The Pennsylvania State University

The Graduate School

Computer Science and Engineering

ENHANCING SVATS WITH INTERACTIVITY, SECURITY AND RELIABILITY

A Thesis in

Computer Science and Engineering

by

Jyoti Bala

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

August 2009

The thesis of Jyoti Bala was reviewed and approved* by the following:

Guohong Cao
Professor, Department of Computer Science and Engineering
Thesis Co-Advisor

Sencun Zhu
Assistant Professor, Department of Computer Science and Engineering
Thesis Co-Advisor

Mahmut Kademir
Director of Graduate Affairs
Associate Professor, Department of Computer Science and Engineering

*Signatures are on file in the Graduate School

ABSTRACT

With reduced cost and increased reliability of the wireless motes, wireless sensor networks (WSNs) are finding applications in a lot of varied fields such as industrial process monitoring and control, machine health monitoring, environment and habitat monitoring, healthcare applications, home automation, and traffic control. One such system for detecting and tracking stolen vehicles is the “Secure Vehicle Antitheft System (SVATS)”. It overcomes the shortcomings of traditional anti-theft systems and aims at making theft detection more reliable and giving better response times.

This research work extends the work done on SVATS. It builds a security framework for SVATS, analyses the various security threats that a SVATS kind of system might be faced with. Further, it builds a framework for integrating the SVATS sensor network with the GSM network for promptly alerting the owner or the police station of the theft via a text message. Finally, this work analyzes the feasibility and commercial viability of the system by running experiments in various topologies and observing the response times of the system. These experiments were run for extended periods of time and hence the results also go on to prove the stability of the system in an actual implementation.

Table of Contents

LIST OF FIGURES vi

ACKNOWLEDGEMENTS viii

Chapter 1 Introduction 1

1.1SVATS Overview 2
1.2 Contributions 3
1.3 Hardware / Software Platforms 4

Chapter 2 SMS Send in SVATS 9

2.1 Introduction 9
2.2 Overview of SMS in GSM Networks 10
2.3 Implementation Options for SVATS 12
2.4 Proposed Solution 13

Chapter 3 SVATS Security Framework 17

3.1 Introduction 17
3.2 Attack Models 17
3.3 SVATS Security Framework Requirements 19
3.4 Proposed Solution Details 21

Chapter 4 Timing Measurements 29

4.1 Introduction 29
4.2 Goals 30
4.3 Measurement Parameters 31
4.4 Measurement Procedure and Results 32
4.5 Conclusion 40

Chapter 5 Packet Loss in SVATS network 41

5.1 Introduction 41
5.2 Packet Loss in SVATS 42
5.3 Measurements and Results 44
5.4 Measurement Analysis 49
5.5 Reliable Message Delivery Requirement in SVATS 50
5.6 Proposed Solution for Reliable Message Delivery 52
5.7 Conclusion 53

Chapter 6 Motion Sensor Based Movement Detection 54

6.1 Introduction 54
6.2 Accelerometer 54
6.3 Motion Sensor scheme 55

6.4 Comparative Study of Motion Sensor scheme with RSSI based scheme 56

Chapter 7 Conclusions and Future Work 58

7.1 Conclusions 58
7.2 Future Work 59

References 65

LIST OF FIGURES

Figure 1-1 SVATS in action 3
Figure 1-2 MicaZ Mote 5
Figure 1-3 Block Diagram of the MPR2400 based MicaZ mote 6
Figure 2-1 GSM Network Architecture 10
Figure 2-2 Simulation Setup for SMS sending 13
Figure 3-1 Message Integrity 23
Figure 3-2 Message Structure modifications 25
Figure 4-1 Single Level Topology 34
Figure 4-2 Results Single Level Topology 35
Figure 4-3 Two Level Topology 36
Figure 4-4 Results Two Level Topology 37
Figure 4-5 Three Level Topology 38
Figure 4-6 Results Three Level Topology 39
Figure 4-7 Varying Alive Message Intervals 40
Figure 5-1 Scenario 1 44
Figure 5-2 Packet Loss for Scenario 1 44
Figure 5-3 Scenario 2 45
Figure 5-4 Packet Loss for Scenario 2 45
Figure 5-5 Scenario 3 46
Figure 5-6 Packet Loss for Scenario 3 46
Figure 5-7 Scenario 4 47
Figure 5-8 Packet Loss for Scenario 4 47
Figure 5-9 Scenario 5 48

Figure 5-10 Packet Loss for Scenario 5 48

Figure 5-11 Mean and Median Values 49

ACKNOWLEDGEMENTS

I would like to thank my advisor Professor Guohong Cao, who provided the overall direction as well as day to day guidance for this research work. For their review, instructive comments and advice, I thank my committee co-chair Professor Sencun Zhu. I would also like to thank graduate students Wenhui Hu and Soumya Jain for their help in the coding and implementation of the system.

Chapter 1

Introduction

A wireless sensor network (WSN) is a wireless network consisting of spatially distributed autonomous devices using sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, at different locations. The original motivation of a wireless network was military applications such as battle field surveillance.

However, of late, WSNs are finding their way into many industrial and civilian applications, such as industrial process monitoring and control, machine health monitoring, environment and habitat monitoring, healthcare applications, home automation, and traffic control. In addition to one or more sensors, each node in a sensor network is typically equipped with a radio transceiver or other wireless communications device, a small microcontroller, and an energy source, usually a battery.

In a project started two years ago, we have developed a sensor network based vehicle anti-theft system (SVATS). SVATS was mainly designed as an automobile theft detection and notification mechanism. This thesis studies the feasibility and durability of the system. Also, it improves the commercial viability of the system by making certain valuable additions.

In the United States, a motor vehicle is stolen every 26 seconds. The national insurance crime bureau has reported that vehicle theft rates have kept steadily high at about 1.2 million since 1997, despite the widespread use and increasing sophistication of anti-theft devices. These systems can be classified roughly into three types – *lock devices, alarm systems and vehicle tracking / recovery systems*. Lock devices are represented by the typical steering wheel lock which, though cheap are easy to disarm. Alarm systems are by far the most popular anti-theft

devices. However, they suffer from very high false positive rates. The most sophisticated are the vehicle tracking systems, usually based on wireless transmitters which can be activated along with a GPS device which broadcasts the current location to all tracking devices. The main drawbacks of this system are the high costs involved and the fact that they can be easily disabled. As an illustration of the former reason, the upfront costs are in the range of $ 500 - $ 1500, and there are other associated periodic payments.

To address these above mentioned limitations, we designed a sensor network based vehicle anti-theft system.

1.1 SVATS Overview

SVATS (Secure Vehicular Anti Theft System) is an anti-theft system developed about 2 years ago. The underlying principle of the system is that the there is a sensor (Mica2 or MicaZ mote) attached to each car. These sensors continuously transmit wireless signals to the other cars .The sensors in the vehicles that are parked within the same parking area first form a sensor network, then monitor and identify possible vehicle thefts by detecting unauthorized vehicle movement. When a vehicle theft is detected, an alert will be reported to the base station, if possible, which can immediately notify the security office or the vehicle owner. Further, the stolen vehicle can be tracked by the theft reports received at roadside access points, which are sent from the sensors in the stolen vehicle.

Here is a basic overview of the how SVATS works

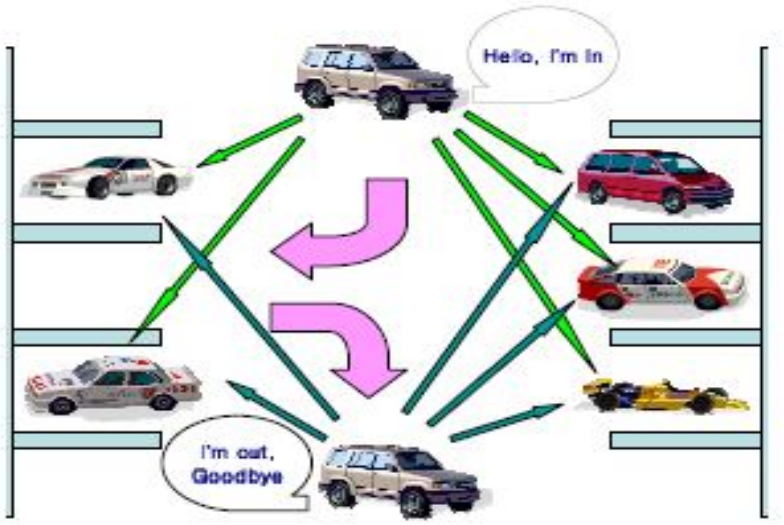

Figure 1-1 SVATS in action

Let us say Bob enters the parking lot in his car. Using a remote control, he instructs the master sensor to send a 'join' message to the network. On receiving this message, the nodes of the existing network add Bob's master sensor to their list of neighbors who they need to monitor.

Further, Bob starts making his own list of neighbors at the same time. After joining the network, Bob's master sensor sends periodic *alive* messages which indicate his presence and help the neighbors create a signature of his RSSI. When the car leaves, it sends a 'leave' message. If it moves without sending a valid 'leave' message, the 'monitors' will immediately know this based on the change in the RSSI signature formed. After voting amongst themselves (to reduce false positives), they will notify the base station of the theft.

1.2 Contributions

1. Theft updates merely reaching the base station is insufficient. They need to reach either the owner of the car or some other authority in real time for some quick action to be

taken. This work integrates the stand alone SVATS sensor network with an already established public network, such as GSM or the GPRS network to achieve this end.

2. The current RSSI scheme is found to have too many false positives. The RSSI changes too frequently and is affected by environmental factors too much. This work suggests a better detection scheme with much lesser false positive rate.
3. Currently, all SVATS message exchange happens in the clear. For the success of the system, a security framework needs to be in place. This work analyses the attacks possible on such a network and implements and designs a basic security framework for the system.
4. Finally, experiments have been run in various parking lot like topologies to prove the durability and sanity of the system. Timing measurements and results of these experiments will prove the feasibility and commercial viability of the system.
5. The prototype implementation was done with Mica2 motes. This work ports the code to MicaZ motes. Though not very different in form factor and range, these motes differ significantly in their data rates. [3][4] Mica2 supports a data rate of 38.4 KBaud. As more and more vehicles join the network, this could prove to be a bottle neck. It would be worthwhile for over all functioning of the system to move to the new MicaZ mote, which supports a data rate of 250 kbps. Problems such as congestion etc can be solved with using the new mote.

1.3 Hardware / Software Platforms

This section provides a brief overview of the hardware and software platforms used for system implementation and evaluation. The hardware used is the Crossbow MicaZ mote. The earlier implementation uses Mica2. Using MicaZ offers a definite advantage in terms of data

rates. The software used is the TinyOS operating system and NesC language. These programs were installed on the MicaZ mote to get the system working.

Hardware Platform – The MICAZ mote

Figure 1-2 MicaZ Mote

MICAZ sensor mote is a 2.4 GHz mote module used for enabling low-power, wireless, and sensor networks. It comprises MPR2400 processor, which is actually based on the low power microcontroller Atmel AtMega128L. The mote has CC2420 radio, with an expansion connector for sensor boards (with various sensing modalities) and LEDs for visual debugging. The radio offers high speed (250 kbps) and high security. MicaZ works on 2 AA size batteries and has a multiyear life. The 51-pin expansion connector can connect to a variety of sensor boards, which provide sensing of quantities such as pressure, temperature, light, acceleration and magnetic fields. The programming board used is a MIB520, which has a USB connection and can be plugged to the laptop directly. The board allows installing the code on the MicaZ motes and also transmits information back to the PC for debugging information.

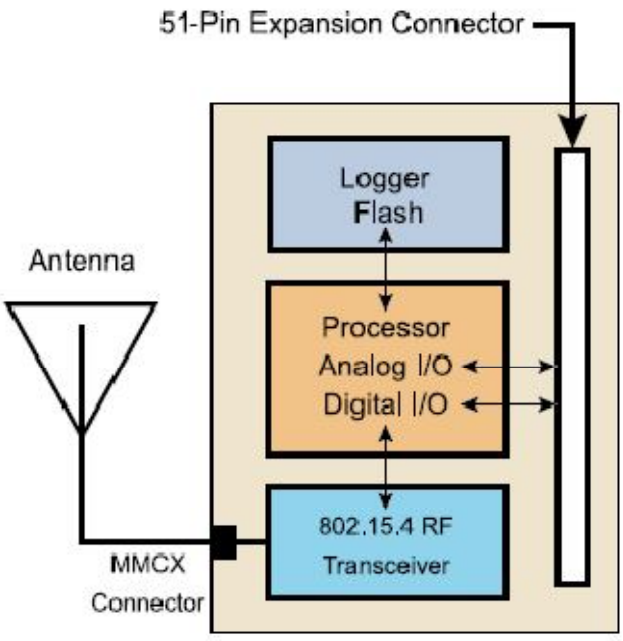

Figure 1-3 Block Diagram of the MPR2400 based MicaZ mote

The Software Platform – TinyOS operating system and nesC programming language

For the purpose of writing the software and installing it on the mote, we use the operating system TinyOS and the programming language NesC. Following describes the basic TinyOS concepts. The TinyOS system, libraries, and applications are written in NesC, which is a C-like language for programming structured component-based applications. The NesC language is primarily intended for embedded systems such as sensor networks. NesC has a C-like syntax, but supports the TinyOS concurrency model, as well as mechanisms for structuring, naming, and linking together software components into robust network embedded systems. The principal goal is to allow application designers to build components that can be easily composed into complete, concurrent systems, and yet perform extensive checking at compile time. TinyOS defines a

number of important concepts that are expressed in NesC. Firstly, NesC applications are built out of **components** with well-defined, bidirectional **interfaces**. Secondly, NesC defines a concurrency model, based on **tasks** and **hardware event handlers**, and detects **data races** at compile time.

