

TRUST BASED ENERGY EFFICIENT LOAD BALANCED NETWORK ROUTING PROTOCOL FOR MOBILE ADHOC NETWORKS

A DISSERTATION

*Submitted in partial fulfillment of the
requirements for the award of the degree*

of

INTEGRATED DUAL DEGREE

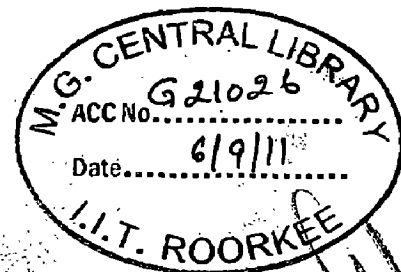
in

COMPUTER SCIENCE AND ENGINEERING

(With Specialization in Information Technology)

By

ROOPALI RAWAT



**DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE -247 667 (INDIA)
JUNE, 2011**

CANDIDATE'S DECLARATION

I hereby declare that the work being presented in the dissertation work entitled "Trust Based Energy Efficient Load Balanced Network Routing Protocol for MANET" towards the partial fulfillment of the requirement for the award of the degree of Integrated Dual Degree in Computer Science and Engineering (with specialization in Information Technology) submitted to the Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, India is an authentic record of my own work carried out during the period from May, 2010 to May, 2011 under the guidance and provision of Dr. A.K.Sarje & Dr. Manoj Mishra, Professor, Department of Electronics and Computer Engineering, IIT Roorkee.

I have not submitted the matter embodied in this dissertation work for the award of any other degree and diploma.

Date: 29 June, 2011

Place: IIT Roorkee



(Roopali Rawat)

CERTIFICATE

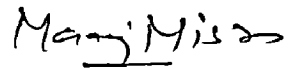
This to certify that the declaration made by the candidate is correct to the best of my knowledge

Date: 29 June, 2011

Place: IIT Roorkee


Dr. A.K. Sarje
Professor

Department of Electronics and Computer Engineering
IIT Roorkee, India


Dr. Manoj Misra
Professor

Department of Electronics and Computer Engineering
IIT Roorkee, India

ACKNOWLEDGEMENTS

I would like to take this opportunity to extend my heartfelt gratitude to my guide and mentor Dr. A.K.Sarje & Dr. Manoj Misra Professor, Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, for their trust in my work, their able guidance, regular source of encouragement and assistance throughout this dissertation work. I would state that the dissertation work would not have been in the present shape without their inspirational support and I consider myself fortunate to have done my dissertation under them. I also extend my sincere thanks to Dr. S.N. Sinha, Professor and Head of the Department of Electronics and Computer Engineering for providing facilities for the work.

I would like to thank all my friends who supported and encouraged me to finish this work. Finally, I would like to say that I am indebted to my parents for everything that they have given to me. I thank them for sacrifices they made so that I could grow up in a learning environment. They have always stood by me in everything I have done, providing constant support, encouragement, and love.

Roopali Rawat

LIST OF FIGURES

Fig 2.1 Different Routing Protocol for MANET

Fig 2.2 Format of AODV Route Request Packet

Fig 3.1 Format for Routing table entry

Fig 3.2: Format of RREQ packet

Fig 3.3: Format of RREP packet.

Fig 3.4 Pseudo-code getCost() Function

Fig 3.5 Entry in neighbor list

Fig 3.6 Structure of send_data table and rcv_data table

Fig 3.7 Data Table Entry Data structure

Fig 3.8 Data Table Entry Data structure after receiving hello packet from A

Fig 3.9 Data Table Entry Data structure after receiving hello packet from B

Fig 3.10 Hello packet Format

Fig 3.11 Trust table entry data structure

Fig 3.12 Flow in the sendHello() function

Fig 3.13 Hello packet traversal

Fig 3.14 Flow in the rcvHello() function

Fig 3.15 Hello packet header structure.

Fig 4.1 File reference of eeaodv.cc.

Fig 4.2 Change in True Positive and False Negative with different allowed_energy_difference value.

List of Tables and Graphs

Table 4.1 Trace file format.

Table 5.1 Simulation Parameters Used for simulation.

Graph 3.1 Trust value of a node vs simulation time.

Graph 5.1 Simulation time vs Number of Energy exhausted nodes.

Graph 5.2 Number of vulnerable nodes vs Network lifetime.

Graph 5.3 Number of Malicious nodes vs. Number of packet Drop.

ABSTRACT

The two most challenging issues in deployment of mobile adhoc networks (MANETs) are trust and energy consumption, since mobile nodes are battery powered and their behavior is unpredictable. Furthermore replacing and recharging batteries and making nodes co-operative are often impossible in critical environments like military applications. So the main issue is how to prolong the lifetime of the network. In this dissertation report, we propose a trust based energy efficient load balanced routing protocol for MANET. During route discovery, neighbor node with more trust and maximum average path energy is selected. Route reply from a node is accepted only if its trust value is high. Otherwise, the route is discarded. This approach forms a reliable route from source to destination thus increasing network life time, improving energy utilization and decreasing the packet loss during transmission.

Table of Contents

CANDIDATE'S DECLARATION.....	i
ACKNOWLEDGEMENTS	ii
LIST OF FIGURES	iii
LIST OF GRAPH AND TABLES	iv
ABSTRACT.....	v
TABLE OF CONTENTS.....	vi
Chapter 1: INTRODUCTION.....	1
1.1 Mobile Adhoc Network.....	1
1.2 Trust and Load Balancing	3
1.3 Problem Statement	4
1.4 Organization of Dissertation.....	4
Chapter 2: BACKGROUND AND LITERATURE SURVEY.....	5
2.1 Proactive energy aware routing protocols and Algorithms.....	6
2.1.1 Destination-sequenced distance vector (DSDV)	6
2.1.2 Optimized Link state routing (OLSR)	6
2.2 Reactive energy efficient routing protocols and algorithms	7
2.2.1 Dynamic Source Routing (DSR)	8
2.2.2 Ad-hoc On demand Distance Vector Routing (AODV)	9
2.3 Summary.....	10
2.4 Research Gap.....	10

Chapter 3: Trust Based Energy Efficient Load Balanced Network Routing Protocol

3.1 Description of Proposed Protocol	11
3.2. Role of Hello Packets	15
3.3. Control Packets.....	19
3.3.1 Hello Packet	19
3.3.2 Route Request Packet.....	20
3.3.3 Route Reply Packet.....	21
3.3.4 Route error.....	21
3.4 Sequence numbers.....	22
3.5. Routing Process.....	22
3.5.1 Route discovery.....	22
3.5.2 Reverse path setup.....	23
3.5.3 Forward path setup.....	23
3.5.4 Link breakage.....	23
3.6 Energy Calculation Model.....	24
3.7 Assumptions	25
3.8 Trust Model	25
3.8.1 Definition/ trust equation	25
3.8.2 Evaluation Process.....	26
3.8.2.1 Trust Initial Phase.....	26
3.8.2.2 Trust Update Phase.....	26
3.8.2.3 Trust Re-establish Phase.....	27
3.9 Parameter for evaluating the protocol.....	27

Chapter 4: IMPLEMENTATION AND SIMULATION DETAILS.....	28
4.1 Simulator.....	28
4.2 Proposed Protocol (eeaodv)	28
4.2.1 File Dependency of EEAODV Protocol.....	28
4.2.2 Trust Management of EEAODV Protocol.....	29
4.2.3 Enabling Hello Packets.....	31
4.3 Creating random traffic-pattern for wireless scenarios	31
4.4 Creating node-movements for wireless scenarios.	32
4.5 Trace File Format.....	33
Chapter 5: RESULTS AND DISCUSSION.....	34
5.1 Performance Metric.	34
5.2 Simulation Setup.....	34
5.3 Effect of network run time on the energy of the nodes.	35
5.4 Effect of number of vulnerable node on network lifetime.....	36
5.5 Effect of number of malicious node on number of packet drop.....	37
Chapter 6: CONCLUSION & FUTURE WORK	38
REFERENCES	39

CHAPTER 1

Introduction

1.1 Mobile Adhoc Network

A **computer network**, often simply referred to as a **network**, is a collection of computers and devices connected by communication channels that facilitates communication among users and allows users to share resources with other users. Initially computer networks were started as a necessity for sharing files and printers but later this has moved from that particular job of file and printer sharing to application sharing and business logic sharing.

A network can be characterized as wired or wireless. Wireless can be distinguished from wired as no physical connectivity between nodes is needed. Advantages of wired over wireless network are that they have high connection speed and they are not prone to interference and fluctuations. Wireless networking is an emerging technology that allows users to be able access a broad range of information and services while user are mobile.

There are two types of wireless networks:

- Infrastructure networks.
- Ad hoc networks.

In infrastructure networks, mobile nodes communicate via base stations which are a part of a fixed wired network. An ad hoc network is a network that is created dynamically without any preexisting network infrastructure. All nodes in an ad hoc multihop network behave like routers, so as nodes move around routes to other nodes in the network will need to be discovered and maintained. Ad hoc networks are very useful in situations like emergency search-and-rescue operations and meetings where people want to quickly share information.

Mobile Adhoc Network (MANET) [1, 2] is a collection of independent mobile nodes that can communicate to each other via radio waves. The mobile nodes that are in radio range can directly communicate, whereas others need the aid of intermediate nodes to route their packets. These networks are fully distributed and can work at any place without the help of any infrastructure. This property makes these networks highly flexible and robust. The characteristics of these networks are summarized as follows:

- Communication via wireless means.
- Nodes can perform the roles of both hosts and routers.
- No centralized controller and infrastructure.
- Dynamic network topology.
- Frequent routing updates.

MANETs are much more vulnerable to attack than wired network because of the following reasons:

- Open Medium - Eavesdropping is easier than in wired network.
- Dynamically Changing Network Topology – Mobile Nodes comes and goes from the network, thereby allowing any malicious node to join the network without being detected.
- Cooperative Algorithms - The routing algorithm of MANETs requires mutual trust between nodes which violates the principles of Network Security.
- Lack of Centralized Monitoring - Absence of any centralized infrastructure prohibits any monitoring agent in the system.

Some of the common attacks in MANETs are:

- Jamming.
- Snooping.
- Flood Storm attack.
- Packet Modifications and Dropping.
- Repeater attack.
- Identity Impersonation.
- Black Hole attack.
- Wormhole attack.

The advantage of Mobile Ad-hoc Networks (MANETs) is to form a wireless network in the absence of fixed infrastructure. During early stages of MANETs, routing protocols were incapable of handling security issues but the introduction of newer techniques like cryptography and trust enabled them to handle the routing securely.

Ad hoc wireless networks find applications in emergency search-and-rescue operations, decision making in the battlefield, data acquisition operations in hostile terrain, etc. It is featured by dynamic topology (infrastructureless), multihop communication, limited resources (bandwidth, CPU, battery, etc.) and limited security. These characteristics put special challenges in routing protocol design. The primary objectives of MANET routing protocols are to maximize network throughput, to maximize energy efficiency and to maximize network lifetime.

In this thesis, we simulate a new trust based energy efficient load balanced MANET routing protocol. The goal of the routing protocol is to maximize the *network lifetime*.

