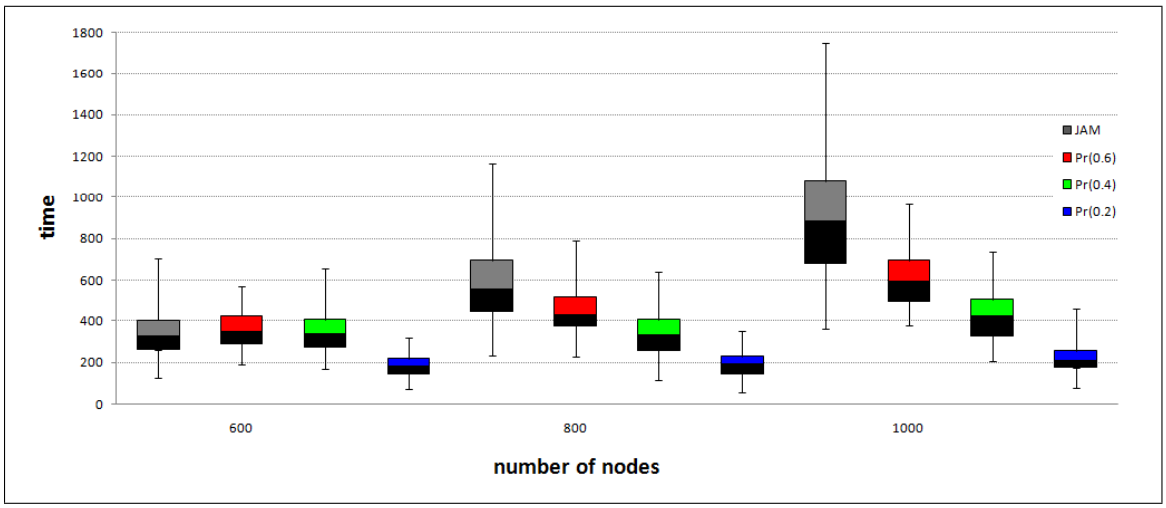


Figure 4.3. Comparison with JAM on time to map jammed regions.

4.1.2 Performance of the system

In the second experiment we evaluate the performance of our system. For this we took a measure of performance by incrementing the probability of a node to become reporter, by varying P_{rep} from 0.2 to 0.6. The performance is presented in terms of precision and recall for the system. These two values are computed by the of number of true positive, false positive and false negative nodes the BS can identify while performing the mapping of the jammed nodes. This experiment is done with three different densities of network of – 600, 800 and 1000 nodes.

The BS calculates the precision and recall values for the jammed nodes found in the network with the increasing probability P_{rep}. Here, precision (Eq. 4.1) gives the probability of finding real jammed nodes classified as jammed by the BS to all then nodes classified as jammed by the BS. On the other hand recall (Eq. 4.2) specifies the probability of original jammed nodes mapped by the BS among all the jammed nodes present in the network. Here, TP, FP and FN stands for true positive, false positive and false negative values for the jammed nodes identified by the BS.

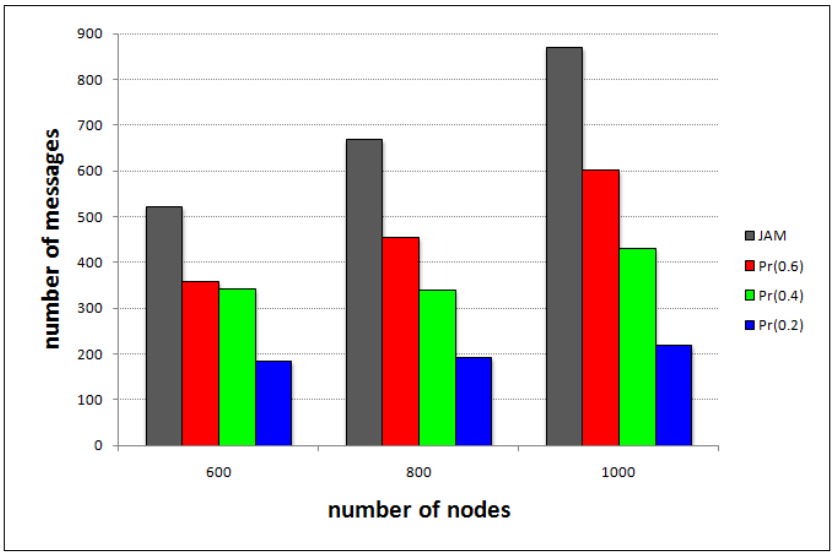


Figure 4.4. Comparison with JAM on message overhead.

$$Precision = \frac{TP}{TP + FP} \tag{4.1}$$

$$Recall = \frac{TP}{TP + FN} \tag{4.2}$$

Results from the second set of experiments are shown in Table 4.3. The recall improves better with increasing probability as when the number of reporter nodes are higher. This is because the BS has more information about the jammed nodes which helps the BS to include more jammed nodes in each region.

