

Security Enhanced Wireless Communication Sensor Networks

Dr. Rama Rao Adimalla¹, Dr. P. Aruna Kumari²

¹ Professor and HOD, Department of Computer Science and Engineering,
Lendi Institute of Engineering and Technology (A), Vizianagaram,
Jawaharlal Nehru Technological University Gurajada, Vizianagaram,
Andhra Pradesh, India.

[1*ramaroadimalla@gmail.com](mailto:ramaroadimalla@gmail.com)

² Assistant Professor, Department of Computer Science and Engineering,
Jawaharlal Nehru Technological University Gurajada, Vizianagaram,
Andhra Pradesh, India.

Abstract: These days, WSN communication security and localization security are major concerns in many security application domains, including military applications that rely on ad hoc wireless sensor networks. On the other hand, a variety of methods are available to offer communication security against malevolent nodes. These algorithms were unable to guarantee communication security in dangerous situations. In order to offer security and guard against intrusion in WSNs, we present a signature-based node authentication mechanism in this work. DSA is what we're utilizing to both create and validate the signature.

Keywords: WSN, Communications, SHARP, Routing Protocols

INTRODUCTION

Introducing relay capabilities in a network has a strong effect on the information flow that extends to all communication levels, from the achievable rates to the routing strategy.

A fundamental understanding of the role that relays play in wireless networks is of paramount importance to the design of efficient protocols for future communication systems. The problem of routing in a traditional multi-hop (TM) network model, where each relay node only listens to the immediately previous node is quite well understood today. For the purpose of routing, these networks are well modeled by directed graphs. Given a routing metric criteria, optimality conditions that guarantee that efficient path search algorithms, such as Dijkstra's algorithm, find the optimal path were studied in [1], [2]. The problem of routing in an accumulative multi-hop (AM) network model, in which we are instead interested, is however far from being understood today. In accumulative multi-hop networks, a single source communicates to a single destination assisted by several relay nodes that can accumulate the received energy/ information from previous relay transmissions.

In our product, the file or data is sent from sender node to the destination node in a network in a shortest path. We have used Dijkstra's algorithm to find the shortest path from one node to another node. The main aim is to present an effective Routing protocol in Accumulative Multi-Hop Networks and to solve the energy routing problem using decode-and-forward relays. The result and energy consumption will be represented in a graph.

LITERATURE SURVEY

Wireless Sensor Networks (WSNs) are used in many applications in military, ecological, and health-related areas. These applications often include the monitoring of sensitive information such as enemy movement on the battlefield or the location of personnel in a building. Security is therefore important in WSNs. However, WSNs suffer from many constraints, including low computation capability, small memory, limited energy resources, susceptibility to physical capture, and the use of insecure wireless communication channels. These constraints make security in WSNs a challenge. In this article we present a survey of security issues in WSNs. First we outline the constraints, security requirements, and attacks with their corresponding counter measures in WSNs. We then present a holistic view of security issues. These issues are classified into five categories: cryptography, key management, secure routing, secure data aggregation, and intrusion detection. Along the way we highlight the advantages and disadvantages of various WSN security protocols and further compare and evaluate these protocols based on each of these five categories. We also point out the open research issues in each subarea and conclude with possible future research directions on security in WSNs.

No secure protocol is using. So there is no secure and dynamic routing. Wireless Sensor Networks are a new class of Ad Hoc networks that will find increase in deployment in coming years, as they enable reliable monitoring and analysis of unfamiliar and untested environments. The advances in technology have made it possible to have extremely small, low powered sensor devices equipped with programmable computing, multiple parameter sensing, and wireless communication capability. But, because of their inherent limitations, the protocols designed for such sensor networks must efficiently use both limited bandwidth and battery energy. In this paper, we develop an M/G/1 model to analytically determine the delay incurred in handling various types of queries using our enhanced APTEEN (Adaptive Periodic Threshold-sensitive Energy Efficient sensor Network protocol) protocol. Our protocol uses an enhanced TDMA schedule to efficiently incorporate query handling, with a

queuing mechanism for heavy loads. It also provides the additional flexibility of querying the network through any node in the network. To verify our analytical results, we have simulated a temperature sensing application with a Poisson arrival rate for queries on the network simulator ns-2. As the simulation and analytical results match perfectly well, this can be said to be the first step towards analytically determining the delay characteristics of a wireless sensor network.