1.2 Trust and Load Balancing

In MANET, node misbehavior due to selfish or malicious reasons can significantly degrade the performance of ad hoc networks. To cope with misbehavior in such self-organized networks, a mechanism must be in place. In this dissertation, a trust-based model on a self-policing mechanism is proposed to make collaboration rational for selfish/malicious nodes. This trust system makes them evaluate the trust of their neighbor locally to mitigate contamination by direct observation and the use of second-hand information available. In our approach, every node maintains a trust rating of their neighbor node by evaluating their published energy value.

The concept of "Trust" originally derives from the social sciences and is defined as the degree of subjective belief about the behaviors of a particular entity. Blaze et al. [3] first introduced the term "Trust Management" and identified it as a separate component of security services in networks. Trust management in MANETs is needed when participating nodes, without any previous interactions, must establish a network with an acceptable level of trust relationships among themselves.

Trust management [4], including trust establishment, trust update, and trust revocation, is much more challenging in a MANET than in traditional centralized environments. For example, collecting trust information (or evidence) to evaluate trustworthiness of a node is difficult due to mobility that changes in network topology. Resource constraints (like battery power) further confine the trust evaluation process to only local information, so that trust establishment would be based on incomplete and incorrect information. The dynamic nature and characteristics of MANETs result in uncertainty and incompleteness of the trust evidence that is continuously changing over time.

As against table driven routing protocols used by wired networks, MANETs commonly use on-demand routing protocols like AODV (ad-hoc on demand distance vector routing) [14] and DSR (destination sequenced distance vector) [11]. Frequent mobility of nodes causes changes in the topology of the network. This makes storing routing information in routing tables unusable. So on-demand routing protocols commonly use flooding to learn new routes. All the routing protocols use minimum number of hops as the criteria for route selection. Nodes which are part of shortest path will be burdened when network traffic increases giving rise to congestion. Consequently overburdened nodes start dropping packets. Moreover energy of these nodes, starts decreasing rapidly. Hence such nodes die earlier resulting in network partitioning. So incorporating load awareness into routing protocols is very essential for distributing traffic uniformly over all portions of the network. Incorporating just load awareness proves to be still not sufficient to improve the performance. For instance, a node may be selected as part of a routing path because of its lower traffic level, but that node may not be having enough energy in it. Therefore very often traffic and battery aware routing metrics are combined. Load balancing on the metric of traffic can be single handled by implementing routing on battery-awareness, because the average battery power of the path decreases where there is more traffic. Hence it is

imperative that a routing protocol does load balancing and energy balancing taking into effect the number of neighbors and their energy.

1.3 Problem Statement

The aim is to simulate a trust based energy efficient load balanced network layer routing protocol for mobile adhoc network which focuses on increasing the lifetime of the network. The protocol should also consider the role of vulnerable nodes in the routing and decrease their use for an increase in network lifetime.

- Should detect malicious nodes (if any).
- Should consider role of vulnerable nodes in routing

1.4 Organization of Dissertation

This report comprises of six chapters including this chapter that introduces the topic and states the problem. The rest of the dissertation report is organized as follows.

Chapter 2 provides a brief description of literature review on routing protocols .The other topics include routing protocols for MAETs, existing energy efficient protocols and research gaps. Chapter 3 provides a detailed description of the design, architecture and working of the proposed protocol. Chapter 4: This chapter gives details of implementation and simulation setup for the proposed protocol. Chapter 5 discusses the results and including discussion on them. It also provides as analysis on important performance parameters. Chapter 6 concludes the work and gives the directions for future work.

CHAPTER 2

Background and Literature Survey

In recent past energy efficient dynamic routing was addressed by many researchers. This has produced much innovation and novel ideas in this field. We discuss reactive and proactive based routing approaches. Most of the work today is based on energy efficient routing because power is the main concern in ad-hoc wireless networks. Each and every protocol has some advantages and shortcomings. None of them can perform better in every condition. It depends upon the network parameters which decide the protocol to be used. Several protocols for energy efficient routing and their modifications have also been proposed to improve the performance of ad-hoc networks.

MANET is formed of mobile nodes (supports wide range of mobility) and so the scenario is changing time to time. This is what makes the MANETs difficult for routing. There are many strategies proposed for routing in MANETs. Mobile ad hoc network's routing protocols [5] are characteristically based on three main approaches. These are On-demand approach, Table driven approach and hybrid approach. Each category has many protocols and some of these protocols are shown in figure (2.1).

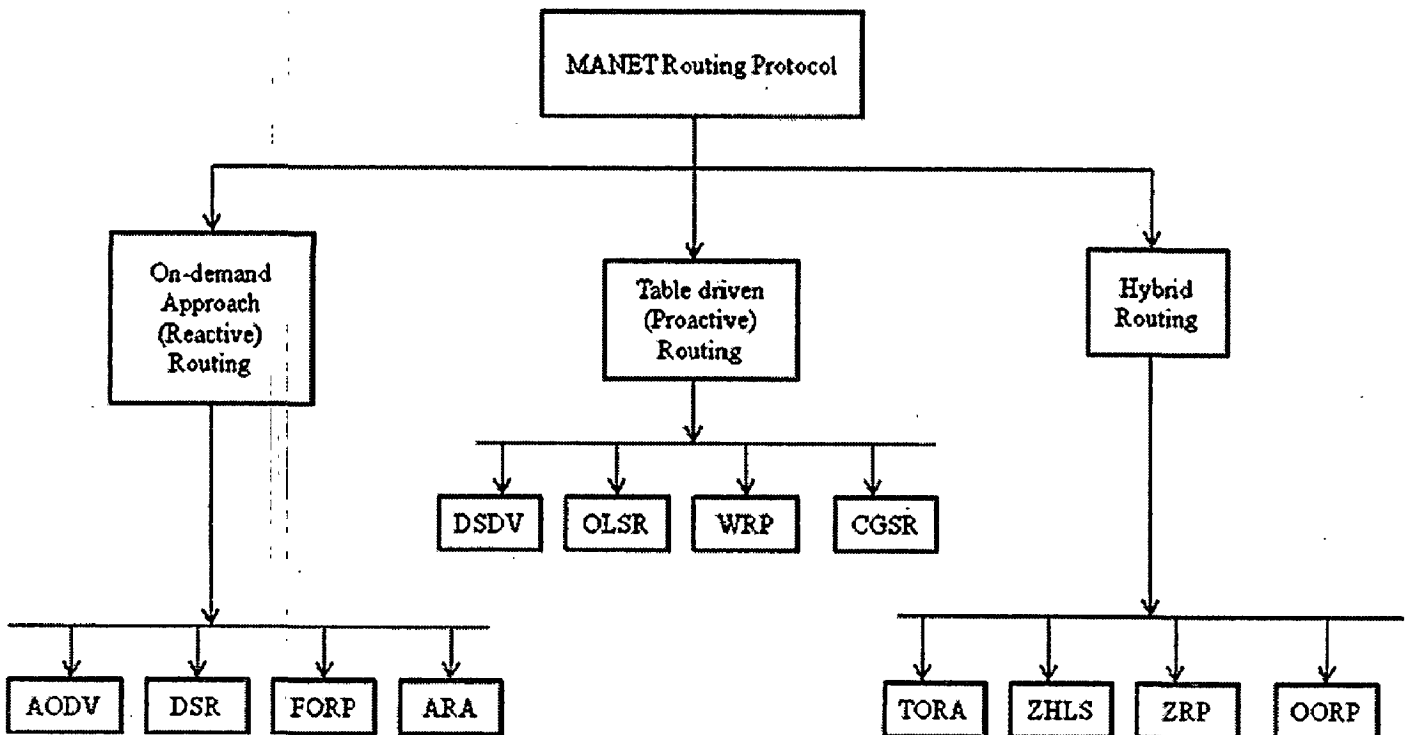


Fig 2.1 Different Routing Protocol for MANET

2.1 Proactive Routing Protocols

First of all we will discuss about proactive routing protocols which are categorized further in a following way on the basis of the algorithms used. This type of protocols maintains fresh lists of destinations and their routes by periodically distributing routing tables throughout the network.

2.1.1 Destination-sequenced distance vector (DSDV)

DSDV [6] (destination sequenced distance vector) is the most obvious proactive protocol which was given by Perkins and Bhagvat. It is based on Bellman Ford algorithm [7]. It removed the shortcomings (loops, count to infinity problem) of contemporary distance vector protocol which was not suited for ad-hoc networks. It is a destination based distance vector routing protocol in which every node maintains a routing table. This routing table contains all available destinations, the next node to reach to destination, and the number of hops between it. Whenever any node changes its position it broadcast the routing updates to the other nodes. Sequence number is used to avoid loop problems. Keeping the simplicity of distance vector protocol it guarantees loop freeness it reacts immediately on topology changes. Since the route for destination is always available at the routing table of each node so there is no latency caused by route discovery. But broadcasting of routing updates may cause high traffic load between the nodes if the density of the nodes are high. So this protocol is best suited if the density of the ad-hoc network is low. However if the mobility of the node is too high broadcasting updates may cause time delay.

2.1.2 Optimized Link state routing (OLSR)

OLSR [8] is a link state proactive protocol which routes to all reachable nodes in the network with minimal delay. This protocol use the concept of selective flooding which reduces the network traffic and power consumption for highly dense network, since it allows only a set of nodes (MPR's) to broadcast the control messages whenever the topology changes. It removes the problem of unnecessary duplication of control messages. The main advantage of OLSR protocol is that it is good for dense network which was not supported by AODV protocol. In OLSR each node periodically broadcast hello messages to learn topology up to 2 hops. Based on this hello messages each node select its set of MPR's. The problem in this type of protocol is to select a minimal set of MPRs each time the topology changes which is a NP hard problem. OLSR protocol does not take energy saving techniques into account.

A new energy efficient unicast routing protocol **EOLSR** [9] was proposed in 2008 .EOLSR increases the network lifetime by selecting the path having minimum cost where the cost is calculated on the basis of residual energy of each traversed node and the energy conserved on this path.

Energy-efficient broadcast OLSR (EBOLSR) [10] adapts the OLSR protocol in order to maximize the network lifetime for broadcast communications. In EBOLSR energy efficient MPR [9] selection is done by the residual energy of nodes, in this protocol the weighted residual energy of energy efficient MPR candidate and its 1 hop neighbors is considered. The basic phenomenon about this EBOLSR protocol is to select the energy efficient multipoint relays [MPR's].

2.2 Reactive energy efficient routing protocols and algorithms

In reactive routing, the routes are discovered only when need of that route arises. This type of protocols finds a route on demand by flooding the network with Route Request packets. The main disadvantages of such algorithms are:

- High latency time in route finding.
- Excessive flooding can lead to network clogging.

There are two types of reactive routing:

Source Routing:

In source routing, data packets carry the complete addresses from source to destination and no routing table in intermediate nodes. Some source routing protocols are: Dynamic Source Routing, Associatively Based Routing, and Signal Stability-based Adaptive Routing. We will discuss only DSR protocol and its energy efficient improvements in the following section.

Hop by Hop Routing

In hop- by- hop routing data packets do not carry complete route from source to destination but carries only the address of destination and the next hop. Some hop-by-hop routing protocols are: Ad-hoc on demand Distance Vector routing (AODV), Temporarily Ordered Routing Algorithm (TORA). In the following chapter we will only discuss AODV and its energy efficient improvements.