Table 4.3. Precision and recall

	Probability	Precision	Recall
600 nodes	0.2	0.77	0.51
	0.4	0.81	0.73
	0.6	0.82	0.75
800 nodes	0.2	0.79	0.56
	0.4	0.83	0.73
	0.6	0.84	0.80
1000 nodes	0.2	0.83	0.59
	0.4	0.84	0.75
	0.6	0.87	0.82

CHAPTER 5

EXPERIMENT

In this chapter we describe our experiments on real sensor motes to test the performance of jammed region mapping technique. We use Crossbow's TelosB motes (TPR2400) that are generally platform for low-power research development for wireless sensor network experimentations and TinyOS 2.0 for programming.

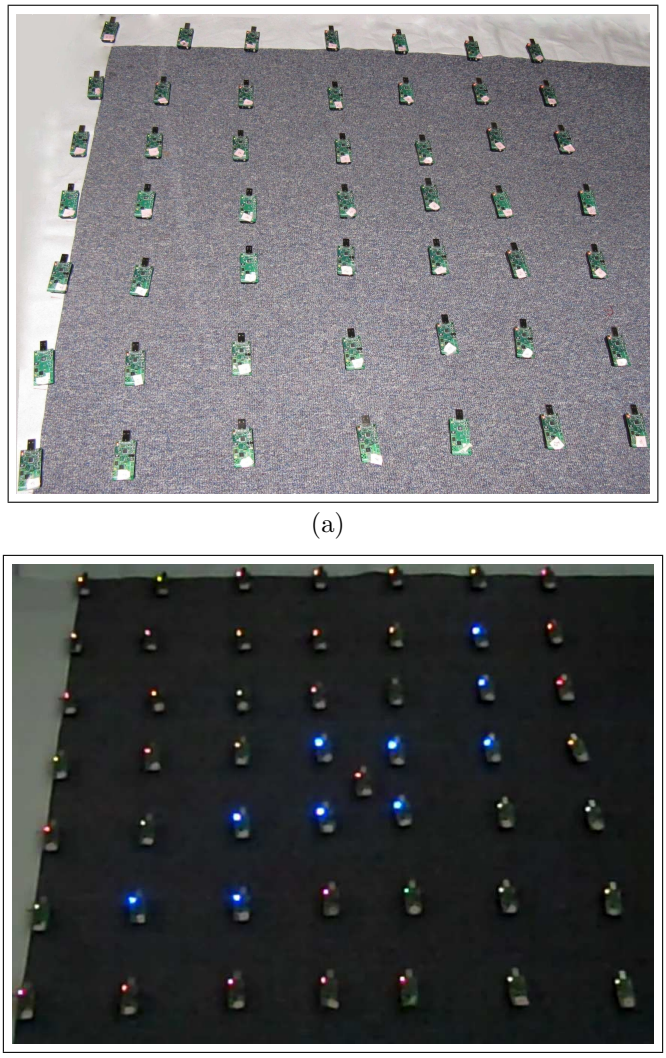
The motes are of following specification [13]:

- IEEE 802.15.4 compliant
  - 802.15.4 is an IEEE standard that defines a MAC and PHY layer targeted to low power wireless sensor networks. TelosB motes uses Chipcon CC2420 RF transceivers that supports the 802.15.4 standard and used in Telos platform. TinyOS uses CC2420 radio stack.
- 250 kbps data rate radio
- TI MSP430 microcontroller with 10kB RAM
- Runs TinyOS 1.1.10 or higher
- Data collection and programming via USB interface
- Default range of RF transceiver at outdoor is 75-100 m and indoor is 20-30 m.

5.1 Setup

For our experiments we use a total of 50 motes, where 49 of them are used for the network, and one is programmed as jamming device. For setting up the network, grid topology of 7×7 motes is applied (Fig. 5.1). The sensors were deployed by placing the adjacent sensors $8''$ apart. The network information is shown in Table 5.1.