A NesC application consists of one or more *components* linked together to form an executable. A component **provides** and **uses** *interfaces*. These interfaces are the only point of access to the component and are bi-directional. An interface declares a set of functions called **commands** that the interface provider must implement and another set of functions called **events** that the interface user must implement. For a component to call the commands in an interface, it must implement the events of that interface. A single component may use or provide multiple interfaces and multiple instances of the same interface.

There are two types of components in NesC: **modules** and **configurations**. Modules provide application code, implementing one or more interface. Configurations are used to assemble other components together, connecting interfaces used by components to interfaces provided by others. This is called **wiring**. Every NesC application is described by a **top-level configuration** that *wires* together the components inside.

Finally, we present an overview of the concurrency model. TinyOS executes only one program consisting of selected system components and custom components needed for a single application. There are two threads of execution: *tasks* and *hardware event handlers*. Tasks are functions whose execution is deferred. Once scheduled, they run to completion and do not preempt one another. Hardware event handlers are executed in response to a hardware interrupt, and also run to completion, but they may preempt the execution of a task or some other hardware event handler. Commands and events that are executed as part of a hardware event handler must be declared with the **async** keyword. Because tasks and hardware event handlers may be preempted by other asynchronous code, NesC programs are susceptible to certain race conditions.

Races are avoided either by accessing shared data exclusively within tasks, or by having all accesses within **atomic** statements. The NesC compiler reports potential *data races* to the programmer at compile-time. It is possible the compiler may report a false positive. In this case a variable can be declared with the **norace** keyword.

Chapter 2

SMS Send in SVATS

2.1 Introduction

SVATS system currently deals with theft detection in the isolated wireless sensor network. Though this in itself is a significant value add, commercial usefulness of this system can be significantly enhanced if SVATS can take care of end to end reporting of the theft to the vehicle owner or security personnel responsible for the parking lot or to local police authority. This end to end reporting will require integration of the isolated SVATS wireless sensor network with the public network which has far and wide reach. Once this integration is done, various useful applications can be built on top of it, which utilize the robust theft detection mechanism provided by SVATS.

As soon as theft is detected, it could be automatically relayed to multiple recipients for whom this information will be of interest. Vehicle owners can be informed that their car has been stolen. Local police authorities could also be informed, once the reliability of the alert has been established by the SVATS wireless sensor network. Instant alarm alert can be sounded at the control room of the parking lot. This alarm alert can also be propagated to the personnel responsible through communication mechanisms supported by today's public network (e.g. cellular phones etc.). Applications utilizing the theft detection mechanism of SVATS could enable tracking of the vehicle's status through internet.

As mentioned above, integration of isolated wireless sensor networks with public networks has tremendous commercial potential. Integration of the existing system with SMS in GSM networks has been demonstrated in the current work. The current work can easily be

extended for other types of integration between isolated wireless sensor networks and other public networks.

Due to the ubiquitous nature of the cellular networks in today's world, they are the most logical choice of a theft reporting mechanism in real time. SMS has been chosen as the real time communication mechanism here but an automated call can also be made using the work demonstrated here.

2.2 Overview of SMS in GSM Networks

The basic network architecture of the SMS is depicted in figure 4.

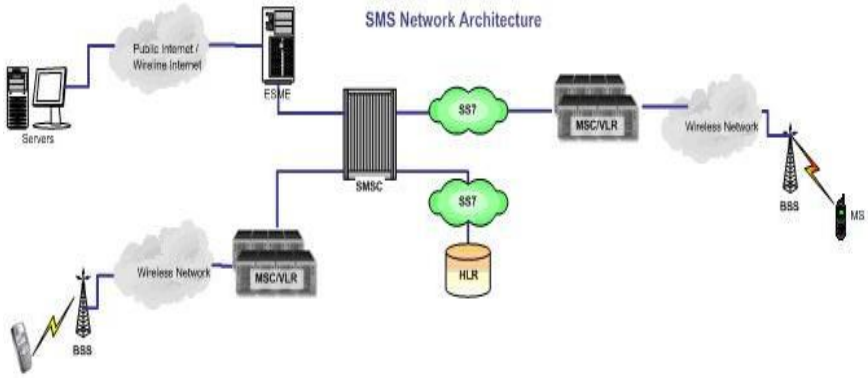

Figure 2-1 GSM Network Architecture

The following are the main entities involved.

Short Messaging Entities are responsible for receiving and sending messages. The location of an SME can vary. For instance, it can be mobile or it can be located in a fixed network

Short Messaging Service Centre is the network element sitting between an SME and the mobile phone. It relays messages to and fro. It can also store messages and forward them later in case the receiving device is switched off or out of coverage area etc.

If the message originates or terminates at an entity other than a cell phone, an SMS gateway is required which interacts with the SMSC to ensure delivery of these SMS messages. These SMS gateways are typically provided by people called aggregators. Some applications have tie ups with aggregators to deliver SMS messages generated by the application. For instance, mobile outlook is an example of such an application for Microsoft.

*SMS-Gateway/Interworking Mobile Switching Center, a*lso known in short as the SMS-GMSC, this entity manages interaction between the sending SMSC and the receiving SMSC. In addition to that, it is the responsibility of the SMS-GMSC to query the home location register (HLR) and extract routing information from it. This component is generally integrated with the SMSC itself.

Home Location Register, also known as the HLR, stores records for each subscriber, including the current location, billing information etc. This helps provides information to route messages, calls etc. It also interacts with the SMSC to give it routing information or inform when a cellular device goes out of the network coverage are or comes back into the network coverage area. HLR storage is permanent and accessible to all SMSCs.

Mobile Switching Center is the network component which sits between the base station controller and the rest of the GSM network. It is responsible for call and data control into the GSM network.

Visitor Location Register (VLR) is a database that contains temporary information about subscribers. This information is needed by the MSC to service visiting subscribers.

The Base Station Subsystem is responsible for providing all the radio related functionality to the network. This forms the radio interface between the cellular devices and the GSM network.

BSS is composed of two main components – the Base Station Controller (BSC) and Base Transceiver Stations (BTSs). BTSs are the first hop from the mobile station. These carry voice and data traffic forward to the BSC and then further to the rest of the network.

The Mobile Station is the cellular deice we are familiar with. It uses the wireless channel to make or receive voice calls and send or receive SMSs. The protocol used for signaling on the wireless channel is SS7.

2.3 Implementation Options for SVATS

SVATS details a mechanism of theft reporting to the base station. It is assumed that the base station or the control center of the parking lot will have connectivity with a network in the outside world. This connectivity will be used to achieve integration between SVATS and other public network.

For experiment purposes it is assumed that the base station is connected to a PC through the serial port. This PC is in turn connected to a bluetooth enabled cellular phone through bluetooth. On receiving an alert over the serial port, the PC triggers sending of SMS to the cellular phone which in turn sends the SMS to the cellular network. The cellular network then delivers SMS to the destination cellular phone.

There are 3 main options of sending a SMS from a device other than the cellular phone.

1. Buying or downloading an application which enables sending SMS from PC to mobile phone. The application, however, should provide an interface which can be used by the SVATS software implementation which reads messages over the serial port to trigger SMS sending. The application thus cannot be just a black box which allows users to send SMSs from a GUI screen. That will not serve the purpose. The application must provide APIs to interface with it.

2. The second option is to take service from an aggregator. The job then will include making an application, complying with the aggregators API. This however would be an expensive option because the aggregator will charge per message. This is normally used for applications which need bulk message sending. The aggregator typically charges per 100 messages or even per thousand messages. SVATS however, does not require such a heavy rate of message sending and hence, it might not be economical to go for this option of message sending.

3. Using a GSM modem: Cellular phones also have a mode in which they can function as a GSM modem. Cellular phone can be connected to the PC using the serial cable, USB connection or a bluetooth connection. AT commands used for controlling a modem can be used to send SMS to the GSM Network. The cost incurred in this option is the cost of sending one SMS in the cellular network. It is assumed that theft detection messages would only be sparingly sent from a Base Station. Thus, the cost incurred in this option will be minimal. This option has been used in current work. Connectivity between cellular phone and PC is achieved through bluetooth.

2.4 Proposed Solution

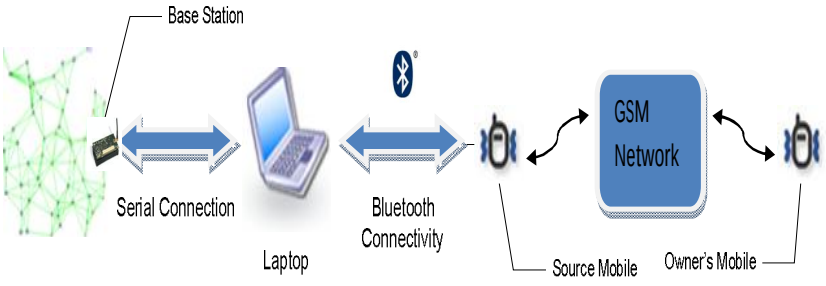

Figure 2-2 Setup for SMS sending

Figure 2-2 details the setup used for SMS sending from SVATS network. For experiment purposes, the base station can be easily plugged to the laptop, which is assumed to have bluetooth capability. It can thus pass on the message to another blue tooth enabled device, which further pushes the message onto the GSM network.

The software running on the laptop is Gnokii [8]. Gnokii is open source software which allows interfacing a PC with the mobile phone which is acting as a GSM modem. It is supported by Nokia. It provides tools and user space drivers for use with mobile phones under various operating systems. Though most of the Gnokii testing is done under Linux, it works fairly well with Solaris and Windows machines too. For the current work Gnokii was used on a Windows XP platform. Gnokii software provides options for connecting the cellular phone to the PC through a serial cable connection, USB connection, infrared connection or bluetooth. Bluetooth connectivity has been used in current work.

It provides functionality in different areas for users to manipulate the mobile phone.

- Users can send SMS, receive them and save them in the phone. Gnokii supports delivery reports, picture messages, concatenated messages, wap pushes, unicode messages. Gnokii allows users to send and receive logos and ringtones over SMS.
- Gnokii offers users the possibility to read and write phonebook. Phonebook entries can be displayed in human readable form or can be exported to comma separated output, vCard version 3.0 format or ldif format.
- Gnokii supports either calendar or todo lists. Gnokii is capable to export calendar to iCal files and import them from the same format. Most calendar features like different types of events, recurrence, start and end times are supported.
- Users can initiate and answer calls with gnokii.

- Among other gnokii capabilities are security options like entering PIN, ringtone and logo handling and others.

Implementation Details

The Base Station is connected to the PC or the laptop via a serial RS232 cable. The laptop and the cellular phone are connected via bluetooth. The software used is Gnokii, the details of which are described in the previous section. The cell phone device used is Nokia 6085. It has bluetooth connectivity, which is crucial for the experiment

Gnokii code was interfaced with SVATS code to implement the complete solution. An interface program was written in C. This was the interface between the mote and the PC through the serial port. The mote, which is the Base Station in this case, sat on the programming board and communicated to the PC via the serial port. The interface program continuously monitored the serial port messages, waiting on a theft report message.

Upon receiving a theft report, the interface function to the GNOKII library is invoked. This uses low level AT commands to instruct the cellular device in bluetooth range to send the text message to another cellular device. After the message is received at the receiving phone, a delivery report is delivered to the sender, which can also be read using AT commands. The status of this message is checked to confirm if report delivery was successful or not.

AT commands [12], mentioned above are instructions used to control a Modem. AT is the abbreviation of attention. Every command line starts with “AT” or “at”. GSM/GPRS modems and mobile phones support an AT command set that is specific to the GSM technology, which includes SMS-related commands. Some of the SMS related commands are the following

AT+CMGS - Send SMS message

AT+CMSS - Send SMS message from storage

AT+CMGL - List SMS messages

AT+CMGR - Read SMS messages

Mobile phone manufacturers usually do not implement all AT commands, command parameters and parameter values in their mobile phones. Also, the behavior of the implemented AT commands may be different from that defined in the standard. In general, GSM/GPRS modems designed for wireless applications have better support of AT commands than ordinary mobile phones.

Some AT commands require the support of mobile network operators. For example, SMS over GPRS can be enabled on some GPRS mobile phones and GPRS modems with the +CGSMS command (command name in text: Select Service for MO SMS Messages). But if the mobile network operator does not support the transmission of SMS over GPRS, this feature cannot be used.

Chapter 3

SVATS Security Framework

3.1 Introduction

In the existing implementation of SVATS, no security mechanisms are implemented. All message transmission happens in the clear. It is fairly simple for an outsider to sniff packets from the network and exploit the vulnerabilities of the system. Due to the absence of any security mechanism, confidentiality and integrity of the messages cannot be ensured. Also, the authenticity of the sender cannot be established at the receiver.

This chapter analyses the attack model for SVATS, then the requirements of a security framework for the system. It then proposes a Security framework for SVATS. Finally, the implementation details of the framework are explained.

3.2 Attack Models

This section outlines the type of attacks that a SVATS like network is prone to. Also some mechanisms to overcome the same are proposed side by side.

Denial of Service Attacks

It is very easy to launch a denial of service attack on a sensor network. All nodes talk to each other at the same frequency. All nodes follow CSMA/CA protocol to transmit. That is, they sense the channel before sending anything. It is fairly straight forward (as shown by

implementation also) for a node to override the CSMA/CA protocol and flood the channel with messages. Because this mote uses the channel to full capacity, other motes cannot send or receive anything from the channel. This effectively destroys the purpose of SVATS. No message exchange is possible in presence of such a jammer node.

