

Trust Formalization in Mobile Ad-Hoc Networks

Raihana Ferdous

Institute for Integrated and Intelligent Systems(IIS)

Griffith University, Australia

Email: Raihana.Ferdous@student.griffith.edu.au

Vallipuram Muthukumarasamy

IIS

Griffith University, Australia

Email: v.muthu@griffith.edu.au

Abdul Sattar

IIS

Griffith University, Australia

Email: a.sattar@griffith.edu.au

Abstract—Recent advances in networking technology have increased the potential for dynamic enterprise collaborations between an open set of entities on a global scale. Mobile ad-hoc networks (MANETs) have created problem in resource sharing as they are constructed via mobile nodes without any prior knowledge of the existing nodes which may not be trustworthy. Trust management appears to be a promising approach to formalize trustworthiness among these anonymous nodes. The outcome of the study is to formalize a trust mechanism for MANETs. The aim of this paper is threefold: to formalize and evaluate trust, to use trust as a basis to establish keys between nodes in MANETs, and to utilize trust as a metric for establishing secure distributed control in MANETs. We define metrics for nodes to establish and manage trust, and use this mutual trust to make decisions on establishing group and/or pair-wise keys in the network. We also review the routing protocols of ad-hoc networks with trust considerations and select Dynamic Source Routing (DSR), a protocol that can be used in distributed ad hoc network settings for path discovery.

mobile ad hoc networks; trust; trust-formalization; routing; security;

I. INTRODUCTION

Mobile ad-hoc networks (MANETs) are dynamically configured, multi-hop wireless networks with varying topology. In order to enforce cooperation within the network, adjacent nodes should build up trust with time. Such trust establishment procedure can improve security, connectivity, and quality of service in the network to enhance its performance.

For example, considering the scenario where a man-made or natural disaster has taken place. The recovery of data of the attacked place has been rapidly deployed using ad-hoc networks and coordination between the constituents of the rescue systems has been initiated. Trust is very important here because an attack has already taken place, and now the adversary may try to destroy the relief operations by compromising the core data of the system.

These schemes only specify mechanisms to prevent false key generation and compromise of other nodes' keys, if a node is eventually compromised. Thus these schemes inherently assume nodes to be non-malicious at the time of key establishment. To the best of our knowledge, there are no efficient formal schemes for functional verification or judgment on key establishment requests from other nodes. Specifically, if a node is unknown, keys would still be established with it as long as the infrastructure for establishing keys is present. Schemes involving a trusted third party for key establishment are deemed impractical for ad-hoc networks due to the limitations

on finding such an authority, and therefore, are not considered as a practical solution.

Additionally, self organization of ad-hoc network nodes into clusters has been studied in the literature [2] to induce distributed control in such networks which are otherwise infrastructure-less. Present ad-hoc network clustering schemes use physical location as a metric to cluster the nodes[3]. Any node can elect to become a cluster head and can propagate cluster joining requests to its k-hop neighbors through various flooding mechanisms. This choice of cluster formation is arbitrary and does not take security into account. A node that is malicious could initiate a cluster formation announcement and could potentially compromise all nodes that elect to join its cluster.

The main problems that we want to address in this paper are the following: (a) To define a metric that the nodes can use to make decisions on whether to establish keys with other nodes in an ad-hoc network, given that the infrastructure for establishing such keys exists, and (b) To define a basis on which nodes in an ad-hoc network can securely group together, so that some kind of distributed control can be introduced in an otherwise infrastructure-less network. We propose to use trust between the nodes, as a mechanism to address both these problems. We present an architecture that uses trust as a metric for nodes to (a) Make decisions on establishing keys with other nodes in the network, and (b) Group together into trust-based domains.

