Environment-Aware Trusted Data Delivery in Multipath Wireless Protocols

Mohit Virendra, Arunn Krishnamurthy, Krishnan Narayanan, Shambhu Upadhyaya
Department of Computer Science and Engineering
State University of New York at Buffalo
Buffalo, NY 14260
{virendra, ack6, kn38, shambhu}@cse.buffalo.edu

Kevin Kwiat
US Air Force Research Laboratory
525 Brooks Road
Rome, NY 13441
kwiatk@rl.af.mil

***Abstract*: Current multipath protocols for Multi-Hop Wireless Networks (MWNs) use routes with the least hop-count as the default for data transmission. Given a set of multiple routes between a source-destination pair, selecting the best route(s) should depend on a variety of factors rather than just the hop-count. In this paper we propose a network environment-aware framework for trust-based route selection for MWNs that makes informed and adaptive route-selection decisions. Though we use the Ad-hoc On-demand Multipath Distance Vector (AOMDV) routing as the underlying representative protocol to build our framework on, the design-logic is generic and suitably modifiable for other specific multipath schemes. We use the term "trust" to encompass all such metrics pertaining to the quality of a route, i.e., the predicted probability of packets reaching the destination if sent on that route. In our scheme a node quantifies trust values for its neighboring nodes and for the routes that pass through it. Our scheme uses an incentive-and-penalty system to encourage non-selfish and cooperative node behavior towards routing network traffic. The trust metric framed in this protocol is self learning and adaptive. It adjusts to varying network conditions and quick convergence of the protocol implies that it works well in mobility scenarios. Glomosim simulations demonstrate the improvement in throughput over conventional AOMDV under scenarios of congestion, link failures and route unreliability. Simulations also demonstrate the quick convergence of route trusts corresponding to changing route conditions and the corresponding rescheduling of traffic resulting in the increased throughput.**

Keywords: AOMDV, Multipath, Routing, Security, Trust

I. INTRODUCTION

Mobile Multi-Hop Wireless Networks (MWNs) have dynamic topology due to node mobility. Frequent link breakages necessitating route discovery are not uncommon. Since route discovery algorithms are flooding-based, finding multiple routes between a source and destination is not significantly expensive compared to finding a single route. Increased robustness and improved data-loss-tolerance to path breakages should offset any additional costs in finding multiple routes. Simple enhancements to well-known routing protocols can provide effective multipath solutions.

Marina et al [1], proposed Ad-hoc On-demand Multipath Distance Vector (AOMDV) routing, a multipath variant of the well-known Ad-hoc On-demand Distance Vector (AODV) [2, 3] routing protocol. AOMDV finds multiple routes between a source-destination pair, but leaves route selection decisions, i.e., which route(s) should be used and how much traffic should be scheduled across each route, on the implementer's discretion.

Route selection for any multipath protocols may be affected by congestion, presence of selfish (or malicious) nodes, and other adverse network or physical conditions. However, existing multipath protocols (including AOMDV) would be transparent to such ambient conditions. For example, AOMDV would only evaluate and utilize information pertaining to number of hops and route freshness. Under real deployment conditions a naive approach towards route selection may translate into severe throughput degradation and may incur heavy losses. On the other hand, additional information on the nature of routes may enhance the probability of packets reaching the destination.

A. *Motivation*

Lack of information regarding participating nodes and route conditions in a network can cause security infringements like colluding black hole attacks [4] and hinder the efficacy of data transfer between nodes. Adaptive schemes for evaluating ambient route conditions are in order for MWNs. Neighboring nodes can be a first-step in determining such conditions. Nodes can recursively query their next hop neighbors along a route to discern the condition of the route further downstream or upstream. This is feasible as nodes can be in direct communication only with other nodes in their radio range.

The definition of the word *trust* in the dictionary is, "reliance on the integrity, strength, ability, surety, etc., of a person or thing; confidence" [5]. Trust should dynamically update, conforming to any changes in confidence on others. We utilize this notion of trust to design a framework for nodes in MWNs to make informed route selection decisions. Dynamic topology and distributed nature of MWNs prohibits utilizing any infrastructure-based solutions. Thus, our trust framework should be infrastructure-less, adaptive, self-learning and functional in the presence of malicious or selfish nodes. It should be able to estimate the degree of reliability/unreliability of routes due to physical and network conditions. Such a selection process should arguably enhance data delivery robustness.

B. Problem Statement

Our goal in this paper is to devise a trust-metric which can be used as a reference for nodes to make routing and traffic scheduling decisions in multipath scenarios. The problem that we want to address here is the effective computation of route trust metrics which are quickly adaptable to changing route conditions. This continuously-evaluated trust should allow nodes to dynamically switch the traffic distribution across different available routes in response to changing route conditions. It is important to distinguish that the design goal of our framework is not to detect malicious nodes. Rather the goal is to quickly detect any *effects* on data transfer, attributable to malicious nodes or other adverse conditions in a route, and to take corrective actions (e.g., use alternate routes).

C. Contributions

The main contribution of this paper is a framework that allows nodes to evaluate route conditions and provides them with metrics to make informed routing decisions in MWNs using multipath protocols. Though AOMDV is the underlying representative protocol and the test bed for the framework in this paper, the design-logic is generic and suitably modifiable for other specific multipath schemes.

The specialty of the proposed framework is its simplicity, yet its effectiveness in evaluating changing route conditions, visible in the enhanced throughput and robust data delivery as compared with native multipath routing. The simplicity is attributable to the logic that all conditions impacting data transmission along a route can be clubbed under the term trust for the purpose of robust data delivery (see Sec. III B). In our scheme a node quantifies trust values for its neighboring nodes and for the routes that pass through it. Our scheme uses an incentive-and-penalty system to encourage non-selfish and cooperative node behavior towards routing network traffic. The trust metric framed in this protocol is self learning and adaptive. It adjusts to varying network conditions and quick convergence of the protocol implies that it works well in mobility scenarios.

D. Paper Organization

The rest of the paper is organized as follows: Section II presents related work. A brief overview of AODV and AOMDV protocols and the underlying assumptions in our model are presented in Section III. Section IV presents an outline of the trust framework and describes its implementation through modifications to AOMDV. Section V describes the framework in detail. Section VI evaluates the performance of our framework and compares it with two variants of AOMDV using Glomosim [6] simulations. Section VII concludes the paper with a discussion of proposed future work.

II. Related Work

Since our framework combines concepts from diverse backgrounds, we: (a) first provide a brief overview of the current trust-based models for WMNs, (b) enumerate the relevant literature in robustness and security of multipath protocols next, and (c) outline the few known works that combine trust and routing.

