P-3540

SECURE DATA COLLECTION IN WSN USING RANDOMIZED DISPERSIVE ROUTES

By

V.RAJESWARI	Reg No: 0710107075
S.SHIVASHANKARI	Reg No: 0710107095
M.SWEETHA	Reg No: 0710107105
S.THENMOZHI	Reg No: 0710107107

of

KUMARAGURU COLLEGE OF TECHNOLOGY, COIMBATORE - 641 049.

(An Autonomous Institution affiliated to Anna University of Technology, Coimbatore)

A PROJECT REPORT

Submitted to the

FACULTY OF ELECTRONICS AND COMMUNICATION ENGINEERING

In partial fulfillment of the requirements for the award of the degree

of

BACHELOR OF ENGINEERING

APRIL 2011

BONAFIDE CERTIFICATE

Certified that this project report entitled **"Secure Data collection in WSN Using Randomized Dispersive Routes"** is the bonafide work of **Ms.V.Rajeswari** [Reg. no. 0710107075], **Ms.S.Shivashankari** [Reg. no. 0710107095], **Ms.M.Sweetha** [Reg. no. 0710107105], and **Ms.S.Thenmozhi [Reg. no. 0710107107]** who carried out the research under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

[illegible] 18/4/11
Project Guide

Mrs.M.Alagumeenaakshi

[illegible]
Head of the Department

Dr. Rajeswari Mariappan

The candidate with university Register no. 0710107075, 0710107095,0710107105,0710107107 is examined by us in the project viva-voce examination held on ..18.-4.-11...

[illegible] 18/4/11
Internal Examiner

[illegible]
External Examiner

ABSTRACT

Security is a very important issue in any wireless network environment. Wireless Sensor Networks is usually consisting of huge number of limited sensor devices which are communicated over the wireless media. Hence sensor devices are exposed to various kinds of attacks such as denial of service attacks (DOS), flooding, node replication, compromised node attacks (CN) at different layers of communication. Therefore, security in WSNs is a challenging task. Of these attacks DOS and CN are the key attacks of the proposed work. Many secure multipath routing protocols are introduced which are vulnerable to these attacks because of their deterministic nature. Hence a mechanism that generates randomized multipath routes is proposed. Under this mechanism, the routes are also highly dispersive and energy-efficient, making them quite capable of bypassing black holes at low energy cost. Extensive simulations are conducted to verify the validity of the mechanism.

ACKNOWLEDGEMENT

First I would like to express my praise and gratitude to the Lord, who has showered his grace and blessing enabling me to complete this project in an excellent manner.

I express my sincere thanks to our beloved Director **Dr.J.Shanmugam**, Kumaraguru College of Technology, for his kind support and for providing necessary facilities to carry out the work.

I express my sincere thanks to our beloved Principal **Dr.S.Ramachandran**, Kumaraguru College of Technology, who encouraged me in each and every steps of the project work.

I would like to express my deep sense of gratitude to our HOD, the ever active **Dr.Rajeswari Mariappan**, Department of Electronics and Communication Engineering, for her valuable suggestions and encouragement which paved way for the successful completion of the project work.

In particular, I wish to thank with everlasting gratitude to the project coordinator **Mrs.K.Kavitha.,** Assistant Professor-SRG , Department of Electronics and Communication Engineering, for her expert counseling and guidance to make this project to a great deal of success.

I am greatly privileged to express my heartfelt thanks to my project guide **Mrs.M.Alagumeenaakshi.,** Assistant Professor, Department of Electronics and Communication Engineering, throughout the course of this project work and I wish to convey my deep sense of gratitude to all the teaching and non-teaching staffs of ECE Department for their help and cooperation.

Finally, I thank my parents and my family members for giving me the moral support and abundant blessings in all of my activities and my dear friends who helped me to endure my difficult times with their unfailing support and warm wishes.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

WSN	Wireless Sensor Networks
CN	Compromised Node
DOS	Denial of Services
PRP	Purely Random Propagation
SQL	Structured Query language
ODBC	Open Database Connectivity
JDBC	Java Database Connectivity
VM	virtual Machine
API	Application Programming Interface
GUI	Graphical User Interface
TCP	Transmission Control Protocol
UDP	User Data gram Protocol
IP	Internet Protocol

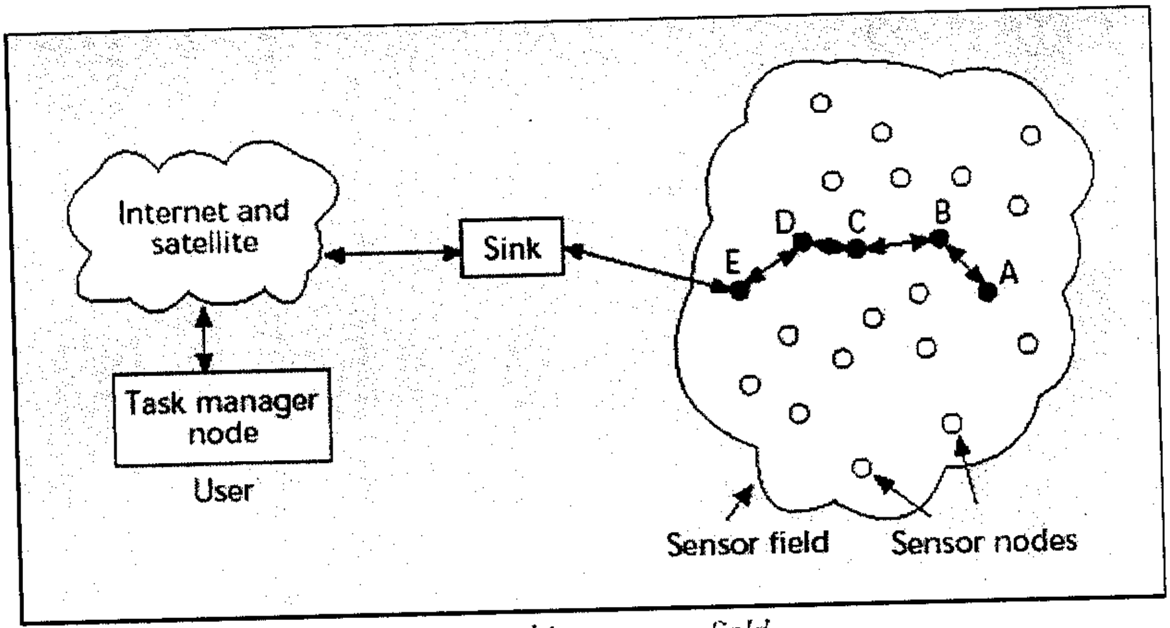
CHAPTER 1
INTRODUCTION

Radio communications are expensive in terms of energy consumption. In wireless sensor networks (WSNs), where nodes might be very limited in resources, it is fundamental that communication overhead be controlled. An interesting approach to achieve such an objective is to perform data aggregation, where relaying nodes exploit the distributed nature of the network and perform in network processing. That is, multiple readings from various sensors are merged into smaller messages as they are conveyed toward the base station.

The Wireless Sensor Network (WSN) system is built on an IEEE 802.15.4 wireless mesh network. The 802.15.4 radio in each WSN device provides for low-power communication of measurement data across a large network of devices. WSN software builds on top of that to provide network configuration and reliable communication from the host PC or Programmable Automation Controller. A wireless sensor network (WSN) is a wireless network consisting of spatially distributed autonomous devices that use sensors to monitor physical or environmental conditions. These autonomous sensor nodes can be densely deployed either inside the phenomenon or very close to it. The position of sensor nodes need not be predetermined. This allows random deployment inaccessible terrains or disaster relief operations. The unique feature of sensor networks is the cooperative effort of sensor nodes. Instead of sending the raw data to the nodes responsible for the fusion, they use their processing abilities to locally carry out simple computations and transmit only the required and partially processed data.

1.1 WSN ARCHITECTURE

The sensor nodes are usually scattered in a sensor field. Each of these scattered sensor nodes has the capabilities to collect data and route data back to the sink. Data are routed back to the sink by a multi hop infrastructure less architecture through the sink. The sink may communicate with the task manager node via Internet or satellite.



■ Figure 1. *Sensor nodes scattered in a sensor field.*

1.1.1 HARDWARE CONSTRAINTS:

A sensor node is made up of four basic components, a sensing unit, a processing unit, a transceiver unit, and a power unit. They may also have additional application-dependent components such as a location finding system, power generator, and mobilizer. Sensing units are usually composed of two subunits: sensors and analog-to-digital converters (ADCs).

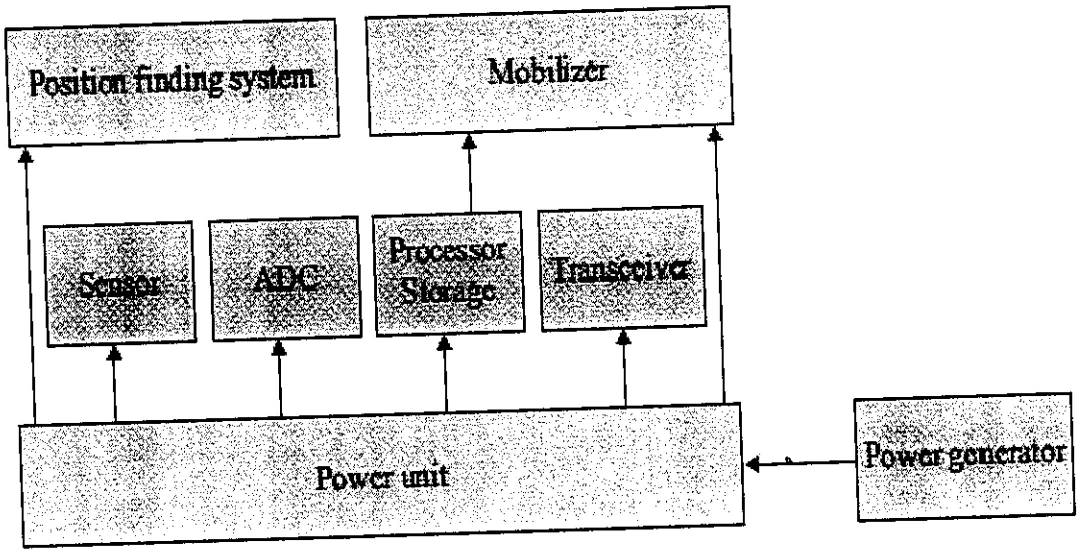


FIGURE: 1.2 HARWARE CONSTRAINS

The analog signals produced by the sensors based on the observed phenomenon are converted to digital signals by the ADC, and then fed into the processing unit. The processing unit, which is generally associated with a small storage unit, manages the procedures that make the sensor node collaborate with the other nodes to carry out the assigned sensing tasks. A transceiver unit connects the node to the network. One of the most important components of a sensor node is the power unit. Power units may be supported by power scavenging units such as solar cells.