In Wireless Sensor Networks (WSNs), authentication is a crucial security requirement to avoid attacks against secure communication, and to mitigate against DoS attacks exploiting the limited resources of sensor nodes. Resource constraints of sensor nodes are hurdles in applying strong public key cryptographic based mechanisms in WSNs. To address the problem of authentication in WSNs, we propose an efficient and secure frame work for authenticated broadcast /multicast by sensor nodes as well as for outside user authentication, which utilizes identity based cryptography and online/offline signature (OOS) schemes. The primary goals of this framework are to enable all sensor nodes in the network, firstly, to broadcast and/or multicast an authenticated message quickly; secondly, to verify the broadcast/multicast message sender and the message contents; and finally, to verify the legitimacy of an outside user. This paper reports the implementation and experimental evaluation of the previously proposed authenticated broadcast /multicast by sensor nodes scheme using online/offline signature on Tiny OS and MICA2 sensor nodes.

In this system we study the securer outing for cluster based sensor networks where clusters are formed dynamically and periodically. We point out the deficiency in the secure routing protocols with symmetric key pairing. Along with the investigation of ID-based cryptography for security in WSNs, we propose a new secure routing protocol with ID-based signature scheme for cluster-based WSNs, in which the security relies on the hardness of the Diffie-Hell man problem in the random oracle model. Because of the communication over head for security, we provide analysis and simulation results in details to illustrate how various parameters act between security and energy efficiency.

The proposed security system for the Wireless Sensor Network (WSN) is based on the WSN security design goal that ‘to design a completely secure WSN, security must be integrated into every node of the system’. This paper discusses on two main components of the security framework viz. these cure key management module and the secure routing scheme. The incorporation of security mechanism during the

routing protocol design phase is the main focus of this paper. The proposed security framework viz. 'Secure and Hierarchical, a Routing Protocol' (SHARP) is designed for the wireless sensor network applications which are deployed particularly for data collection purpose in a battle field where the security aspect of the network cannot be compromised at any cost. SHARP consists of three basic integrated modules and each module performs a well defined task to make the whole security framework a complete system on its own.

Proposed System

In Proposed System, we studied the routing problem in accumulative multi-hop networks. We showed that a supposed to traditional multi-hopping where the network is well modeled by a graph, for routing in accumulative networks, the network needs to be modeled by a hyper graph.

We studied the properties that guarantee that Dijkstra's algorithm finds the optimal path in such networks, and presented sufficient conditions for the optimality.

System Analysis

4.1 System requirement specifications

A requirement is a feature that the system must have or a constraint that it must to be accepted by the client. Requirement engineering aims at defining the requirements of the system under construction. Requirement engineering include two main activities, requirement elicitation, which results in the specification of the system that the client understands ,and analysis which in analysis model that the developer can unambiguously interpret. A requirement is a statement about what the proposed system will do. Requirements can be divided into two major categories: Functional requirements and Non Functional requirements.

4.1.1 Functional Requirements

Functional requirements describe the interactions between the system and its environment independent of its implementation. The environment includes the user and any other external system with which the system interacts. Functional requirements capture the intended to behavior of the system, this behavior may be expressed services, tasks or functions the system is required to perform.

In product development, it is useful to distinguish between the base line functionality necessary for any system to compete in that product domain, and features that differentiate the system from competitors products and from variants in your company own product line/family. Features may be additional functionality, or differ from the basic functionality along some quality attribute (such as performance or memory utilization).

A system must send a files sent by the sender throughout the network to the destination whenever a certain condition is met i.e. browse the file, encryption and decryption of the file. Functional requirements specifies a function that a system or system component must be able to perform. It can be documented in various ways. The most common ones are written descriptions in documents, and use cases.