By conducting experiments, it was observed that though MicaZ motes can transmit at the rate of 250 kbps but another mote listening to this mote can hear only 17 kbps of this data. The mote transmitting at 250 kbps does not follow any MAC layer protocol. It was observed that two such motes, with CSMA/CA turned off and transmitting at full capacity can cut almost entire communication between legitimate SVATS nodes.

A detailed scheme to detect and avoid jamming is proposed in Section 3.4

Eavesdropping

Since packets are transmitted in clear, an adversary can eavesdrop on them. Packet fields can be altered and packets can be retransmitted. For instance, AM_MTLEVELDISCOVERY message from the Base Station helps the other motes level themselves with respect to the Base Station. The motes which listen to this message directly know that they are one hop away from the Base Station. But altering this packet and making the level something other than zero can kill the intent of the system. Because, now no node knows that it is one level away from the Base Station. Hence, there is no way that alert messages can reach the Base Station

Also, an adversary can listen to alert messages and can later alter node IDs and play it to the Base Station and cause chaos by reporting false alerts and causing the whole system to fail. The problem of eavesdropping can be effectively countered by encrypting and integrity protecting the messages. The details of the scheme are provided in the next few sections.

Attacks by malicious nodes

Malicious nodes, located on the route to the base station can choose to silently drop alert messages. Also, malicious nodes can chose to not participate in voting and hence alert messages not getting generated. Both of these can have harmful effects on the system. The security framework proposed in this chapter, along with the new detection scheme proposed in the last chapter can avoid the problem caused by malicious nodes to a great extent.

3.3 SVATS Security Framework Requirements

Most of the above mentioned problems can be solved by bringing in place a simple security mechanism for SVATS. This would provide the basic security protection to all SVATS messages, which includes authenticity, confidentiality and integrity.

SVATS, being a wireless sensor network has its own constraints and the proposed security scheme should take those constraints into account. At the very least, the security framework should provide the following features.

Confidentiality, Integrity, Authenticity

A security mechanism for SVATS should provide, at a minimum, confidentiality and integrity of the messages and authenticity of the sender to the receiver. This would help tackle the problem of eavesdropping and also aid in jamming detection, mentioned in the previous section.

Suitability for a resource constrained system

The proposed security mechanism though robust should also be simple for a resource constrained system like MicaZ mote to implement. Since motes are battery powered and hence power constrained, using asymmetric key protocols could be an expensive operation to afford.

Since message exchange rate between motes in SVATS is quiet high, encrypting/ decrypting these with asymmetric keys could deplete the mote of power and battery too soon.

Catering to various message types

Currently implemented SVATS has mainly three types of messages:

- Broadcast messages (AM_MTLEVELDISCOVERYMSG, AM_MTJOINREQMSG etc)
- Unicast messages (AM_MTJOINREPMSG)
- Multicast messages (AM_MTALIVEMSG)

The broadcast messages can be sent by any node and every node must be able to interpret these messages. Hence, for these messages, a global key needs to be used. A global key is one, which all nodes in the network are aware of. All broadcast messages are encrypted using this key and decrypted using it too. Since, all the nodes, including the base station possess the global key, everyone can send messages encrypted with the global key and all nodes can also decrypt messages encrypted with the global key.

For one to one messages, for example, when a node receives a join request, it sends back a join reply. The join reply is intended to be heard only by the node it is addressed to. For unicast messages such as these, all nodes need to share a key with all other nodes they will need to communicate with. Mechanism for such one to one communication needs to be chosen carefully because it might become practically impossible, as the number of nodes increases.

Finally, messages such as alive messages need to be heard only by nodes which are monitoring a particular node. These nodes start a timer to periodically check if it is receiving sufficient alive packets from the nodes it is currently monitoring. Other nodes need not listen to these alive messages.

Thus the security mechanism needs to implement three types of keys:

- Global – shared by everybody in the network and used to encrypt broadcast messages in

network

- Individual pair wise – shared only between two individuals who are involved in a one to one communication

- Group specific – shared only between individual parts of a group (e.g. monitor group) and is required for any communication happening within the group

3.4 Proposed Solution Details

Keeping in mind the requirements of a security framework for SVATS, we implement the general encryption and decryption mechanisms for all messages. The following are the details of the solution proposed.

Key Generation

To implement the keys with which various messages will be encrypted and decrypted, we use a modified version of Blundo scheme. Blundo scheme proposes a key management protocol for wireless sensor networks. This protocol has been used for generation of broadcast, group and individual network keys for message encryption. These keys are symmetric in nature. The following few paragraphs detail how the Blundo scheme works.

The scheme uses bivariate polynomial to generate one to one keys between nodes. A bivariate polynomial has 2 variables. It also has the special property that interchanging the variables in the polynomial does not alter the final value of the polynomial. That is, f(x, y) = f(y, x). It is this special property of the polynomial, which aids in key negotiation between 2 nodes without the symmetric key ever been exchanged over the air. For example, if the node id of a particular node is 1, the polynomial stored on that node shall be f (1, y). Similarly, polynomial

stored on node id 2 shall be f (2, y). Now whenever node 1 wants to talk to node 2, it puts y = 2 in its polynomial and derives the shared key between 1 and 2, which is f (1, 2). Similarly, node 2 puts y = 1 and calculates f (2, 1).

Since f(x, y) = f(y, x), it implies that f (1, 2) = f (2, 1).

Thus, nodes 1 and 2 actually end up computing the same key and this symmetric key can be used for encrypting and decrypting messages exchanged exclusively between nodes 1 and 2.

Blundo scheme is more relevant for static kind of networks, where node movement is minimal and configuration, once formed, remains static throughout. It uses polynomial pool-based key pre-distribution scheme. In this scheme, all motes at compile time are assigned a subset of polynomials from a polynomial pool. At the time of key establishment, motes look for other motes that share a polynomial with them and keys can be directly established between such motes. However, motes which don't share a polynomial cannot establish a key amongst themselves. Such motes need to look for other motes with which both the motes share a key. With the help of this new mote, a key can be established.

SVATS however is a fairly dynamic network. Since the underlying assumption of Blundo scheme is static topology of networks, the scheme has been modified for its use in SVATS.

There is no guarantee that a mote will share a key with another mote that it wants to talk to. Also, it is not guaranteed that there will be a third mote there with the help of which the key can be negotiated. For instance, when there are just two cars in the lot and they need to talk to each other, they must be able to do so and the probability of their being able to establish a key should be 100 per cent. This can be ensured if all possible node combinations share a polynomial.

To satisfy the above SVATS related requirement, changes have been made to the key management protocol code. Instead of assigning multiple polynomials to each mote and then computing the key by checking if there are shared polynomials, only a single polynomial is assigned to each mote. That ways, there is no probability required that two motes that want to talk

to each other share a key. Since every mote is assigned the same polynomial, they are bound to be able to negotiate a key amongst themselves. It is envisioned that such a polynomial can be provided to an incoming car by an initial exchange with the parking lot's base station at car's entry. However, for simulation purposes the polynomial was burnt on each mote.

Integrity Protection

A simple message integrity scheme has been proposed. For broadcast messages, a global flag is agreed upon. All messages begin with that global flag. Since it is known to all nodes, they decrypt the message and check if the first byte is indeed the global flag. If it is, the message integrity is intact then the rest of the fields are read. If not, the message will be discarded.

For unicast messages, it is proposed that the receiver id will be the first byte of the message. That ways, the receiver decrypts the message with a pair wise key and then checks if the first byte of the message indeed contains his own id. If it does, the message integrity is assumed. If not, the message is considered corrupt and discarded.

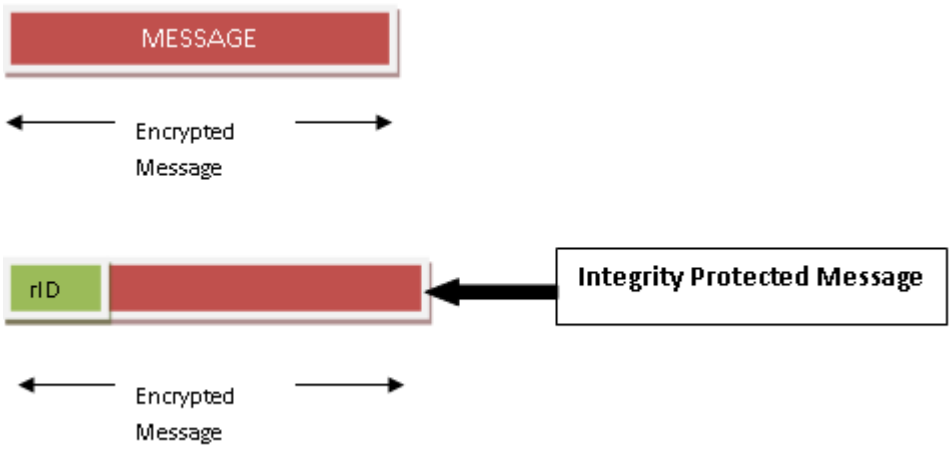

Figure 3-1 Message Integrity

In the above figure, the first message is simply encrypted and hence no integrity protection. In the second message however, the receiver id is present in the first byte of the message. This whole message is now encrypted. The presence of the receiver id in the first byte offers integrity protection.

The message integrity algorithm can then be made public. Since the integrity check value is encrypted using a secret key, it is ensured that no unintended node will be able to generate a valid message in the network.

Message Structure Changes

Certain changes have been made to the current message structure of the messages in SVATS for security implementation.

The main cause of these changes is that, by looking at the message header, the receiver should be able to figure out which key to use to decrypt the message. Also, the message identity should not go in the clear. Because that would reveal a communication pattern and also help attackers clearly identify alert messages being sent. For the receiver to figure out which key to use to decrypt a particular message, it needs to know two things – the sender ID and whether the message is a unicast or a broadcast. Global key is used to decrypt broadcast messages and pair wise key for unicast messages.

Currently the message header contains a 16 bit field which is the message id and sender and the receiver ids are present in the message body. These fields are now interchanged. The message ID is now a part of the message body and is hence encrypted. The sender id is now the header of the message. Also, this never occupies all 16 bits, because the sender ID can never be so huge.

That why, the most significant bit (MSB) of the sender id indicates whether the message is a broadcast or a one to one message. In case, the MSB is 1, the sender ID is not needed as far as decrypting the message is concerned. The message is decrypted with the global key.

If the MSB is 0, the message is not a broadcast and hence it is meant for this particular node. Then this node can read the sender ID and decrypt the message with the pair wise key that it shares with this particular sender.

Thus, the new message header gives the receiver complete information on how to decrypt the received message correctly. After decrypting, the first byte of the message can be read by the receiver to ensure that the message integrity is maintained and that the message is not corrupted.

If the first byte contains a pre agreed value, then the rest of the message is decrypted and further action taken. Else the message is discarded.

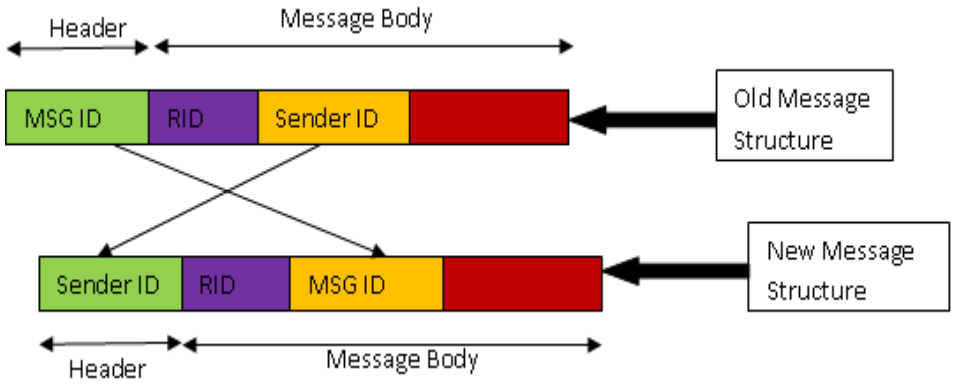

Figure 3-2 Message Structure modifications

Detecting and Avoiding Jamming Attacks

Jamming is a relatively easy attack to launch on part of the attacker and it can make the entire system ineffective. [14] and [15] are some of the available literature on how to simulate jamming attacks and how to prevent them. But most of the existing work is simulation based.

There is very little work, involving actual implementation. A very relevant work in this regard is [14] “The Feasibility of Launching and Detecting Jamming Attacks in Wireless Networks”. This work discusses various ways of launching a jamming attack, detecting a jamming attack and ways of preventing it.

The attack simulation is done by simply having 2 nodes override their MAC layer protocol and push data into the network at a maximum rate. This jams the channel very effectively and hinders communication between legitimate SVATS nodes to a great extent.

This work implements a new scheme for jamming detection. It is assumed that the jammer node does not have any legitimate keys that SVATS nodes use. Thus, after decrypting a message from a jammer, the receiving node will be able to determine that the message was junk and hence discard the message. In the attack launched, the jammer is able to hog the network and hence, all nodes in the jammed area will receive a lot of junk messages. A threshold value, let’s say t, is decided. When t junk messages are received by a node, it concludes that it is in an area that is under a jamming attack.

The following simple scheme can be used to avoid jamming. The nodes in the jammed area can switch to another channel. These nodes before switching to a new channel transmit messages indicating the new channel that they are switching to. Nodes in the jammed area may not be able to receive these messages. But nodes outside the jammed area can hear these messages and relay them further, till the whole network switches to a new frequency. At the new frequency, all nodes, including those in the jammed area, can talk to each other and not loose messages.