2.2.1 Dynamic Source Routing (DSR)

DSR [11] is a source routing protocol. It means that the sender node knows the complete route to the destination. These routes are stored in the route cache. If a node has data to send and no route is present then route discovery process will go on. Route discovery is basically based on flooding mechanism in which route request (RREQ) packets is sent to all its neighbors. Each intermediate node rebroadcasts it unless it is the destination or it has a route to the destination. This type of node replies to the request with a route reply packet that is routed back to the source node. If the node has already treated this route request it rejects the new received request. Route maintenance will go on if a link of route is broken then it deletes each route having this link from its cache, then it generates a route error packet to inform the source node and all intermediate nodes about this link failure until this route error packet reaches to the destination. After that a new route request launched by source to find a new route or check in its route cache. Due to caching DSR is more effective at low mobility and at low loads. But, it has many limitations such as it doesn't take into consideration the capacity of each node as power computing and no security mechanism is defined for DSR.

Weight based DSR (WBDSR) [12] is an improvement of conventional DSR. In this protocol, the weight of each route is considered as a metric for route selection. Weight of each route can be calculated as follows:

- Compute the node weight of each node weight = battery level of this node + Stability of this node
- Compute the route-weight as the minimum of all node weights included in this route.
- Select the main route as the one having the maximum route - weight.
- If two or more routes have the same route-weights then choose the route which has minimum hops.

Thus WBDSR always gives the longest network life time in both high by mobile networks and static networks because it timely change the used route with another one which maintains the use of the nodes which enhances the network life time.

Energy Dependent DSR (EDDSR) [13] is energy dependent DSR algorithm which helps node from sharp and sudden drop of battery power. EDDSR avoids use of node with less power supply and residual energy information of node is useful in discovery of route. Residual battery power of each node is computed by itself and if it is above the specific threshold value then node can participate in routing activities otherwise node delays the rebroadcasting of route request message by a time period which is inversely proportional to its predicted lifetime. With help of

ns-2 simulator author performed simulation which shows EDDSR is better than DSR in terms of node lifetime.

2.2.2 Ad-hoc On demand Distance Vector Routing (AODV)

AODV [14] Ad-hoc On Demand Distance Vector Routing is a protocol which combines some properties of DSR and DSDV routing protocols. In this protocol when a source needs to send data packets, it checks the route table, if there is a route to the destination, the data packets will be transmitted to the next node following the route in the route table. Else if the route is not present in the route table, source node starts the route discovery process. The source node broadcasts a route request [RREQ] message.

A RREQ message contains following important fields:

Source address	Source sequence number	Broadcast ID	Destination address	Destination sequence number	Hop counter

Fig 2.2 Format of AODV Route Request Packet

Source address and broadcast ID uniquely identifies a RREQ packet. When an intermediate node gets a RREQ message, it first checks that the RREQ has been received already according to the source address and broadcast ID. If this RREQ message has been already received, discards the RREQ message. Else it records the information in RREQ, increases the hop counter and broadcast the RREQ to its neighbors. This process continues until the RREQ message reaches to the destination or the value of Time to Live (TTL) exceeds the maximum allowed. When a RREQ message is received by an intermediate node from a node, a reverse link is formed between these nodes. When the destination node gets the RREQ, a reply message RREP will be transmitted back to the original source along the established reverse route path and after receiving the reply message, the source node gets a path from source to destination and source is ready to send the data packets. AODV broadcasts a HELLO message with regular intervals to check the connectivity of the active route. When neighboring nodes receive HELLO message, they update corresponding routes. If HELLO message is not received in the definite time interval, the link is considered to be broken. When a link is broken, a route error message RERR is transmitted to inform the source node that a link has been broken. Then route discovery process restarts. AODV performs well in high mobility and high loads.

In [15], author proposes two energy efficient algorithms based on AODV as: AODVE and AODVM. In AODVE, to increase the lifetime routing is based on the minimum remaining energy metric and that route is selected in which there is a maximum of minimum remaining energy (MIN_RE) and this field is added in RREQ as well as in the RREP. Other parameters are same as in AODV. Similar to AODVE, the latter AODVM also considers the residual energy but

it also considers the hop count value. It increases the lifetime of a network by arranging almost all nodes to involve in data transfer. It also shows improvement in delay and energy consumption of node.

In [16] New-AODV protocol has been proposed. The energy state of each node as well as of the entire network has been considered. New field is added to the RREQ message which carries the collected remaining energy of nodes participating between source and the destination. In this, Destination node does not give an immediate reply to the request but waits for some time and in the mean time, calculate the mean energy of the network and is stored in each node. In case of a new route, this Mean energy is then compared with the energy remaining in the node and if it is less, then RREQ message is delayed by some time and by this the entire lifetime is extended.

While [17] proposes an algorithm which selects the nodes on the basis of their energy status, which help in discovering alternate paths and to solve the problem of asymmetric links. In this, neighboring nodes (Backbone nodes) of active route having energy above than some threshold value are selected for route establishment between source and destination. It shows delay in starting phase but is best for high density networks.

Some modifications [18] have been done to improve the performance of AODV. It controls the transmission power consumption of nodes by listening to only their messages; this is done by adding two fields to RREQ and RREP packets. It also increases the lifetime of a network by judging about the duplicity of broadcast but work better in low mobile network.

2.3 Summary

In the above section we discussed some conventional protocols and their modification which includes energy efficiency with the importance of energy efficient routing protocols. Performance of the protocol varies according to the variation in the network parameters. Sometimes the mobility of the node of the network is high sometimes energy of the node is our prime concern.

2.4 Research Gap

All of the above protocols use the information of remaining node energy as a parameter to select the route. But energy of a node is a node's attribute and no other node has the information unless it is broadcast. This gives malicious node an edge to deny the routing services and still be trustworthy. None of the above work is able to handle this issue. M. Pushpalatha et al. [19] discusses the use of trust management in energy efficient/ energy aware protocol but doesn't handle the authenticity of published remaining node energy by a node.

In the next chapter we propose an improvement in the protocol design given in [17] and add the trust management method on nodes for the authenticity of the published remaining node energies.

CHAPTER 3

Trust Based Energy Efficient Load Balanced Network Routing Protocol

This chapter gives the complete architecture, design and working of the proposed Trust Based Energy Efficient Load Balanced Network Routing Protocol.

3.1 Description Of Proposed Protocol

The proposed Trust Based Energy Efficient load balanced MANET protocol (EEAODV) is an improvement over AODV [12]. EEAODV determines a route to a destination only when a node wants to send a packet to that destination. Routes are maintained as long as they are needed by the source. Sequence numbers ensure the freshness of routes and guarantee the loop-free routing.

When a source needs to send data packets, to the destination it checks the route table, if there is a route to the destination, the data packets will be transmitted to the next node following the route in the route table. Same as AODV each routing table entry in EEAODV contains the following information (refer fig 3.1):

Destination Node id	Next hop	Number of hops	Destination sequence number	Expiration time for this route table entry	Active neighbors for this route
---------------------	----------	----------------	-----------------------------	--	---------------------------------

Fig 3.1 Format for Routing table entry

Expiration time, also called lifetime, is reset each time the route has been used. The new expiration time is the sum of the current time and a parameter called active route timeout. This parameter, also called route caching timeout, is the time after which the route is considered as invalid, and so the nodes not lying on the route determined by RREPs delete their reverse entries. If active route timeout is big enough route repair function will maintain routes. Destination node id is the node id for which the route entry is made and next hop is the neighbor node id which should be the next node in the route. The entry is updated on the basis of trust value of the node, cost, indicator and hop count (in decreasing order of priority)

When a route is not available for the destination in the source node route table, a route request packet (RREQ) is flooded throughout the network. The RREQ contains the following fields (refer Fig 3.2):

Source address (rq_src)	Request ID	Source sequence n.o	Destination address (rq_dest)	Destination sequence no.	Hop count (rq_hop)	Cost (rq_cost)	time to live (TTL)	Indicator (rq_ind)
-------------------------	------------	---------------------	-------------------------------	--------------------------	--------------------	----------------	--------------------	--------------------

Fig 3.2: Format of RREQ packet

The request ID is incremented each time the source node sends a new RREQ, so the pair (source address, request ID) identifies a RREQ uniquely. On receiving a RREQ message each node checks the source address and the request ID. If the node has already received a RREQ with the same pair of parameters the new RREQ packet will be discarded. Otherwise the RREQ will be either forwarded (broadcast) or replied (unicast) with a RREP message:

- if the node has no route entry for the destination, or it has one but this is no more an up-to-date route, the RREQ will be rebroadcasted with incremented hop count.
- if the node has a route with a sequence number greater than or equal to that of RREQ, a RREP message will be generated and sent back to the source.

Every RREQ carries a time to live (TTL) value that specifies the time this message should be in the network. This value is set to a predefined value at the first transmission and is increased at retransmissions. Retransmissions occur only if no replies are received.

Cost (rq_cost) parameter carries the average path cost of the path travelled by the RREQ packet. The cost of a node is calculated by the function getCost() given in fig 3.6. rq_cost is the average of the cost of the nodes travelled. Indicator rq_ind carries the count of vulnerable nodes (nodes whose remaining energy is less than threshold energy). Whenever a RREQ is forwarded then the node add its cost and indicator value.

If a RREQ packet reaches a node which is the destination, or has a valid route to the destination, then the node unicasts a route reply message (RREP) back to the source. If a node is destination then it leaves the value of rp_cost and rq_ind as zero. The node gets the next hop from the route table entry of the source node and forwards the RREP packet to next hop node which further add these (hop count, rp_cost, rp_ind) information about the destination. This message has the following format (refer Fig 3.3):

source address	destination address	destination sequence no.	hop count	Cost (rp_cost)	Indicator (rp_ind)	life time
----------------	---------------------	--------------------------	-----------	----------------	--------------------	-----------

Fig 3.3: Format of RREP packet.

t_node	t_value	energy_cal	energy	max_sq_no
--------	---------	------------	--------	-----------

Fig 3.5 Entry in neighbor list

Every node in the network keeps a record of its neighbors in neighbor list. 't_node' is the node id for which the entry is made and t_value is its calculated trust value. 'energy_cal' of t_node x is the calculated independently based on the collected data by the node n, 'energy' of x is the energy published by x, 'max_sq_no' is the sequence number of the latest hello packet arrived at node n from node x.

For calculating the energy and trust value of the neighbors the protocol first stores the number of data received and sent by the node in the send_data and rcv_data tables.

Node	Number of packets
A ₁	5
A ₂	3
.	.
.	.
.	.
A _x	4

Fig 3.6 Structure of send_data table and rcv_data table

dt_src	dt_dst	dt_nop
--------	--------	--------

Fig 3.7 Data Table Entry Data structure

In the data table (format shown in fig 3.7) the count of packet is stored for which there has been no confirmation The data table stores the number of packets (dt_nop) exchanged between the dt_src(packet source node) and dt_dst(packet destination node). Remaining energy for the nodes is calculated based on the positive decrement in dt_nop. Every count of packets received from dt_src by dt_dst is subtracted from dt_nop and every count of packet sent is added.

Let's say node had received a hello packet from node A stating that node A has sent 10 data packets to node b. The data table will have the entry as shown in fig 3.8.