A. Trust in Networks

Some of the earliest trust models to improve security and confidentiality were proposed by Beth et al. [7], and Rahman et al. [8]. These models envisioned applications like web based negotiations, third party arbitrations and e-commerce. Zhou and Haas [18] proposed a technique that uses threshold cryptography to distribute trust in ad-hoc networks for the purpose of key management. Their work mainly focused on confidentiality and node authentication. Several trust models for assessing node dependability, reliability and security in ad-hoc networks and other P2P systems have since been proposed in literature [10, 11, 12, 13]. In a recent article Li et al. [9] provide a summary of such models. These models develop means to evaluate trustworthiness of nodes based on their behavior. Promiscuous monitoring by peer nodes plays a significant role in trust computation in all these models. Promiscuous monitoring is expensive, resource consuming and preferably avoidable in WMNs. A salient feature of our framework is that it minimizes promiscuous monitoring by making nodes accountable for cooperation in data forwarding. Besides, we take a holistic approach by evaluating route-performance rather than focusing on detecting individual node-misbehavior. Our concern is to quickly detect the potential *impact* of any node misbehavior (or other adverse route conditions) on data forwarding, and to take swift remedial measures.

B. Multipath Protocols and Routing Misbehavior

Minimizing the impact of attacks and misbehavior on routing protocols in MWNs [21] has been an area of active research. Zapata et al. [22], use one-way hash chains and digital signatures to make AODV secure. However, their work does not provide any information on route dependability. Some recent works have evaluated the robustness and attack resiliency of multipath protocols in WMNs. Kotzanikolaou et al. [19] perform a comparative analysis of the impact of invisible node attacks and sybil [20] attacks on multipath protocols versus conventional single-path protocols. Duan et al. [18] demonstrate the superiority of multi-path routing protocols over traditional single-path protocols in terms of resiliency against blocking and node isolation-type attacks. Our framework assumes that data sent on multipath protocols can be secured using the above cryptographic and security frameworks. These schemes assure confidentiality; our framework aims for robustness.

C. Trust and Routing

Marti et al. [23] proposed the watchdog and the pathrater mechanisms to collect statistics and compute trust on routes for the Dynamic Source Routing (DSR) protocol. Li, Lyu and Liu [15] proposed a model which uses both cryptography and trust. Zouradaki et al. [16] propose an observation and opinion based trust and confidence framework which incorporates "second hand trustworthiness from third party values". Their framework performs extensive computations to gather and evaluate trust opinions from nodes that are not direct neighbors, and substantiates it with accumulating evidence from neighbors. All these works again rely extensively on continuous promiscuous monitoring of neighboring nodes. In Sun et al.'s framework [14], the nodes request recommendations and select a route based on route quality. A node has to select a subset of nodes with trust values higher than a threshold to determine whose recommendations to use. It is mentioned in the paper that the Trust Recommendation Request (TRR) messages constitute a significant share of overhead and this overhead increases with the size of the trust chains. Besides the rules for trust updating and management are extensive and this may prevent quick computation in mobility scenarios. Our framework does not introduce any extra control overhead for trust computation, and a node does not seek trust recommendations from multiple nodes. The convergence time for trust computation in our framework is the same as that of AOMDV, i.e. equivalent to the underlying routing protocol.

III. Background and Assumptions

For conceptual understanding, we present a brief overview of the AODV and AOMDV protocols and then outline the assumptions of our framework.

A. Overview: AODV and AOMDV

AODV has two basic operations: Route Discovery and Route Maintenance. Route Discovery is initiated when a source node wants to find a route to a new destination or when the lifetime of an existing route to a destination has expired. The source node broadcasts a *Route Request* (RREQ) packet which is in turn re-broadcasted by the nodes' neighbors until the sought route is discovered. Upon receiving an RREQ, an intermediate node with a 'fresh enough' route to the destination or the destination node itself unicasts a *Route Reply* (RREP) packet back to the source node. This is possible because each node receiving the RREQ caches the route back to the originator of RREQ. AODV also uses the *Route Error* (RERR) and *Route Reply Acknowledgement* (RREP-ACK) control packets for route management.

AOMDV provides options for discovering link-disjoint or node-disjoint multi-paths between pairs of nodes. The crux of AOMDV lies in ensuring loop freedom and path disjointedness through modifications to the route update rules of the parent AODV protocol. AOMDV ensures loop freedom through Sequence Number Rule, Route Advertisement Rule and Route Acceptance Rule. It maintains link-disjoint paths by restricting each node in the route to have a unique next hop as well as a unique last hop. If common nodes in a set of link-disjoint paths prevent other upstream nodes from having more than one path through them, node-disjoint paths can be obtained. This can be achieved by stipulating that for a given destination sequence number, every node advertises one single designated path to other nodes.

B. Assumptions

Malicious node detection is outside the scope of this paper. We do not deem it important to distinguish between packets dropped due to malicious or selfish behavior and packets dropped due to congestion. As long as such actions are inhibitive towards the ultimate goal, i.e., efficient data transfer, they are all worthy of loss of trust. Our model ensures that as long as such behavior is exhibited, data is not sent through those routes. However, we do assume message confidentiality and tamper-proofing through schemes mentioned in Sec. II B, when necessary. Thus, information will not be compromised when our protocol is executing in the presence of malicious nodes. Nodes have unique ids that are non-forgeable. All links are bidirectional, though the cost/capacity of sending data, channel conditions and node

behavior maybe different in each direction. Multiple loop-free braided paths are discovered using AOMDV which, by default offers node or link disjoint paths. Link capacities can be estimated through techniques proposed by Li et al. [24], and other related works. Node and Route trust are assumed to be non-transitive i.e., Txy ≠ Tyx, x and y being the nodes under consideration. Source and destination nodes in a data transfer session are presumed to be non-malicious. Trust is computed on a continuous scale of 0 to 1. Initial node trust values are assumed to be 1 for the destination and 0.5 for all other participating nodes (explained in Sec. V).

IV. Framework Outline and Implementation through AOMDV Modifications

A. Framework Outline

A node maintains two types of trust values, for routes passing through it and for its one-hop neighbors.

- **Route Trust:** Measure of the reliability that packets would reach the destination if forwarded on a particular route. This is computed by each node for all routes in its routing table.
- **Node Trust:** Measure of confidence on one-hop neighbors for accurately assessing and reporting the condition of the route downstream (towards the destination) from them.

Initial node trust is pre-assigned and initial route trust is formalized by Effective Link Capacity (ELC) and Effective Route Capacity (ERC) values (See Sec. V). Subsequent updates to node and route trusts are interdependent. Route trust is recursively computed by each node starting at the destination and moving upstream towards the source. At each hop, a node's assessment of its downstream reporting neighbor (i.e., node trust) is factored into the route trust. In turn, the difference between predicted route trust and the eventual route performance governs an upstream node's trust (node trust) on its downstream neighbor. Thus nodes are accountable for providing an accurate assessment of route conditions. In subsequent sections we see that selfish behavior harms a node's prospects for routing its self-data. Additionally, nodes are not unduly punished if they accurately report route conditions like congestion which are beyond their control. Incentives and penalties are awarded to each neighbor based on the accuracy of their assessment.

To design such an operable trust model, we propose modifications to the existing AOMDV protocol that are described below.