II. RELATED WORK AND PAPER ORGANIZATION

The idea of using trust to mitigate security threats has been an important area of research [4]. Trust establishment and management between entities (nodes or agents) can be done through a central trusted authority or in a distributed fashion by nodes [5], or a combination of both. Related work in this area [6], [7], [8], [9], employ both these techniques. For example, Zhou et al. [10] propose the idea of utilizing threshold cryptography to distribute trust in ad-hoc networks, Davis [11] proposes the use of certificates based on hierarchical trust model to manage trust, and Eschenauer et al. [1] contrast between trust establishment in ad-hoc networks and the Internet. Our approach is new in that we use trust as a metric to address the problems.

This paper is organized as follows: Section 3 formalizes the notion of trust between two nodes as a combination of self trust and group trust. Section 4 quantifies trust between two nodes and describes schemes for trust management. Section 5,

6 and 7 describe the notion of Trust Domains, organization of nodes in ad-hoc networks into domains based on trust values and integration of trust model for secure routing in MANETS. We finally conclude the paper in Section 8 with a summary of its contributions, limitations and proposed future work.

III. TRUST FORMALIZATION

This section describes the trust formalization. Our schemes draw ideas from the Watchdog and Pathrater schemes [13], utilized for cooperation of nodes in ad-hoc networks. We define a node n 's trust on another node m :

$$T_{nm} = \alpha_1 nT_s^m + \alpha_2 nT^m_o \quad (1)$$

In the above equation, $T_{n,m}$ is evaluated as a function of two parameters:

- (a) nT_s^m : Node n 's self evaluated trust on m ; n computes this by directly monitoring m .
- (b) nT^m_o : Weighted sum of other nodes' trust on m evaluated by n .

In eq. (1), α_1 and α_2 are weighting factors such that $\alpha_1 + \alpha_2 = 1$. Thus, by varying α_1 and α_2 , n can vary the weight of self evaluated vs. others trust in calculating its total trust on m . Here, $0 \leq T_{nm}, nT_s^m, nT^m_o \leq 1$, and thus eq. (1) is normalized.

A. Evaluating nT_s^m

Node n computes this value by directly monitoring m when m is in its radio range. We define nT_s^m as:

$$nT_s^m = f(\Phi, \Omega) \quad (2)$$

Node n 's self trust on m is a function (f) of traffic statistic functions Φ and Ω computed by monitoring m . Precise definition of f can be implementation dependent. We assume f to be a weighted sum of Φ and Ω . Here, Φ is a function of monitored traffic statistics pertaining purely to traffic volume and Ω is a function of monitored traffic statistics pertaining to information integrity. Lee et al. [14] compile node monitoring statistics for one hop neighbors in ad-hoc networks. Thus, node n can monitor the following statistics for a one-hop neighbor m : Incoming packets on m , outgoing packets from m , outgoing packets of which m is the source, incoming packets of which m is the destination, incoming packets on m from n , etc. Based on these monitored statistics we define Φ and Ω as:

$$\Phi = g(\gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6) \quad (3)$$

$$\Omega = h(\lambda_1, \lambda_2) \quad (4)$$

Here g and h can again be defined based on the implementation. Like f , we assume them to be weighted summation of their constituent parameters. These parameters are defined below:

- γ_1 : packets sent by m to n that are dropped by m
- γ_2 : total packets dropped by m
- γ_3 : packets dropped by m due to congestion
- γ_4 : packets dropped by m due to unknown reasons

- γ_5 : n 's assessment of m 's priority to m 's self packets vs. all other nodes' packets
- γ_6 : packet forwarding delay by m
- λ_1 : packets misrouted by m
- λ_2 : packets falsely injected by m

B. Evaluating nT^m_o

In the representation nT^m_o , O is the set of other nodes whose trust on m is utilized by n in evaluating its own trust on m . O is defined as:

$O = \forall \text{ node } o \in O \Rightarrow o \text{ is in the range of both } m \text{ and } n, \text{ and } \exists T_{no}, s.t. T_{no} \geq \text{"good"}$.

"good" is a threshold value for demarcating Unknown and Good trust-regions and this is further explained in Sec. 4. In this section we present four schemes for computing the value of nT^m_o , where n , m and O have their usual meanings and nT^m_o is defined as above.