A	B	10
---	---	----

Fig 3.8 Data Table Entry Data structure after receiving hello packet from A

When after some time a hello packet from node B is received at node stating it has received 7 data packets from node A, then the data entry at node is modified (as given in fig 3.9) and energies of node A and node B is decreased.

A	B	3
---	---	---

Fig 3.9 Data Table Entry Data structure after receiving hello packet from B

In EEAODV hello packets (format given in fig 3.10) have an additional role i.e to verify the data information sent by the neighbor node so that trust value can be calculated. Following section discusses this role.

main_node	node_energy		src_seqno	
num	ptype[10]	psrc[10]	pdst[10]	nop[10]
	rp_dst	*ttable		

Fig 3.10 Hello packet Format

When a hello packet is generated, the main_node is the node id of the hello packet generating node and node_energy is its energy. It carries a src_seqno with it, which combined with main_node is its unique identity. A hello packet also carries a pointer to the trust_table of the main_node which has all the information about the trust of all the nodes in the network. It helps the protocol to perform efficiently in high mobility scenarios.

Fig 3.11 shows the structure for a single entry for a trust table. Every node in the network has its own table and has an entry for every other node in the network. Calculation of the trust is discussed in sec 3.8.

t_node	t_value	energy_cal	max_sq_no
--------	---------	------------	-----------

Fig 3.11 Trust table entry data structure

't_node' is node id for which the entry is made, t_value is trust value generated for t_node by the node, 'energy_cal' is the calculated energy by the node for t_node and 'max_sq_no' is the maximum value of sequence number of packet if received from t_node. Detailed trust management is discussed in

3.2. Role of Hello Packets.

In the network whenever a node sends or receives a data packet, it makes the entry in the send_data table or rcv_data table (fig 3.6). Whenever the timer for a hello packet is triggered, sendHello() function is called and procedure as given in the fig 3.12 takes place.

When a sendHello() is triggered the send_data and receive_data are checked for any data. If the node has neither sent nor received any data in between the hello interval then hop_count value is

Let us say node A sends a data packet to node B (refer Fig 3.13) then A does an entry in send_data table for B and B does it in recv_data for A. When the sendHello() function is called A broadcasts the hello packet with hop_count = 1 to inform its neighbors. But B broadcasts the hello packet with hop_count is initialized as 2 so that it reaches neighbors of A and they can verify the information of data transaction between A and B.

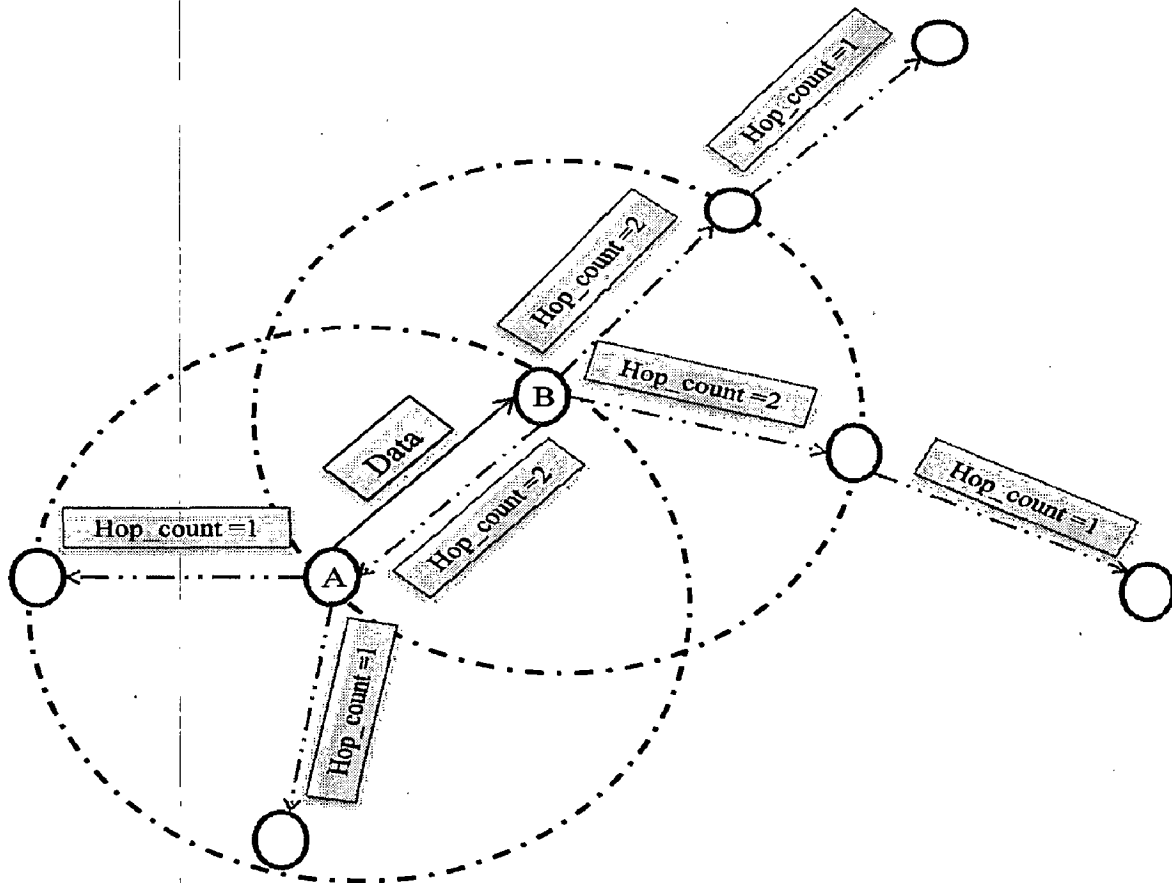


Fig 3.13 Hello packet traversal

Whenever a hello packet arrives at a node `recvHello(p)` function is called. The `src_seqno` value of the packet is matched with the `max_seq_no` of the `main_node`. If the `src_seqno` is less than `max_seq_no`, then the packet has previously arrived at the node and is dropped.

The value of `num` gives the information about the number of links whose data for number of packet received/sent is in the packet. For every value of `num`, the values in `psrc[]` (array packet source), `pdst[]` (array packet destination) and `nop[]` (array number of packet) is updated in `data_table`. Variable `ptype[]` states the type of entry (receive or send). But the entry is only made when the packet source or packet destination is the neighbor. When all the information is extracted and saved from the hello packet then its `hop_count` is decremented and if `hop_count` is 1 then the packet is broadcasted. Fig 3.14 more clearly explains the flow.

3.3. Control Packets

3.3.1 Hello Packet

In simple AODV protocol there is no separate structure for Hello packets. They are initialized and implemented via AODV_REPLY packets. In EEAODV this control packet has 2 main utility:

- To detect the link failure (the conventional function like in AODV).
- To verify the data information sent by the neighbor nodes.

The detail of how it is implemented is already discussed in sec 3.2. Fig 3.15 shows the main header structure of hello packet.

```
Struct hdr_aodv_hello {
    u_int32_t      num;                // not greater than 10
    bool          ptype[10];          // 0-> send 1-> recieve
    nsaddr_t      psrc[10];           // packet source
    nsaddr_t      pdst[10];           // packet destination
    u_int32_t      nop[10];           // number of packets

    nsaddr_t      main_node;          // main packet generator
    float         node_energy;        // packet generator's energy
    u_int32_t      src_seqno;         // Destination Sequence Number

    nsaddr_t      rp_dst;             // Destination IP Address
    double        rp_lifetime;        // Lifetime
    u_int8_t      rp_hop_count;       // Hop Count
    trustTable    *ttable;

};
```

Fig 3.15 hello packet header structure

Description of the fields:

- **main_node**: the node id of the packet generating the hello packet
- **node_energy**: remaining energy of the main_node
- **src_seqno**: sequence number of the packet.
- **rp_dst**: desitnation node id.
- **rp_lifetime**: time of which the node can roam in the network.
- **rp_hop_count**: maximum number of hop it can travel. It can be 1 or 2.

3.3.2 Route Request Packet

As described in section 3.1, a route request packet (RREQ) is broadcasted when the source doesn't have a route entry for destination in its routing table. Fig 3.16 shows the RREQ packet header structure. Fields in detail has been explained earlier.

```
struct hdr_aodv_request {
    u_int8_t    rq_type;
    u_int8_t    rq_hop_count;

    nsaddr_t    rq_dst;
    u_int32_t   rq_dst_seqno;
    nsaddr_t    rq_src;
    u_int32_t   rq_src_seqno;

    double      rq_timestamp;
    double      rq_cost;
    double      rq_ind;
}
```

Fig 3.16 Route Request packet header structure

Description of fields:

- **rq_hop_count** : at first its value is one and it gets incremented till a route reply is generated
- **rq_dst** : destination node's ip address.
- **rq_dst_seqno**: destination sequence number. When an intermediate node receives a RouteRequest, it either forwards it or prepares a RouteReply if it has a valid route to the destination. The validity of a route at the intermediate node is determined by comparing the sequence number at the intermediate node with the destination sequence number in the RouteRequest packet
- **rq_src**: source node's ip address
- **rq_src_seqno**: packet sequence number
- **rq_timestamp**: time at which request was sent
- **rq_cost**: average cost of the path
- **rq_ind**: number of nodes in path who have remaining energy less than threshold

3.3.3 Route Reply Packet

If a node is the destination, or has a valid route to the destination, it unicasts a route reply message (RREP) back to the source. This message has the following format (refer Fig 3.17):

u_int8_t	rp_type;
u_int8_t	reserved[2];
u_int8_t	rp_hop_count;
nsaddr_t	rp_dst;
u_int32_t	rp_dst_seqno;
nsaddr_t	rp_src;
double	rp_lifetime;
double	rp_timestamp;
double	rp_cost;
double	rp_ind;

Fig 3.17 Route Reply packet header structure

Description of fields:

- **rp_hop_count:** number of nodes between the source and destination node
- **rp_dst:** destination ip address
- **rp_dst_seqno :** destination sequence number
- **rp_src :** source ip address
- **rp_lifetime:** time for which it can be in the network
- **rp_timestamp:** time at which the RREQ was generated, corresponding to which the RREP packet is sent. It is used to compute route discovery latency
- **rp_cost:** average cost of the path
- **rp_ind:** number of nodes in path who have remaining energy less than threshold

3.3.4 Route error

All nodes monitor their own neighborhood. When a node in an active route gets lost, a route error message (RERR) is generated to notify the other nodes on both sides of the link of the loss of this link.

3.4 Sequence numbers

Counting to infinity

The core of the problem is that when X tells Y that it has a path somewhere, Y has no way of knowing whether it itself is on the path - as Tanenbaum[18] notes. So if Y detects a link to Z is broken, but X still has a "valid" path to Z, Y assumes X in fact does have a path to Z. So X and Y will start updating each other in a loop, and the problem named "counting to infinity" arises. EEAODV avoids this problem by using sequence numbers for every route, so Y can notice that X's route to Z is an old one and is therefore to be discarded.

Time stamping

The sequence numbers are the most important feature of AODV for removing the old and invaluable information from the network. They work as timestamps and prevent the EEAODV protocol from the loop problem. The destination sequence number for each destination host is stored in the routing table, and is updated in the routing table when the host receives the message with a greater sequence number. The host can change its own destination sequence number if it offers a new route to itself, or if some route expires or breaks.