Destination	Seq. #	Adv. Hop Count	QRY Flag	Route List

Node ID	Node Trust
------	------
------	------

Fig. 1a. Route Table

hop_count1	next_hop1	Last_hop1	timeout1	Route Trust1
hop_count2	next_hop2	Last_hop2	timeout2	Route Trust2
:	:	:	:	:
:	:	:	:	:

Fig. 1b. Neighbors' Trust Table

B. AOMDV Modifications

Additions/modifications to AOMDV are made for piggy-backing ELC and ERC values in the RREP packets upstream and maintaining trust details for all routes in each node's routing table. The details are as follows:

- Each node maintains an additional data structure called the Neighbors' Trust Table. It contains neighboring node IDs, and their corresponding node trust (Fig 1a).
- The RREP packet has an additional field to accommodate the route trust (Fig 2a). For every RREP, the intermediate node caches the route trust sent by the downstream node from the Route Trust field. The node then computes its own trust value on the route and updates the Route Trust field with its own route trust value in the RREP packet. The newly computed route trust is also updated against the corresponding destination entry in the routing table.
- A *Type* field in the packet header in AOMDV identifies the control packet type (RREQ, RREP, RERR and RREP-ACK). In our scheme, we introduce two new control packets, the *Query* (QRY) packet and the *Query-Acknowledgement* (QRY-ACK) packet (Fig 2b). They are the modified versions of the RREP-ACK packets of AOMDV. When an intermediate/source node wants to reassess its route trust for a particular destination, it sends out a QRY packet to the destination. The destination, upon receiving the QRY packet sends back a QRY-ACK packet that contains the number of packets it received so far since the last transmission of QRY-ACK. The scheme can be modified so that the destination periodically sends the received-packet count information without any node specifically requesting it. The nodes can continually assess and update the trust values in this scenario. *Type* field values 0-3 are used by the existing control packets in AOMDV. We use 4 and 5 as the value of *Type* to send the QRY and QRY-ACK packets respectively. These packets also have a checksum field that contains the checksum

computed over the entire packet and this field is encrypted by the packet creator and cannot be tampered without detection.

- Every route table entry corresponding to a destination also stores the corresponding route trust value computed by that node and a *Query Flag* bit (Fig. 1b). This bit is set when a QRY packet has been sent by that node and the corresponding QRY-ACK has not yet been received.

V. FRAMEWORK DETAILS

First the notations used in the description of the framework are listed. Trust initialization, trust update, localized repair for circumventing malicious nodes and the incentives and penalties for nodes are then described in detail.

A. Notations

- *S, D*: Principals, such as the communicating source and destination nodes.
- *L [X, Y]*: link between neighboring nodes *A* and *B*.
- *ELC [X, Y]*: normalized ELC of L[X, Y] computed at node *Y*.
- *ERC [Y, D]*: normalized ERC of the route between node *Y* and the destination *D*; originally received by Y from its downstream neighbor(s), it is updated by node *Y* and forwarded to its upstream neighbor(s).
- Θ: Operator which computes the new ERC value at each node by combining the ELC value computed at that node with the ERC value received from the downstream neighbor.
- S→X→D: denotes a route between source *S* and destination *D* via an intermediate node *X*.

Note on ELC and ERC: Link capacities can be estimated by using existing schemes [24]. We define ELC as the *effective* estimated capacity of a link between a pair of nodes. It is an indicator of the traffic that can be scheduled on the link for a *particular* route/flow. For example ELC [X, Y] would be node *Y*'s estimate of traffic it can successfully receive from node *X* corresponding to *a single route*. Similarly, ERC is the *effective* capacity of the route from an intermediate node to the destination. ERC [Y, D] would not just be computed over the actual link capacities of the links constituting the route from node *Y* to the destination *D*. Rather, it would factor in the effective capacities of links computed at each intermediate node as described below. An ERC value corresponding to a route at a node can thus be analogous to that node's trust on that route downstream.

B. Initialization of Trust values at StartupTime

At startup (bootstrap) time, node trusts are initialized to 0.5 for intermediate nodes and 1 for destination. Route trusts are unknown and computed recursively based on the ELC and ERC values as shown in the following example scenario. In Fig. 3, assume node *S* wants to communicate with the node *D* and it broadcasts RREQ packets. When the RREQs reach node *D*, it first computes ELC [Z, D], link capacity of the link between itself and the immediate last-hop node. Unless there is localized congestion at D, this value would usually be 1. This is also the ERC [D, D]. This value is piggybacked on the RREP sent by *D* to *Z* as ERC [Z, D], D's estimate of Z's traffic that D can successfully send to the destination (itself, in this case). In other words this is the trust that *D* claims to have on the route downstream. The upstream node *Z* stores this as the route trust corresponding to the destination *D*. Node *Z* updates and reports the new ERC values to each of its upstream neighbors on the routes between S and D (in this example it is nodes A, Y and B). For example, the value reported to node A would be:

$$ERC\ [A,Z] = ELC\ [A,Z]\ \Theta\ ERC\ [Z,D]$$

Node A would similarly update this value and send the updated value(s) to (each of) its upstream neighbor(s) in turn.

Destination ID
DSN
Originator ID
Lifetime
: :
Hop Count
Route Trust

Type
Originator ID (Destination Node)
Destination ID (Source/Interim node)
No. of Packets received
Timestamp
Checksum

Fig. 2a. RREP Fig. 2b. QRY/QRY-ACK

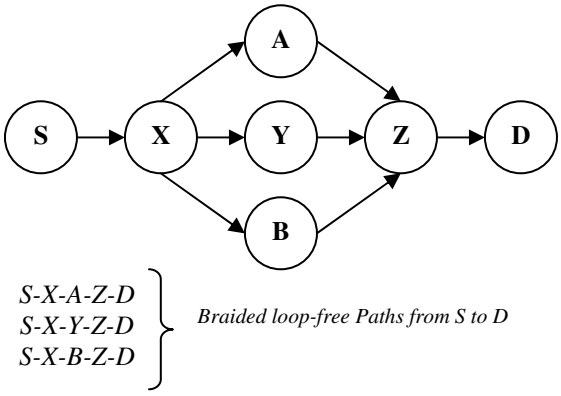


Fig. 3: Multiple Paths discovered between S and D

Now nodes A, Y and B update their ERC values and piggyback these on RREPs to node X. Since X receives ERC values from multiple downstream neighbors, it sends the following ERC value to node S:

ERC [S, X] = ELC [S, X] Θ Weighted_Average {ERC [X, A], ERC [X, Y], ERC [X, B]}

This ERC value acts as the initial route trust that node S has on the route S→X→D. In the event that an intermediate node already has fresh enough route to the destination and sends back an RREP instead of the destination, the trust computation would be unaffected.