- 1) *Optimistic or Greedy Approach*: This is the simplest approach. nT^m_o is computed by selecting the largest value of the product $T_{i,m} \times T_{n,i}$ for all values of i in the set O . In other words, node n uses the highest value of trust that nodes in the set O assign to the node m , weighting it with its own trust on the nodes in the set O .

$$nT^m_o = \max_{i \in O} (T_{i,m} \times T_{n,i}) \quad (5)$$

- 2) *Simple Average of Weighted Products*: The value of nT^m_o is the simple average of the product of $T_{i,m}$ and $T_{n,i}$ over all the nodes i in the set O .

$$nT^m_o = \frac{\sum_{i \in O} (T_{i,m} \times T_{n,i})}{|O|} \quad (6)$$

where $|O|$ is the cardinality of the set O .

- 3) *Weighted Average*: The value of nT^m_o is the weighted average of $T_{i,m}$ over all the nodes in the set O . The weight associated with each $T_{i,m}$ is $T_{n,i}$.

$$nT^m_o = \frac{\sum_{i \in O} (T_{i,m} \times T_{n,i})}{\sum_{i \in O} T_{i,m}} \quad (7)$$

Alternatively, if we normalize with respect to $T_{i,m}$ we get,

$$nT^m_o = \frac{\sum_{i \in O} (T_{n,i} \times T_{i,m})}{\sum_{i \in O} T_{n,i}} \quad (8)$$

Thus, in the weighted average approach the weighted products of $T_{i,m}$ and $T_{n,i}$ are normalized either with $T_{i,m}$ or $T_{n,i}$ for all values of i in the set O .

- 4) *Double Weighted Approach*: In this approach, we further try to improve on the value of nT^m_o computed from the weighted average approach by normalizing the product of $T_{i,m}$ and $T_{n,i}$ with respect to both $T_{i,m}$ and $T_{n,i}$.

$$nT^m_o = \frac{\sum_{i \in O} (T_{i,m} / \sum_{j \in O} T_{j,m}) \times T_{n,i}}{\sum_{i \in O} T_{n,i}} \quad (9)$$

Alternatively,

$${}_nT^m_o = \frac{\sum_{i \in o, j \in o} (T_{n,i} / \sum T_{n,j}) x T_{i,m}}{\sum_{i \in o} T_{i,m}} \quad (10)$$

It should be noted that the first scheme is the simplest, but it is based on the accuracy of trust on one node. This scheme would be the most vulnerable to misdecisions and failure due to malicious misrepresentation of trust by a single node, or a collusion of nodes. Schemes 2, 3, and 4 not only increase in complexity of evaluation, but should also provide a corresponding enhanced accuracy in the evaluation of trust. We are currently performing simulations to verify the validity of this assumption.

IV. TRUST EVALUATION

This section describes trust *evaluation*, i.e., trust establishment and management between nodes. We explain the initial trust establishment procedure between a pair of nodes that are one-hop neighbors and explain the consequences of node mobility on the existing trust between a pair of nodes. We define trust to be non-transitive. Thus $T_{n,m} \neq T_{m,n}$. Both m and n independently evaluate $T_{m,n}$ and $T_{n,m}$ respectively, through the schemes described above. The associability of trust will be addressed in Sec. 5. Trust evaluation between the nodes is defined as a four phase process: Initiation and Monitoring, Query and Evaluation, Updating, and the Restructuring phase. There is an optional fifth phase: Re-establishment after declared malicious. The five trust phases of the trust evaluation process are outlined by tracing a sample trust value of a node m continuously evaluated by another node $nT_{n,m}$ over a period of time. Here we explain three trust regions: Good, Uncertain, and Bad. Nodes above the good trust threshold, i.e., in the Good region are highly trusted one-hop neighbors of n , and if they are one-hop neighbors of m also, then their trust will be utilized by n in evaluating ${}_nT^m_o$. The nodes in Uncertain region are those with intermediate trust values, and their trust is not utilized in evaluating ${}_nT^m_o$. Nodes in the Bad region are those marked as malicious by n . This is further explained in the subsections below.