Each host keeps its own sequence number, which is changed in two cases:

- Before the node sends RREQ message, its own sequence number is incremented.
- When the node responds to a RREQ message by sending a RREP-message, its own sequence number becomes the maximum of the current sequence number and the node's sequence number in the received RREQ message.

The reason is that if the sequence number of already registered is greater than that in the packet, the existing route is not up-to-date. The sequence numbers are not changed by sending HELLO messages.

3.5. Routing Process

When a network is initialized every node sends a hello packet to its neighbor node to inform about itself.

3.5.1 Route discovery

Route discovery process starts when a source node does not have routing information for a node to be communicated with. Route discovery is initiated by broadcasting a RREQ message. The route is established when a RREP message is received. A source node may receive multiple RREP messages with different routes. It then updates its routing entries if and only if the RREP has a greater sequence number, i.e. fresh information or lower rp_cost (average path cost).

3.5.2 Reverse path setup

While transmitting RREQ messages through the network each node notes the reverse path to the source. When the destination node is found the RREP message will travel along this path, so no more broadcasts will be needed. For this purpose, the node on receiving RREQ packet from a neighbor records the address of this neighbor.

3.5.3 Forward path setup

When a broadcast RREQ packet arrives at a node having a route to the destination, the reverse path will be used for sending a RREP message. While transmitting this RREP message the forward path is setting up. One can say that this forward path is reverse to the reverse path. As soon as the forward path is built the data transmission can be started. Data packets waiting to be transmitted are buffered locally and transmitted in a FIFO-queue when a route is set up. After a RREP was forwarded by a node, it can receive another RREP. This new RREP will be either discarded or forwarded, depending on its destination sequence number or the rq_cost or the trust value of the neighbor node:

- if the new RREP has a greater destination sequence number, then the route should be updated, and RREP is forwarded
- if the destination sequence numbers in old and new RREPs are the same, but the new RREP has a smaller hop count or lower average path cost or the difference between trust value is greater than the allowed then this new RREP should be preferred and forwarded
- Otherwise all later arriving RREPs will be discarded

3.5.4 Link breakage

Because nodes can move link breakages can occurs. If a node does not receive a HELLO message from one of his neighbors for specific amount of time called HELLO interval, then

- the entry for that neighbor in the table will be set as invalid.
- the RERR message will be generated to inform other nodes of this link breakage.

RERR messages inform all sources using a link when a failure occurs.

3.6 Energy Calculation Model

Energy Model, as implemented in ns2, is a node attribute. The energy model represents level of energy in a mobile host. The energy model in a node has a initial value which is the level of energy the node has at the beginning of the simulation known as initialEnergy_. It also has a given energy usage for every packet it transmits (txPower_) and receives (rxPower_).

Data table helps in calculating the current energy of the neighbor node. By calculating energy of the node we can determine that it is trustable or not.

There are two types of malicious nodes considered by the protocol:

- a. Nodes which publishes its energy less than original remaining energy.
The primary aim of these types of nodes is to deny the routing service by show themselves as vulnerable. In a simple load balancing protocol these type of node may take advantage and free them from the routing work.
- b. Nodes which publishes its energy more than original remaining energy.
The basic logic of a load balanced routing protocol is to select the node which has maximum energy and route packet through it so that the total lifetime of network increases. These types of nodes can take advantage of this for overhearing the data traffic and launch attack on the basis of collected data. The trust definition equation is different for these types of nodes. It is as follow:

$$T_{x(new),y} = \psi \{ T_{x,y}, k \}$$

$$\psi = \begin{cases} T_{x,y} - \alpha & t_x \geq k \\ T_{x,y} - (k_2 - T_{x,y}) & t_x < k \end{cases} \quad (1)$$

$T_{x(new),y}$ = new trust value of node x (x should be a neighbor) for node y.

k = Threshold trust value.

α, k_2 = constant.

The trust pattern of these types of nodes is shown in graph 3.1.

3.7 Assumptions

While designing the proposed protocol following scenarios are assumed:

- A node never reappears after leaving the network.
- A malicious node case discussed in 3.6.a is assumed to be a data packet dropping node.

3.8 Trust Model

3.8.1 Definition/ trust equation

We have to calculate trust for two types of nodes:

(a) Neighbor nodes:

The following equation is for nodes which are malicious and publish energy less than original.

$$T_{x(new),y} = \psi \{T_{x,y}, k\}$$

$$\psi = \begin{cases} T_{x,y} - \alpha & T_{x,y} \geq k \\ 2T_{x,y} - k^2 & T_{x,y} < k \end{cases} \quad (2)$$

Meaning of the above used symbols is as follow:

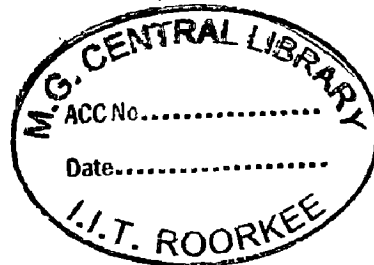
$T_{x(new),y}$ = new trust value of node x (x should be a neighbor) for node y.

k = Threshold trust value.

α, k^2 = constant.

(b) Non neighbor nodes:

$$T_{x(new),y} = \sum_{i=0}^n [W_i(y) \times T_x(i)] \quad (3)$$



Description:

$T_{x(new),y}$ = new trust value of node x (x should be a neighbor) for node y.

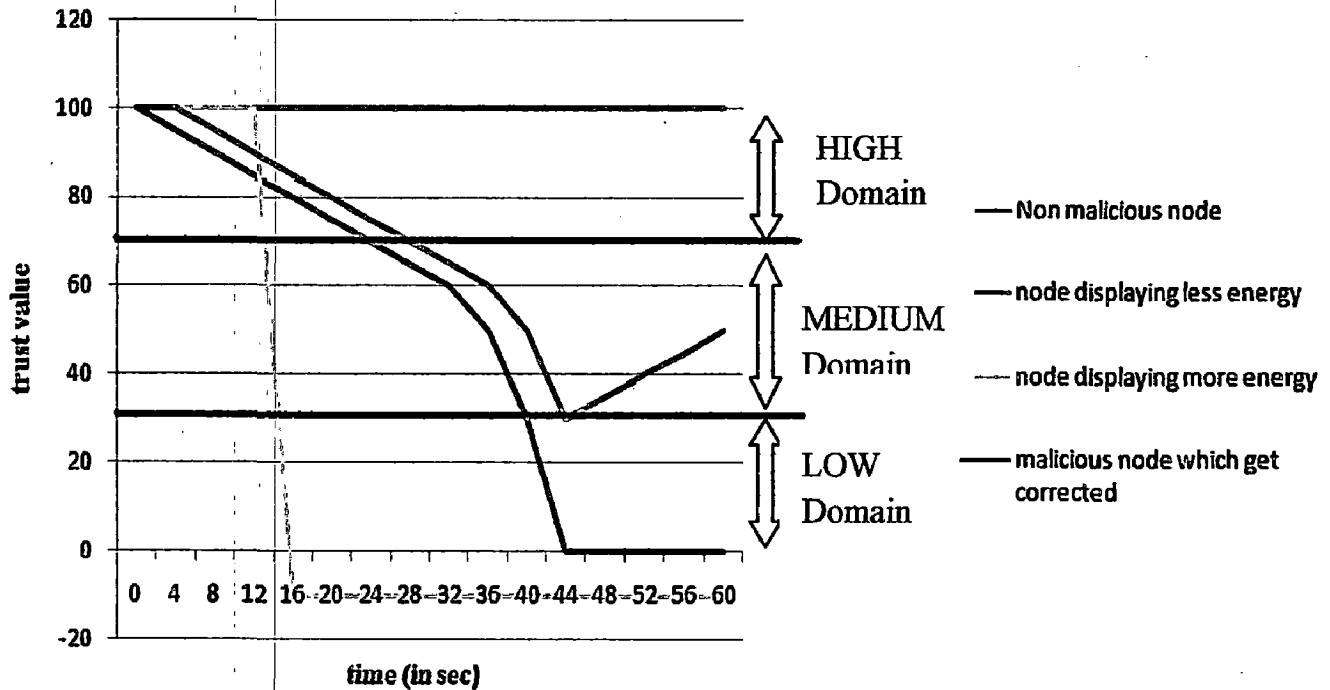
$W_i(y)$ = weight of node i for node y. (ratio of contribution)

$$W_i(y) = T_x(y) / \sum_{a=0}^n T_x(y)$$

$T_x(i)$ = trust value of node x for node i.

3.8.2 Evaluation Process

The trust evaluation process is classified into three phases: initial phase, update phase and re-establish phase as shown in graph 3.1. In the graph $k=70$. $K_2=80$. Here we are considering only the neighbor nodes.



Graph 3.1 Trust value of a node vs. simulation time

3.8.2.1 Trust Initial Phase

When a new node joins a network which already exists, other nodes in this network have no traffic statistics about this new node. As a result, the new node does not have any trust information about its neighbors and vice-versa. Initially the trust value (t_value) of every node is 100.

3.8.2.2 Trust Update Phase

As discussed in sec 3.3 malicious nodes are of two types and hence their trust update phase graph is different.

- a. *Nodes which publishes its energy less than original.*
In this phase the trust gets updated according to equation (2).
- b. *Nodes which publishes its energy more than original.*
In this type of node the trust gets updated according to equation (1).

3.8.2.3 Trust Re-establish Phase

Some nodes will be defined as malicious nodes because of the linking errors or the battery exhausting. Therefore, a redemption mechanism is needed for “malicious” nodes to regain the trust of other nodes. According to the value $T_{x,y}$, N_x won't choose N_y as its forwarding node when N_y , whose trust is in “Low” domain, intends to establish trust with its neighbor N_x . Consequently, the trust value needs increasing periodically to TM , which is the re-establishment process for nodes' trust. The re-establishment function is defined as:

$$T_{x(new),y} = \delta \{T_{x,y}, k\}$$

$$\delta = T_{x,y} + \alpha \quad (4)$$

The slope the re-establishment phase is constant α . The slope when the malicious node crosses the threshold trust value is more than the re-establishment phase. So the trust decrement is faster compared to the increment.

3.9 Parameter for evaluating the protocol

There are many parameters which affect the affects a protocol performance. But for the proposed protocol vulnerable node density and malicious node density should be the basic evaluating parameter as its basic aim is to measure the trustworthiness of the node and to increase network lifetime.

The network lifetime can be defined in several ways, such as the time to the first node failure due to battery outage, the time to the unavailability of application functionality, or the time to the first network partitioning. But here we will consider it as time to the failure of 80% of the nodes.

CHAPTER 4

Implementation and Simulation

4.1 Simulator

For simulation we have used ns2.34 simulator [21][22]. It stands for network simulator. It is used in the simulation of routing protocols, among others, and is heavily used in ad-hoc networking research, and support popular network protocols, offering simulation results for wired and wireless networks alike. It is coded in C++ and provides a simulation interface through OTcl [23], an object-oriented dialect of Tcl[22]. The user describes a network topology by writing OTcl scripts, and then the main ns-2 program simulates that topology with specified parameters.

4.2 Proposed Protocol (EEAODV)

4.2.1 File Dependency of EEAODV Protocol

Fig. 5.1 shows the file dependency of EEAODV Protocol. It is derived from the class Agent, see agent.h.