C. Trust Maintenance

The intermediate nodes maintain the number of data packets forwarded by them to the destination. These nodes periodically reassess their route trust values by sending QRY packets to the destination. The node initiating a QRY packet sets the QRY Flag bit in its route table pertaining to that destination. The destination node, upon receiving the QRY packet sends back a QRY-ACK containing the number of data packets it received since the last transmission of QRY-ACK to the QRY initiator. When the QRY-ACK reaches the QRY initiator, it resets the QRY-ACK flag bit, discards the packet and re-computes the route trust as:

$$Route\,Trust = \frac{Data\ Packets\ reaching\ Destination}{Data\ Packets\ \ forwarded\ by\ the\ node}$$

All intermediate nodes that forwarded the QRY-ACK packet, recompute their route trust based on the packet count in the QRY-ACK packet. The QRY and QRY-ACK packets are assumed to be tamper proof. If multiple QRY or QRY-ACK packets are lost along a route, then the route trust would automatically decrease. We evaluate this in Sec. VI through simulations. The node trust on the immediate downstream node is computed using the formula:

$$Node\,Trust = \frac{Actual\ Data\ Rate\ achieved\ on\ Route}{Data\ Rate\ \ promised\ by\ the\ downstream\ node}$$

A node informs its upstream neighbors about any changes in the route trusts downstream. These upstream nodes recompute their route trust values for the route downstream and inform their own upstream neighbors in turn. Accuracy of such updates would factor in revaluating node trusts on downstream neighbors. For example, if decrease in route trust due to congestion is accurately reported by a downstream node then the node trust on the downstream node should not be reduced. This avoids unduly penalizing the downstream nodes based on a decrease in route trust for which they should not be accountable.

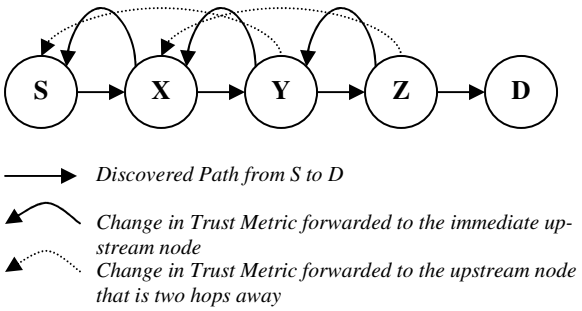
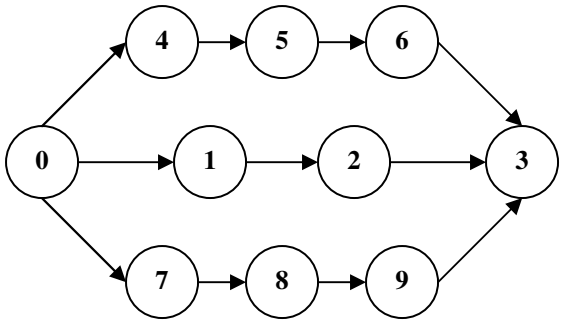


Fig. 4: Reporting of Route Trust Values



Source: Node 0, Destination: 3
Next Hop from Source: 4, 1, 7

Fig. 5: Simulation Scenario

However, this feature could be exploited by malicious/selfish nodes that might periodically inform their upstream nodes, about bogus route congestion and thereby maintain their node trust values. To circumvent this problem, we employ the two-hop reporting scheme that helps in reducing false trust reporting by malicious intermediate nodes. The scheme is explained using a sample scenario as shown in the above figure (Fig 4). Let us assume that node *Y* is malicious. If we had used one hop reporting, then *Y* would be the only gateway for the upstream nodes to determine the reliability of the downstream route. *Y* being malicious may silently discard any trust change advertisements sent by *Z* or could send a spurious trust change report to *X*. In the two-hop reporting scheme, *Z* reports changes in trust to both *Y* and *X*. Node *X* thus receives two copies of reports which it can compare, one from *Z* and the other from *Y*. The report from *Z* uses the *Localized Repair* feature of AOMDV/AODV to find an alternate sub-route between *X* and *Z* to bypass *Y*. Depending on the topology, the level of accuracy desired, or the length of sub-route where node behavior is under suspect, localized repair maybe be extended to two-hops or more for an alternate sub-route between *X* and *Z*.

If no alternate sub-routes can be found, then two things can be done:

- The Report to *X* can be encrypted and sent via *Y* itself, which if tampered would be detected. This is the only time when promiscuous monitoring in the framework might be required, i.e., if *Z* suspects *Y* to be silently dropping these reports, then *Z* may promiscuously monitor *Y*.
- *Z* can initiate a fresh route discovery to the source node and alert the source. This can be done in conjunction with the above, or independently if promiscuous monitoring is to be completely avoided.

If node *Y* claims route-congestion for more than time a threshold $T_{congmax}$, or when there is ambiguity between reports sent by the two downstream nodes, then all the routes with next hop Y are invalidated and purged from the routing table and RERR messages are sent to the destination. The node trust on *Y* would be made 0.

D. Incentives and Penalties

Each node maintains a Neighbors' Trust table that contains the node trust value for its neighbors. Depending upon the node trust values, the neighboring nodes maybe awarded incentives or penalties. For example, if the node trust on S maintained by the intermediate node *X* is greater than a threshold, NT_{Good}, node *S* can be awarded with enhanced Quality of Service (QOS) for its self-traffic whenever channel conditions permit. Such a scheme would encourage the nodes to work in unison with their neighboring nodes to obtain maximum throughput. On the contrary, if the node trust on a neighboring node is below a certain threshold, NT_{Bad}, this node can receive the lowest preference for its self-traffic if there is congestion or contention for the channel.

VI. PERFORMANCE EVALUATION

A. Simulation Setup

The baseline objectives of the simulation are to evaluate the following for our framework:

- Packet Delivery Ratio
- Trust Convergence latency

Using Glomosim-2.02, the topology of Fig. 5 was simulated over a field size of 100m X 100m. This scenario is an enhanced version of the network topologies that were used by Marina and Das [1] and Yuan et al. [25]. Their setup comprises of only two link-disjoint paths between the source and destination. We increased the number of available paths to three so as to effectively compare our protocol with an AOMDV variant that implements a Round Robin Path selection. The simulation was run for 150 Seconds with a Seed value of 6. Each node has a transmission range of 30 Meters using a Free-Space propagation-path loss model. Constant Bit Rate (CBR) traffic at 512 Bytes per second with an inter-departure time of 5 ms was injected between the source node 0 and the Destination node 3 from the beginning till the end of the simulation.

B. Varying network Conditions

To simulate general congestion in the network, we introduced CBR traffic of 1024 Bytes per second at the links [2, 3], [7, 8] and [4, 5] during the interval of 10-30S. This represents the scenario when all available routes are experiencing congestion.

Localized congestion was created through 2048 Bytes per second CBR traffic in the following discrete time intervals: 30-40S for the links [2, 3] and [4, 5], 60-70S for the link [2, 3], 75-85S for the link [7, 8], 85-99S for the link [4, 5]. This is to simulate a variety of scenarios: simultaneous congestion in two routes, congestion on one route, different times the congestion eases, etc.

Finally, neighbors of the source node were failed to simulate node failures during the following time intervals: Node 1:100-125S, Node 7:115-125S, Node 4:130-140S. This was done to study the adaptability of our protocol and achieve a fine grained comparison with AOMDV.