Use cases can be textual enumeration lists as well as diagrams, describing user actions. Each use case illustrates behavioral scenarios through one or more functional requirements. Often, though, an analyst will begin by eliciting a set of use cases, from which the analyst can derive the functional requirements that must be implemented to allow a user to perform each use case.

A typical functional requirement will contain a unique name and number, a brief summary, and a rationale. This information is used to help the reader understand why the requirement is needed, and to track the requirement through the development of the system.

4.1.2 Non-Functional requirements

Non-functional requirements are any other requirement than functional requirements .This are the requirements that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors.

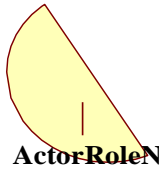


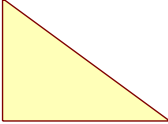
Non-functional requirements are in the form of "system shall be ", an overall property of the system as a whole or of a particular aspect and not a specific function. The system's overall properties commonly mark the difference between whether the development project has succeeded and failed.Non-functionalrequirements-canbedividedintotwomaincategories:

Execution qualities:

Execution qualities, such as security and us ability, which are observable a turn time.

- **Security:** The state of being free from danger or threat.
- **Usability:** It is very easy to learn and operate the system.

Table.1: Use case Diagram Components

Actor	An Actor as mentioned is a user of the system and is depicted using a stick figure. The role of the user is written beneath the icon. Actors are not limited to humans. If a system communicates with another application and expect sinputor delivers output then that application can also be considered as anactor.	
Use case	A Use Case is the functionality provided by the system typically described as verb+ object (eg: Register Car, Delete User). Use Cases are depicted with an ellipse. The name of the Use Case Is written within the ellipse.	
Directed Association	Associations are used to link Actors with use cases and indicate that an actor participates in the Use Case in some form. Directed Association is same as association but difference is that it represented by a line having an arrowhead.	
System boundary boxes	You can draw a rectangle around the use cases, called the system boundary box, to indicate the scope of your system. Anything within the box represents functionality that is in scope and anything outside the box is not.	

Level1DFD:

It is still a general overview, but they go into more detail than a context diagram. In a level1dataflow diagram, the single process node from the context diagram is broken down into sub processes. As these processes are added, the diagram will need additional data flows and data stores to link them together.

Level-1

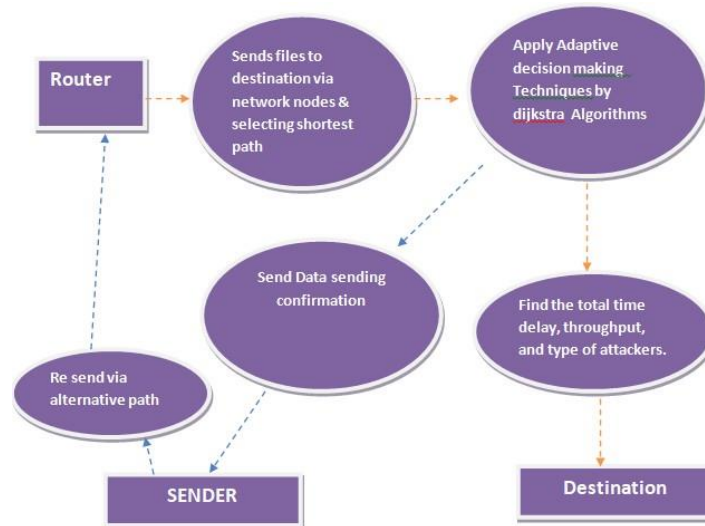


Figure5.4(b)Level 1Data flow Diagram

Level2+DFDs:

It simply breaks processes down into more detailed sub processes. In theory, DFDs could go beyond level 3, but they rarely do. Level 3 data flow diagrams are detailed enough that it doesn't usually make sense to break them down further.