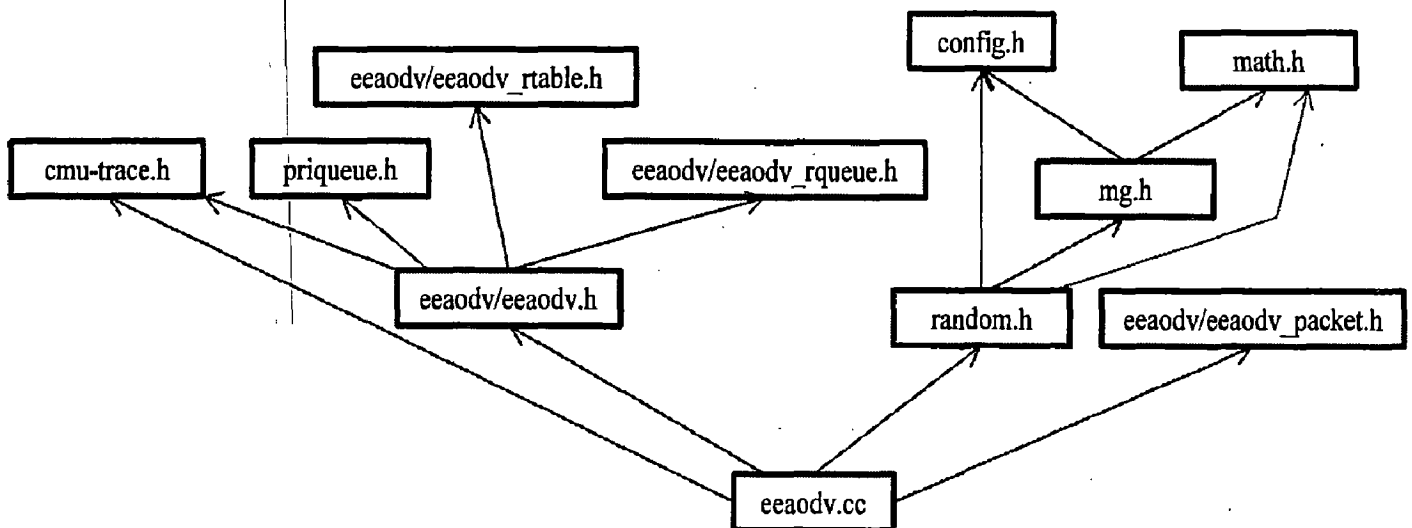


Fig 4.1 File reference of eeadv.cc

4.2.2 Trust Management in EEAODV Protocol

Whenever the hello packet reaches the node, the function `recv_Hello()` is invoked. As described in earlier chapter route selection is on the basis of trust value of neighbor, cost, indicator and `hop_count` (all in decreasing order of priority).

It is obvious that the calculated `energy_cal` is never the very exact value of the energy of the node. So we provide some relaxation in the trust update. Algorithm for trust update is:

Step 1: if $((\text{node.energy_cal} - \text{node.energy}) \leq \text{allowed_energy_difference})$

then goto step 2

else goto step 3

Step 2: decrease the trust value as given in equation (2)

Step 3: increase the trust value as given in equation (4)

By applying the above algorithm we simulated a network of 50 nodes with random coordinates and mobility and repeated it with different `allowed_energy_difference` values. Simulation parameters and procedures are discussed in next section. The Fig 4.2 shows the result. The metrics of measurement are True Positive and False Negative.

$$\text{True Positive} = \frac{\text{Number of malicious nodes declared malicious}}{\text{Total number of malicious nodes in network}}$$

$$\text{False Negative} = \frac{\text{Number of non-malicious nodes declared malicious}}{\text{Total number of non-malicious nodes in network}}$$

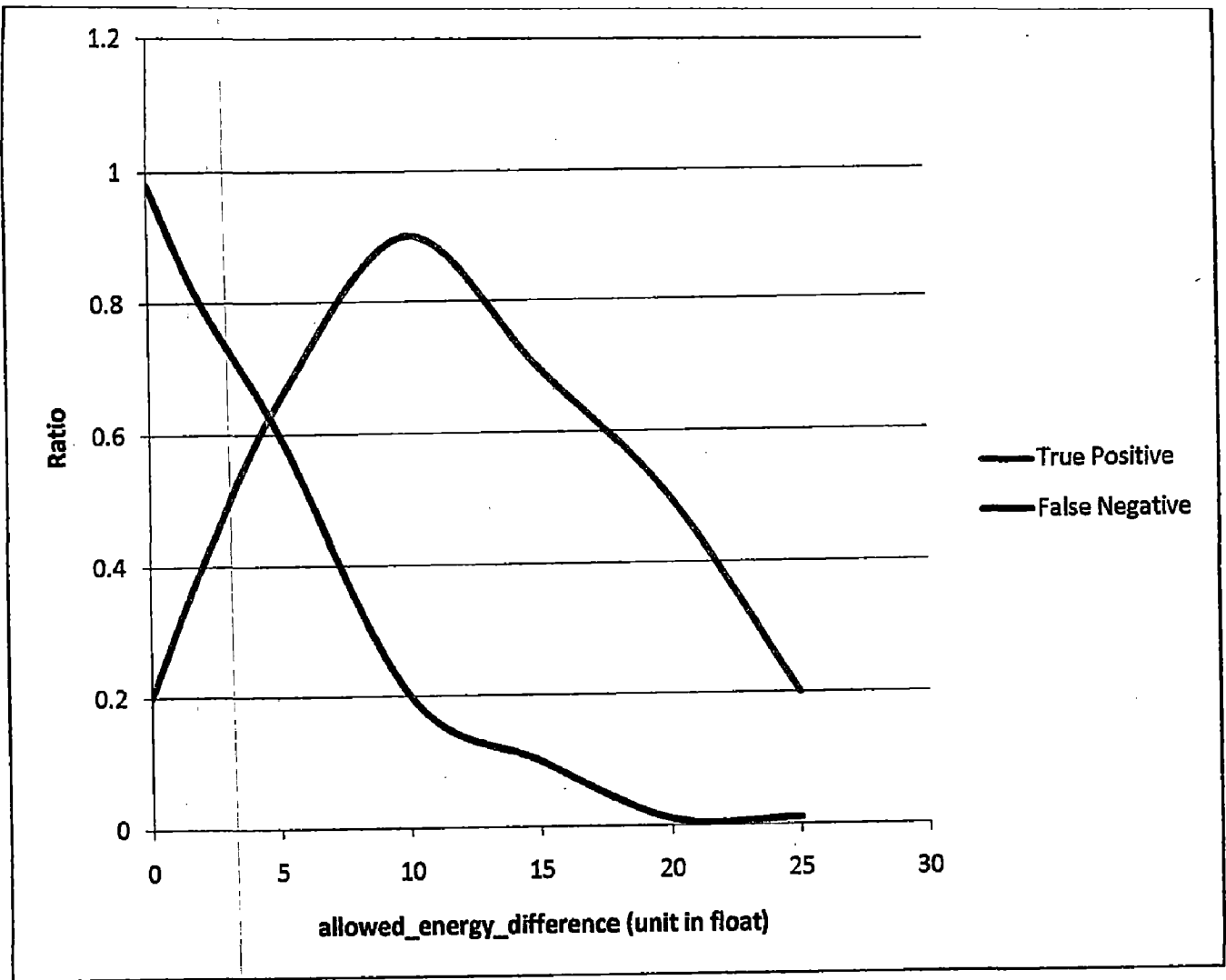


Fig 4.2 Change in True Positive and False Negative with different allowed_energy_difference value.

As the allowed_energy_difference is increased the liberty for comparing the cal_energy and published energy also increases hence as expected the false negative graph decreases. As discussed in chapter 3 the hello packet takes a 2 hop travel time to update the cal_energy of the neighbor node, until then it has the stale data in its routing table. So a minimum time is required by a node to have correct entry for the energy. But at any point of time it is very difficult to calculate exact energy of a node as node's energy decreases variably depending on its state (sleep or active).

As deduced from the graph in fig 5.2 the true positive increases for a value but starts decreasing after achieving maxima. The network has variable malicious node which publish variable wrong energies. So with maxima as allowed_energy_difference we can say that a node almost correctly calculates the trust. After the maxima point true positive decreases because now the incorrect published energies start lying in allowed_energy_difference's range.

4.2.3 Enabling Hello Packets

By default HELLO packets are disabled in the aodv protocol. To enable broadcasting of Hello packets, comment the following two lines present in aodv.cc.

```
#ifndef AODV LINK LAYER DETECTION
```

```
#endif LINK LAYER DETECTION
```

4.3 Creating random traffic-pattern for wireless scenarios.

Random traffic connections of TCP and CBR can be setup between mobilenodes using a traffic-scenario generator script. This traffic generator script is available under `~ns/indep-utils/cmu-scen-gen` and is called `cbrgen.tcl`. It can be used to create CBR and TCP traffics connections between wireless mobilenodes. In order to create a traffic-connection file, we need to define the type of traffic connection (CBR or TCP), the number of nodes and maximum number of connections to be setup between them, a random seed and incase of CBR connections, a rate whose inverse value is used to compute the interval time between the CBR pkts. So the command line looks like the following:

```
ns cbrgen.tcl [-type cbr|tcp] [-nn nodes] [-seed seed] [-mc connections] [-rate rate]
```

The start times for the TCP/CBR connections are randomly generated with a maximum value set at 180.0s. Go through the tcl script `cbrgen.tcl` for the details of the traffic-generator implementation.

For example, let us try to create a CBR connection file between 10 nodes, having maximum of 8 connections, with a seed value of 1.0 and a rate of 4.0. So at the prompt type:

```
ns cbrgen.tcl -type cbr -nn 10 -seed 1.0 -mc 8 -rate 4.0 > cbr-10-test
```

From `cbr-10-test` file (into which the output of the generator is redirected) thus created, one of the cbr connections looks like the following:

```

#
# 2 connecting to 3 at time 82.557023746220864
#
set udp_(0) [new Agent/UDP]
$ns_ attach-agent $node_(2) $udp_(0)
set null_(0) [new Agent/Null]
$ns_ attach-agent $node_(3) $null_(0)
set cbr_(0) [new Application/Traffic/CBR]
$cbr_(0) set packetSize_ 512
$cbr_(0) set interval_ 0.25
$cbr_(0) set random_ 1
$cbr_(0) set maxpkts_ 10000
$cbr_(0) attach-agent $udp_(0)
$ns_ connect $udp_(0) $null_(0)
$ns_ at 82.557023746220864 "$cbr_(0) start"

```

4.4 Creating node-movements for wireless scenarios.

The node-movement generator is available under `~ns/indep-utils/cmu-scen-gen/setdest` directory and consists of `setdest{.cc,.h}` and `Makefile`. CMU's version of `setdest` used system dependent `/dev/random` and made calls to library functions `initstate()` for generating random numbers. That was replaced with a more portable random number generator (class `RNG`) available in `ns`. Run `setdest` with arguments as shown below:

```

./setdest [-n num_of_nodes] [-p pausetime] [-s maxspeed] [-t simtime] \
          [-x maxx] [-y maxy] > [outdir/movement-file]

```

Let's say we want to create a node-movement scenario consisting of 20 nodes moving with maximum speed of 10.0m/s with an average pause between movement being 2s. We want the simulation to stop after 200s and the topology boundary is defined as 500 X 500. So our command line will look like:

```

./setdest -n 20 -p 2.0 -s 10.0 -t 200 -x 500 -y 500 > scen-20-test

```

The output is written to `stdout` by default. We redirect the output to file `scen-20-test`. The file begins with the initial position of the nodes and goes on to define the node movements.

```

$ns_ at 2.000000000000 "$node_(0) setdest 90.441179033457 44.896095544010
1.373556960010"

```

This line from `scen-20-test` defines that `node_(0)` at time 2.0s starts to move toward destination (90.44, 44.89) at a speed of 1.37m/s. These command lines can be used to change direction and speed of movement of mobile nodes.