Also, another very easy to launch and effective jamming attack could be to jam the area around the base station. That ways, all other nodes in the network would continue to work normally and monitor each other for any abnormal behavior etc. However, since the area around the base station is jammed, no alert messages shall reach the base station. Thus, the whole

purpose of the system is defeated. Because it is only when alert messages are reported to the base station that further action can be taken.

To overcome the problem of the area around the base station being jammed, base station should have more participation in the protocol than it currently does. Base station should volunteer to monitor nodes that are reachable from it. It should monitor alive messages received from these nodes and check periodically. That ways, if in several consecutive cycles, if it hasn't heard alive messages from all the nodes that are reachable from it, it can detect that the area around it is jammed. The area can be physically explored to look for the source of the jamming or the base station can command the rest of the network to switch to a new frequency.

Code Overhead

The security code adds about a 2000 lines of code, to the existing SVATS code.

Communication Overhead

Since encrypting the message causes it to become 8 bytes, every message being transmitted within SVATS is now 8 bytes long. Message length varied earlier from 2 bytes to 7 bytes. Thus, every message incurs some overhead in terms of length, the overhead being more for shorter messages and less for longer messages.

However, the message integrity or authentication part mentioned earlier does not incur any extra overhead. This is because in any case, 8 byte messages are transmitted and received. Since maximum original Message length is 7 bytes (AM_MTLEVELDISCOVERY message), adding an extra byte does not increase the message size beyond 8 and hence, no extra bytes need to be transmitted than were already being transmitted in case of encrypted Messages.

The following is a step by step procedure of how message exchange happens in SVATS when the security framework is in place.

1. All communication is encrypted, either with the global key or the one-to-one key, depending on whether the message is unicast or broadcast.
2. On receiving a packet, the receiver looks at the most significant bit (MSB) of the header of the received message.
3. If the MSB is 1, then the message is interpreted as a broadcast message. The message is decrypted with the global key. The first byte of the encrypted message is compared to a pre defined value. If it matches, the message is further processed. Otherwise, the message is discarded.
4. If the MSB is 0, the message is interpreted as a unicast message. The header is read and is treated as the sender id. The one to one key with the sender is computed. The message is decrypted with the newly computed key. The first byte of the decrypted message is read. If this is the id of the receiver, the message is processed and further action is taken. If this byte does not the match the receiver id, the message is treated as corrupted and discarded.

Chapter 4

Timing Measurements

4.1 Introduction

It is very important to run the system for long periods to prove the feasibility of SVATS in an actual environment. Experiments were run in various topologies and leveling time, joining time and monitoring time for the nodes was calculated and plotted. Also, several of these nodes were made to stop sending alive messages periodically and pretend to be stolen. The timings between stopping the alive messages and a theft being reported are also calculated. To be able to conduct these experiments in the lab, the transmission power of the radio was reduced. This caused the MicaZ range to drop to 2-3 meters. This way, multilevel parking could also be simulated.

This chapter details the set ups in which the experiments were run. Also, followed by the setups are the graphs for various timings. 16 MicaZ motes were kept all around the base station mote as shown in the figures. They can all hear the base station, so that they are all at Level 1 theoretically. The motes cannot hear all other motes because of the reduced range, but can easily hear at least 3(number of motes required to be monitored) other nodes, so that they all come in monitored mode. Some of these motes periodically stop sending alive messages, that is, they pretend to be stolen. Theft reporting time measurements are taken. Code is written to periodically shut down some of these motes, take time measurements and restart the motes.

4.2 Goals

The reasons for running these experiments are manifold. We are trying to prove the stability of the system over long periods of time. In reality, cars may be parked in parking lots for hours all together. Running the system for long in the lab proves its long term stability and that the code won't break if run for large periods of time.

In addition to this, we are trying to determine optimum values for various parameters involved. For instance, how long should the alive message period be, what is an average theft detection time for various topologies in the parking lot, after how many alive message periods should we check for theft detection. All these parameters represent an offset between the energy spent by the mote and the increase in accuracy of detection.

With running this system, we are trying to observe various system characteristics for real – the timings involved, the stability of the system, the energy spent by the motes, and compare these to theoretically calculated values, based on which the system is designed and coded.

Also, we are trying to observe system behavior in various topologies. Cars in a parking lot can be standing in a lot of different topologies and that might affect the functioning of SVATS. For instance, some cars might be several hops away from the base station and theft reporting from them might take longer, as compared to cars that can hear the Base Station directly. Keeping different scenarios and situations in mind, 3 test topologies have been designed and all the experiments were run on these. These include nodes up to level 2 from the base station. These are representative of an actual parking lot.

4.3 Measurement Parameters

Important parameters associated with SVATS are

- *Leveling time*: Leveling time is the time at which a mote finalizes its level with respect to the base station. When the node is leveled, it basically knows how many hops away it is from the base station. It also knows about its upstream neighbors. In case the node needs to get a message to the base station, it has to send the message via these upstream neighbors. Leveling time is thus the time taken by a mote to get leveled.
- *Joining time*: Joining time is the time at which the mote enters joined state. Joined state is said to be the state when the node can hear one more node. The presence of a single node, other than itself, causes the node to enter joined state. These nodes can now monitor each other.
- *Monitoring Time*: All nodes keep sending out join requests at fixed intervals, till they have enough nodes monitoring them. This is a configurable parameter in code. The count is currently set to 4. Thus, a node will go into monitored state from joined state, once it has got Join replies from 3 other nodes. Monitoring time is the time taken for a node to enter monitored state. It is a crucial parameter, because if the node does not enter monitored state fast enough, it might be prone to theft before entering that state. The experiments determine the time taken for a node to enter monitored state.
- *Theft detection time*: Theft detection time is defined as the time taken to detect theft. It is measured as the time between when a node stops sending alive messages and when the theft report is actually being sent to the base station. This is also a crucial parameter, because this time should be short enough, so that the car is not driven out of the parking

lot, by the time a theft is detected.

- *Alive message period*: This period determines the interval at which the alive messages are sent. This is an important parameter and needs to be carefully determined. If the alive message period is too high, it might increase the theft detection time too much. If it is too low, it might cause congestion in the network in presence of too many nodes. Also, extra battery is consumed in sending those extra messages. Thus, the parameter is a careful tradeoff between the mote battery consumption and the speed of detection.

The times at which nodes enter leveled, joined and monitored states are crucial to the effective functioning of the system. Also, theft detection time is highly influenced by the alive message period. The objective of these experiments is to determine the time durations taken by motes to enter these states in a running system under topologies which might be occurring in an actual parking lot.

4.4 Measurement Procedure and Results

MicaZ indoor range is 20 to 30 meters. Hence, it was difficult to reproduce scenarios which involved multiple levels i.e. distances from the base station where the base station cannot be heard. To create these levels, transmission power of the motes was reduced to the minimum possible transmission level. When the transmission power is set to minimum, the transmission range of the MicaZ reduces to a couple of meters. Thus, it is possible to simulate several levels in the lab itself.

Several nodes in every configuration were made to stop sending alive messages and restart every 10 alive messages. This emulated their theft. To calculate various timings mentioned above, timestamps were taken at various places in code. An initial timestamp was taken when the

mote started up, in the init () function. Another timestamp was taken when the mote received either the level reply message or a level discovery message i.e. when the mote got leveled. This is the Leveling Time. A third timestamp was taken when the mote received a Join reply from another mote. This is the joining time. Another timestamp was taken upon receiving the third join reply. This would be the monitoring time. After being monitored, some chosen motes stopped sending alive messages. They took a timestamp right then. Then they sniffed the channel to wait for an alert message regarding them. On hearing this alert message, the mote took another timestamp. The difference between this and the previous timestamp would be the theft reporting time. Upon hearing the alert, the mote would restart itself and take those set of timestamps again and transmit a packet over the serial containing the values.

SysTime interface is used for taking the timestamps on events. SysTime provides a gettime32 () method. This method returns a 32 bit value, which is the time elapsed in milliseconds since the mote was started. By getting the difference between timestamps, the elapsed time is calculated. The values are written on the serial in milliseconds instead of microseconds. Millisecond precision is good enough for the application.

The experiment was run in each topology for 4 to 5 hours. Some 1200 - 1300 readings were registered for each topology for each of the timing values. These were then plotted on a scatter graph to visually be able to see easily in which range most of these timings lie. The peaks and the troughs were clipped as they distorted an average timing reading. In order to restart some selected motes, once they heard an Alert about themselves, the Sensor.init () function was called, which caused the mote to restart.

The topologies have been designed to cover cases and positions, as may be present in an actual parking lot. There might be motes in direct contact with the base station and others several hops away from the base station. A group of cars standing close to each other might all be able to hear each other. And there might be another group which can hear each other, but cannot hear this

group. There might be cars which cannot hear the base station directly, but need other cars in the middle to be able to transmit their messages to the base station. All these situations and conditions have been thought of and 3 representative topologies decided.

It is assumed that to prove the stability of the system in these topologies would mean to prove the stability of the system in most circumstances because topologies occurring in an actual parking lot would be a subset of these. The following section shows the topologies and the results obtained in these topologies. The graphs show the distributions of the readings taken.

Topology 1(Single Level)

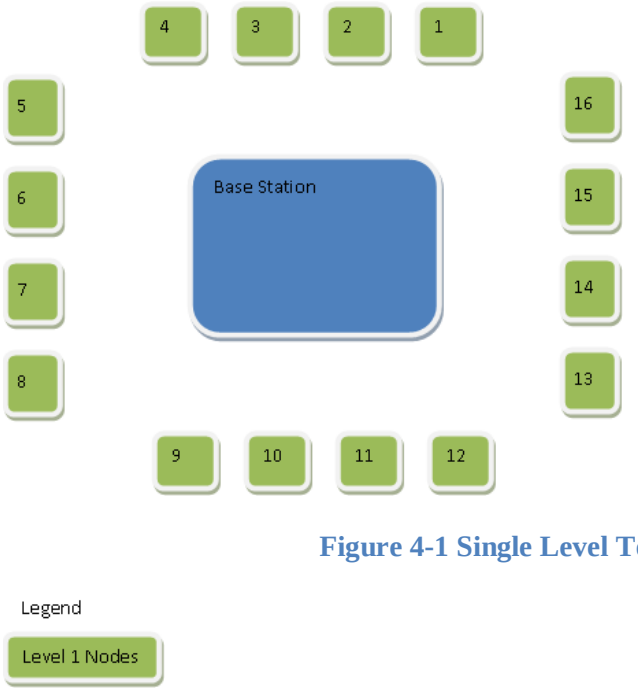

Figure 4-1 Single Level Topology

The following graphs show the measurements taken. All times are in milliseconds.

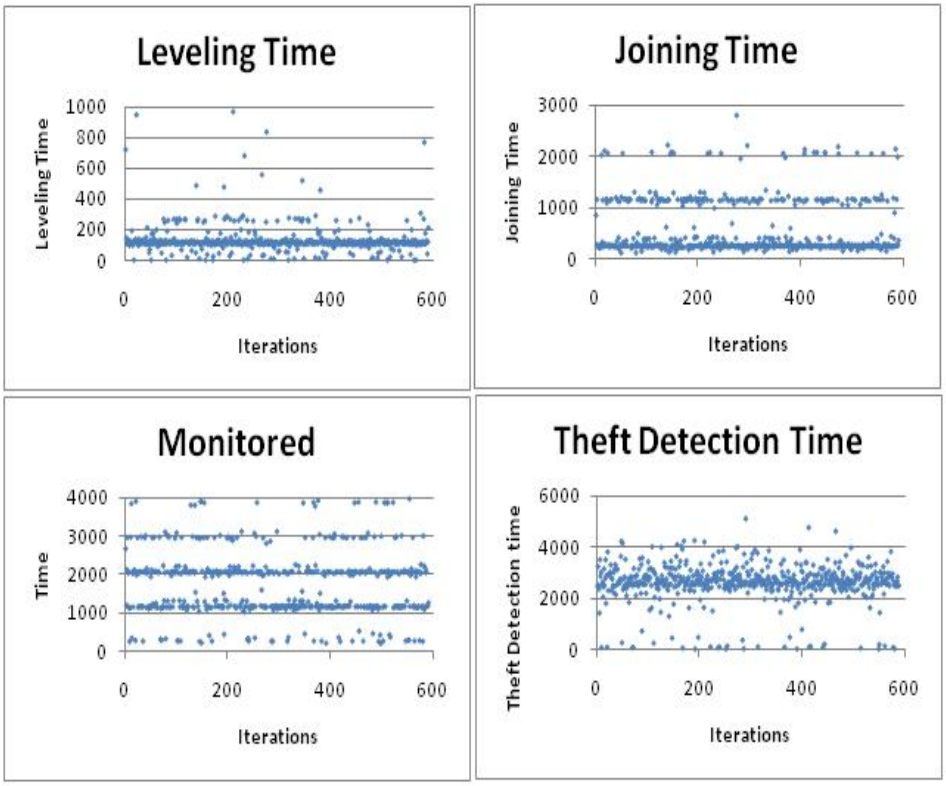

Figure 4-2 Results Single Level Topology

All leveling time readings are cluttered at about 100 milliseconds. This is because the mote sends out a level request at 100 milliseconds and gets leveled in response to the message. However, if that gets missed then the mote gets leveled with the AM_MTLEVELDISCOVERY packet coming from the Base Station. The Base Station sends out this message every 500 msec. That explains those peaks in the graph.

Joining time is the time when the mote knows that there is at least one neighbor in its radio range. As we can see in the graph, it is either cluttered at 250, then at 1250 and then at 2250 milliseconds. These correspond to retrials by the mote and re sending join requests.