Level-2

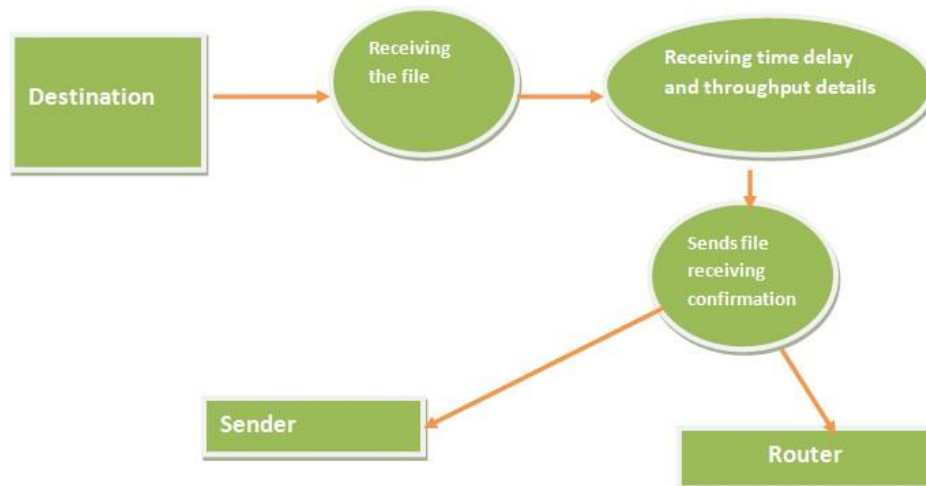


Figure5.4(c)Level2Dataflow Diagram

5.1 SYSTEM ARCHITECTURE

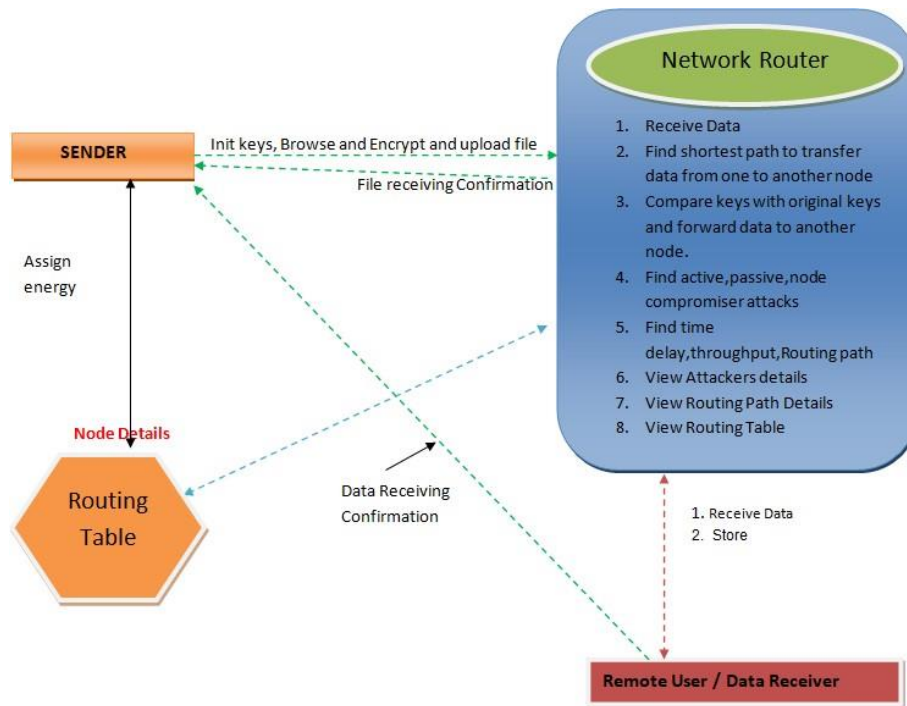


Figure 5.5 System Architecture

Sender:

In this module, the Source browses the required file, initializes nodes with digital signature and uploads to the end user (node a, node b, node c, node d, node e, node f) via Router.