4.5 Trace File Format

In NS-2, the general energy model trace format is given as below:

```
s 0.000000000 0 RTR -- 0 AODV 44 [0 0 0 0] [energy 96.897890 ei 0.010 es 0.000 et 0.000 er
0.003] ----- [0:255 -1:255 1 0] [0x1 1 [0 2] 4.000000] (HELLO)
s 10.000000000 0 RTR -- 0 AODV 48 [0 0 0 0] [energy 96.897890 ei 0.010 es 0.000 et 0.000 er
0.003] ----- [0:255 -1:255 30 0] [0x2 1 1 [1 0] [0 4]] (REQUEST)
```

```
f 1.014124803 _8_ RTR -- 0 AODV 44 [13a 8 1 800] [energy 99.974077 ei 0.010 es 0.000 et
0.007 er 0.008] ----- [1:255 0:255 29 7] [0x4 2 [1 4] 100000.000000] (REPLY)
```

A generalized explanation of trace format is given in table 5.1.

Sr.No	Purpose	Remarks
1	It shows the occurred event	's' SEND, 'r' RECEIVED, 'D' DROPPED
2	Time at which the event occurred	10.000000000
3	Node at which the event occurred	Node id like 0
4	Layer at which the event occurred	'AGT' application layer, 'RTR' routing layer, 'LL' link layer, 'IFQ' Interface queue, 'MAC' mac layer, 'PHY' physical layer.
5	show flags	
6	shows the sequence number of packets	0
7	Packet type	'cbr' CBR packet, 'DSR' DSR packet, 'RTS' RTS packet generated by MAC layer, 'ARP' link layer ARP packet.
8	shows size of the packet	Packet size increases when a packet moves from an upper layer to a lower layer and decreases when a packet moves from a lower layer to an upper layer
9	[.....]	It shows information regarding the node's energy
10	[.....]	It shows information about packet duration, mac address of destination, the mac address of source, and the mac type of the packet body.
11	[....]	It shows information about source node ip : port number, destination node ip (-1 means broadcast) : port number, ip header ttl, and ip of next hop (0 means node 0 or broadcast).

Table 4.1 Trace file format

The processing of the generated trace file is done by using awk [24] files.

CHAPTER 5

Results and Discussion

This section discusses the simulation results of the protocol proposed in Chapter 3.

5.1 Performance Metric.

The performance metric we have used are network lifetime and packet drop. They were briefly discussed in Section 3.5. We have performed experiments by running different simulation scenario files and different mobility patterns by following the procedures given in Section 4.3 and Section 4.4 and compared AODV and EEAODV protocol.

5.2 Simulation Setup

The simulation file (program.tcl) is a OTCL script. Fig 5.1 shows the initialization and definition of the basic simulation parameters.

<i>set val(chan)</i>	<i>Channel/WirelessChannel ;</i>	<i># channel type</i>
<i>set val(prop)</i>	<i>Propagation/TwoRayGround ;</i>	<i># radio-propagation model</i>
<i>set val(netif)</i>	<i>Phy/WirelessPhy ;</i>	<i># network interface type</i>
<i>set val(mac)</i>	<i>Mac/802_11;</i>	<i># MAC type</i>
<i>set val(ifq)</i>	<i>Queue/DropTail/PriQueue ;</i>	<i># interface queue type</i>
<i>set val(ll)</i>	<i>LL;</i>	<i># link layer type</i>
<i>set val(ant)</i>	<i>Antenna/OmniAntenna ;</i>	<i># antenna model</i>
<i>set val(ifqlen)</i>	<i>50 ;</i>	<i># max packet in ifq</i>
<i>set val(nn)</i>	<i>50;</i>	<i># number of mobile nodes</i>
<i>set val(rp)</i>	<i>AODV;</i>	<i># routing protocol</i>
<i>set val(x)</i>	<i>1000;</i>	<i>#X dimension of the topography</i>
<i>set val(y)</i>	<i>1000;</i>	<i>#Y dimension of the topography</i>
<i>set val(energymodel)</i>	<i>EnergyModel ;</i>	
<i>set val(initialenergy)</i>	<i>100;</i>	<i># Initial energy in Joules</i>
<i>set val(sleeppower)</i>	<i>0.0005;</i>	<i>#sleep power W 15 uA P=I*I*R=11.25*10E-9 W</i>
<i>set val(tp)</i>	<i>0.0002;</i>	<i>#transition power consumption (Watt)</i>
<i>set val(tt)</i>	<i>0.0005 ;</i>	<i>#transition time(second) use instate transition from sleep to idle (active)</i>
<i>set val(ip)</i>	<i>0.01;</i>	<i>#idle power</i>

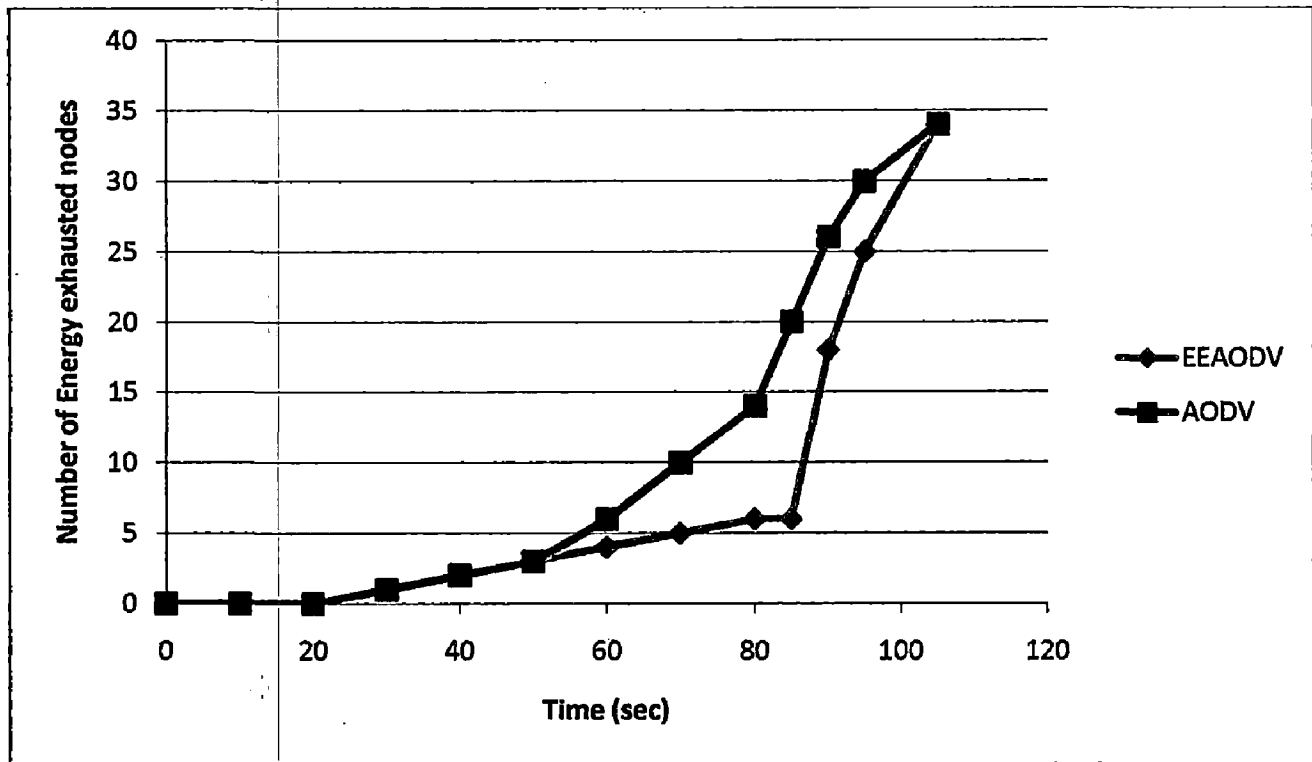
Fig 5.1 Simulation Parameters Used for simulation

5.3 Effect of network run time on the energy of the nodes.

The below graph compare number of exhausted nodes (node whose remaining energy is zero) with respect to time by applying AODV and EEAODV protocol on a network.

Simulation time: 150 sec.

Total Number of nodes: 50.

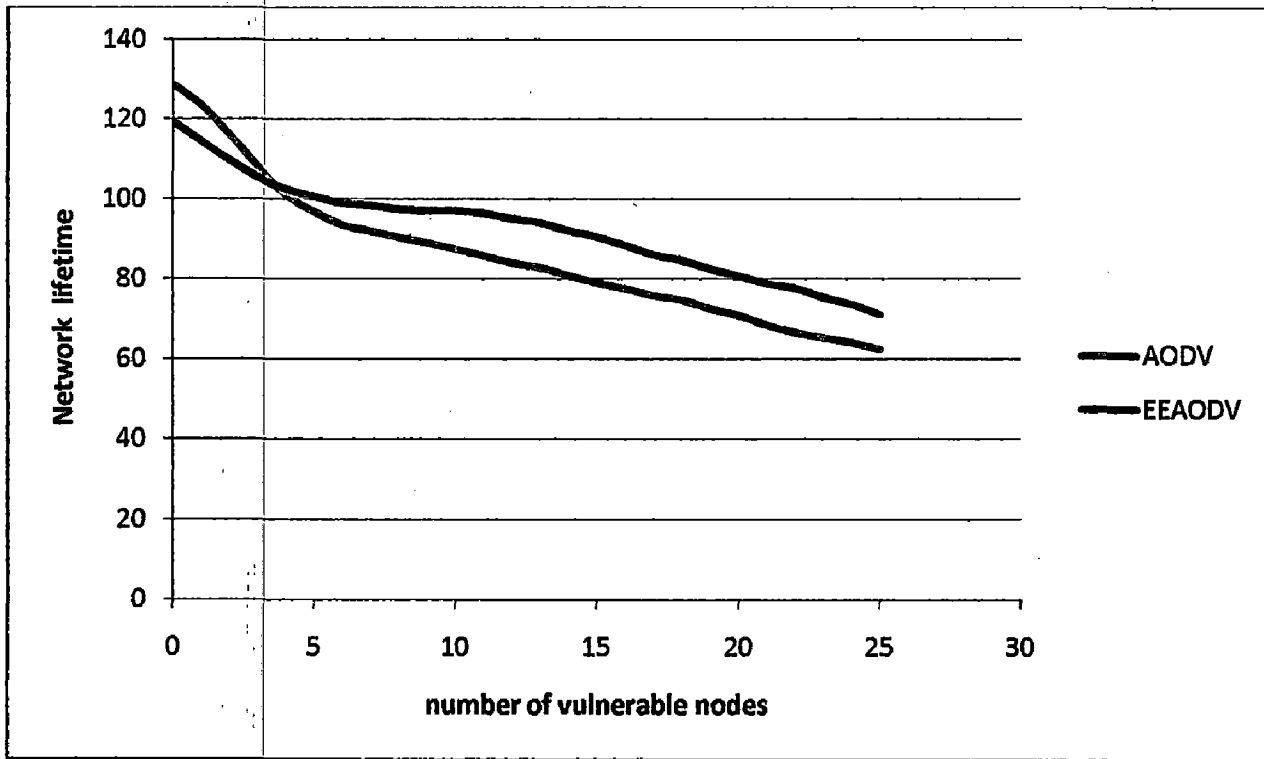


Graph 5.1 Simulation time vs Number of Energy exhausted nodes

Graph 5.1 clearly shows that energy exhaustion rate of network by applying AODV protocol rapidly increased after 30 % of the simulation time. On contrary by applying EEAODV protocol the rate is stable till 60 % of the simulation time, but it rapidly catches up AODV after that.