A mote moves to monitored state when it knows it has 3 other nodes monitoring it. It has to wait for 3 join replies before switching to monitored state.

Theft detection time is the time from which the mote gets stolen (alive messages stop coming) to the time at which the alert is reported at the base station. Most theft detection times are cluttered between 2000 to 3000 milliseconds. This implies it takes between 2 to 3 seconds between when a mote stops sending the alive messages and the theft report being delivered to the base station.

Topology 2(Two levels)

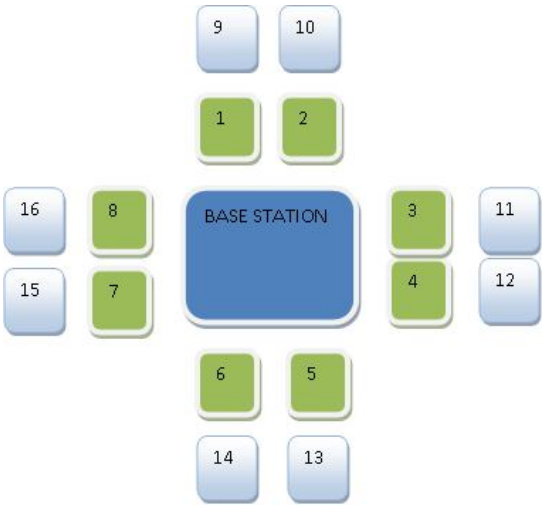

Figure 4-3 Two Level Topology

In the above topology, nodes 1 to 8 are at level 1(they can hear the base station directly). Nodes 9 to 16 are at Level 2 i.e. they cannot hear the base station but they can hear the motes at Level1. This constitutes the 2 level topology. While all these motes are not able to hear each other, the topology ensures that any mote can hear at least 3(minimum number required for the

mote to go into monitored state) others. Readings were taken at several of these motes - some at level 1 and some at level 2. Leveling times, joining times, monitoring times and theft times were measured. Following graphs show the results. All readings are in milliseconds.

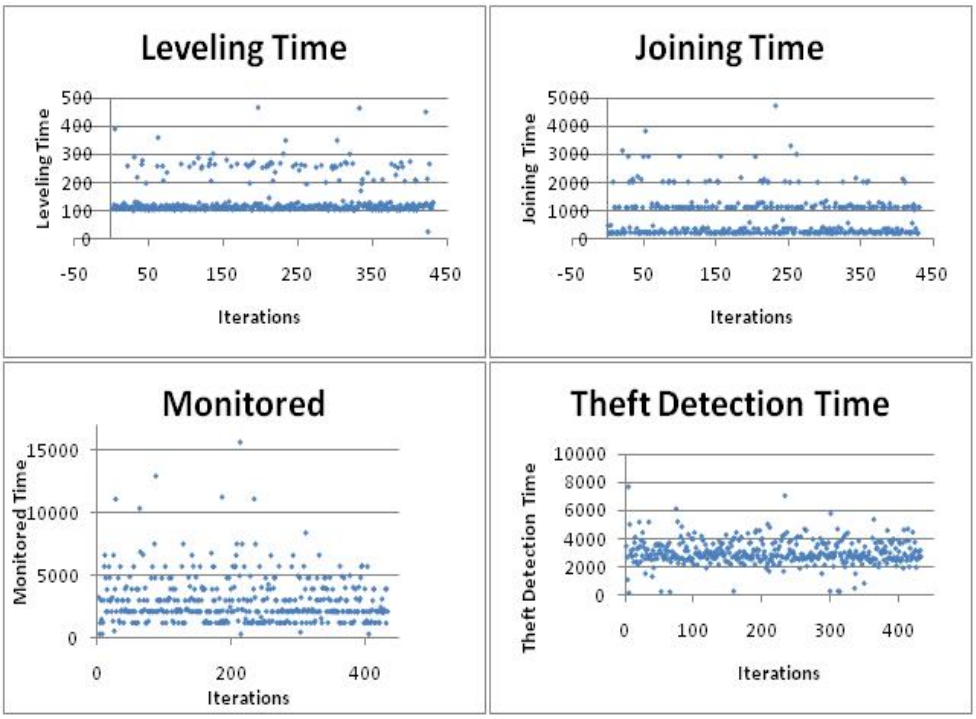

Figure 4-4 Results Two Level Topology

The graph shows the time to level is approximately 100 milliseconds. This is because that is the interval at which the level request is sent and then the reply comes, either from the base station or a lower level mote, depending on where the mote is located.

The theft detection time for this topology stays mostly between 2000 to 3000 milliseconds. Theft detection in a 2 level topology too remains between 2 to 3 seconds.

Topology 3(Three levels)

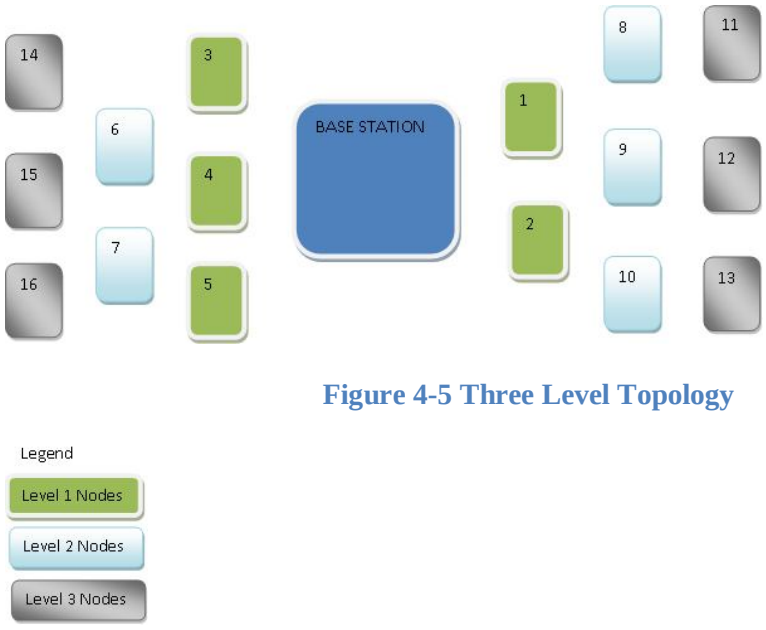

Figure 4-5 Three Level Topology

Shown above is a 3 level topology. Nodes 1 to 5 are at level 1. They can hear the base station, as well as nodes at Level 2 (nodes 6 to 10). Nodes 6 to 10 are at Level 2. They cannot hear the base station directly, but can hear nodes at level 1 and nodes at level 3. Nodes 11 to 16 are at Level 3. They cannot hear the base station. They can also not hear the nodes at level 1. They can only hear nodes at level 2 and hence their level is level 3.

Measurements were taken at several of these nodes at several different levels. Following graphs show the results of those measurements.

.

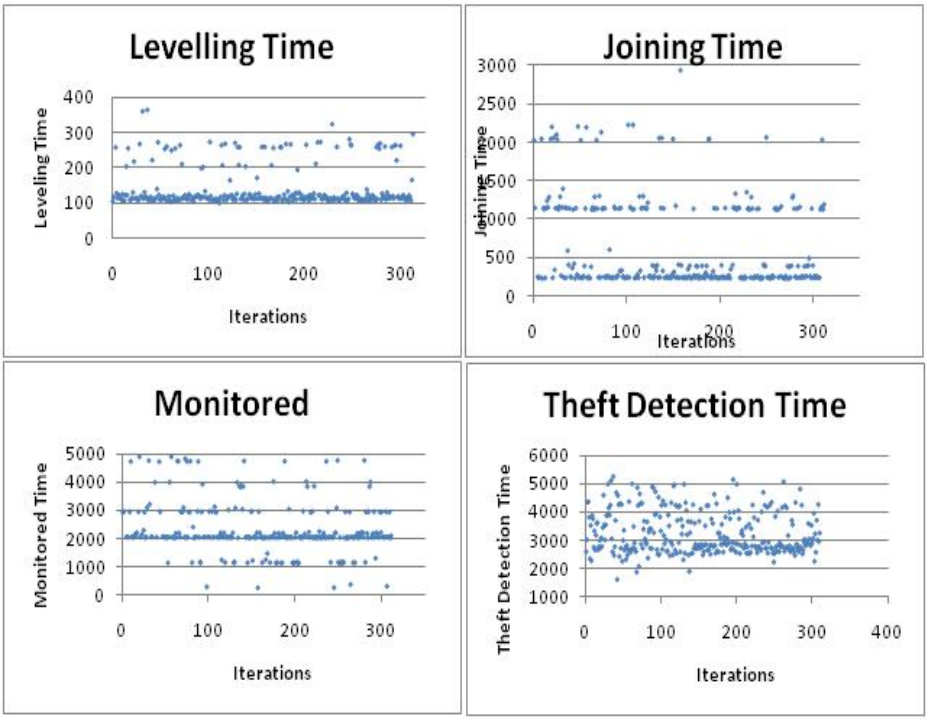

Figure 4-6 Results Three Level Topology

Another experiment that was run was to see the affect of alive message period on the theft reporting time. Experiments were run in topology 1 for alive message periods of 100 milliseconds, 200 milliseconds and 300 milliseconds. The difference in theft reporting time, as can be seen from the graph is huge. But it is offset by the battery consumed in sending the extra messages. The battery consumption doubles with making the alive message period half. The gain in theft reporting time, however, is not so huge. Also, 2 to 3 seconds is a decent response time for the system. Hence, alive message periods up to 300 milliseconds are acceptable, as far as the system performance is concerned.

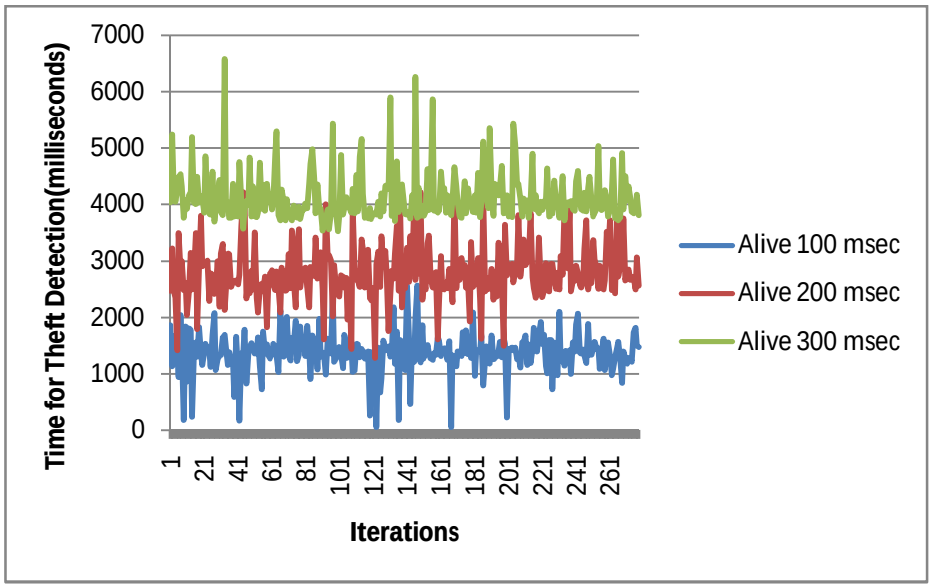

Figure 4-7 Varying Alive Message Intervals

4.5 Conclusion

Thus, we conclude that the leveling times, the joining times and the monitoring times, on an average for these nodes remain the same, and are independent of topologies. All these times remain within 5 seconds. This is a fairly good response time for a car anti-theft system. If the theft can be reported within 5 seconds of the theft happening, the system can be treated as reliable. This is because within 5 seconds, the car is bound to be still in the parking lot or definitely not very far away to take preventive action.

Chapter 5

Packet Loss in SVATS network

5.1 Introduction

Wireless sensor networks these days are used for a variety of applications. Due to the inherent characteristics of the wireless transmission medium used in sensor networks, packet loss is inevitable. Some of the wireless sensor network applications like temperature monitoring; habitat tracking, environment tracking etc. are tolerant to occasional packet loss. However, other applications such as industrial plant tracking or data flows from sink to sources for the purpose of control and management etc. are sensitive even to occasional packet loss.

While it is inevitable to avoid packet loss in a wireless transmission medium various network layer mechanisms can be used to mitigate the effects of packet loss. Different aspects of packet loss can be addressed at various layers of network protocol stack. The data link and MAC layers can take care of the reliable message delivery on a hop by hop basis and can also use different coding schemes to correct errors experienced at the physical layer. The network layer can be used for appropriate routing taking network congestion into consideration. The transport layer can take care of end to end message delivery guarantees. In addition to these layers, mechanisms can be built in the application layer protocols to take care of application specific scenarios for which sensor network is being used.

5.2 Packet Loss in SVATS

SVATS too is sensitive to packet loss. While occasional loss of alive messages between nodes can be tolerated, packet loss for messages like theft report is not acceptable. This work focuses on measurement of packet loss under various conditions to determine the effects of these conditions on packet loss. Subsequently, this work proposes reliability mechanisms for SVATS to mitigate the effects of packet loss.

Packet loss measurements are done on a per link basis. Since packet loss in a wireless sensor network could vary with different conditions, this work analyzes the effects of these conditions on measured packet loss. After filtering out the aberrations in measurements, a range for observed packet loss is determined.

The following were the considerations while measuring packet loss.