CHAPTER 2
SECURITY IN WSN

WSN is mostly used for gathering application specific information from the surrounding environment, it is highly essential to protect the sensitive data from unauthorized access. WSNs are vulnerable to security attacks due to the broadcast nature of radio transmission. Sensor nodes may also be physically captured or destroyed by the enemies. The uses of sensor network in various applications emphasis on secure routing. Various protocols are proposed for routing and data gathering but none of them are designed with security as a goal.

2.1 Security requirements

The resource limitation of sensor networks poses great challenges for security. A sensor network is a special type of Ad hoc network. So it shares some common property as computer network. The security requirements of a wireless sensor network can be classified as follows:

• **Authentication**: As WSN communicates sensitive data which helps in many important decisions making. The receiver needs to ensure that the data used in any decision-making process originates from the correct source. Similarly, authentication is necessary during exchange of control information in the network.

• **Integrity**: Data in transit can be changed by the adversaries. Data loss or damage can even occur without the presence of a malicious node due to the harsh communication environment. Data integrity is to ensure that information is not changed in transit, either due to malicious intent or by accident.

• **Data Confidentiality**: Applications like surveillance of information, industrial secrets and key distribution need to rely on confidentiality. The standard approach for keeping confidentiality is through the use of encryption.

• **Data Freshness**: Even if confidentiality and data integrity are assured, we also need to ensure the freshness of each message. Data freshness suggests that the data is recent, and it ensures that no old messages have been replayed. To ensure that no old messages replayed a time stamp can be added to the packet.

• **Availability**: Sensor nodes may run out of battery power due to excess computation or communication and become unavailable. It may happen that an attacker may jam communication to make sensor(s) unavailable. The requirement of security not only affects the operation of the network, but also is highly important in maintaining the availability of the network.

• **Self-Organization**: A wireless sensor is independent and flexible enough to be Self-organizing and self-healing according to different hassle environments. Due to random deployment of nodes no fixed infrastructure is available for WSN network management. Distributed sensor networks must self-organize to support multihop routing. They must also self organize to conduct key management and building trust relation among sensors.

- **Time Synchronization**: Most sensor network applications rely on some form of time synchronization. In order to conserve power, an individual sensor's radio may be turned off periodically.

- **Secure Localization**: The sensor network often needs location information accurately and automatically. However, an attacker can easily manipulate non secured location information by reporting false signal strengths and replaying signals, etc

2.2 THREAT MODELS

There are two types of threat models in WSN. They are internal threats and external threats.

External threats: They cause passive eavesdropping on data transmissions, as well as can extend to injecting bogus data into the network to consume network resources and raise Denial of Service attack.

Internal threats: It is an authorized participant in the sensor network which has gone hostile. Insider attacks may be mounted by either compromised sensor nodes running malicious code or adversaries who have stolen the key material, code, and data from legitimate nodes and who then use one or more laptop-class devices to attack the network.

2.3 LAYERING-BASED ATTACKS AND POSSIBLE SECURITY APPROACH

Various security threats of WSN in different layers are:

LAYERS	ATTACKS	SECURITY APPROACH
Physical Layer	Jamming and tampering	Use spread-spectrum techniques and MAC layer admission control mechanisms
Data Link layer	jamming and collision	Use error correcting codes and spread spectrum Techniques
Network Layer	Packet drop, selective forwarding, sink hole attack, Worm hole attack	Authentication, encryption
Transport Layer	Flooding	Limiting connection numbers, client puzzles
Application Layer	Clone attack	Unique pair-wise keys

TABLE 2.3.1 VARIOUS SECURITYATTACKS IN WSN

JAMMING

Jamming attack is done by a device which can partially or entirely disrupt a node's signal. An adversary can cause jamming of the entire network by employing k randomly distributed jamming nodes, paralyzing the N nodes where k is much lesser than N .The parameters such as signal strength of a jammer, the location and the type influence the performance of the network and each jammer has different effect on the node.

COLLISION

In a network if one node is transmitting, other nodes listen, and wait for the transmission to end, before they start transmitting themselves. But if two nodes start transmitting at the same time then the collision occurs. Adversaries need only to induce a collision in one octet of transmission to disrupt an entire packet which would cause a checksum mismatch at some other.

BLACK-HOLE ATTACK

The black hole positions a node in range of the sink and attracts the entire traffic to be routed through it by advertising itself as the shortest route. The adversary drops packets coming from specific sources in the network. This attack can isolate certain nodes from the base station and creates a discontinuity in network connectivity.

SELECTIVE FORWARDING ATTACK

Multi-hop mode of communication is commonly preferred in wireless sensor network data gathering protocols. Multi-hop networks assume that participating nodes will faithfully forward and receive messages. However a malicious node may refuse to forward certain messages and simply drop them, ensuring that they are not propagated any further. This attack can be detected if packet sequence numbers are checked properly and continuously in a conjunction free network. Addition of data packet sequence number in packet header can reduce this attack.

SINKHOLE ATTACK

In sinkhole attack, the adversary tries to attract nearly all the traffic from a particular area through a compromised node. A compromised node which is placed at the centre of some area creates a large "sphere of influence", attracting all traffic destined for a base station from the sensor nodes. The attacker targets a place to create sinkhole where it can attract the most traffic, possibly closer to the base station so that the malicious node could be perceived as a base station.

WORM HOLE ATTACK

In this attack an adversary could convince nodes who would normally be multiple hops from a base station that they are only one or two hops away via the wormhole. The simplest case of this attack is to have a malicious node forwarding data between two legitimate nodes. Wormholes often convince distant nodes that they are neighbours, leading to quick exhaustion of their energy resources. An adversary situated close to a base station may be able to completely disrupt routing by creating a well-placed wormhole.

FLOODING

An attacker can cause flood attack by advertising a very high quality route to the base station. So, every node in the network could cause a large number of nodes to attempt to use this route thereby sending the legitimate packets beyond the actual destination. Flooding can be as simple as sending many connection requests to a susceptible node. In this case, resources must be allocated to handle the connection request. Eventually a node's resources will be exhausted, thus rendering the node useless.

CLONE ATTACK

In clone attack, an attacker seeks to add a node to an existing sensor network by copying the node of an existing sensor node. A node replicated in this fashion can severely disrupt a sensor network's performance, packets can be corrupted or even misrouted. This can result in a disconnected network, false sensor readings, etc. By inserting the replicated nodes at specific network points, the attacker could easily manipulate a specific segment of the network, perhaps by disconnecting it altogether.

Here DOS include Jamming in physical layer, compromised node attack like flooding in transport layer which are keys attacks in WSN.

COMPROMISE NODE ATTACK

When an adversary physically compromises a subset of nodes to eavesdrop information then it is called DOS attack.

DENIAL OF SERVICE ATTACK

The adversary interferes with their normal operation of a network actively disrupting, changing or even paralysing the functionality of subset of nodes.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

The existing systems that were proposed to enhance security in WSN are as follows:-

- SPREAD algorithm
- Dijkstra algorithm
- Distributed Bound algorithm
- Lex-Control algorithm
- Gossiping algorithm

SPREAD ALGORITHM

Scalable Platform for Reliable and Efficient Automated Disrtibution algorithm attempts to find multiple most-secure and node-disjoint paths. The security of a path is defined as the likelihood of node compromise along that path, and is labeled as the weight in path selection.

DIJKSTRA ALGORITHM

Dijkstra algorithm is used to iteratively find the top- K most secure node-disjoint paths. It calculates the shortest path between two points on a network using graphs made up of nodes and arcs.

BOUND-CONTROL AND LEX-CONTROL ALGORITHMS

Bound –control and Lex-control algorithm computes multiple path, in such a way that the performance degradation (e.g., throughput loss) is minimized when a single-link attack or a multi-link attack happens, respectively

GOSSIPING ALGORITHM

To reduce unnecessary retransmissions and improve energy efficiency, the Gossiping algorithm was proposed as a form of controlled flooding, whereby a node retransmits packets according to a pre-assigned probability.

Parametric Gossiping was proposed in to overcome the percolation behavior by relating a node's retransmission probability to its hop count from either the destination or the source. A special form of Gossiping is the Wanderer algorithm, whereby a node retransmits the packet to one randomly picked neighbor.

3.1.1Disadvantages

- Existing randomized multi-path routing algorithms in WSNs have not been designed with security considerations in mind, largely due to their low energy efficiency.

- Multi-path routing mechanism, Gossiping algorithm has a percolation behavior, in that for a given retransmission probability, either very few nodes receive the packet, or almost all nodes receive it.

- The Wanderer algorithm has poor energy performance, because it results in long paths.