Router:

The Router is responsible for forwarding the data file in shortest distance to the destination; the Router consists of Group of nodes, the each and every node (n1,n2,n3,n4,n5,n6,n7,n8,n8,n10,n11,n12,n13) consist of energy and Digital Signature. If router finds shortest path then it intimates to the neighbor nodes to form the routing path to send data to destination. The routing path uses Disktra's algorithms to find the shortest path to destination.

Destination:

In this module, the Destination can receive the data file from the Source which is sent via Router, if shortest path found via intermediate nodes. As soon as data receives then the destination will save data to the local disk.

5.2 Algorithm

Step1: Assign every node a tentative distance.

Step2: Set initial as current and mark all other nodes as UN visited.

Step3: For current node, consider all unvisited nodes and calculate tentative distance, Compare current distance with calculated distance and assign the smallest value.

Step4: When all the neighbors are considered of the current node, mark it

visited. Step5: If the destination node is marked visited, stop.

Step6: End

5.2.1 Algorithm Description

Dijkstra's algorithm initially marks the distance (from the starting point) to every other intersection on the map with infinity. This is done not to imply there is an infinite distance, but to note that those intersections have not yet been visited; some variants of this method simply leave the intersections' distances unlabelled. Now, a each iteration, select the current intersection. For the first iteration, the current intersection will be the starting point, and the distance to it (the intersection's label) will be zero. For subsequent iterations (after the first), the current intersection will be a closest unvisited intersection to the starting point (this will be easy to find).

From the current intersection, update the distance to every unvisited intersection that is directly connected to it. This is done by determining the sum of the distance between an unvisited intersection and the value of the current intersection, and replace the unvisited intersection with this value (the sum), if it is less than its current value. In effect, the intersection is relabeled if the path

to it through the current intersection is shorter than the previously known paths. To facilitate shortest path identification, in pencil, mark the road with an arrow pointing to their label led inter section if you label/ re label it, and erase all others pointing to it. After you have updated the distances to each neighboring intersection , mark the current intersection as visited, and select an unvisited intersection with minimal distance (from the starting point) – or the lowest label—as the current intersection. Intersections marked as visited are labeled with the shortest path from the starting point to it and will not be revisited or returned to.

Continue this process of updating the neighboring intersections with the shortest distances, then marking the current intersection as visited and moving onto a closest un visited intersection until you have marked the destination as visited. Once you have marked the destination as visited (as is the case with any visited intersection) you have determined the shortest path to it, from the starting point, and can trace your way back, following the arrows in reverse; in the algorithm's implementations ,this is usually done(after the algorithm has reached the destination node) by following the nodes' parents from the destination node up to the starting node; that's why we also keep track of each node's parent.

This algorithm makes no attempt of direct" exploration “towards the destination as one might expect. Rather, the sole consideration in determining the next "current" intersection is its distance from the starting point. This algorithm therefore expands outward from the starting point, interactively considering every node that is closer in terms of shortest path distance until it reaches the destination. When understood in this way, it is clear how the algorithm necessarily finds the shortest path. However, it may also reveal one of the algorithm's weaknesses: its relatives lowness in some to apologies.

IMPLEMENTATION

In our project we have use java Language Developing the front end Java, RMI, Swings, And AWT and some Networking processed them. Java is a high-level programming language developed by Sun Microsystems. It was originally designed for developing programs for set-top boxes and hand held

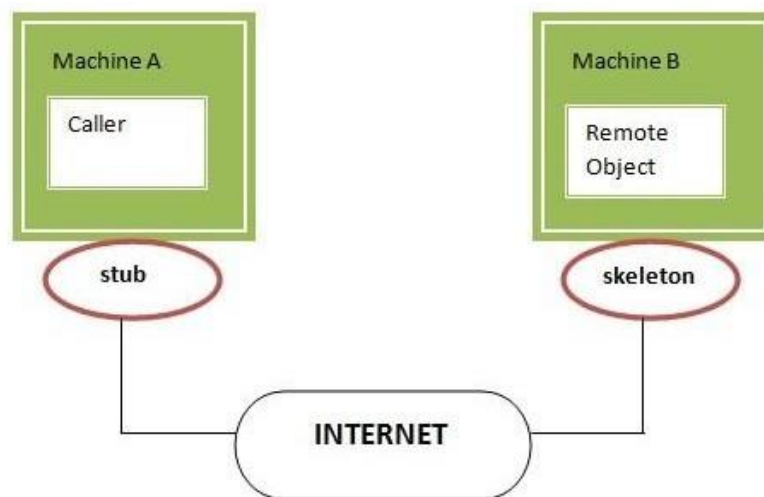
devices, but later became a popular choice for creating web applications.