The network area is a square of size $48'' \times 48''$. Since the default radio range of the sensor motes are quite high, We set the parameter DCC2420-DEF-RFPOWER of the motes to 1, 2 and 3 and found $9''$, $20''$ and $213''$ as corresponding ranges. For our indoor experiments, we used the lowest range (DCC2420-DEF-RFPOWER = 1).



(a)

(b)

Figure 5.1. (a)Wireless network of size 7×7 using grid topology, (b)Jamming.

Table 5.1. Basic setup of the network

Size of Network	49
Average number of neighbors	5.22
Total number of jammed nodes	10
Sensor node signal range	$9''$
Jammer signal range	$9''$
Average number of jammed regions	1

5.1.1 Jammer node

To make a regular sensor node to work as the jammer, we bypass the MAC protocol for that mote by disabling channel sensing and radio back off operation. So, this jammer mote can send continuous signals and jam reception of all the motes those are within its transmission range. Fig. 5.1 shows jamming in the network.

5.1.2 Neighbor setup

Each sensor node in the network sends a beacon packet to its neighbors each time interval ($0.4s$). We determine the neighbors of a node by setting a threshold for the number of messages it receives from another node at unit time.

5.1.3 Detection

The beacon packets sent by the nodes to its neighbors are used to detect jamming in the network. When the jammer node is on, the jammed sensors fails to receive the regular beacon packets from its neighbors and will be able to detect that its being jammed. The boundary nodes of the jammed region notify their un-jammed neighbors that they are jammed and the un-jammed nodes starts the mapping protocol.

5.2 Simulation

We give this setup of network with each node and its corresponding neighbors and also the list of jammed nodes from the experiment, as input to our simulator (section 4) and see how our region mapping protocol performs for actual jammed regions. Since the jammed regions that are generated are of irregular sizes, we used convex-hull for final mapping, instead of the circular shapes. For each of the experiment we run our simulation of region mapping for 1000 times to measure the performance.

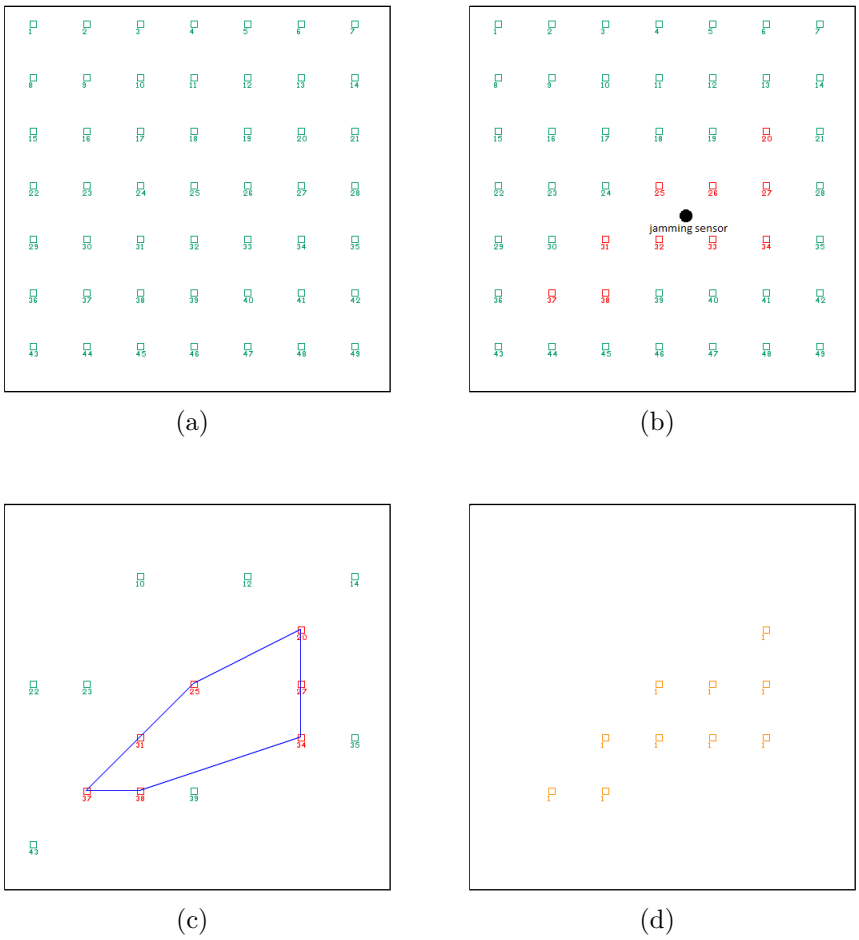


Figure 5.2. Simulation results from real experiment. (a)WSN, (b)Jamming, (c)Reporter, jammed nodes and convex hull of the detected region, (d)Mapped area.