- *Topology*: Typical rectangular topology has been chosen which is representative of the shape of actual parking lots. Details of various topologies used is present as part of description of individual test scenarios.
- *Distance between nodes*: Since distance between nodes determine wireless signal strength and the fading effects, measurements have been taken with varying distances between different nodes. The distances chosen are multiples of nine feet which approximately is a representative of actual parking lot dimensions.
- *Physical Obstructions*: Physical obstructions could significantly vary wireless signal strength in sensor networks. Packet loss measurements have been taken by enclosing some of the nodes in a metal container. This scenario enables the study of effects of physical obstructions while keeping all other conditions same.
- *Environmental Surroundings*: Packet loss measurements have been taken both indoors and in outdoor surroundings. Care was taken to ensure that unusually

high environmental interferences to not affect the packet loss measurements. Careful selection for outdoor location and time of day (late evening) helped to eliminate unusually high environmental interferences.

- *Mote Specific Effects*: Packet loss measurements could be dependent on manufacturing characteristics of a specific mote. To eliminate dependence on this parameter it was ensured that same mote is used at the same location in all experiments. Unusual packet loss measurements, if any, from a mote were carefully monitored and to generalize measurements packet loss from other motes was also considered before determining packet loss ranges for a specific experiment. Since battery power plays an important role in these measurements, it was ensured that before running each experiment, batteries were charged to their full capacity.
- *Network Congestion*: Alive messages are the most frequent messages exchanged in a SVATS system. To simulate effects of more or less number of packets on the transmission medium, alive message period was varied for different experiments.
- Duration of Experiments: Each experiment was conducted for durations ranging from 4 hours to 12 hours. Packet loss measured is expressed as a percentage of total packets received during that duration.
- *Pair wise Measurement at each link*: Pair wise measurements were taken for each monitored link in the experiments. This maximized the number of available packet loss measurements. It also provided an opportunity to analyze the reasons for observed packet loss for each link, instead of getting a generalized view of packet loss for the whole SVATS network.

5.3 Measurements and Results

This section details the scenarios and the packet losses observed in them.

Scenario – 1

Motes were placed in topology as given in the figure below and the experiment was run indoors with an alive message period of 300 milliseconds.

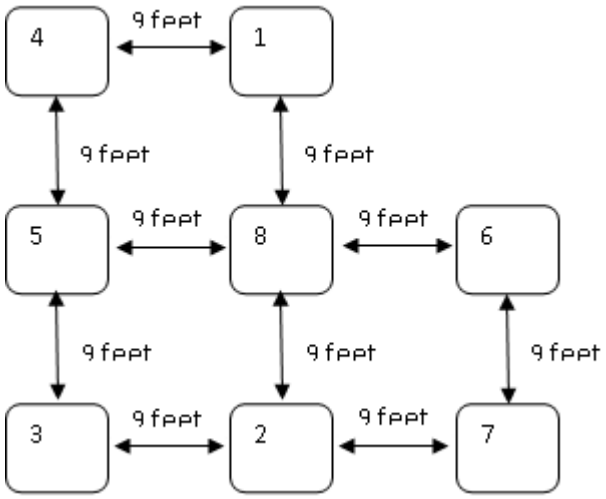

Figure 5-1 Scenario 1

The following is the packet loss observed.

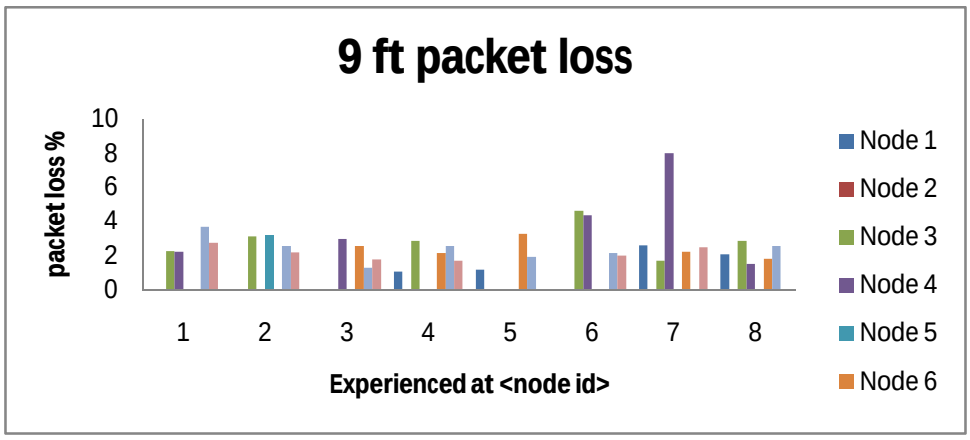

Figure 5-2 Packet Loss for Scenario 1

Scenario – 2

Motes were placed in topology as given in the figure below and the experiment was run indoors with an alive message period of 300 milliseconds.

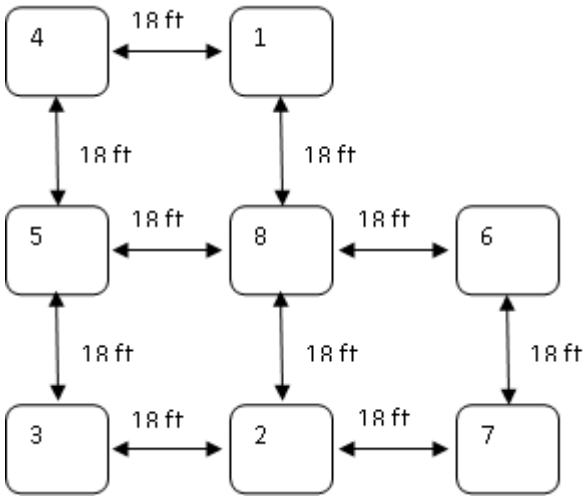

Figure 5-3 Scenario 2

The following is the packet loss observed.

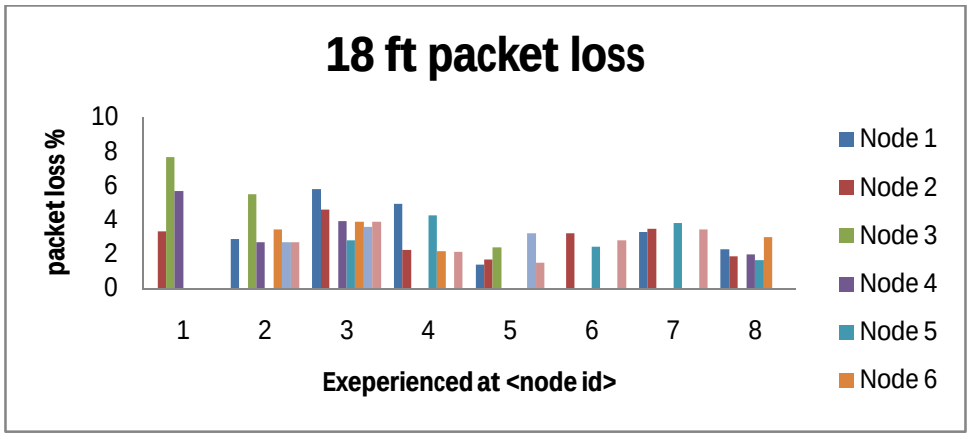

Figure 5-4 Packet Loss for Scenario 2

Scenario – 3

Motes were placed in topology as given in the figure below and the experiment was run indoors with an alive message time period of 300 milliseconds. Nodes 5 and 8 were enclosed in a metal enclosure.

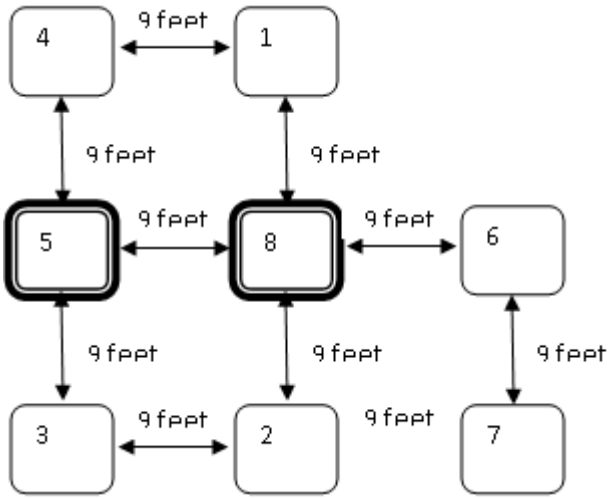

Figure 5-5 Scenario 3

The following is the packet loss observed.

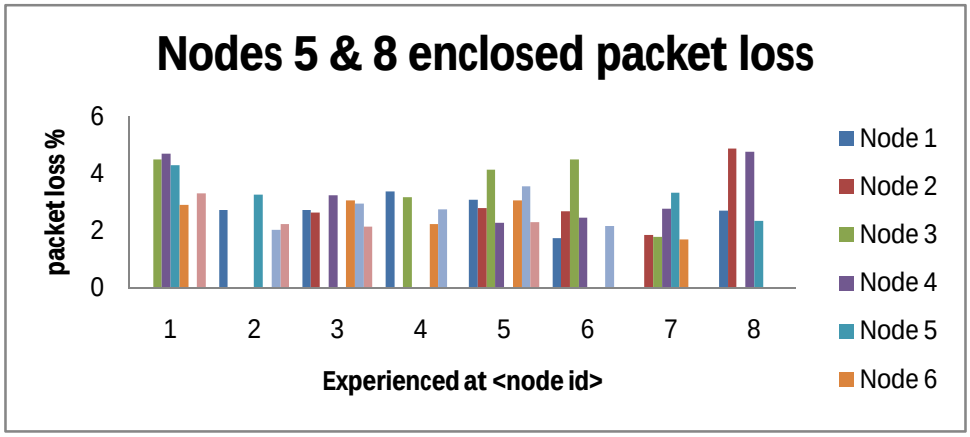

Figure 5-6 Packet Loss for Scenario 3

Scenario – 4

Motes were placed in topology as given in the figure below and the experiment was run outdoors for with an alive message period of 300 milliseconds.

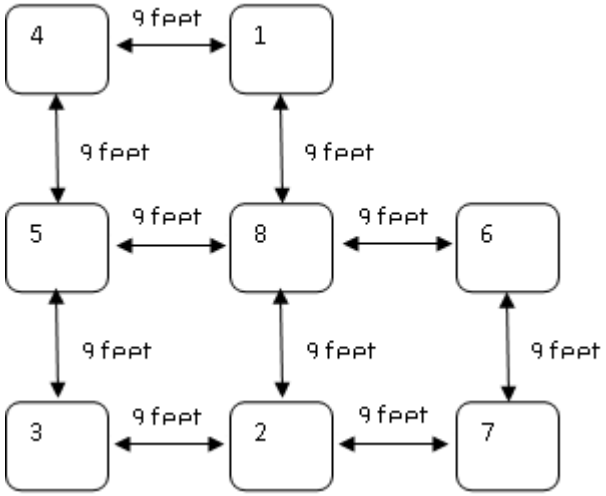

Figure 5-7 Scenario 4

The following is the packet loss observed.

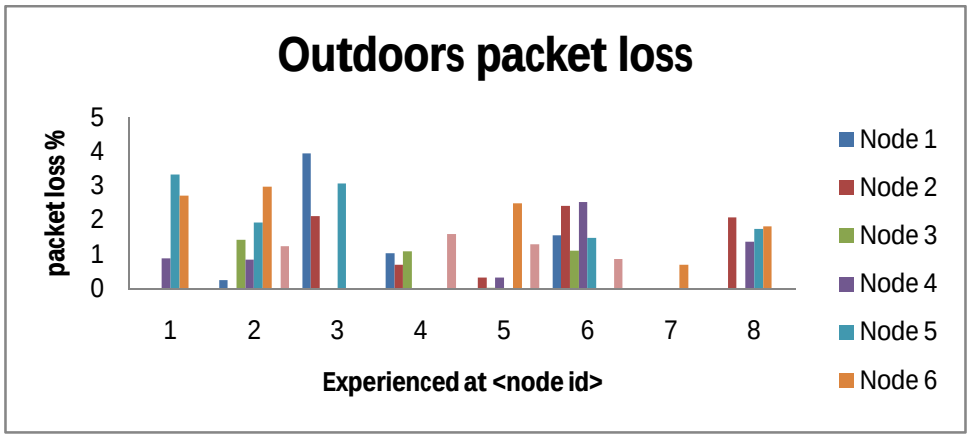

Figure 5-8 Packet Loss for Scenario 4

Scenario – 5

Motes were placed in topology as given in figure below and the experiment was run indoors for with an alive message period of 3000 milliseconds.

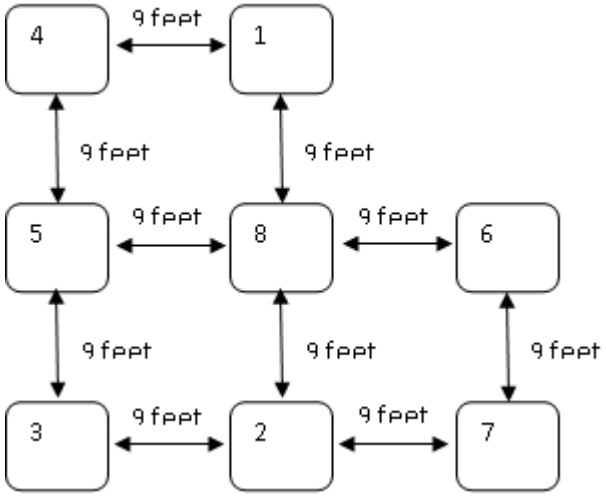

Figure 5-9 Scenario 5

The following is the packet loss observed.