The Java syntax is similar to C++, but is strictly an object-oriented programming language. For example, most Java programs contain classes, which are used to define objects, and methods, which are assigned to individual classes. Java is also known for being more strict than C++, meaning variables and functions must be explicitly defined. This means Java source code may produce errors or "exceptions" more easily than other languages, but it also limits other types of errors that may be caused by undefined variables or unassigned types.

The skeleton is an object, acts as a gateway for the server side object. All the incoming requests are routed through it. When the skeleton receives the incoming request, it does the following tasks:

1. It reads the parameter or the remote method
2. It invokes the method on the actual remote object, and
3. It writes and transmits (marshals) the result to the caller.

In the Java 2 SDK, a stub protocol was introduced that eliminates the need for



skeletons.

Figure 6.2 RMI

For Back End Purpose We are using SQL, SQL is a standardized query language for requesting information from a database. SQL has been the favourite query language for database management systems running on minicomputers and main frames. Increasingly, however, SQL is being supported by PC database systems because it supports distributed databases

(databases that are spread out over several computers systems). This enables several users on a local-area network to access the same database simultaneously.

An IDE normally consists of a source code editor, build automation tools, and a debugger. Most of the modern IDEs have intelligent code completion. Some IDEs, such as Net Beans and Eclipse, contain a compiler, interpreter, or both

6.1 Eclipse

Eclipse is an integrated development environment (IDE) used in computer programming, and is the most widely used Java IDE.^[6] It contains a base workspace and an extensible plug-in system for customizing the environment. Eclipse is written mostly in Java and its primary use is for developing Java applications, but it may also be used to develop applications in other programming languages via plug-ins,

including Ada, ABAP, C, C++, C#, Clojure, COBOL, D, Erlang, Fortran, Groovy, Haskell, Java Script, Julia,^[7] Lasso, Lua, NATURAL, Perl, PHP, Prolog, Python, R, Ruby (including Ruby on Rails framework), Rust, Scala, and Scheme. It can also be used to develop documents with and packages for the software Mathematics. Development environments include the Eclipse Java development tools (JDT) for Java and Scala, Eclipse CDT for C/C++, and Eclipse PDT for PHP, among others.