A. Initiation and Monitoring Phase

This is the phase when the network is newly deployed, or a new node joins the network. This is the period when the new nodes have not established keys with their one-hop neighbors (or any other nodes). During this phase, the new node(s) monitors its one-hop neighbors. The monitoring node (n) switches over to promiscuous mode and listens for all packets transmitted by the monitored node (m). It collects the statistics mentioned in Sec. 3.1. For example, node m joins the network at time $T = 0$, and becomes a one-hop neighbor of n . Thus a new or unknown node is given a ‘‘bare’’ trust value. During this phase, nodes will not send any sensitive data to their neighbors, unless timely delivery is absolutely essential (e.g., in disaster management scenarios it might be critical to exchange information immediately after network deployment).

B. Query and Evaluation Phase

During this phase, the nodes evaluate their self trust on their one-hop neighbors (e.g., ${}_nT^m_s$) through a challenge response system. The nodes query their neighbors regarding the statistics they have already assimilated (defined in Sec. 3.1). This is akin to truth verification, as the monitoring nodes already know the correct answers to their queries. The neighbors’ trust evaluation is based on the accuracy of their responses. This has been explained in Sec. 3.1. Evaluation of ${}_nT^m_o$ is done as explained in Sec. 3.2.

C. Updating Trust

As long as a node remains in the radio range, its trust is continuously evaluated and updated. Thus, monitoring and querying is performed even after trust has been established between the nodes. However, the periodicity of querying and monitoring is decreased with time if the trust value stabilizes and is maintained at a certain level.

D. Restructuring Phase

This phase takes into account two different scenarios and their effect on inter-node trust values:

- Trusted one-hop neighbors move out of radio range due to node mobility: A node, say m , which was previously in the radio range of a node n , now moves out of its radio range due to node mobility. The value of $\alpha 1$ (the proportion of self trust in overall trust) now decays exponentially as:

$$\alpha 1 = c.e^{-\lambda,t} \quad (11)$$

Parameter λ is the decay factor which is determined by the infrastructure and mobility constraints of the network, and c is some constant. Node n now fixes ${}_nT^m_s$, to the value at time T_{dis} (just before m moved away). But since $\alpha 1$ exponentially decays, n ’s importance on ${}_nT^m_s$ in calculating $T_{n,m}$ decreases with time. If the node m is outside n ’s radio range for a time period Δt_{max} , and if $T_{n,m} > T_{good}$, then at $T_{tsh} = (T_{dis} + \Delta t_{max})$, $\alpha 1$ is forced to 0 (i.e., $\alpha 1 = 0$), and $T_{n,m}$ is reduced to T_{good} (i.e., $T_{n,m} > T_{good}$). This is shown in Fig. 1 by the curve C’D and the line DF. If the value of $T_{n,m}$ is below the Good Region then it is left unchanged. This value is kept constant as the history information of node n for the scenario that m and n eventually return to each other’s radio range.

- Trusted one-hop neighbors that had previously moved out of radio range are now back in radio range: the node m , after moving out of n ’s radio range, eventually returns back in the range of n (i.e., again becomes a one-hop neighbor of n). Re-evaluation of $T_{n,m}$ by n is now required for potentially restoring $T_{n,m}$ to the highest trust value, as m becomes directly monitored again. This re-evaluation of $T_{n,m}$ is similar to the Initiation and Monitoring phase (Sec. 4.1). The only difference is that this re-evaluation does not begin from the bare trust value, but starts from the value of $T_{n,m}$ previously fixed by n (after m had moved out of its radio range).