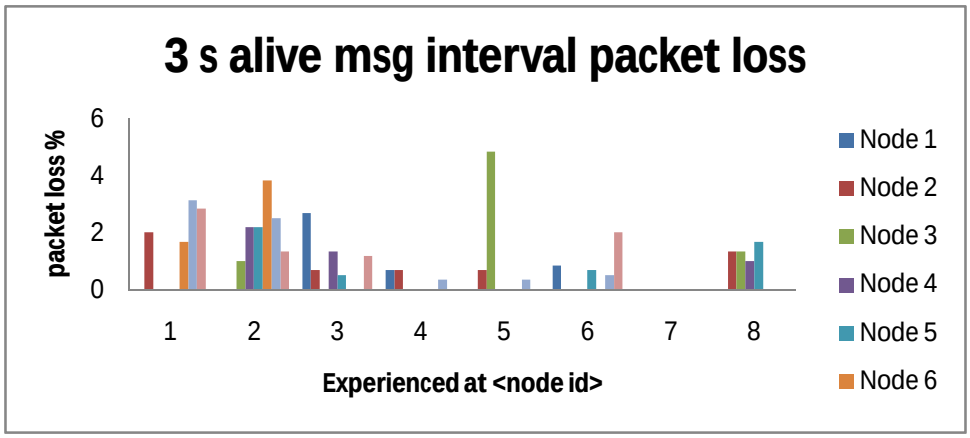

Figure 5-10 Packet Loss for Scenario 5

5.4 Measurement Analysis

	9 ft	18 ft	enclosed nodes 5 & 8	outdoors	3 s alive interval
Range	1.08 to 8	1.38 to 7.66	1.68 to 4.88	0.01 to 3.95	0.33 to 4.83
Range for mid 80% values	1.55 to 3.7	1.88 to 4.93	2.03 to 4.5	0.24 to 2.98	0.5 to 2.67
Mean	2.62	3.27	2.97	1.51	1.58
Mean of mid 80% values	2.44	3.12	2.91	1.46	1.43
Median	2.40	3.12	2.78	1.40	1.33

Figure 5-11 Mean and Median Values

The table above gives a detailed data analysis of measurements taken for all scenarios. From this analysis following conclusions can be drawn:

1. The mean value of packet loss for 9 ft measurements is close to 2.5%. The median value is also close to this figure.
2. The ranges for mid 80% increase for 18 ft measurements as compared to 9 ft measurements. It can be attributed to signal power degradation over a longer distance of 18 ft as compared to 9 ft. This signal power degradation results in more packet loss at various links of SVATS network. Correspondingly the mean values of packet loss (both for entire range and for mid 80% range) also increase for 18 ft measurements. The median value also increases because of reasons mentioned in this bullet.
3. The mid 80% range values for scenario 3 (enclosed nodes 5 & 8) increase when compared to 9 ft measurements. This can be attributed to greater packet loss experienced for packets sent and received by nodes 5 & 8 which are enclosed. Correspondingly, the mean (for both entire range and for the mid 80% range) values

also increase as compared to 9 ft measurements. The median value also increases as compared to 9 ft measurements. However, the mean and median values remain lower than 18 ft measurements. This can be attributed to signal degradation (due to longer link distances between nodes) experienced by all nodes in scenario 2 as compared to more packet loss experienced by only two nodes (5 & 8 which are enclosed) in scenario 3.

4. The packet loss in outdoor environments is the lowest amongst all scenarios and is close to 1.5%. The median value is also close to this value. It is lower than scenario 1 in which the nodes were placed in the same topology and were separated by the same physical distance as in the outdoor environment.
5. The packet loss experienced in scenario 5 is between outdoor and indoor measurements of 9 ft physical distance between nodes. This can be attributed to lesser number of packets in the SVATS network since the alive message interval had been made tenfold to 3 seconds as compared to other scenarios.

5.5 Reliable Message Delivery Requirement in SVATS

This section analyzes various requirements for reliable message delivery in a SVATS network and proposes modifications in SVATS protocols to mitigate the effects of packet loss. .

Nodes in SVATS can get leveled through either of the following mechanisms:

1. *Response to level discovery request sent by it*

Loss of level discovery request does not matter as SVATS retransmits the level discovery request if it does not receive a level discovery response. Loss of level discovery response also does not matter because either the new mote will receive a level discovery response

from some other mote or if it does not hear from any other mote, it will retransmit the level discovery request.

2. *On receiving the level discovery request from the base station*

Loss of individual level discovery requests from base station does not matter as base station periodically retransmits the level discovery request. Thus, if the mote does not receive one of the messages, it would receive a subsequent request.

Thus, for the leveling phase, functionality of SVATS is not affected by packet loss.

A node sends out a join request and gets joined when it receives a join reply from any of the peer nodes. Since the nodes retransmit join requests, loss of either the join request or the join reply does not matter. Thus, for the joining phase, functionality of SVATS is not affected by packet loss.

When a node has received sufficient number of join replies, it implies that there are sufficient numbers of peer nodes which are monitoring alive messages from this node. Here again the node retransmits the join request if it does not receive sufficient number of join replies. Thus, for the monitoring phase, functionality of SVATS is not affected by packet loss.

Loss of individual alive messages does not matter as SVATS has a robust mechanism to conclude theft detection and this mechanism is resilient of occasional packet loss of alive messages. Waiting for multiple alive messages before concluding theft detection and vote solicitation by peer nodes are two aspects of theft detection algorithm because of which it is resilient to occasional packet loss of alive messages. Thus, for the alive message tracking phase, functionality of SVATS is not affected by packet loss.

5.6 Proposed Solution for Reliable Message Delivery

Theft report message (from individual nodes to the base station) is the single most important message in SVATS protocol as it is used to report a theft to base station. Prompt and reliable reporting of theft is the intent of whole SVATS system. Thus, it is imperative that end to end reliable delivery of this message is ensured. Currently, there is no such mechanism in SVATS protocol. This work proposes several mechanisms to ensure end to end reliable delivery of this message.

Theft report message is sent to all neighboring nodes which are one level lesser than node who is sending this message. Thus, the sending node attempts to send this message to the base station through multiple routes by sending it to multiple nodes which are closer to the base station than the sending node. This work proposes an ACK mechanism wherein the base station sends an acknowledgement for the theft report to the node which originally sent the theft report. If the sending node does not receive this acknowledgement it should retransmit the theft report message. This would ensure reliable end to end delivery of the theft report message.

However, the retransmission of theft report message will not help in case of network congestion. To ensure end to end reliable delivery of this message this work proposes to introduce a message priority field in the theft report message. This field will be used by the network layer in the SVATS network to forward this packet through links which it knows are not congested. Also, while dropping packets during congestion, the priority field will ensure that the theft report message receives preferential treatment and is not dropped by nodes which are on route to base station.

The ACK mechanism coupled with the priority field will ensure end to end reliable delivery of the theft report message even in cases of network congestion.

5.7 Conclusion

This work proposes mechanisms for end to end reliable delivery of theft report message even in case of network congestion. Forwarding the theft report message to the base station through different routes significantly increases the probability of theft report message getting delivered to the base station. However, it also increases network congestion and in cases where network is already congested, the flood of message thus created will significantly add to congestion. As future work, mechanisms should be devised which significantly reduce the number of theft report messages that are generated as a result of forwarding this message through multiple routes. At the same time these mechanisms should ensure that the probability of delivering the theft report messages to the base station is as high as in the current mechanism which achieves it by creating a flood of messages. Also future works should take measurements which prove the viability of proposed schemes.

A related aspect of packet loss is detection and avoidance of congestion in SVATS network. Future works can devise a framework which is capable of detecting congestion in a SVATS network and take appropriate measures to correct congestion while at the same time ensuring that the SVATS network remains functional.

Chapter 6

Motion Sensor based movement detection

6.1 Introduction

SVATS relies on tracking of unauthorized movement of a vehicle (i.e. movement without sending a leave message) to detect a potential theft. To ensure the reliability of system it is essential that the theft detection rate is close to 100% and also the false positive rate is as low as possible, desirably 0%. SVATS currently uses an RSSI based schemes for tracking of vehicle movements. RSSI based schemes represents a passive monitoring system since the movement of the stolen vehicle is being tracked by other vehicles. Also, the simulation results of the RSSI based scheme has shown that wireless signal strength, which is the primary criteria being monitored, is highly dependent on environmental signal interferences and other factors such as an object coming between the two vehicles etc. This dependence leaves room for false positives and for missing theft detection instances. This work (jointly with [11]) proposes a motion sensor based scheme for detection of unauthorized vehicle movement. Details of in depth analysis and simulation results of this proposed method can be found as part of [11].

6.2 Accelerometer

An accelerometer is a device which measures the acceleration that it experiences. Since, any static object experiences an acceleration of 1g, any reading of acceleration from the accelerometer needs to be adjusted for it. Accelerometers are capable of detecting even small changes in acceleration. Depending on the granularity of the measurements required, there are

different types of accelerometers available. For application in SVATS, the accelerometer available with MTS310 sensor board suffices.

Accelerometers are already being used for detecting movements for seismic activities, structural and building vibrations, navigation, consumer electronics (in smart phones and gaming consoles etc.) etc.

6.3 Motion Sensor scheme

This work reuses the concept of movement detection of accelerometers which is already being used in other applications of accelerometers. Sensor motes installed in vehicles are proposed to have a motion sensor capability also. Thus, any small movement of the vehicle will be detected by the motion sensors. This movement will be analyzed against the current state of SVATS system. If the movement is an authorized one e.g. after the vehicle has been unlocked, the movement will not be reported as a theft. However, if the movement is an unauthorized one, a theft report will immediately be broadcasted to the neighboring vehicles by the stolen vehicle. The accelerometer could be sensitive to even small vibrations of the static vehicle. Thus, to eliminate the false positives generated because of it, such movements should not be considered as displacement for SVATS tracking purposes.

Since accelerometers are sensitive to smallest of movements, it is essential that movements other than actual physical displacement of the entire vehicle are not considered for SVATS implementation. To eliminate such small movements, “movement” of vehicle should not be concluded from one single measurement of the accelerometer. Instead, some consecutive readings should be considered and if all of them indicate continuous movement, only then an actual physical displacement of the vehicle should be concluded. This would eliminate possibilities like, accidental shaking of vehicle by an individual or due to structural movements of

the building housing the parking, minor seismic activity etc. The environment in which the vehicle is parked might be experiencing small vibrations (e.g. due to other vehicles in the parking lot coming in or out etc). The actual physical displacement of vehicles can be concluded from accelerometer measurements with a high degree of accuracy. Simulation results done as part of [11] indicate the same.

6.4 Comparative Study of Motion Sensor scheme with RSSI based scheme

Detection Rate and False Positives

As mentioned previously, the primary parameter being monitored in a RSSI based scheme is the signal strength. Since the signal strength is highly dependent on external factors such as environmental interference, obstructions between source and sink of signals etc. there is a possibility that a RSSI based scheme could have some false positives. However, the motion sensor scheme (along with all the measures mentioned in the previous section) is a more reliable scheme. Since accelerometers are highly sensitive to motion detection, there is virtually no possibility that an unauthorized movement of the vehicle can go unnoticed. Also, the measures mentioned in the previous section provide enough reliability to eliminate any false positives. Thus, we can conclude that though both schemes can reliably detect unauthorized movement, the motion sensor based scheme has a higher probability of reporting less, if not none at all, false positives.

Speed of Detection

Motion sensor based scheme is an active movement detection scheme since the movement of the node is being monitored at the node itself. On the other hand RSSI based scheme is a passive movement detections scheme since the movement of a particular node is being detected by the neighboring nodes. Due to this and other surrounding logic in RSSI based schemes, they have a slower speed of movement detection as compared to motion sensor based scheme.

Cost

RSSI based scheme does not require any additional component apart from the wireless capability of a mote. However, motion sensor based scheme requires an additional motion sensor. Thus the cost of the motion sensor based scheme is more than of the RSSI based scheme.

A detailed comparison of these schemes is present as part of [11].

Chapter 7

Conclusions and Future Work

7.1 Conclusions

This chapter concludes this work and talks about future work. This work builds upon an already existing car theft system – "Secure Vehicle Anti Theft System – SVATS". It works on improving the commercial viability of the system and also, the experiments conducted vouch for the stability of the system and the realistic response times which makes the system deployable and trust worthy in real parking lot kind of scenarios.

Firstly, a security framework is developed for the system. In the existing implementation, it was fairly simple for an intruder to disarm the system. There was no built in security for the system. All messages exchange; include theft report delivery, happened in the clear. This work implements a key establishment protocol between the nodes. Further, messages are encrypted/decrypted with agreed upon keys. Also, integrity protection of the data is ensured and the potential security attacks to the system are analyzed.

Secondly, a framework is built to interface the sensor network based SVATS network with the GSM network. This allows reporting theft alerts in real time to either the owner of the car or the police station. This is a significant enhancement to the existing system because it makes the system more responsive and allows a quick action once a text message is delivered to the cellular device, alerting the user of the theft.

Thirdly, this work involves running experiments, testing the system, in various topologies, the results of which go a long way in proving the feasibility and commercial viability of the system. The experiment results show the response times are very well acceptable for a real

time system. It also proves that the system is stable and can run for long periods of time without any serious failures and software crashes.

Finally, this work involved upgrading the entire system to MicaZ motes, from Mica2 motes, on which the original prototype was done. This makes the system more reliable and robust, as MicaZ has a more reliable and powerful radio, as compared to the Mica2 [3][4]. This involved porting the code to the MicaZ platform.

Overall, the system works reliably and this work enhances the security of the system.