3.2 PROPOSED SYSTEM

One remedial solution to resolve DOS and CN attacks is to exploit the network's routing functionality. Specifically, if the locations of the black holes are known a priori, then data can be delivered over paths that circumvent these holes, whenever possible. In practice, due to the difficulty of acquiring such location information, the above idea is implemented in a probabilistic manner, typically through a two-step process.

1. Packet splitting

The packet is broken into M shares using a (T,M) threshold secret sharing mechanism such as the Shamir's algorithm. The original information can be recovered from a combination of at least T shares, but no information can be guessed from less than T shares.

2. Routing

Multiple routes from the source to the destination are computed according to some multipath routing algorithm. These routes are node-disjoint or maximally node-disjoint subject to certain constraints such as min-hop routes. The M shares are then distributed over these routes and delivered to the destination. As long as at least M-T+1 (or T) shares bypass the compromised or jammed nodes, the adversary cannot acquire or deny the delivery of the original packet. But three security problems exist in the above counter-attack approach.

1. This approach is no longer valid if the adversary can selectively compromise or jam nodes. This is because the route computation in the above multipath routing algorithms is deterministic in the sense that for a given topology and given source and destination nodes, the same sets of routes are always computed by the routing algorithm.

As a result, once the routing algorithm becomes known to the adversary through memory interrogation of the compromised node the adversary can compute the set of routes for any given source and destination.

Then the adversary can pinpoint to one particular node in each route and compromise or jam these nodes. Such an attack can intercept all shares of the Information, rendering the above counter-attack approaches ineffective.

2. Very few node-disjoint routes can be found when the node density is moderate and the source and destination nodes are several hops apart.

3. The set of routes are computed under certain constraints, the routes may not be spatially dispersive enough to circumvent a moderate-size black hole.

Thus the solution is proposed such that instead of selecting paths from a pre-computed set of routes, it computes multiple paths in a randomized way such that each time an information packet needs to be sent, set of routes taken by various shares of different packets keep changing over time. This can overcome the black holes formed by Compromised-node and denial-of-service attacks

As a result, a large number of routes can be potentially generated for each source and destination. To intercept different packets, the adversary has to compromise or jam all possible routes from the source to the destination, which is practically not possible.

So even if the routing algorithm becomes known to the adversary, the adversary still cannot pinpoint the routes traversed by each packet. Because nodes are randomly generated they may no longer be node-disjoint.

Besides randomness, the routes generated by these mechanisms are also highly dispersive and energy-efficient, making them quite capable of bypassing black holes at low energy cost.

Due to security considerations the routes computed are implemented in a distributed way, such that the final route aggregate decision of all the nodes participate in the route selection Extensive simulations are conducted to verify the validity of the mechanism.

3.2.1 Advantages:

- Provides highly dispersive random routes at low energy cost without generating extra copies of secrete shares.
- If the routing algorithm becomes known to the adversary, the adversary still cannot pinpoint the routes traversed by each packet
- Energy efficient

CHAPTER 4

SYSTEM REQUIREMENTS

4.1 HARDWARE REQUIREMENTS

Processor	: Any Processor above 500 MHz
RAM	: 128Mb.
Hard Disk	: 10 GB.
Compact Disk	: 650 Mb.
Input device	: Standard Keyboard and Mouse.
Output device	: VGA and High Resolution Monitor

4.2 SOFTWARE REQUIREMENTS

Operating System	: Windows Family.
Language	: JDK 1.5
Data Bases	: Microsoft SQL Server
Front End	: Java Swing

CHAPTER 5

SOFTWARE AND TECHNOLOGIES DESCRIPTION

5.1 JAVA TECHNOLOGY

Java technology is both a programming language and a platform.

5.2 THE JAVA PROGRAMMING LANGUAGE

The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

- Simple
- Architecture neutral
- Object oriented
- Portable
- Distributed
- High performance
- Interpreted
- Multithreaded
- Robust
- Dynamic
- Secure

With most programming languages, either compilation or interpretation is done to a program so that it can run on a computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first program is translated into an intermediate language called Java byte codes. Java byte codes are platform-independent codes interpreted by the interpreter on the Java platform.

The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.

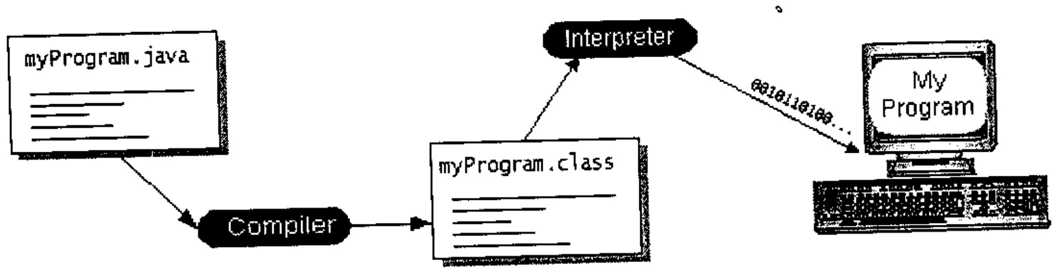


FIGURE 5.2.1 WORKING OF JAVA

The Java bytecodes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it's a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java bytecodes help make "write once, run anywhere" possible. The byte codes can be complied on any platform that has a Java compiler. The bytecodes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.

5.3 THE JAVA PLATFORM

A platform is the hardware or software environment in which a program runs. The most popular platforms are Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

- The Java Virtual Machine (Java VM)
- The Java Application Programming Interface (Java API)

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as packages..

The following figure depicts a program that's running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.

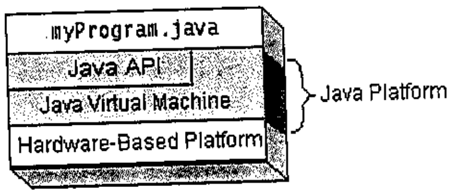


FIGURE 5.3.1 JAVA PLATFORM

Native code is code that after compliation, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring performance close to that of native code without threatening portability.

5.3.1 FEATURES OF JAVA:

The most common types of programs written in the Java programming language are applets and applications. An applet is a program that adheres to certain conventions that allow it to run within a Java-enabled browser.

However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

An application is a standalone program that runs directly on the Java platform. A special kind of application known as a server serves and supports clients on a network. Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a servlet. A servlet can almost be thought of as an applet that runs on the server side. Java Servlets are a popular choice for building interactive web applications, replacing the use of CGI scripts. Servlets are similar to applets in that they are runtime extensions of applications. Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server.

How does the API support all these kinds of programs? It does so with packages of software components that provides a wide range of functionality. Every full implementation of the Java platform gives you the following features:

- **The essentials**: Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.

- **Applets**: The set of conventions used by applets.

- **Networking**: URLs, TCP (Transmission Control Protocol), UDP (User Data gram Protocol) sockets, and IP (Internet Protocol) addresses.

- **Internationalization**: Help for writing programs that can be localized for users worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.

- **Security**: Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.

- **Software components**: Known as JavaBeans ™, can plug into existing component architectures.

- **Object serialization**: Allows lightweight persistence and communication via Remote Method Invocation (RMI).

- **Java Database Connectivity (JDBC™)**: Provides uniform access to a wide range of relational databases.

The Java platform also has APIs for 2D and 3D graphics, accessibility, servers, collaboration, telephony, speech, animation, and more. The following figure depicts what is included in the Java 2 SDK.

5.4 ODBC

Microsoft Open Database Connectivity is a standard programming interface for application developers and database systems providers. Before ODBC became a de facto standard for Windows programs to interface with database systems, programmers had to use proprietary languages for each database they wanted to connect to. Now, ODBC has made the choice of the database system almost irrelevant from a coding perspective, which is as it should be.

Application developers have much more important things to worry about than the syntax that is needed to port their program from one database to another when business needs suddenly change.

Through the ODBC Administrator in Control Panel, a particular database that is associated with a data source that an ODBC application program is written to use. Think of an ODBC data source as a door with a name on it. Each door will lead you to a particular database. For example, the data source named Sales Figures might be a SQL Server database, whereas the Accounts Payable data source could refer to an Access database. The physical database referred to by a data source can reside anywhere on the LAN.

The ODBC system files are not installed on your system by Windows 95. Rather, they are installed when you setup a separate database application, such as SQL Server Client or Visual Basic 4.0. When the ODBC icon is installed in Control Panel, it uses a file called ODBCINST.DLL. It is also possible to administer your ODBC data sources through a stand-alone program called ODBCADM.EXE.

From a programming perspective, the beauty of ODBC is that the application can be written to use the same set of function calls to interface with any data source, regardless of the database vendor. The source code of the application doesn't change whether it talks to Oracle or SQL Server. We only mention these two as an example. There are ODBC drivers available for several dozen popular database systems.

Even Excel spreadsheets and plain text files can be turned into data sources. The operating system uses the Registry information written by ODBC Administrator to determine which low-level ODBC drivers are needed to talk to the data source (such as the interface to Oracle or SQL Server). The loading of the ODBC drivers is transparent to the ODBC application program. In a client/server environment, the ODBC API even handles many of the network issues for the application programmer.

The advantages of this scheme are so numerous that you are probably thinking there must be some catch. The only disadvantage of ODBC is that it isn't as efficient as talking directly to the native database interface. ODBC has had many detractors make the charge that it is too slow. Microsoft has always claimed that the critical factor in performance is the quality of the driver software that is used. In our humble opinion, this is true. The availability of good ODBC drivers has improved a great deal recently. And anyway, the criticism about performance is somewhat analogous to those who said that compilers would never match the speed of pure assembly language. Maybe not, but the compiler (or ODBC) gives you the opportunity to write cleaner programs, which means you finish sooner. Meanwhile, computers get faster every year.