As the simulation time increases in the AODV applied network energies of the node from a single path heavily decreases and hence get exhausted (no remaining energy), but in the EEAODV applied network comparatively less number of nodes get exhausted because it finds alternate path whose remaining energy is more. But as the simulation time ends the total exhausted number nodes will be coincides.

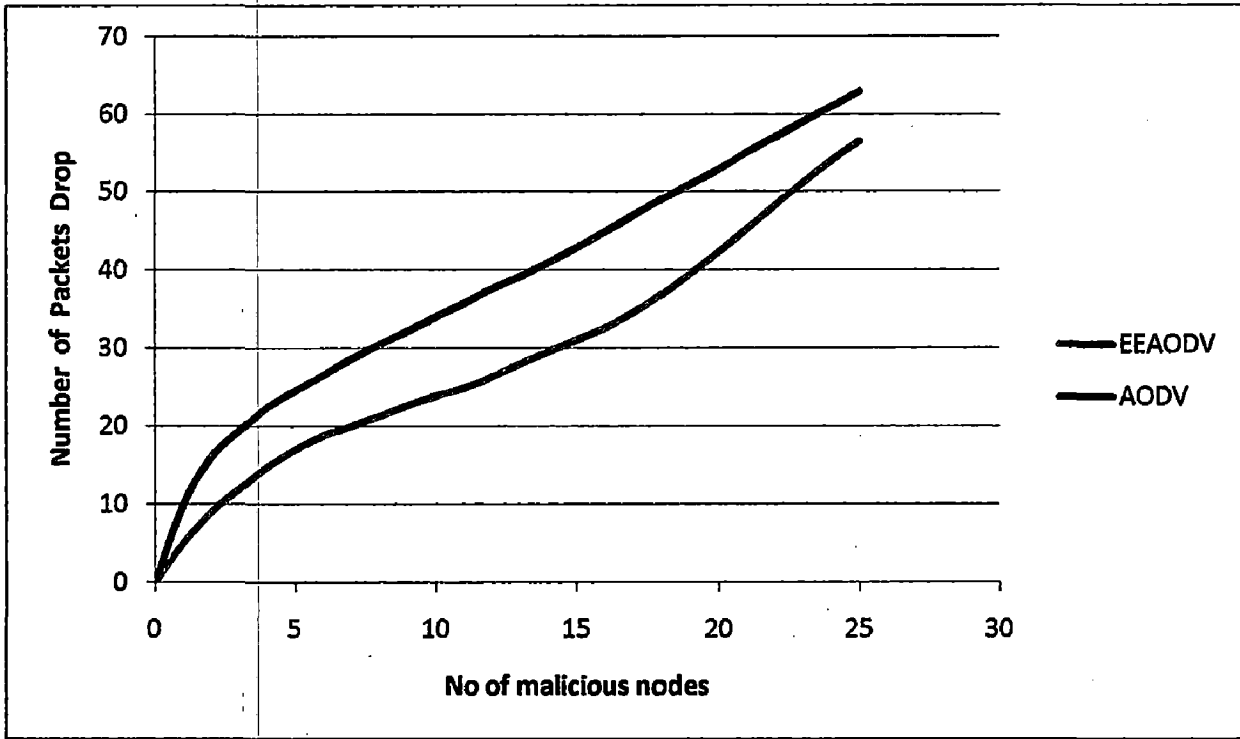
5.4 Effect of number of vulnerable node on network lifetime



Graph 5.2 number of vulnerable nodes vs network lifetime

From the above graph we can see that as at the starting the network lifetime of AODV is more than EEAODV this is due to the extra overhead EEAODV introduces to the routing which initially decreases the network lifetime. But as the number of vulnerable nodes increases EEAODV protocol comparatively increases the network lifetime by successfully applying load balancing. In the range of number of vulnerable nodes {10-15} i.e. 20-30% EEAODV shows the best network lifetime difference. But as the vulnerable nodes crosses 40% limit the network lifetime difference decreases.

5.5 Effect of number of malicious node on number of packet drop



Graph 5.3 Number of Malicious nodes vs. number of packet Drop

As deduced from the graph the EEAODV is best in performance when the number of malicious node is in {10, 20} range i.e. 20-40 % range (total number of node = 50).

Chapter 6

Conclusion & Future Work

Battery life is one the most challenging resources of mobile adhoc networks. Many protocols have been proposed to save the battery power and to increase the network lifetime. But they all rely on the information of remaining energy given by the respective node. So their performance decreases if some of the nodes are malicious in the network.

To overcome this problem a trust based energy efficient load balanced protocol has been proposed. It detects the malicious nodes hence maintaining a descent network throughput and increases the network lifetime by performing energy efficient load balanced routing.

FUTURE WORK

In the network we have maintained energy_cal for every node in the network but when a node move out of the network for a significant time and returns back then the calculated energy (energy_cal) may be wrong. Future work should be a mechanism to timestamp the last trace of the node and to calculate the remaining energy according to the time difference.

In the proposed protocol we have not considered the case when a malicious node may broadcast a hello packet originating from its neighbor to validate its false remaining energy information. A mechanism can be proposed to validate the hello packet originator.

References

- [1] C. Sivaram Murthy and B.S Manoj, "Ad Hoc Wireless Networks", Pearson Education, Second Edition India, 2001.
- [2] "Tutorial on wireless ad hoc networks" by David Remondo, Second International Conference in Performance Modeling and Evaluation of heterogeneous networks, July 2004.
- [3] M. Blaze, J. Feigenbaum, and J. Lacy, "Decentralized Trust Management," Proc. IEEE Symposium on Security and Privacy, 1996.
- [4] K.Seshadri Ramana, Dr. A.A. Chari and Prof. N.Kasiviswanth "A SURVEY ON TRUST MANAGEMENT FOR MOBILE AD HOC NETWORKS", Proc of International Journal of Network Security & Its Applications (IJNSA), Volume 2, Number 2, April 2010
- [5] Title: "List of Routing Protocols for an ad-hoc networks" online available at "http://en.wikipedia.org/wiki/List_of_ad_hoc_routing_protocols"
- [6] Charles E. Perkins and Pravin Bhagwat, "Highly Dynamic Distance Destination-Sequenced-Vector Routing," Volume 24 , Issue 4 (October 1994) Pages: 234 - 244 Year of Publication: 1994 ISSN:0146-4833.
- [7]Bellman ford tutorial (http://www.csanimated.com/animation.php?t=Bellman-Ford_algorithm)
- [8] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum and L. Viennot, "Optimized Link State Routing Protocol for Ad-hoc Networks," Multi Topic Conference, 2001. IEEE INMIC 2001, Technology for the 21st Century. Proceedings. IEEE International. Issue Date: 2001.
- [9] Radhika D. Joshi and Priti P. rege, "Distributed Energy Efficient Routing in Ad-hoc Networks," in 978-1-4244-3328-5/08 in IEEE 2008.
- [10] Xiaoying Zhang, Thomas Kunz, Li Li and Oliver Yang, "An Energy-efficient Broadcast Protocol in MANETs," Communications Networks and Services Research Conference, Proceedings of the 2010 8th Annual Communication Networks and Services Research Conference, Pages: 199-206 SBN: 978-0-7695-4041-2.
- [11] David B. Johnson and David A. Maltz, "Dynamic Source Routing in Ad-hoc Wireless Network," The Kluwer International Series in Engineering and Computer Science, 1996, Volume 353, 153-181, DOI: 10.1007/978-0-585-29603-6_5 .
- [12] Benamar KADRI, Mohammed FEHAM and Abdallah M'HAMED, "Weight based DSR for Mobile Ad Hoc Networks," in 3rd International Conference on Information and Communication Technologies: From Theory to Applications, 2008. ICTTA 2008. pp. 1- 6, 7-11 April 2008.

- [13] J.-E. Garcia, A. Kallel, K. Kyamakya, K. Jobmann, J.-C. Cano and P. Manzoni, "A Novel DSR-based Energy-efficient Routing Algorithm for Mobile Ad-hoc Networks," in vehicular technology conference 2003 IEEE.
- [14] Charles E. Perkins and Elizabeth M. Royer, "Ad-hoc on demand distance vector Routing," in Proceedings of the Second IEEE Workshop on Mobile Computer Systems and Applications in 1999
- [15] Radhika D. Joshi and Priti P. Rege, "Energy Aware Routing in Ad Hoc network" Proceedings of the Sixth International Conference on Circuits, Systems, Electronics, Control and Signal Processing (WSEAS), Cairo (Egypt), pp 469-475, 2007.
- [16] Jin-Man Kim, Jong-Wook Jang, "AODV based Energy Efficient Routing Protocol for Maximum Lifetime in MANET," aict-iciw, pp.77, Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT-ICIW'06), 2006.
- [17] Vinay Rishiwal and S. Verma, "Energy Aware On Demand Routing for MANET's", Proc. Of IEEE Third International conference on Wireless Communication and Sensor Network (WCSN-2007), pp148-153, 2007
- [18] Chen Jie, Chen Jiapin, Li Zhenbo, "Energy efficient AODV for low mobility Ad hoc Network", Proc of International Conference on Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007. pp1512-1515.
- [19] M. Pushpalatha, R. Venkataraman, and T. Ramarao, "Trust based energy aware reliable reactive protocol in mobile ad hoc networks", World Academy of Science, Engineering and Technology 56 2009
- [20] Tanenbaum A.S Computer Networks. Prentice Hall International, 4th edition, 2003.
- [21] Title: "Marc Greis's tutorial" online Available at (<http://www.isi.edu/nsnam/ns/tutorial/index.html>)
- [22] Title : "Rp-lip Documentation of ns-doc" online Available at (<http://www-rp.lip6.fr/ns-doc/ns226-doc/html/classes.htm>).
- [23] Title: OTcl Tutorial online Available at (<http://www.isi.edu/nsnam/otcl/doc/tutorial.html>).
- [24] Title: Tcl online Available at (<http://en.wikipedia.org/wiki/Tcl>).
- [25] Title: Awk - A Tutorial and Introduction - by Bruce Barnett online Available at (<http://www.grymoire.com/Unix/Awk.html>)