RESULTS

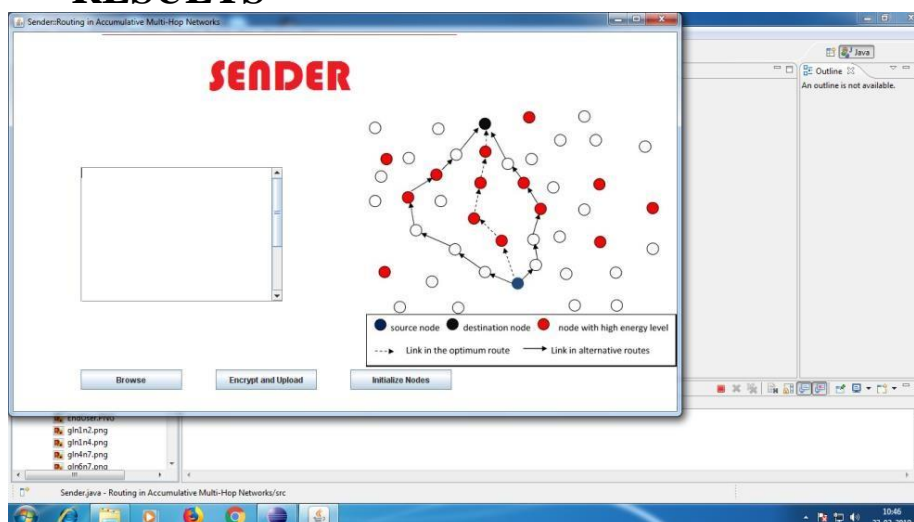


Figure8(a)Sender

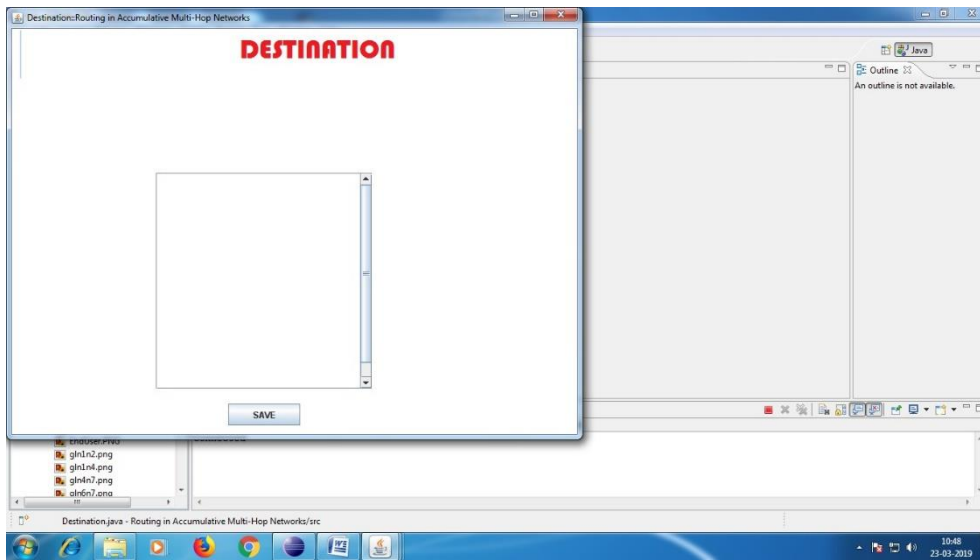


Figure8(b)Destination

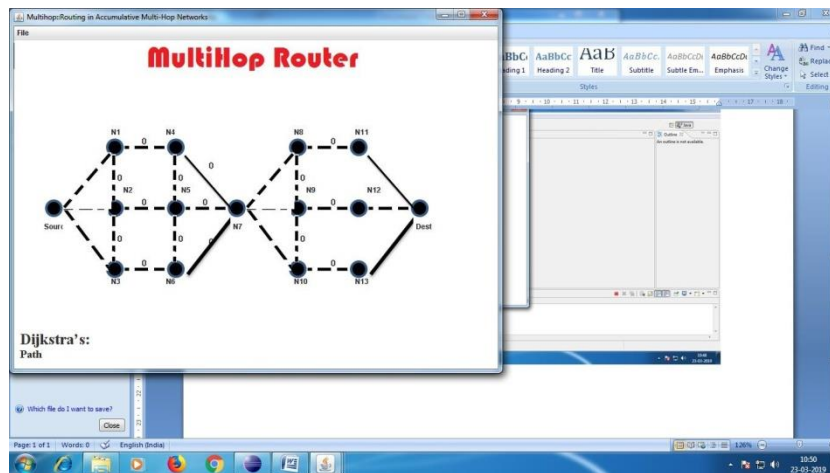


Figure8(c)MultiHopRouter

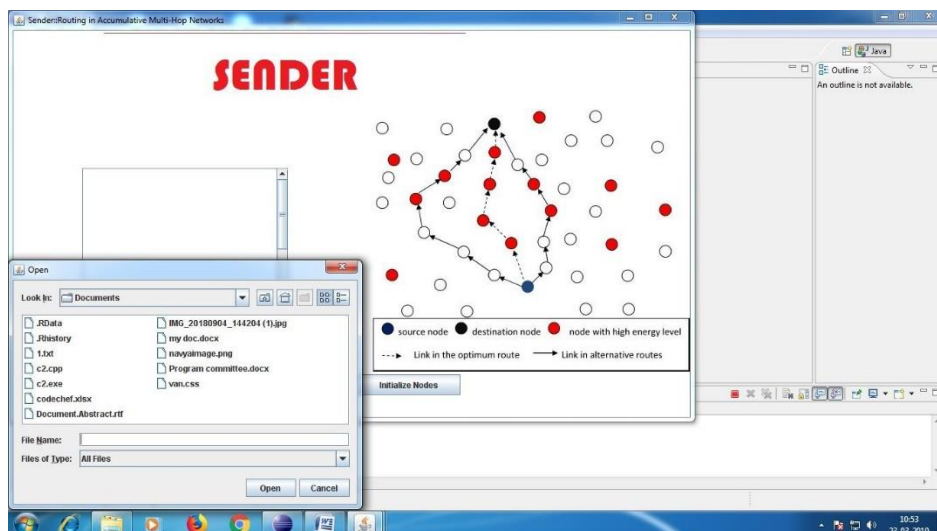


Figure8(d)BrowsingFiles**CONCLUSION**

We studied the routing problem in accumulative multi-hop networks. We showed that a supposed to traditional multi-hopping where the network is well modeled by graph, for routing in accumulative networks, the network needs to be modeled by a hyper graph. We studied the properties that guarantee that Dijkstra's algorithm finds the optimal path in such networks, and presented sufficient conditions for the optimality. These conditions are particularized for the minimum energy routing problem with decode-and-forward relays. We considered the directing issue in accumulative multi-hop networks. We demonstrated that rather than customary multi-hopping where the network is all around displayed by a chart, for directing in accumulative networks, the network should be demonstrated by a hypergraph. We examined the properties that certification that Dijkstra's algorithm finds the ideal way in such networks, and displayed adequate conditions for the optimality.

BIBLIOGRAPHY

- [1] J. Sobrinho, "An algebraic theory of dynamic network routing," *Networking*, IEEE/ACM Transactions on, vol. 13, no.5, pp. 1160–1173, oct.2005.