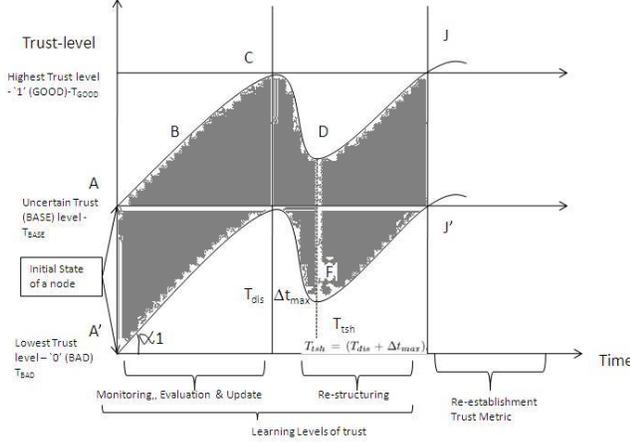


Fig. 1. Trust Evolution

E. Re-establishment Phase

This phase explains the scenario for a node m that was declared malicious previously by a node n and now it wants to re-associate with n . Consider the scenario when n 's trust on m is good, say at point H. Assume that after monitoring m for some time node n discovers to have become malicious. This can be deduced by the challenge response scheme as describe in Sec. 4.1. The point of time when node n makes this conclusion is depicted by point I. Now, depending upon the nature of malicious operations performed by m , node n drops the trust of node m into the bad region. The two levels J and J' show this drop in trust in the figure. For serious malicious activities having a critical impact on the functioning of n itself, the trust of m is dropped to zero as shown by the point J . Zero represents the absolute minimum trust value possible (or highest distrust) in the network. For a malicious operation with a less severe effect, the trust can be dropped to any point in the Bad region as shown by the point J' . The two levels shown are just for illustration, but the new trust value of the malicious node can lie anywhere in the Bad region or it can be quantified into a number of levels based on the seriousness of offense. Trust re-establishment is based on the discretion of the node evaluating the trust (in this case node n). If node n does not want to immediately re-establish trust with node m , then the value of $T_{n,m}$ is unchanged till n decides to reconsider the trust establishment process. Trust increase after reevaluation, if at all initiated by the node n , is linear, provided that m does not perform any more malicious operations.

V. TRUST MODEL AND TRUST DOMAINS

So far we have described the trust establishment and trust evaluation between pairs of nodes that have been one-hop neighbors at some point of time. In this section we extend our pair-wise trust model to include other nodes in the network which are not one-hop neighbors. We define a model through which non-neighbor nodes can establish and manage trust utilizing the five-phased trust evaluation procedure described

in Sec. 4, thus providing a basis for establishing pair-wise keys between any pair of nodes, and also establishing group keys in the network. This model organizes nodes into trust-based clusters called Physical- Logical Trust Domains (PLTDs)[15], thus securely grouping nodes to induce distributed control in the otherwise infrastructure-less network. Member nodes in a PLTD can establish and share a domain (group) key. We use node mobility to propagate trust throughout the network. In our scheme, nodes can belong to multiple PLTDs and there can be several overlapping PLTDs in a physical region. PLTD formation can be initiated by any node. A node n can announce its intention to form a PLTD by requesting nodes in the set P to join its PLTD (PLTD- n). P is defined as: $P = \forall \text{ node } p \in P \Rightarrow p \text{ is in the range of } n$, and $\exists T_{n,p}, s.t. T_{n,p} \geq \text{"good"}$.

Based on its individual trust on n , $T_{p,n}$, each node in P may either accept or decline to join PLTD- n , or it could invite n to join its own PLTD if it has already initiated its own domain formation procedure. If at any time, the trust value of a node in PLTD- n , say m ($T_{n,m}$), falls below "bare", then its domain membership is revoked by n , and this is announced to other members of PLTD- n . Now, if n wants to include a node z (non-neighbor) in PLTD- n , and z is a one hop neighbor of, say node m which is already a member of PLTD- n , then n can request m to invite z to join PLTD- n . Based on its own trust on z ($T_{m,z}$), m might accept or decline to forward this invitation. If m forwards this invitation, then z can make its decision based on m 's evaluation of trust on n ($T_{m,n}$), and its own trust on m ($T_{z,m}$). Thus, the simplest evaluation of $T_{z,n}$ could be:

$$T_{z,n} = T_{m,n} * T_{z,m} \quad (12)$$

This scheme assumes m 's willingness to provide z with $T_{m,n}$. If z is included in PLTD- n , then since n is the request initiator, $T_{n,z}$ is initially set to "good" by default. $T_{n,z}$ is continuously evaluated afterwards. Simplest evaluation of $T_{n,z}$ could be:

$$T_{n,z} = T_{n,m} * T_{m,z} \quad (13)$$

Again this scheme assumes n 's knowledge of $T_{m,z}$ provided by m . If $T_{n,z}$ falls below "bad" at any time, then n revokes the membership of z in PLTD- n , and announces this decision to other member nodes. Node m can unilaterally decide to end its domain membership in PLTD- n at anytime based on its trust on n $T_{m,n}$ falling below a certain threshold. This scheme is significant in both maintaining an admissible level of trust within a PLTD (because domain members share a group key), and in limiting the domain size. It is important to have an upper bound on the membership size of a PLTD for control, overhead and management purposes. Domain size can also be limited by having an absolute upper bound, say k , on the number of member nodes.

This scheme is also extensible for establishing trust with nodes in other parts of the network, by utilizing trusted one-hop neighbors which move away to other parts of the network due to node mobility. If node m moves away to a different part of the network, then n can utilize this to establish trust

with nodes in the immediate vicinity of m 's new location, provided it is still able to communicate with m . Such a trust establishment procedure would be strictly controlled by the minimum thresholds on pair-wise trust values as mentioned above in this section, and in Sec. 4.

The goal is to provide nodes with a mechanism to evaluate the trust level of its direct neighbors. Our model can be divided in two distinct layers as shown in Figure 1. The Learning layer is responsible for gathering and converting information into knowledge. The Trust layer defines how to assess the trust level of each neighbor using the knowledge information provided by the Learning layer and the information exchanged with direct neighbors. Both layers can interact with all layers of the TCP/IP model. In order to know how trustworthy a given neighbor is, each node assigns a so-called trust level for each direct neighbor. We propose a continuous representation for the trust level, ranging from 0 to 1 where 0 means the least reliable node and 1 means the most reliable node. The transmission of a personal opinion about a specific node i is defined as a recommendation. Neighbor nodes take into account this recommendation while calculating the local trust levels for node i . For that purpose, we introduce the concept of relationship maturity, which is based on the age of the relationship between two nodes.

VI. TRUST MODELS IN MANET'S ROUTING PROTOCOLS

DSR[12] is a routing protocol that is designed for use in a multi-hop environment like wireless mobile ad hoc networks. It allows mobile nodes to organize and configure themselves to form connections between them without any aid of an existing infrastructure or administration. DSR routing protocol reacts to the change of topology of the mobile ad hoc network caused by mobility of mobile nodes in the network or by interferences on the wireless communication links[16].

In DSR, like in most routing protocols, two mechanisms are implemented together for the functionality of the protocol: *route discovery* and *route maintenance*.

Route discovery: if the source node wants to communicate to a destination node to which it does not know the route to, the source node has to use the route discovery techniques to find the route to the intended destination. As shown in figure 2, Route request (RTREQ) message is broadcasted by an initiator of the route discovery process to all nodes within its transmission range. If a node receive the RTREQ and it is not the intended destination or the "target" it also broadcasts the RTREQ further and the process goes on till the target receives the RTREQ. The two important information in the RTREQ are "route record" and "request id". Each RTREQ message contains "route record"; the sequence number of all nodes through which the RTREQ was sent during the route discovery process. Each route contains a unique "request id" set by the initiator of the route discovery process. Each node keeps a list with two entries: initiator's address and request-id for keeping track of the duplicated routes. Upon receipt of a RTREQ message, a node checks first its list. In the first case, if a route with the same information as one of the entries in the