5.5 JDBC

In an effort to set an independent database standard API for Java, Sun Microsystems developed Java Database Connectivity, or JDBC. JDBC offers a generic SQL database access mechanism that provides a consistent interface to a variety of RDBMSs. This consistent interface is achieved through the use of "plug-in" database connectivity modules, or drivers.

If a database vendor wishes to have JDBC support, he or she must provide the driver for each platform that the database and Java run on.

To gain a wider acceptance of JDBC, Sun based JDBC's framework on ODBC. As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a completely new connectivity solution.

JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after.

5.5.1JDBC GOALS

Few software packages are designed without goals in mind. JDBC is one that, because of its many goals, drove the development of the API. These goals, in conjunction with early reviewer feedback, have finalized the JDBC class library into a solid framework for building database applications in Java.

The goals that were set for JDBC are important. They will give you some insight as to why certain classes and functionalities behave the way they do.
The six design goals for JDBC are as follows:

1. SQL Level API

The designers felt that their main goal was to define a SQL interface for Java. Although not the lowest database interface level possible, it is at a low enough level for higher-level tools and APIs to be created. Conversely, it is at a high enough level for application programmers to use it confidently. Attaining this goal allows for future tool vendors to "generate" JDBC code and to hide many of JDBC's complexities from the end user.

2. SQL Conformance

SQL syntax varies as you move from database vendor to database vendor. In an effort to support a wide variety of vendors, JDBC will allow any query statement to be passed through it to the underlying database driver. This allows the connectivity module to handle non-standard functionality in a manner that is suitable for its users.

3. JDBC must be implemental on top of common database interfaces

The JDBC SQL API must "sit" on top of other common SQL level APIs. This goal allows JDBC to use existing ODBC level drivers by the use of a software interface. This interface would translate JDBC calls to ODBC and vice versa.

4. Provide a Java interface that is consistent with the rest of the Java system

Because of Java's acceptance in the user community thus far, the designers feel that they should not stray from the current design of the core Java system.

5. Keep it simple

This goal probably appears in all software design goal listings. JDBC is no exception. Sun felt that the design of JDBC should be very simple, allowing for only one method of completing a task per mechanism. Allowing duplicate functionality only serves to confuse the users of the API.

6. Use strong, static typing wherever possible

Strong typing allows for more error checking to be done at compile time; also, less error appear at runtime.

CHAPTER 6

PROPOSED SYSTEM ARCHITECTURE

System architecture mainly includes topology creation, randomized multipath routing and secret delivery of packets, in order to enhance data collection using randomized dispersive routes.

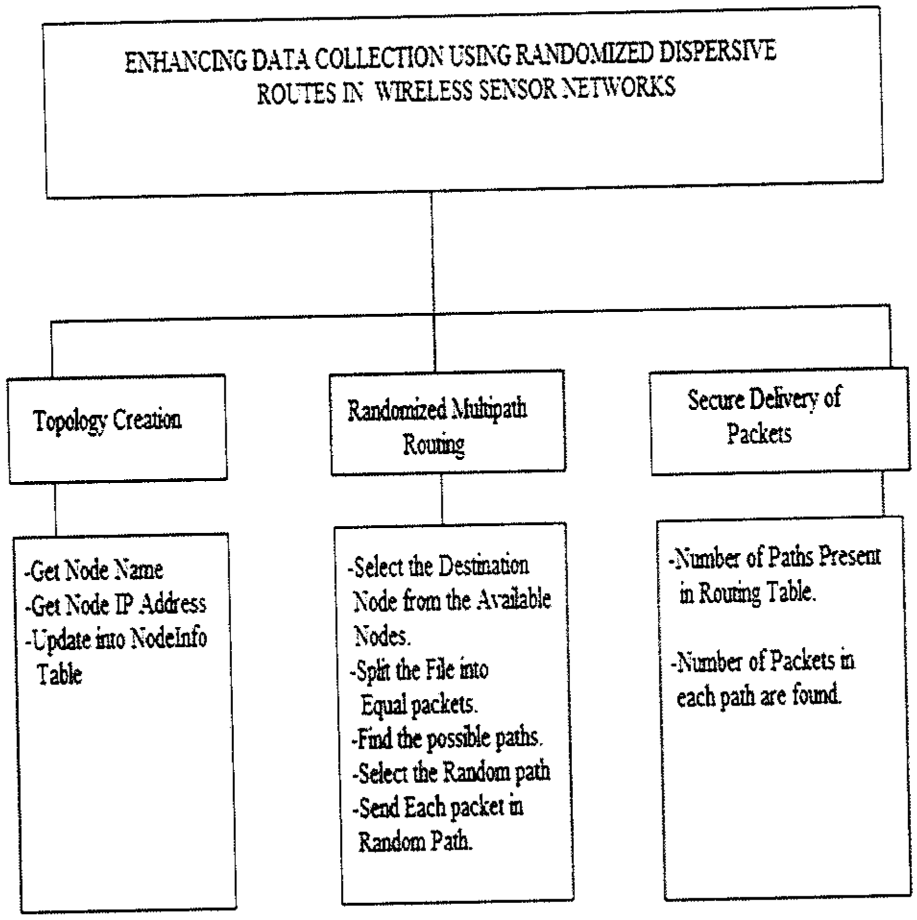

ENHANCING DATA COLLECTION USING RANDOMIZED DISPERSIVE ROUTES IN WIRELESS SENSOR NETWORKS
Topology Creation
Randomized Multipath Routing
Secure Delivery of Packets
-Get Node Name
-Get Node IP Address
-Update into NodeInfo Table
-Select the Destination Node from the Available Nodes.
-Split the File into Equal packets.
-Find the possible paths.
-Select the Random path
-Send Each packet in Random Path.
-Number of Paths Present in Routing Table.
-Number of Packets in each path are found.

FIGURE6.1 SYSTEM ARCHITECTURE

6.1 SYSTEM DESCRIPTION

6.1.1Topology Creation

The name of the node and IP address of the node are obtained to create the topology. That information is stored in node information table. Topology is constructed by getting the names of the nodes and the connections among the nodes as input from the user. While getting each of the nodes, their associated port and IP address is also obtained. For successive nodes, the node to which it should be connected is also accepted from the user. While adding nodes, comparison will be done so that there would be no node duplication. Then the source and the destinations are identified

6.1.2 Randomized Multipath Routing

Randomized multipath routing that can overcome the Compromised Node attack Denial of Service attack is achieved. Here multiple paths are computed in a randomized way each time an information packet needs to be sent, such that the set of routes taken by various shares of different packets.

6.1.3 Secure Delivery of Packets

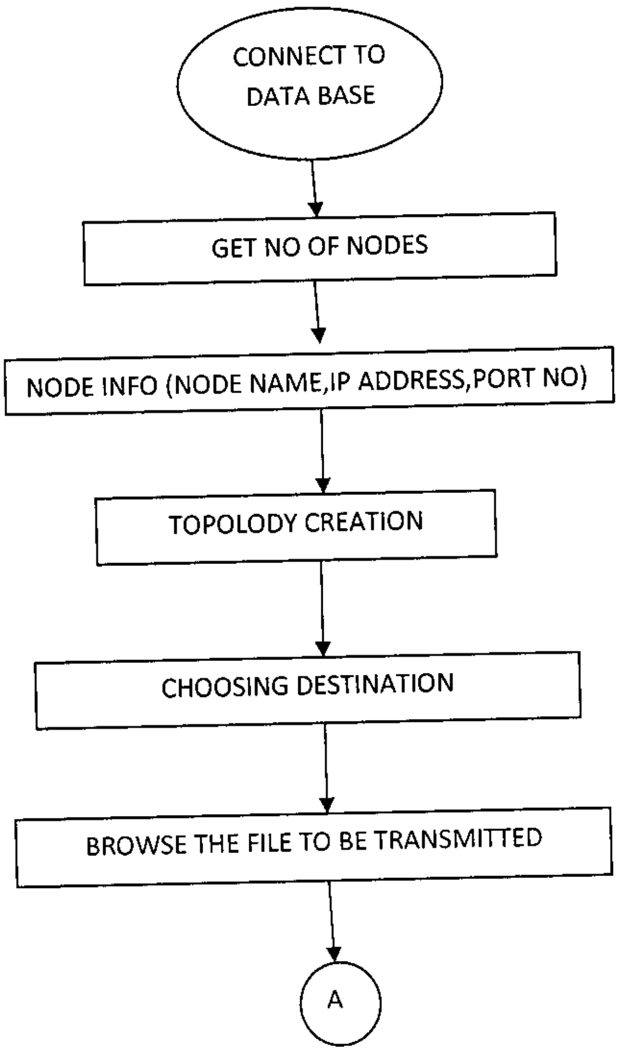
To achieve secure delivery a routing table is maintained. A column to added to maintain the packet delivery ratio. In this the number of many packets are transmitted over each path can be identified. It is useful to determine the number of packets transmitted over a path, so that transmission over particular path can be stopped for some time if it is found to frequently send packets through it. So the hacker cannot identify in which path the message is transmitted and also the data can be transmitted the data securely.