Table 5.2. Results from simulation of the network

Pr. of selection	Precision	Recall
0.4	0.50	0.25
0.6	0.81	0.54
0.8	0.94	0.78

A sequence of simulation images are shown in Fig. 5.2. And the performance in terms of precision and recall is presented in Table. 5.2.

CHAPTER 6

CONCLUSION

Jamming is one of the possible attacks at the physical layer of a WSN, performed by the radio frequency noise to disrupt communication in the network. Jamming can easily cause a denial-of-service attack, preventing the network from performing its regular monitoring functions. Also, a powerful attacker can jam random parts in a network and create a path for himself to come back and forth inside the network and thus creating critical security breach to the WSN. So, it is important to map out the jammed regions so that this information can be used in the network for routing or power management. Thus jammed region mapping in the network helps dealing with jamming and to take effective measures against it. In this paper we proposed an efficient mapping protocol to map the jammed region in a network under such attacks for different application scenarios. We applied the idea of base station computing the approximate mapping of the jammed regions in the network and relieving the sensor nodes from draining out of battery power. This mapping results can be improved by more nodes sending jamming notification messages to base station, which can be a trade off between performance of mapping versus the network overhead and can be achieved by increasing the chances of a nodes surrounding the jammed region being selected as reporter nodes. Our simulation results demonstrate that this system requires less interactions of the sensor nodes and thus have less overhead by the overall network and also provides faster mapping of the jammed regions.

We developed our intruder model considering jamming devices randomly placed in the network with varying signal range creating circular interference around it. In

future, we will be working on jamming regions introduced by jamming devices of more asymmetric and irregular signal range. We will be working on improved k-means algorithm [35, 36], so that we can have better selection of the initial centroids and improved clustering results in the presence of clusters of irregular size, density and shape.

REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, pp. 393–422, 2002.

[2] Z. Haas, J. Deng, B. Liang, P. Papadimitratos, and S. Sajama, "Wireless ad hoc networks," 2002.

[3] T. He, S. Krishnamurthy, J. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, and L. Gu, "Energy-efficient surveillance system using wireless sensor networks," in *Mobisys*. ACM Press, 2004, pp. 270–283.

[4] S. Han, E. Chang, L. Gao, and T. Dillon, *Taxonomy of Attacks on Wireless Sensor Networks*. Springer London, 2006, pp. 97–105.

[5] A. Wood and J. Stankovic, "Denial of service in sensor networks," *IEEE Computer*, vol. 35, no. 10, pp. 54–62, 2002.

[6] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: attacks and countermeasures," in *First IEEE International Workshop on Sensor Network*, 2003, pp. 113–127.

[7] Y. Z. W. Xu, W. Trappe and T. Wood, "The feasibility of launching and detecting jamming attacks in wireless networks," in *6th ACM international symposium on mobile ad hoc networking and computing (MobiHoc'05)*, 2005, pp. 46–57.

[8] I. K. M. Li and R. Poovendran, "Optimal jamming attacks and network defense policies in wireless sensor networks," in *26th IEEE International Conference on Computer Communications (INFOCOM'07)*, 2007.

[9] A. Wood, J. Stankovic, and S. Son, "JAM: A Jammed-Area Mapping Service for Sensor Networks," in *24th IEEE Real-Time Systems Symposium*, 2003, pp. 286–97.

[10] "Sensor node," http://en.wikipedia.org/wiki/Sensor_node.

[11] D. L. Adamy and D. Adamy, *EW 102: A Second Course in Electronic Warfare*. Norwood, MA: Artech House, Inc., 2004.

[12] "Ieee 802.15 wpan task group 4 (tg4)," http://www.ieee802.org/15/pub/TG4.html.

[13] "Telosb," http://www.willow.co.uk/TelosB_Datasheet.pdf.