C. Route Selection and Trust Reassessment Intervals

Route selection was done in a weighted round robin fashion. The number of data packets sent via a particular next hop is directly proportional to the route trust metric pre-computed for that node. Revaluation of trust metrics by the source node was done by sending QRY packets at time intervals dictated by the already computed route trust values for the next hop nodes. For example, if the trust value on the selected next hop node was less than 0.5, QRY packets were sent out after every 20 data packets. Similarly, if the trust was in between 0.5 and 0.8, the querying frequency was every 100 packets and if the trust was above 0.8, the frequency was every 200 packets. The destination node, in response, generated QRY-ACK packets were forwarded to all the upstream nodes that participated in the data transmission. Trust computation was performed as described previously.

D. Results

Simulations were run with the above setup and the results were compared with the AOMDV protocol. Trust values for each next hop node and the Packet Delivery Ratio were plotted against time. The Packet Delivery ratio was obtained by capturing the sent and received data packet count.

As seen from the Figures 6a, 7a and 8a, during the initial interval, 10-30S, local congestion simulated in all the available paths considerably affected the overall packet delivery ratio. As a consequence the route trust for all the next hops fluctuated during this time frame. Since there was no alternate path with better route trust, data packets were sent over all the paths and hence the overall protocol suffered due to this congestion. During the interval 60-70S, only the route via node 1 suffered congestion resulting in packet loss. The trust metric re-computation latency (time interval between the onset of congestion and the time at which the source obtains the QRY reply) was approximately 1.5 sec. This number was an averaged output of several test runs. Since the route trust on route via node 1 was greater than 0.8 before the 60th second, the QRY packets were sent out only after 200 data packets, and hence the trust convergence interval was large as indicated by pointer 1 in Fig. 6b. Once the congestion was realized at the source, the route trust on node 1 was decreased and the traffic was diverted through alternate paths via nodes 4 and 7. Thus, the route trust follows the packet delivery ratio computed using the QRY-ACK packet(s) from the destination. During this time interval of 60-70S, 5 data packets were sent through the congested route periodically to check if the congestion got cleared. Thus when the localized congestion between nodes 2 and 3 subsided after the 70th sec, the source was able to reassess the trust within the next 0.5 sec. This was because of the reduced QRY request frequency that was set to be every 20 data packets. Thus, the trust convergence interval was less as indicated by pointer 2 in Fig. 6b. In this duration, traffic was redirected through alternate paths and hence the overall packet delivery ratio was not affected significantly as can be seen from the overall packet delivery ratio Fig. 9. Likewise, when the trust metrics were maintained in-between 0.5 and 0.8, the trust convergence interval was approximately 1 sec. This could be visualized during the 30-31st sec in Fig. 6b. Similar localized congestion, reduction in route trust and diversion of data through highly trusted routes were monitored during the intervals 75-85S and 85-99S for the alternate paths and plotted in the Figures 7a, 7b and 8a, 8b, respectively. These plots too, show strict resemblance to the trust convergence latencies observed in Fig. 6b.

It is clear from Figures 6b, 7b and 8b, that route/node failures (Node 1:100-125S, Node 7:115-125S, Node 4:130-140S) are quickly detected by the source in our framework. As shown earlier, the convergence interval for this detection is fairly small.

The same simulation setup was also used to run the AOMDV protocol. We used two variants of AOMDV, first, the round robin route selection option in which, all the available multiple routes were selected in a round robin manner for packet transfer and second, the single route option, where only one route was selected for packet transfer. Comparing our proposed scheme's "overall packet delivery ratio"(Fig. 9) with that of both types of AOMDV (Figures 10 and 11), it is evident that even with round robin route selection, AOMDV suffered approximately 50% decline in throughput when there was congestion in the routes downstream. The performance of single route AOMDV was even worse. Additionally, our protocol quickly sensed node failures and diverted traffic via alternate paths as against AOMDV that kept attempting to send traffic via routes with failed nodes.

Though simulations were run using a baseline setup, the end results assure the effectiveness of our proposal when adapted to Multi-Path protocols. It is a self learning scheme which adapts to ambient conditions.

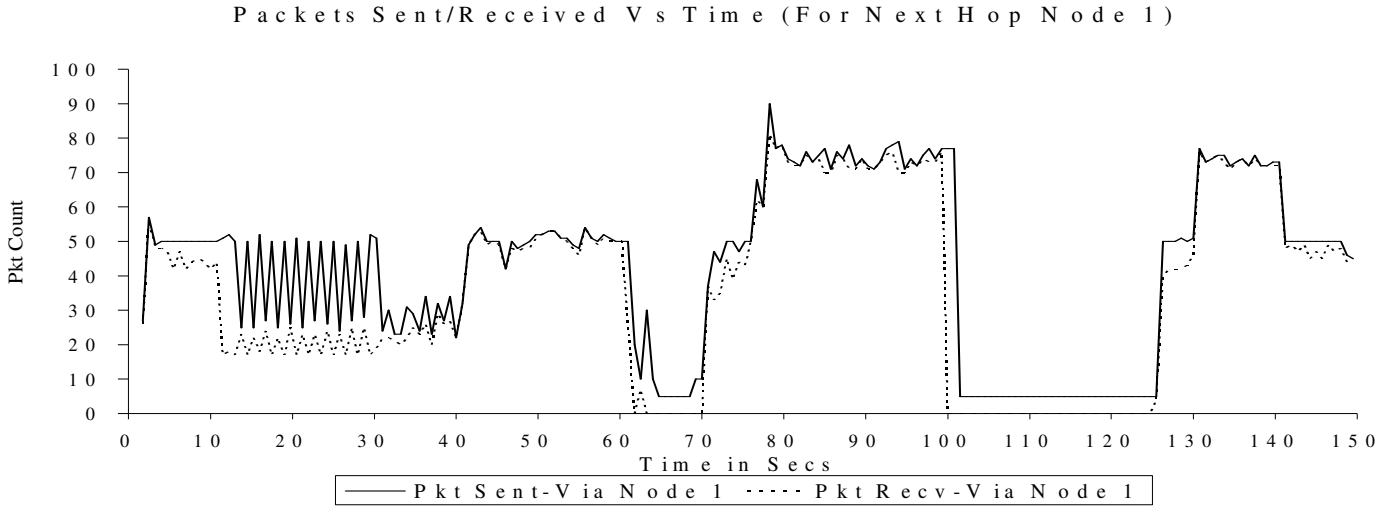


Fig. 6a.