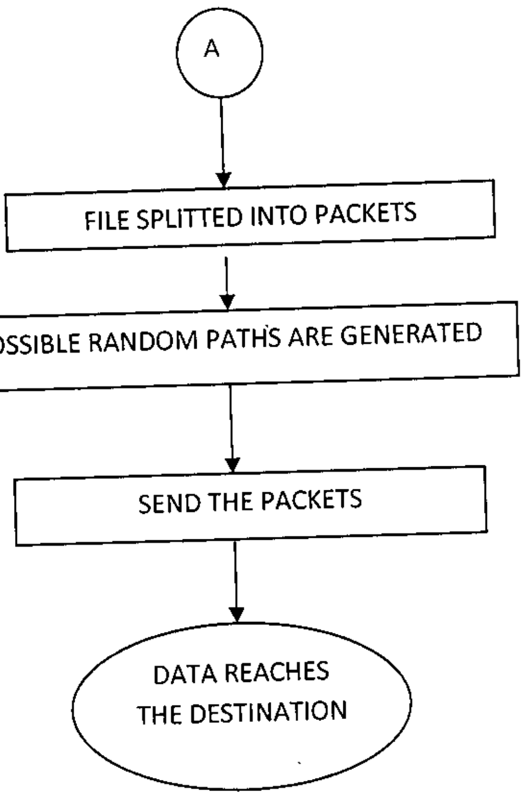
CHAPTER 7

ALGORITHM AND FLOW CHART OF THE PROPOSED SYSTEM

7.1 FLOW CHART

In this chapter the diagrammatic representation of the simulation is explained. The data flow includes establishing a connection with the data base, choosing number of nodes that are to be included in the network and updating those node information in data base. With the help of randomized path the packets are delivered with security


CONNECT TO DATA BASE
GET NO OF NODES
NODE INFO (NODE NAME,IP ADDRESS,PORT NO)
TOPOLODY CREATION
CHOOSING DESTINATION
BROWSE THE FILE TO BE TRANSMITTED
A


A
FILE SPLITTED INTO PACKETS
POSSIBLE RANDOM PATHS ARE GENERATED
SEND THE PACKETS
DATA REACHES THE DESTINATION

7.2 ALGORITHM:

Connect to data base:

- Initially the header files are declared.
- Create a class named db connection.
- ODBC driver is initiated and imported.
- The driver manager intimates the data base to get registered and connected.

Get Address:

- A input variable IP is initialized as string.
- The given IP address is stored in a file.
- The file is converted into bytes that can be read by the system.
- Then it returns the IP address in the form of a string.

Get Number of nodes:

- Class named get number is created.
- Get number establishes a connection with the database.
- An option pane is created to enter the number of nodes.
- If the number of nodes is not entered then a loop statement executes to display a message to enter the values.
- The given number of nodes is called by the object node info.

Get Node Information:

- ➢ A content pane is created with three text fields and two buttons.
- ➢ The text fields are node name , IP address and port number.
- ➢ The buttons are entered and clear.
- ➢ The class get info is established
- ➢ The inputs are given in the text fields
- ➢ When the enter button is clicked it calls the function insert node info
- ➢ It checks weather all the text fields are filled if not it displays a message to be filled.
- ➢ When all the inputs are given it establishes a connection with database and checks weather the given node already exists
- ➢ If the node name already exists in the database the it displays the message "the node already exists "and the text fields are cleared, otherwise the given node information's are updated in the database
- ➢ Once when the number of nodes becomes zero then it displays as "the nodes are successfully inserted".

Topology Creation:

- Create a class called topology creation.
- An Content pane is created with two combo boxes and two buttons
- The combo boxes consists of list of nodes and has entered and cancel button.
- The nodes are listed in the combo boxes by assigning a vector called node list.
- When the enter button is clicked database connection is established and the node information such as starting node, ending node and their weights are inserted in the database and the topology is created.

Login Form:

- Here the nodes which are to participate in the environment is logged in
- It consists of a label called node name and two buttons such as login and clear
- Once login button is clicked a socket is open for input and output stream.
- Then the respective node establishes a db connection and obtains its ip address and the port number.
- If the node name exists in the database then a message "login successfully" is displayed. Otherwise it displays the message "enter the correct node name".

Choosing Destination:

- Once a topology is created the destination is chosen for the respective source
- Hence a login form is created.
- This consists of labels to choose the destination, the file to be send and buttons such as send, clear.
- Once the destination is chosen then the possible paths are displayed with respective to the chosen destination.
- Then the file to be transmitted is browsed and the obtained file is splitted into packets.
- Once when the send button is clicked then the number of splitted packets is displayed along with the path.
- Then the packets are sent one by one.

Secure Delivery of Packets:

- A routing table is maintained.
- A column is maintained to note the packet delivery.
- With the help of this the number of packets delivered overreach path is determined.
- It will be useful for to identify any path can handle number packets.
- The transmission can be stopped for some amount of time period over that path.
- So the hacker cannot identify in which path the message is transmitted and the data is transmitted safely.

CHAPTER 8
SIMULATION RESULTS

8.1 NUMBER OF NODES

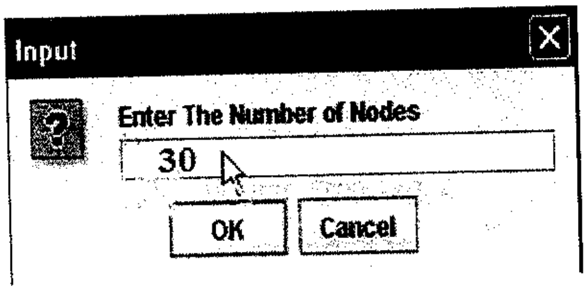
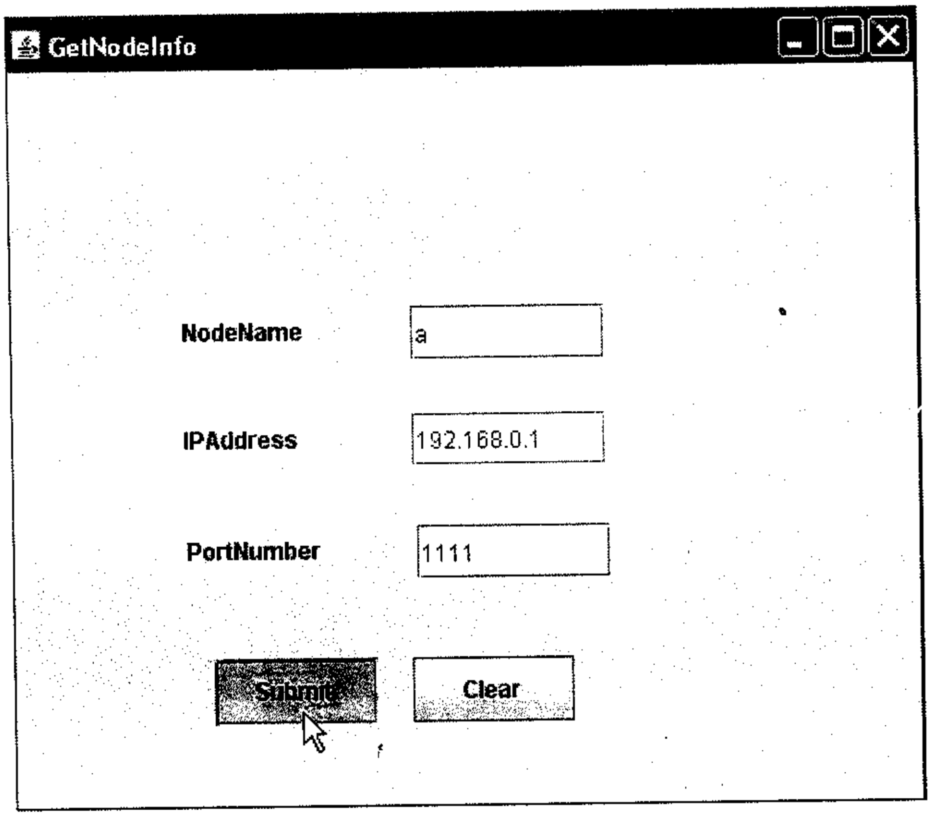


FIGURE 8.1 NUMBER OF NODES

The number of nodes that are to be included in the network is given here. After entering the number of nodes, ok button is clicked to obtain get node information.

For example here the number of nodes created are 30.