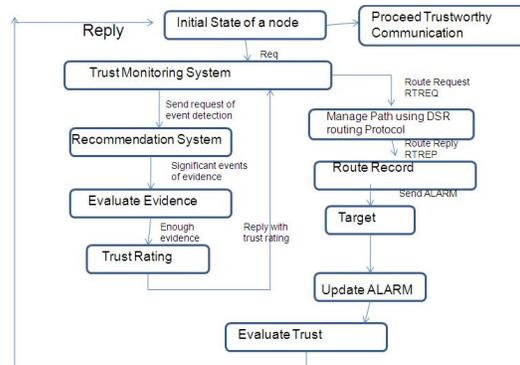


Fig. 2. The Trust Model

list, then the route is not processed further and it is discarded. Discarding a route when it is already in the list ensures that one single RTREQ does not get propagated endlessly, thereby forming loops. In the second case, if a node receives a RTREQ and its address is contained in the "source record", that route is also discarded and it is not processed further, this avoids redundant routes for a single node. In the third case, if the node's address corresponds to the target address in the "route record" then the RTREQ has reached its destination and the route reply (RTREP) message is sent with a copy of the route to the initiator of the route discovery process. If the target receives the RTREQ and if it already has a route to the initiator of the RTREQ, it may use that route to send the RTREP. Otherwise the target will reverse the route used to send the RTREQ and sends the RTREP via that route but this is only possible if the links on all intermediate nodes are bidirectional. If the target does not know the route to the initiator of the route request and if links are not bidirectional, then the piggybacking approach is used. The RTREP messages are piggybacked on a RTREQ message targeted at the initiator of the route discovery to which it is replying. If none of the three cases are true, the node then adds its own address to the "route record" and rebroadcasts the RTREQ message.

Route maintenance: If a route fails at any hop in the path during a communication between source and destination, the node encountering the error sends an error message to the originator of the route. The failing route is detected when a node fails to forward a packet by using periodic broadcast messages. When a route error is received, the node experiencing the error is removed from the cache of the node receiving the error message. All routes containing the node in error must be shortened at that point. If links do not work equally well in both directions then end-to-end acknowledgment is used to detect the failing link.

VII. INTEGRATING TRUST MODULES USING THE EXISTING DSR

To communicate in a network, all mobile nodes must have a unique identifier, usually the IP address. However, in MANET the topology changes dynamically, thus creating difficulties

TABLE I
PATH CHOSEN IN PROPOSED SCHEME

Next hop neighbor in the best path P1	T	U	T
Next hop neighbor in the best path P2	U	U	T

for centralized administration to distribute IP addresses or any other identifier. This situation leads to a distributed, dynamic and automatic service.

Towards establishing trust in MANETs, an integrated approach for auto-configuration, authentication and certification is needed. Auto-configuration provides a service that renders MANET more efficient and robust. Even though there are many approaches related to auto-configuration, none has been standardized. One solution is detecting anomalous activities.

At the Wireless Application layer can use service as the class and can contain following features such as- the total number of requests to the same service, total number of services requested, the average duration of service, the number of nodes that requested service, the total number of service errors etc. A classifier for each service then be described the for each service a normal behavior for its requests. As it mentioned earlier that any node wishes to send messages to a distant node, its sends the ROUTE REQUEST(RTREQ) to all the neighboring nodes. The ROUTE REPLY(RTREP) obtained from its neighbor is sorted by trust ratings. The source selects the most trusted path. If its one hop neighbor node is a friend, then that path is chosen for message transfer. If its one-hop neighbor node is an acquaintance and if the one hop neighbor of the second best path is a Trustworthy then T is chosen otherwise it is U as depicted on the table 1. Similarly an optimal path is chosen based on the degree of trustworthiness existing between the neighbor nodes. The source selects the shortest and the next shortest path. Whenever a neighboring node is rated as trustworthy, the message transfer is done immediately. This eliminates the overhead of invoking the trust estimator between trusted partners. If it is an acquaintance or stranger, transfer is done based on the ratings. This protocol will converge to the DSR protocol if all the nodes in the ad-hoc network are being rated as trustworthy, T .