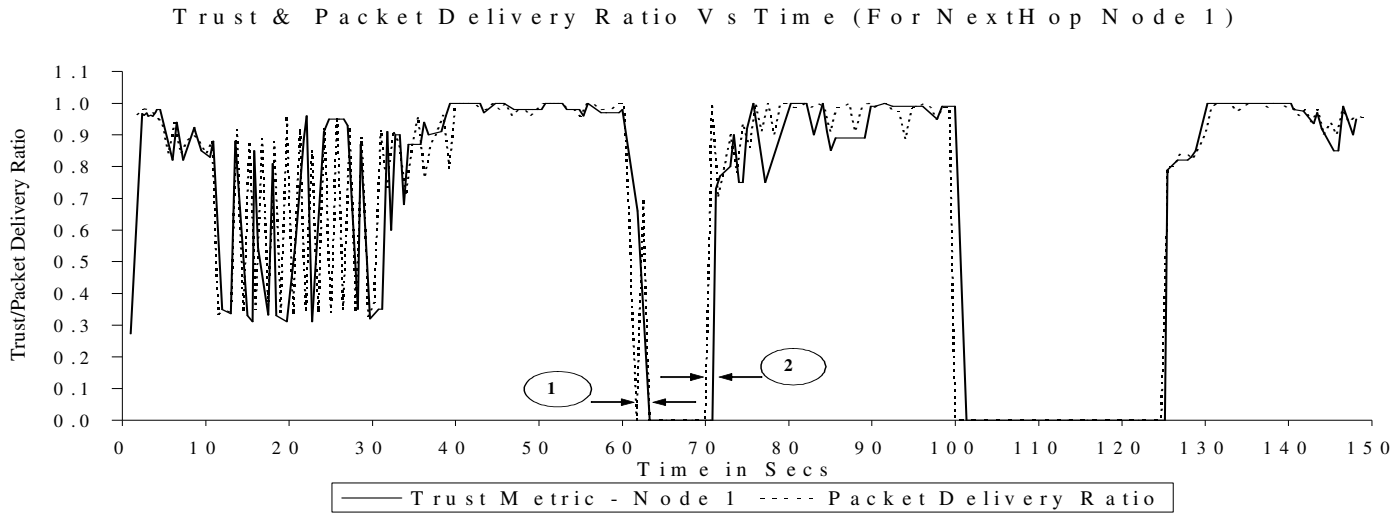


Fig. 6b.

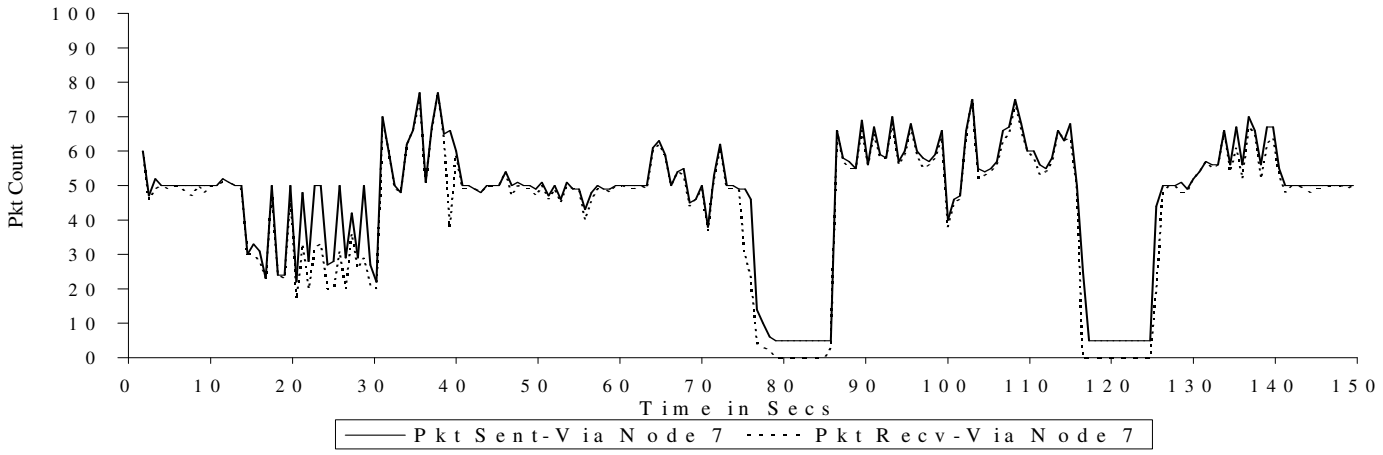


Fig. 7a.

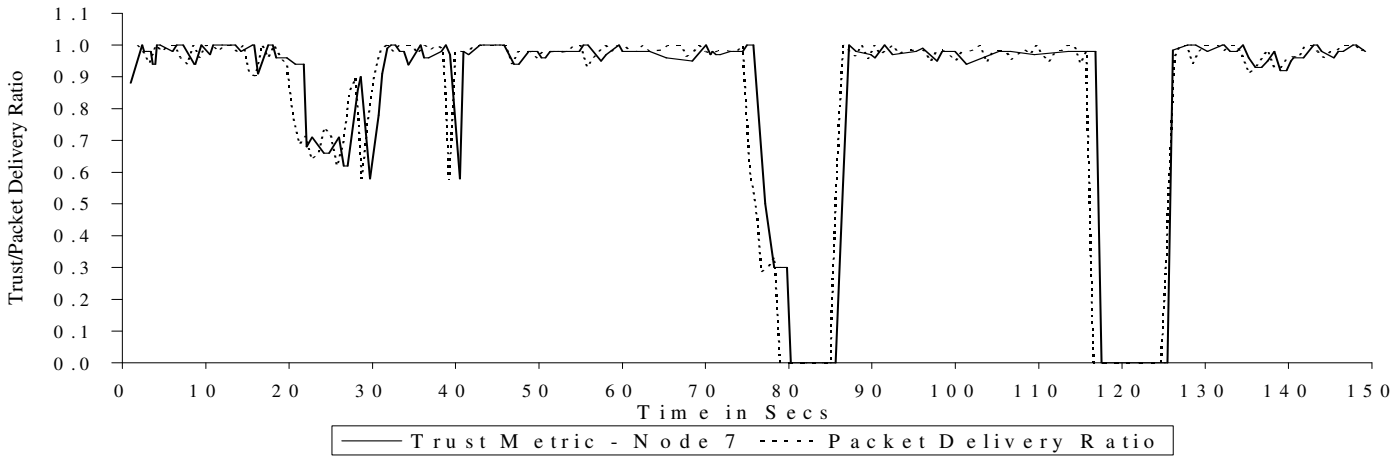


Fig. 7b.