8.2 GET NODE INFORMATION


GetNodeInfo

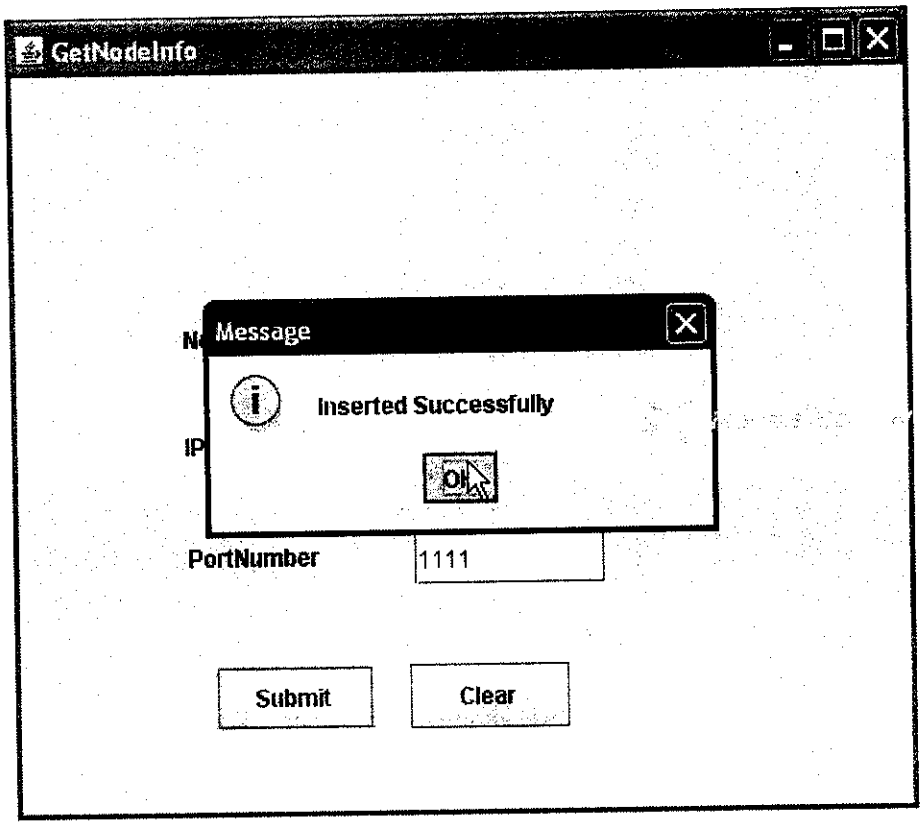
NodeName a

IPAddress 192.168.0.1

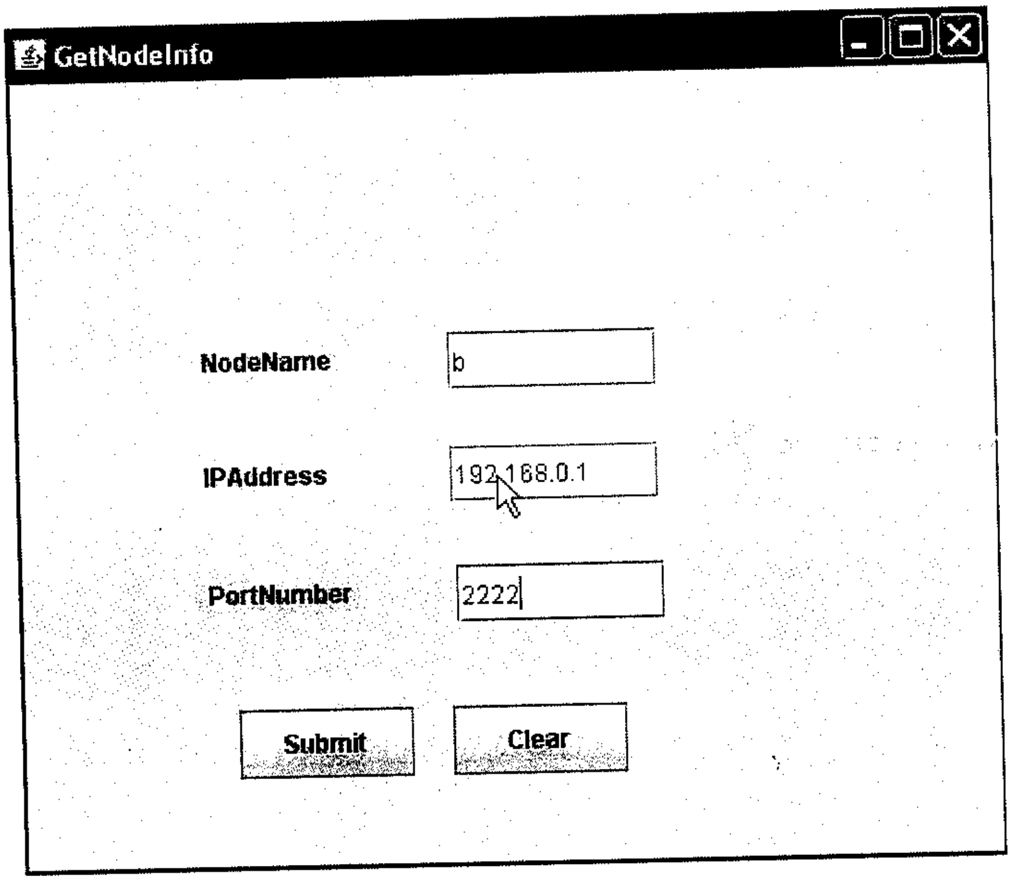
PortNumber 1111

8.2 GET NODE INFORMATION

After specifying the number of nodes in the network, the node info content pane appears. The name of the node, its internet protocol addresses and port number is obtained which are updated in the data base. If the node name is reaped then a notification message appears.



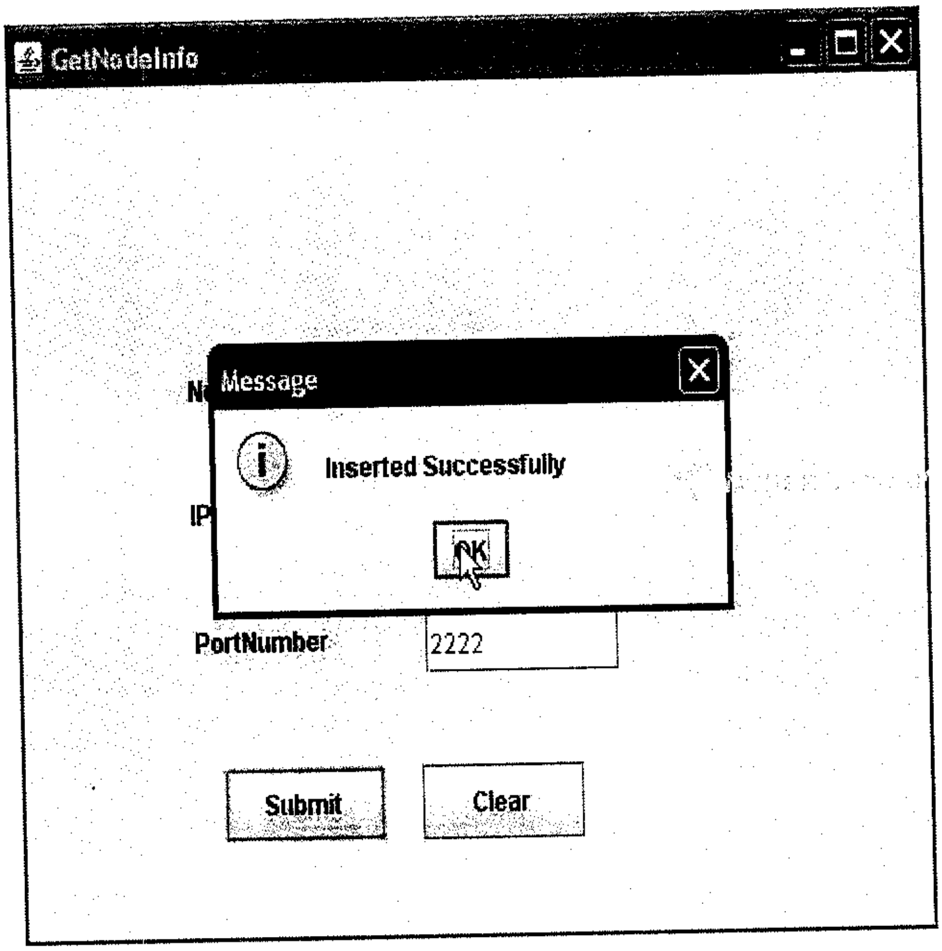
If the node information is correctly given then a message indication shows that the nodes are successfully updated or inserted in the database.