7.2 Future Work

Monitor and Neighbor List reorganization

Current SVATS protocol updates the monitor and neighbor lists of various nodes only when a node joins the SVATS network. Thus, these lists become static in nature i.e. they do not change with changes in topologies as and when new nodes join or leave the SVATS network. For instance during early hours a car (let's call it car X) enters a parking lot. Since a lot of cars are not parked in the parking lot, the car might be getting monitored by nodes (let's call them cars A, B and C) which are far off from the monitored car. As the day passes by more cars get parked into the parking lot and thus the neighborhood around car X is now densely populated with cars. Due to this change in topology, changes are required in the monitor and neighbor list of cars A, B, C and X. Cars A, B and C should not be monitoring car X now. Instead they should be monitoring other cars which are closer to them as compared to car X. Also, car X should now be monitored by cars which have been parked in its vicinity and thus are closer to car X as compared cars A, B and C. This dynamic reorganization will lead to more optimal pockets of nodes being created. Better tracking can be achieved by this reorganization since tracking will be done by stronger

signals and a pocket will consist of cars in close vicinity only. On the other hand if the parking lot becomes sparsely populated, it will give the parked vehicles an opportunity to be monitored by vehicles which though are not close but still capable of monitoring a lone vehicle.

Here are some initial thoughts on how this reorganization can be implemented. Any node should determine vehicles in its vicinity by measuring the strength of received signals from these nodes. If sufficient readings of high signal strength have been received the node should take a decision whether a better topology is possible as compared to the one present in its monitor and neighbor list. If the node decides to do reorganization, it should send appropriate messages to the concerned nodes so that they too can reorganize nodes at their ends.

Lone vehicle protection

SVATS requires that for any node to enter into monitored state, it should receive a minimum number of replies to the join request. However, there might be cases when a vehicle is parked in an isolated area where it cannot receive the minimum number of join replies required for it to become monitored. Such a vehicle will not be monitored by neighboring nodes and thus will be an easy target for theft. Future works can focus on how to identify such unprotected nodes and suggest mechanisms through which even such nodes can be monitored.

One mechanism of doing this can be assisting the incoming vehicle and suggesting a few parking spots that would be optimum for it from the SVATS protocol point of view. This way, sparse network problems can be avoided.

[38] is a relevant work in this regard. Ideas can be taken from here, as to how to track the empty parking spaces in the parking lot. Also, the new vehicles coming into the parking lot can be assisted, as to what would be optimum place for them to park their car, in terms of the distance between their current position and the parking spot and in terms of the safety of the vehicle. This

could take into account the fact that the vehicle should be monitored by a threshold number of other vehicles to be protected by the SVATS protocol. Thus, the suggested parking spaces could be the spaces where the vehicle will be in signal range of a threshold number of other vehicles. This list could then be ordered by increasing distance of the parking spot from the vehicle's current location.

Network Security

A basic security scheme has been proposed and implemented as a part of this work. However, to make the SVATS system completely secure much needs to be done for the security aspects of the protocol. Current work proposes to derive a secret key for unicast messages using the node ids of the communicating nodes. This key is derived from a bivariate polynomial which is burnt into all the motes. It would be impractical to assume that all parking lots will be using the same bivariate polynomials. Thus, there needs to be a mechanism to have multiple bivariate polynomials available for any node. Alternatively, a node could become aware of a bivariate polynomial when it is entering a parking lot. Future works can concentrate on devising a framework for nodes to have multiple bivariate polynomials and also provide nodes the capability to choose amongst these polynomials when they are communicating with other nodes.

For broadcast messages this work proposes to have a common key which is available with all the nodes. Here again it is impractical to assume that all parking lots will be using the same key for broadcast messages. Future works can concentrate on devising a framework for nodes to have multiple broadcast message keys and also provide nodes the capability to choose amongst these keys when they are broadcasting messages to other nodes.

Also, when a node gets compromised, all the keys and bivariate polynomials stored on the node are compromised. A rekeying mechanism needs to be in place. In case keys like the

broadcast key gets exposed, it can become a major task to re distribute the new key to all the existing nodes. Thus, management of keys and their distribution and revoking can prove to be an interesting are of future work.

User Privacy

Another aspect of future works to concentrate on is the privacy of visited parking lots data for a vehicle. SVATS network of a parking lot have a signature of each vehicle visiting that parking lot. Thus, a collection of data from various parking lots in an area can reveal a history of visited parking lots for a vehicle. While this data could be useful for law enforcement agencies, it also raises concerns for privacy of vehicles.

Reducing the number of messages exchange in voting and theft report

Currently, the number of message exchanged in the SVATS protocol – voting and theft reporting is unnecessarily high. Whenever nodes detect missing alive messages from a particular node, they all detect the missing node simultaneously. All nodes start sending out voting requests and then the network is flooded with vote requests and vote replies. Some simplification can be thought of. Nodes can use some sort of back off and then send vote request only if they haven't heard another vote request regarding the missing alive messages about the same node. Also, all nodes need not initiate the sensing of theft report. A head node can be elected among the group of nodes, which will be responsible for sending the theft report. This will reduce the number of messages in the network significantly. Also, it will cause fewer collisions and less missing packets.

Using the GPS Sensor to enhance SVATS functionality

It was determined with experiments that the GPS sensor that comes with the MTS310 board has a fairly good accuracy, even indoors. Using the GPS sensors in cars could enhance SVATS functionality. All nodes can pass their exact location coordinates to the base station. And the base station could build an exact map of the parking lot. Further enhancements can be thought of, using this GPS sensor. Also, its feasibility can be studied in terms of the cost of the GPS sensor and the additional benefits it provides.

Building Location Data

Another direction of future work for SVATS could be to build the location data of all the existing nodes at the Base Station. Since GPS mentioned in the previous section might turn to be a very expensive option and it requires extra hardware too. Existing SVATS code could be enhanced, so that the nodes can use triangulation technique and the fact that signal strength is inversely proportional to the distance between the nodes. Using this information, the nodes can find their distance relative to each other. The Base Station can build the whole topology of the network using this data from the nodes. This could prove to be a useful piece of information in case of theft reports. With some fixed nodes in the lot, the topology can then be constructed in absolute terms. In case of theft report, the alert being passed to the police or the personnel concerned can then give absolute location where the vehicle was parked, instead of just reporting theft. This could aid in tracking the vehicle down and taking some quick action.

Tracking Mechanism

Once the theft has been reported and the vehicle has started moving, a tracking mechanism needs to be in place to track the movement of the vehicle and know its exact coordinates at various points of time, after the vehicle has been reported as stolen. [11] is a relevant work in this regard. The tracking mechanism is implemented, along with the slave sensor functionality. As a part of the future work, the existing SVATS code can be integrated with the tracking code and the system tested as a whole.

References

[1]. Hui Song, Sencun Zhu and Guohong Cao, “SVATS: A Sensor network based Vehicle Anti-Theft System", IEEE INFOCOMM, April 2008.

[2]. C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro and M. Yung, “Perfectly-Secure Key Distribution for Dynamic Conferences”, Lecture Notes in Computer Science, vol.740, pp. 471–486, 1993.

[3]. Crossbow MICA 2 mote – datasheet available at www.xbow.com/products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf

[4]. Crossbow MICA Z mote – datasheet available at www.xbow.com/products/Product_pdf_files/Wireless_pdf/MICAZ_Datasheet.pdf

[5]. Analog Devices ADXL202 Accelerometer – data sheet available at http://www.analog.com/en/mems-and-sensors/imems-accelerometers/adxl202/

[6]. Crossbow MTS310 Sensor Board data sheet

[7]. C. Karlof, N. Sastry and D. Wagner, “TinySec: A Link Layer Security Architecture for Wireless Sensor Networks”, Proceedings of the 2nd international conference on Embedded networked sensor systems (2004), pp. 162-175..

[8]. GNokii help page available at http://gnokii.org/

[9]. Sencun Zhu, The Pennsylvania State University and Sanjeev Setia, George Mason University And Sushil Jajodia, George Mason University – “LEAP+: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks”.

[10]. G. Lin, G. Noubir, Wireless Security Laboratory, College of Computer Science, Northeastern University – “Low Power DoS Attacks in Data Wireless LANs and Countermeasures”.

[11]. Stolen Vehicle Tracking Using Sensor Network – MS Thesis by Soumya Jain.

[12] .AT Commands Tutorial available at http://www.developershome.com/sms/atCommandsIntro.asp

[13]. Wenyuan Xu, Timothy Wood, Wade Trappe, Yanyong Zhang – "Channel Surfing and Spatial Retreats: Defenses against Wireless Denial of Service".

[14]. Wenyuan Xu, Wade Trappe, Yanyong Zhang and Timothy Wood – "The Feasibility of Launching and Detecting Jamming Attacks in Wireless Networks".

[15] .Wenyuan Xu, Ke Ma, Wade Trappe, and Yanyong Zhang, Rutgers University – "Jamming Sensor Networks: Attack and Defense Strategies".

[16] .Syed Obaid Amin, Muhammad Shoaib Siddiqui and Choong Seon Hong - "Detecting Jamming Attacks in Ubiquitous Sensor Networks".

[17] .Anthony D. Wood, John A. Stankovic, and Gang Zhou – "DEEJAM: Defeating Energy-Efficient Jamming in IEEE 802.15.4-based Wireless Networks".

[18] .Rajani Muraleedharan and Lisa Ann Osadciw – "Jamming Attack Detection and Countermeasures In Wireless Sensor Network Using Ant System".

[19] .Bu_gra Gedik, Member, IEEE, and Ling Liu, Senior Member, IEEE – "Protecting Location Privacy with Personalized k-Anonymity: Architecture and Algorithms".

[20]. W. Zhang and G. Cao, "Group Rekeying for Filtering False Data in Sensor Networks: A Predistribution and Local Collaboration-Based Approach," IEEE INFOCOM, March 2005.

[21]. M. Shao, Y. Yang, S. Zhu, and G. Cao, "Towards Statistically Strong Source Anonymity for Sensor Networks" IEEE Infocom, 2008.

[22]. Chieh-Yih Wan, Shane B. Eisenman, Andrew T. Campbell, "Demo Abstract: CODA + PSFQ + Virtual Sinks = Enabling Technologies for Resilient Sensor Networking", Proceedings of the 2nd international conference on Embedded networked sensor systems.

[23]. Sukun Kim, Rodrigo Fonseca, David Culler, “Reliable Transfer on Wireless Sensor Networks”.

[24]. Seung-Jong Park, Raghupathy Sivakumar. “MobiHoc Poster: Sink-to-Sensors Reliability in Sensor Networks”, Mobile Computing and Communications Review, Volume 7, Number 3.

[25]. Holger Karl, Andreas Willig, “A short survey of wireless sensor networks”.

[26]. Chonggang Wang, Kazem Sohraby1, Bo Li, and Weiwen Tang, “Issues of Transport Control Protocols for Wireless Sensor Networks”.

[27]. Chonggang Wang and Kazem Sohraby, Bo Li, Mahmoud Daneshmand, Yueming Hu, “A Survey of Transport Protocols for Wireless Sensor Networks”, IEEE Network 2006.

[28]. I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci Broadband, “Wireless sensor networks: a survey”, Elsevier Science B.V.

[29]. Justin Jones, Mohammed Atiquzzaman , “Transport Protocols for Wireless Sensor Networks: State-of-the-Art and Future Directions”, International Journal of Distributed Sensor Networks.

[30]. Stann, F. and Heidemann, J., “RMST: Reliable Data Transport in Sensor Networks”. IEEE International Workshop on Sensor Net Protocols and Applications.

[31].Chieh-Yih Wan, Andrew T. Campbell, Lakshman Krishnamurthy , **“**PSFQ: a reliable transport protocol for wireless sensor networks**”,** International Workshop on Wireless Sensor Networks and Applications.

[32]. Yogesh Sankarasubramaniam , Özgür B. Akan, Ian F. Akyildiz, “ESRT: event-to-sink reliable transport in wireless sensor networks”**,** International Symposium on Mobile Ad Hoc Networking & Computing.

[33]. Eugenia Giancoli, Filippe Jabour, Aloysio Pedroza, “Collaborative Transport Control Protocol for Sensor Networks”, ICCEE Proceedings of the 2008 International Conference on Computer and Electrical Engineering.

[34]. Faisal Karim Shaikh, Abdelmajid Khelil, Neeraj Suri, “A comparative study of data transport protocols in wireless sensor networks”, WOWMOM , Proceedings of the 2008 International Symposium on a World of Wireless, Mobile and Multimedia Networks - Volume 00.

[35]. Kaushalya Premadasa , Bjorn Landfeldt, “Dynamically Re-Configurable Transport Protocols for Low-End Sensors in Wireless Networks”, Proceedings of the Fifth Annual Conference on Communication Networks and Services Research.

[36]. Christopher Ho, Katia Obraczka, Gene Tsudik, Kumar Viswanath, “Flooding for reliable multicast in multi-hop ad hoc networks”, Proceedings of the 3rd international workshop on Discrete algorithms and methods for mobile computing and communications.

[37]. Elena Pagani, Gian Paolo Rossi, “Reliable broadcast in mobile multihop packet networks”, Proceedings of the 3rd annual ACM/IEEE international conference on Mobile computing and networking.

[38]. Rongxing Lu, Xiaodong Lin, Haojin Zhu, Xuemin (Sherman) Shen, “SPARK: A New VANET-based Smart Parking Scheme for Large Parking Lots”, IEEE INFOCOM 2009.

[39]. Kebin Liu, Mo Li, Yunhao Liu, Minglu Li, Zhongwen Guo, Feng Hong, “Passive diagnosis for wireless sensor networks”, Proceedings of the 6th ACM conference on Embedded network sensor systems.