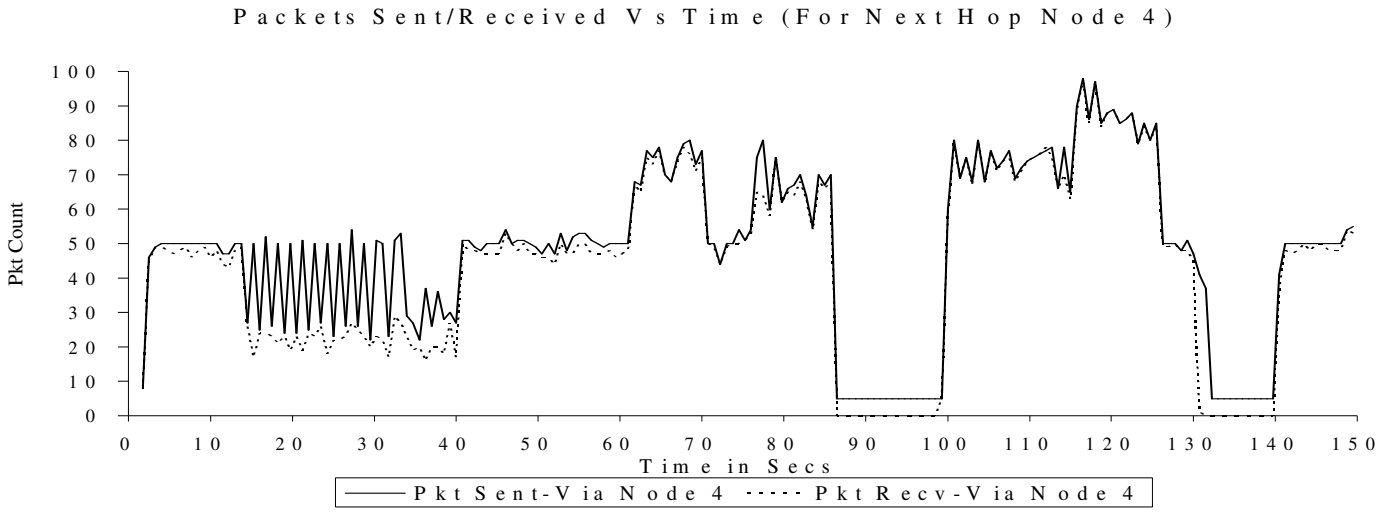


Fig. 8a.

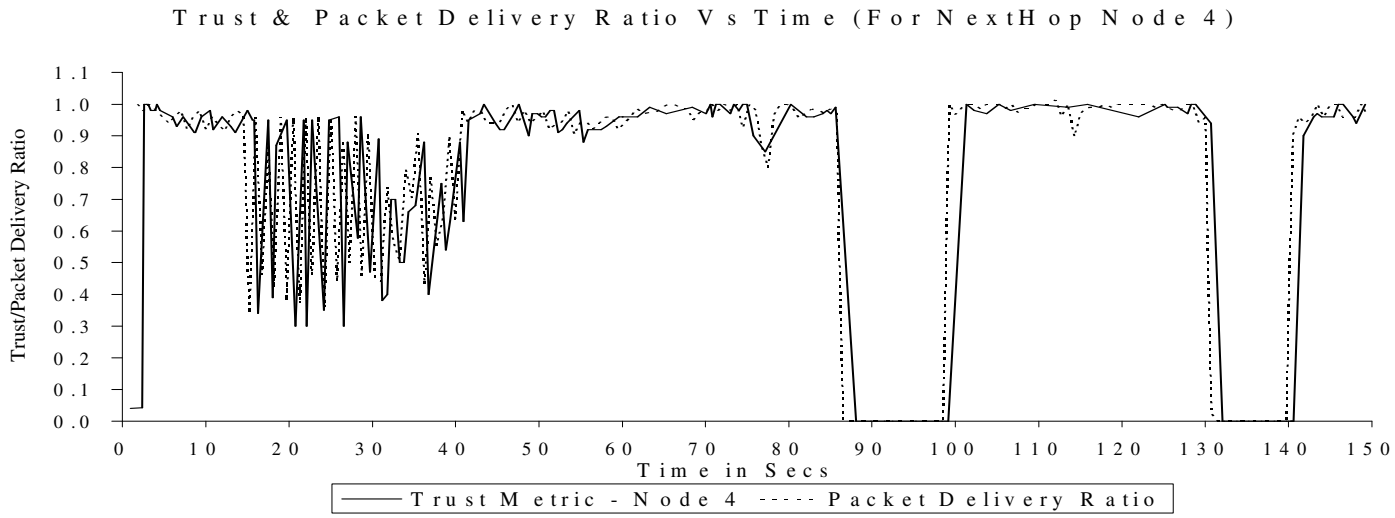


Fig. 8b.

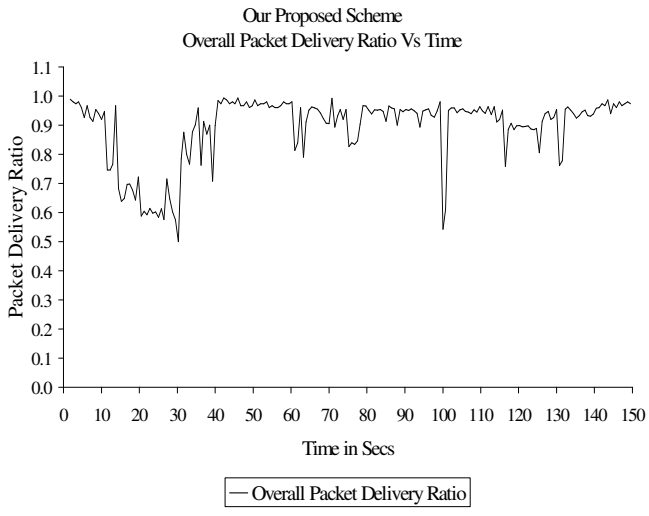


Fig. 9.

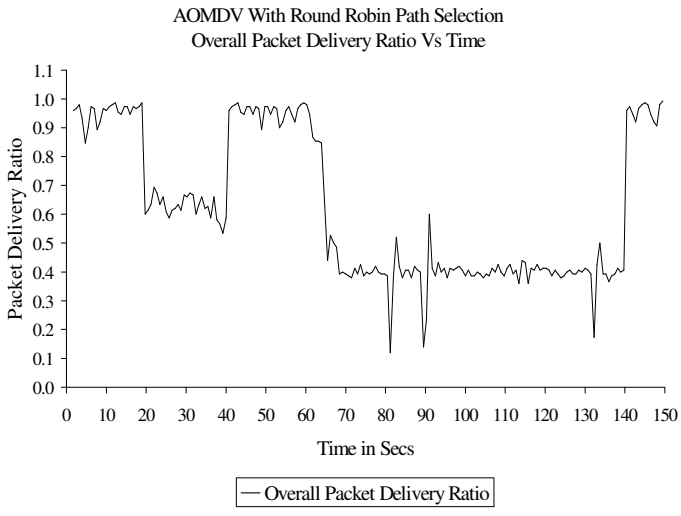


Fig. 10.

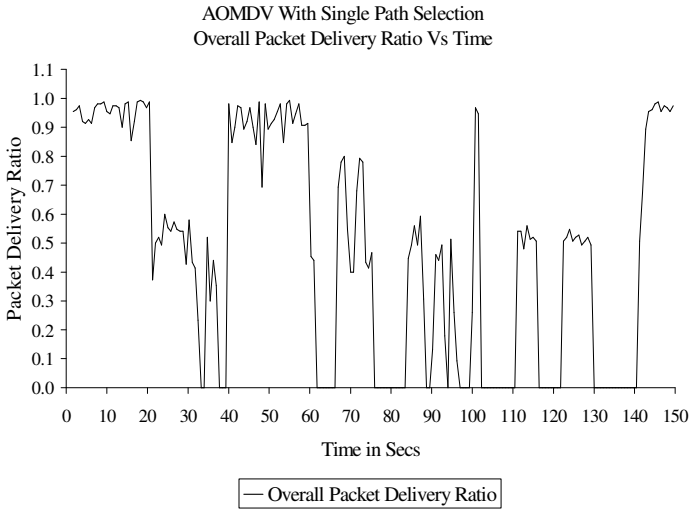


Fig. 11.