GetNodeInfo

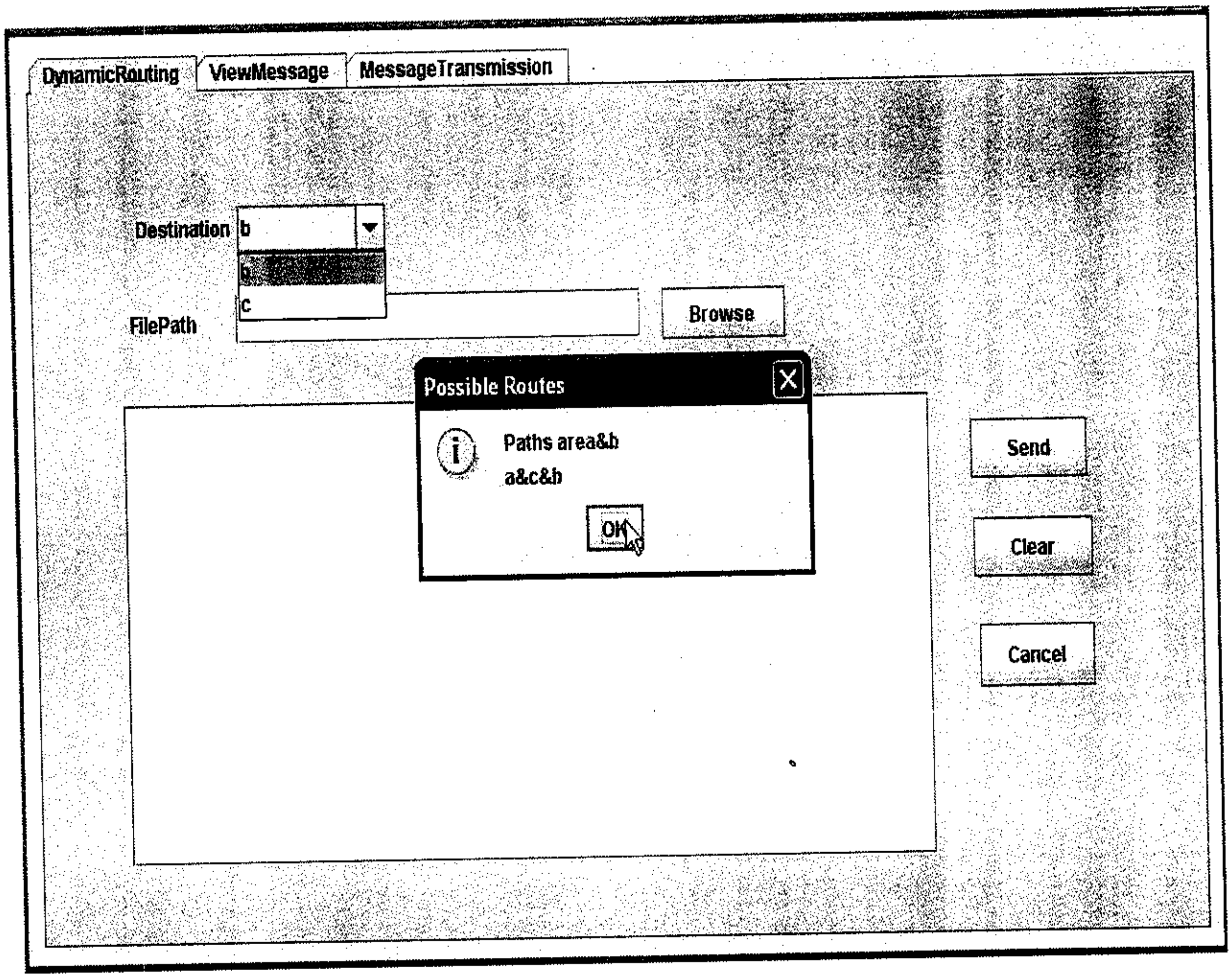
NodeName: b

IPAddress: 192.168.0.1

PortNumber: 2222


GetNodeInfo
Message
Inserted Successfully
OK
PortNumber
2222
Submit
Clear

8.3 SELECTION OF DESTINATION



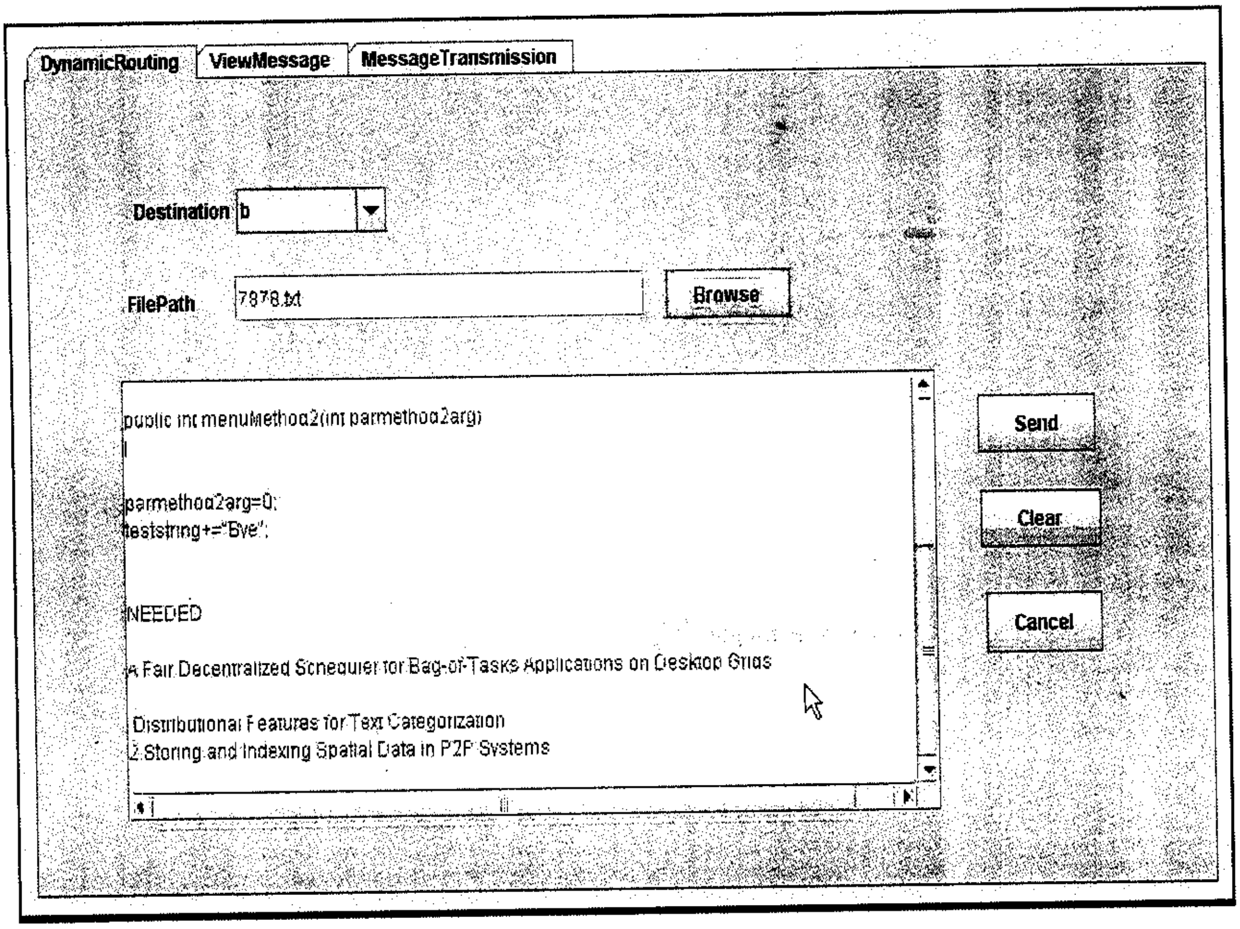
Possible Routes

Paths area&b
a&c&b

FIGURE 8.3 SELECTION OF DESTINATION

After deciding the source node the destination nodes are chosen. For the chosen destination node corresponding random path appears if the paths are satisfied then ok button is clicked.

8.4 SELECT THE FILE TO BE TRANSMITTED



Destination b

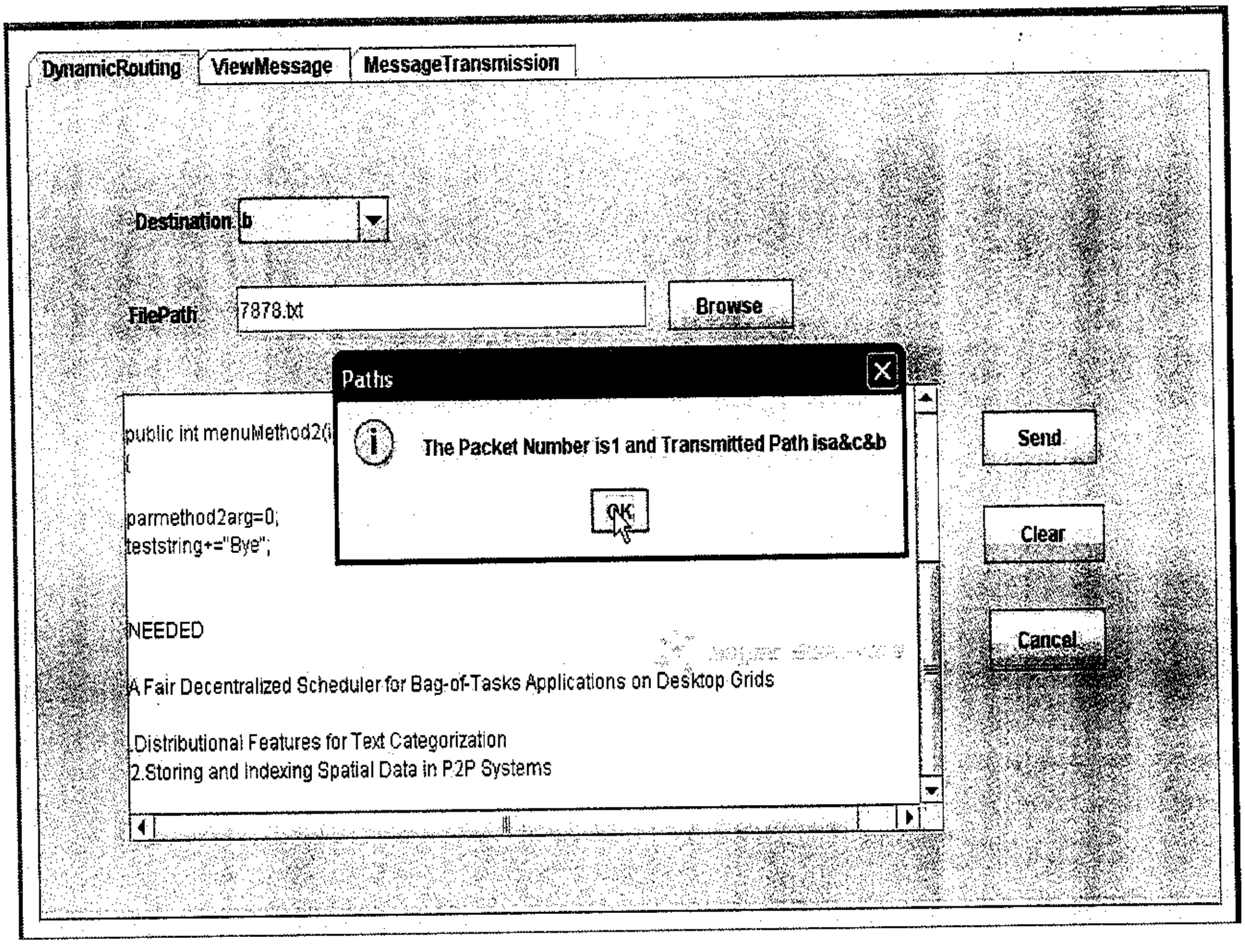
FilePath 7878.txt

```
public int menuMethod2(int parmethod2arg)
{


parmethod2arg=0;
teststring+="Bye";



NEEDED

A Fair Decentralized Scheduler for Bag-of-Tasks Applications on Desktop Grids

Distributional Features for Text Categorization
2.Storing and Indexing Spatial Data in P2P Systems
```

FIGURE 8.4 SELECT THE FILE TO BE TRANSMITTED

Once the ok button is clicked, the file which is to be transmitted is browsed and sent. Suppose if an alternate file has to be selected instead of the previous one then the clear button is clicked to remove the previously selected file.