- [2] Y. Yang and J. Wang, "Design guidelines for routing metrics in multihop wireless networks," in *IEEE INFOCOM*, 2008, pp. 1615–1623.
- [3] I. Maric and R. Yates, "Cooperative multihop broadcast for wireless networks," *IEEE J. Select. Areas Commun.*, vol. 22, no.6, pp. 1080–1088, Aug. 2004.
- [4] J.Chen,L.Jia,X.Liu,G.Noubir,andR.Sundaram,"Minimumenergyaccumulative routing in wireless networks," in *Proc. IEEE INFOCOM*, vol. 3, Mar.2005, pp. 1875 – 1886.
- [5] A.Molisch,N.Mehta,J.Yedidia,andJ.Zhang,"Cooperativerelaynetworksusing fountain codes," in *Proc. IEEE Global Communications Conference (GLOBECOM)*, Nov. 2006.
- [6] J. Castura and Y. Mao, "Rateless coding over fading channels," *Communications Letters, IEEE*, vol. 10, no. 1, pp. 46–48, Jan 2006.
- [7] S. C. Draper, L. Liu, A. F. Molisch, and J. S. Yedidia, "Cooperative routing for wireless networks using mutual-information accumulation," *IEEE Trans. Inform. Theory*, vol. 57, Aug. 2011.
- [8] Z. Yang and A. Høst-Madsen, "Routing and power allocation in a synchronous gaussian multiple-relay channels," *EURASIP Journal on Wireless Communications and Networking*, 2006.
- [9] T. Girici and A. C. Kazez, "Energy efficient routing with mutual information accumulation," in *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, 2012 10th International Symposium on, may 2012, pp. 425–430.
- [10] R. Yim, N. Mehta, A. F. Molisch, and J. Zhang, "Progressive accumulative routing in wireless networks," in *Proc. IEEE Global Communications Conference (GLOBECOM)*, Nov. 2006.