VII. DISCUSSION AND CONCLUSION

When multiple paths are available for every source-destination pair, one or more routes can be selected for data transfer in order to improve the throughput and reduce the latency. Policies can be put in place to efficiently schedule the traffic among the multiple paths. In this paper, we proposed a scheme that implements a scheduling mechanism based on trust. It is to be noted, that our work concentrated only on quickly adopting to changing route conditions, and not on traffic distribution and load balancing.

For the purpose of simulations we used a proportional scheme where the amount of traffic sent on routes was dependent on trust. Extensions of the scheme can be proposed that may support priority or classification of data. For example, data can be classified based on trust and confidentiality as Critical, Confidential and less Critical/Confidential. This classification can be done by the source. Critical data is time bound and needs to be delivered with little latency. When an intermediate node sees the critical data, it may attempt to forward the packets downstream as quickly as possible. If there are multiple paths available, all the paths maybe utilized for quick transmission. Confidential data requires a reliable channel for transfer. Hence, only the route that has the highest route trust value in the route table maybe used to forward the confidential data. The third data type, less critical/confidential data doesn't suffer from latency or reliability issues. So, the intermediate nodes may have the liberality of sending the data over less trusted routes. By choosing routes that have low trust value, the intermediate nodes can use the less critical data packets as probe packets to reassess their trust on routes with lower trust. The nodes that lie in the low trusted route try to forward these packets dutifully to gain back the confidence of the upstream nodes.

In this paper we proposed a framework for Trust-based route selection for MWNs. We utilized AOMDV as the underlying protocol. The goal was to design a lightweight framework that makes informed route-selections rather than making a default selection of routes with minimum hop count. Though our framework is not comprehensive, it can serve as a good starting point and guide for designing environment aware protocols. Even in its limited form, the utility of the nature of this work is demonstrated by the simulation results. Our future goal is to make this framework more fine grained and perform more extensive simulations representing a broader range of network conditions.

REFERENCES

[1] M. K. Marina, S. R. Das, "Ad-hoc On-demand Multipath Distance Vector Routing". In the Proceedings of IEEE international Conference on Network Protocols (ICNP) 2001, 14-23.

[2] C. Perking, E. Royer, "Ad Hoc On Demand Distance Vector Routing", IEEE Workshop on Mobile Computing Systems and Applications (WMCSA), 1999, pp. 90-100.

[3] C. Perkins, E. Royer and S. Das, "Ad hoc on-demand distance vector routing", RFC-3651, (Jul. 2003)

[4] H Deng, W Li, DP Agrawal, "Routing Security in Wireless Ad-Hoc Networks", IEEE Communications Magazine, Volume: 40, Issue10, pp.70-75, ISSN: 0163-6804.

[5] The online multi-source dictionary search service, Lexico Publishing Group, "http://dictionary.reference.com/browse/trust".

[6] X Zeng, R. Bagrodia, and M. Gerla. "GloMoSim: a library for parallel simulation of large-scale wireless networks", in Proceedings of the 12th Workshop on Parallel and Distributed Simulations, May 1998.

[7] T. Beth, M. Borcherding and B. Klein, "Valuation of trust in open networks", ESORICS, November 1994.

[8] A. Rahman and S. Hailes, "A Distributed Trust Model", New Security Paradigms Workshop 1997, ACM, 1997.

[9] H. Li, M. Singhal, "Trust Management in Distributed Systems", Computer, vol. 40, no. 2, pp. 45-53, Feb 2007.

[10] A. Pirzada, C. McDonald, “Establishing Trust in Pure Ad-hoc Networks”, 27th Australasian Conference on Computer Science 2004, 47-54.

[11] Z. Liu, A. Joy and R. Thompson, “A Dynamic Trust Model for Mobile Ad-hoc Networks”, 10th IEEE International Workshop on Future Trends of Distributed Computing Systems (FTDCS) 2004, pp. 80-85.

[12] L. Eschenuer, V. D. Gligor, J. Baras, “On Trust Establishment in Mobile Ad-Hoc Networks”, 10th International Workshop on Security Protocols, 2002, pp. 47-62.

[13] M. Virendra, M. Jadliwala, M. Chandrasekaran, S. Upadhyaya, “Quantifying Trust in Ad-Hoc Networks”. IEEE international Conference on Integration of Knowledge Intensive Multi-Agent systems (KIMAS) 2005, pp. 65-71.

[14] Y. L. Sun, Z. Han, W. Yu and K. J. R. Liu, “A Trust Evaluation Framework in Distributed Networks: Vulnerability Analysis and Defense against Attacks”, IEEE INFOCOM 2006.

[15] X. Li, M. R. Lyu, J. Liu, “A Trust Model Based Routing Protocol for Secure Ad Hoc Networks”. IEEE Aerospace Conference (IEEEAC) 2004, pp. 1286-1295.

[16] C. Zouridaki, B. Mark, M. Hejmo, R. Thomas, "A Quantitative Trust Establishment Framework for Reliable Data Packet Delivery in MANETs,” ACM workshop on Security of ad-hoc and sensor networks SASN '05, Alexandria, VA, Nov 2005.

[17] L. Zhou and Z.J. Haas, “Securing ad-hoc networks,” IEEE Network Magazine, vol. 13, no. 6, November 1999, pp. 24–30.

[18] Q. Duan, M. Virendra, S. Upadhyaya, “On the Hardness of minimum Cost Blocking Attacks on Multi-Path wireless Routing Protocols”, IEEE International conference on Communications (ICC) 2007.

[19] P. Kotzanikolaou, R. Mavropodi, C. Douligeris, "Secure Multipath Routing for Mobile Ad-hoc Networks", Second Annual Conference on Wireless On-demand Network Systems and Services (WONS’05).

[20] J. Douceur, “The Sybil Attack”, International Workshop on Peer-to-Peer Systems (IPTPS ’02), American Mathematical Society, Mar 2002.

[21] J. Marshall, V. Thakur, A. Yasinsac, “Identifying Flaws in the Secure Routing Protocol”, 22nd IPCC Conference, Apr 2003, pp. 167–174.

[22] M. Zapata and N. Asokan, “Securing Ad-hoc Routing Protocol”. In the proceedings of 3rd ACM workshop on Wireless Security 2002,

[23] S. Marti, T. Giuli, K. Lai, M. Baker, “Mitigating Routing Misbehavior in Mobile Ad Hoc Network”. In the Proceedings of 6th annual conference on Mobile Computing and Networking 2000, 255-265.

[24] J. Li, C. Blake, D. De Couto, H. Lee, R. Morris, “Capacity of Ad Hoc wireless networks”, 7th Annual International Conference on Mobile Computing and Networking (MOBICOM), Rome, Italy, 2001.

[25] Y. Yuan, H. Chen, M. Jia, “An Optimized Ad-hoc On-demand Multipath Distance Vector (AOMDV) Routing Protocol”, 2005 Asia-Pacific Conference on Communications, Perth, Australia, October 2005.