[14] Y. Zhang and P. Kitsos, *Security in RFID and Sensor Networks*. Auerbach Publications, 2009.

[15] W. Xu, K. Ma, W. Trappe, and Y. Zhang, "Jamming sensor networks: Attack and defense strategies," *IEEE Network*, vol. 20, no. 3, pp. 41–47, 2006.

[16] W. Xu, T. Wood, and Y. Zhang, "Channel surfing and spatial retreats: Defenses against wireless denial of service," in *Proceedings of the 2004 ACM workshop on Wireless security, 2004*. ACM Press, 2004, pp. 80–89.

[17] "Ultra-wideband," http://en.wikipedia.org/wiki/Ultra-wideband.

[18] A. Spyropoulos and C. Raghavendra, "Energy efficient communications in ad hoc networks using directional antennas," in *IEEE Infocom*, 2002, pp. 220–228.

[19] G. Zhou, T. He, J. Stankovic, and T. Abdelzaher, "RID: Radio interference detection in wireless sensor networks," in *INFOCOM*, 2005.

[20] Y. Law, L.Hoesel, J. Doumen, P. Hartel, and P. Havinga, "Energy-efficient link-layer jamming attacks against wireless sensor network mac protocols," in *Wireless Sensor Network MAC Protocols SANS05*, 2005.

[21] A.Wood, J.Stankovic, and G. Zhou, "DEEJAM: Defeating Energy-Efficient Jamming in IEEE 802.15.4-based Wireless Networks," in *Sensor, Mesh and Ad Hoc Communications and Networks (SECON'07)*, 2007, pp. 60–69.

[22] M. Cagalj, S. Capkun, and J. Hubaux, "Wormhole-based antijamming techniques in sensor networks," *IEEE Transactions on Mobile Computing*, vol. 6, no. 1, pp. 100–114, 2007.

[23] W. Xu, W. Trappe, and Y. Zhang, "Anti-jamming timing channels for wireless networks," in *WISEC*, 2008, pp. 203–213.

[24] A. K. V. Pham, "Mobile software agents: an overview," in *IEEE Communications Magazine*, vol. 36, no. 7, 1998, pp. 26–37.

[25] A. Mpitziopoulos, D. Gavalas, C. Konstantopoulos, and G. Pantziou, "Jaid: An algorithm for data fusion and jamming avoidance on distributed sensor networks," *Pervasive and Mobile Computing*, vol. 5, no. 2, pp. 135–147, 2009.

[26] C. Savarese and J. Rabaey, "Locationing in distributed ad-hoc wireless sensor networks," in *ICASSP*, 2001, pp. 2037–2040.

[27] T. He, C. Huang, B. Blum, J. Stankovic, and T. Abdelzaher, "Range-free localization schemes for large scale sensor networks," 2003, pp. 81–95.

[28] E. Wittmann, *Routing in Wireless Sensor Networks*. Saarbrücken, Germany, Germany: VDM Verlag, 2008.

[29] C. Schurgers and M. Srivastava, "Energy efficient routing in wireless sensor networks," in *The MILCOM Proceedings on Communications for Network-Centric Operations: Creating the Information Force*, 2001, pp. 357–361.

[30] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[31] J. MacQueen, "Some methods for classification and analysis of multivariate observations," *5th Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297, 1967.

[32] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, 2nd ed., ser. The Morgan Kaufmann series in data management systems. Morgan Kaufmann, 2006.

[33] "Convex hull," http://en.wikipedia.org/wiki/Convex_hull.

[34] "Graham scan," http://en.wikipedia.org/wiki/Graham_scan.

[35] T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, and R. S. ands A. Wu, "An efficient k-means clustering algorithm: Analysis and implementation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 881–892, 2002.

[36] P. Bradley and U. Fayyad, "Refining initial points for k-means clustering," in *15 International Conference on Machine Learning (ICML '98)*. Morgan kaufmann, 1998, pp. 91–99.

BIOGRAPHICAL STATEMENT

Nabila Rahman completed her Bachelors in Computer Science and Engineering from Bangladesh University of Engineering and Technology (BUET) in 2007. She worked as software engineer in several well-known software outsourcing companies in Bangladesh since 2006. She started her graduate studies to pursue Masters in Computer Science at the University of Texas at Arlington in fall 2008. She joined ISec Lab (Information Security Lab) at UTA and her research involves security, specially jamming attacks in wireless sensor networks.