8.5 PACTETS SENT OVER A PARTICULAR PATH



Destination: b

FilePath: 7878.txt

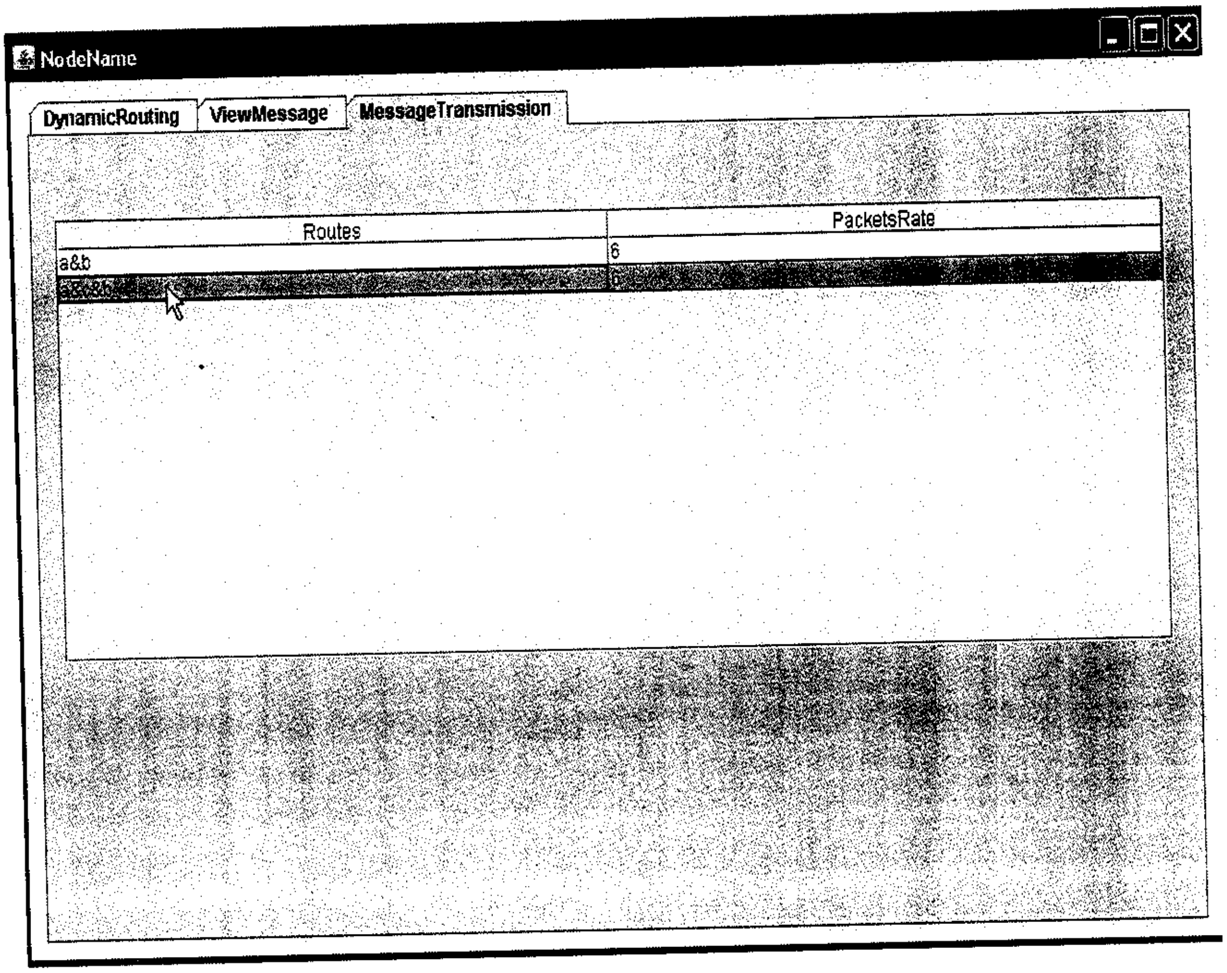
The Packet Number is1 and Transmitted Path isa&c&b

```
public int menuMethod2(i
{

parmethod2arg=0;
teststring+="Bye";


NEEDED

A Fair Decentralized Scheduler for Bag-of-Tasks Applications on Desktop Grids

Distributional Features for Text Categorization
2.Storing and Indexing Spatial Data in P2P Systems
```

FIGURE 8.5 PACTETS SENT OVER A PARTICULAR PATH

When the sent button is clicked the packet number and the path it takes to reach the destination is displayed. Once the ok button is clicked then the second path and its corresponding path is shown and this continues till the last packet is transmitted.

8.6 ROUTING TABLE



Routes	PacketsRate
a&b	6
a&c&b	6

FIGURE 8.6 ROUTING TABLE

The routing table shows the various paths for the file to be transmitted and the total number of packets transmitted over that path. This table helps to determine the random paths and the dispersive routes.

CHAPTER 9

CONCLUSION

The simulation results have shown the effectiveness of randomized dispersive routing in combating CN and DOS attacks. By appropriately setting the secret sharing and propagation parameters, the packet interception probability can easily be reduced by the proposed algorithm, which is at least one order of magnitude smaller than approaches that use deterministic node-disjoint multi-path routing. At the same time, this improved security performance comes at a reasonable cost of energy.

FUTURE ENHANCEMENT

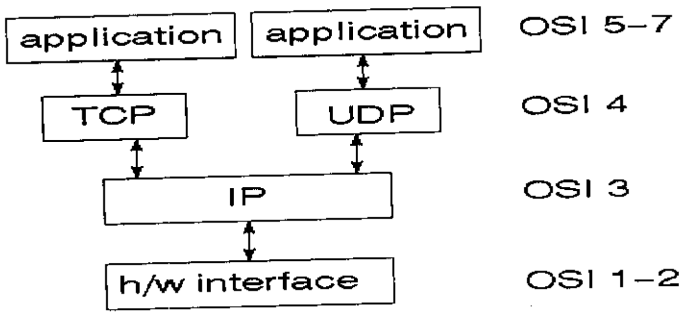
The current work is based on the assumption that there are only a small number of black holes in the WSN. In reality, a stronger attack could be formed, whereby the adversary selectively compromises a large number of sensors that are several hops away from the sink to form clusters of black holes around the sink. Collaborating with each other, these black holes can form a cut around the sink and can block every path between the source and the sink. Under this cut around- sink attack, no secret share from the source can escape from being intercepted by the adversary. Our current work does not address this attack. Its resolution requires extending our mechanisms to handle multiple collaborating black holes, which will be studied in future work. Moreover the current mechanism does not provide any means for error control hence further enhancement for this is also to be handled in future.

APPENDIX

NETWORKING

TCP/IP stack

The TCP/IP stack is shorter than the OSI one:



TCP is a connection-oriented protocol; UDP (User Datagram Protocol) is a connection less protocol.

IP DATAGRAM'S

The IP layer provides a connectionless and unreliable delivery system. It considers each datagram independently of the others. Any association between datagram must be supplied by the higher layers. The IP layer supplies a checksum that includes its own header. The header includes the source and destination addresses. The IP layer handles routing through an Internet. It is also responsible for breaking up large datagram into smaller ones for transmission and reassembling them at the other end.

TCP

TCP supplies logic to give a reliable connection-oriented protocol above IP. It provides a virtual circuit that two processes can use to communicate.

INTERNET ADDRESSES

In order to use a service, you must be able to find it. The Internet uses an address scheme for machines so that they can be located. The address is a 32 bit integer which gives the IP address. This encodes a network ID and more addressing. The network ID falls into various classes according to the size of the network address.

NETWORK ADDRESS

Class A uses 8 bits for the network address with 24 bits left over for other addressing. Class B uses 16 bit network addressing. Class C uses 24 bit network addressing and class D uses all 32.

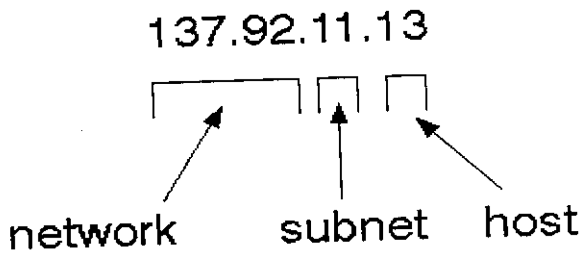
SUBNET ADDRESS

Internally, the UNIX network is divided into sub networks. Building 11 is currently on one sub network and uses 10-bit addressing, allowing 1024 different hosts.

HOST ADDRESS

8 bits are finally used for host addresses within our subnet. This places a limit of 256 machines that can be on the subnet.

TOTAL ADDRESS



The 32 bit address is usually written as 4 integers separated by dots.

PORT ADDRESSES

A service exists on a host, and is identified by its port. This is a 16 bit number. To send a message to a server, you send it to the port for that service of the host that it is running on. This is not location transparency! Certain of these ports are "well known"

SOCKETS

A socket is a data structure maintained by the system to handle network connections. A socket is created using the call socket. It returns an integer that is like a file descriptor. In fact, under Windows, this handle can be used with Read File and Write File functions. Two process wishing communicate over a network create a socket each. These are similar to two ends of a pipe - but the actual pipe does not yet exist.

REFERENCES

1. I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks," IEEE Comm. Magazine, vol. 40, no. 8, pp. 102-114(Aug. 2002).

2. Behrouz A.Forouzan "Data communication and Networking".

3. Herbert Schildt "The complete Reference For Java".

4. Prabhudutta Mohanty,Sangram Panigrahi,Nityananda Sarma and Siddhartha Sankar Satapathy" security issues in wireless sensor network data gathering protocols: a survey",Tezpur University, India.

5. Tao Shu, Marwan Krunz, Fellow and Sisi Liu"Secure Data Collection in Wireless Sensor Networks Using Randomized Dispersive Routes" IEEE transactions on mobile computing, vol. 9, no. 7(july 2010)

6. Zoran S. Bojkovic, Bojan M. Bakmaz, and Miodrag R. Bakmaz." Security Issues in Wireless Sensor Networks" International journal of communications Issue 1, Volume 2(2008)