VIII. CONCLUSION

This paper presented schemes to formalize the notion of pair-wise trust between two nodes in an ad-hoc network. It also presented schemes to evaluate pair-wise trust as a combination of self trust and group trust. This would be helpful in establishing group keys in the network and would also serve as a means of securely grouping nodes into domains in MANETS and would induce distributed control in such networks. In future study, we would like to evaluate the validity of the schemes proposed in this paper through simulations. We are working on integrating node trust models with link and path trust models. Our goal is to design a comprehensive trust based model for mobile ad-hoc networks that can assure an admissible level of security through the use of trust. The choice of DSR as the protocol to use for our work is shown to be based on

its properties of permitting a complete path to be discovered between the source and destinations before communication is initiated between them. However, the concept of ‘trust’, itself remains an open subject. It implies that ‘the need’ for more study in areas such as collecting and distributing information specific to trust, monitoring nodes behavior, storing memories of trust, proving and establishing the reputation of nodes e.g. amongst other subjects that we are considering for future work.

ACKNOWLEDGMENT

This work is made possible through resources and funding provided by the Institute for Integrated and Intelligent Systems(IIS), Griffith University.

REFERENCES

- [1] L.Eschenauer, V.Gligor and J. Baras. *On Trust Establishment in Mobile Ad-Hoc Networks*, Proceedings of 10th International Workshop of Security Protocols, Springer Lecture Notes in Computer Science (LNCS), Apr. 2002.
- [2] P. Krishna, M. Chatterjee, N. Vaidya, D. Pradhan. *A Cluster-based Approach for Routing in Ad-Hoc Networks*. Proc 2nd Symposium on Mobile and Location-Independent Computing. Apr. 1995.
- [3] M. Bechler, H.-J. Hof, D. Kraft, F. Phlke, L. Wolf. *A Cluster-Based Security Architecture for Ad Hoc Networks*. IEEE Infocom. 2004.
- [4] T. Beth, M. Borcherdig and B. Klein. *Valuation of trust in open networks*. Proceedings of ESORICS 1994, November 1994.
- [5] A. Rahman and S. Hailes. *A Distributed Trust Model*. New Security Paradigms Workshop 1997. ACM, 1997.
- [6] D. Balfanz, D. Smetters, P. Stewart and H. Wong. *Talking to Strangers: Authentication in Ad-hoc Wireless Networks*. NDSS. San Diego, 2002.
- [7] T. Hughes, J. Denny, P. Muckelbauer, J. Etl. *Dynamic Trust Applied to Ad Hoc Network Resources*. Autonomous Agents and Multi-Agent Systems Conference, Melbourne, Australia, 2003.
- [8] J. Kong, P. Zerfos, H. Luo, S. Lu and L. Zhang. *Providing Robust and Ubiquitous Security Support for Mobile Ad-hoc Networks*. ICNP, Riverside, CA, 2001.
- [9] M. Virendra and S. Upadhyaya. *Securing Information through Trust Management in Wireless Networks*. Workshop on Secure Knowledge Management (SKM 2004). Buffalo, NY, 2004.
- [10] L. Zhou and Z.J. Haas. *Securing ad hoc networks*. IEEE Network Magazine, vol. 13, no. 6, pp. 24-30, November 1999.
- [11] C. Davis. *A localized trust management scheme for ad hoc networks*. Proceedings of 3rd International Conference on Networking (ICN'04). Mar. 2004.
- [12] David B. Johnson. *Routing in Ad Hoc Networks of Mobile Hosts*. Proceedings of the Workshop on Mobile Computing Systems and Applications, pp. 158-163, IEEE Computer Society, Santa Cruz, CA, December 1994.
- [13] S. Marti, T.J. Giuli, K. Lai, and M. Baker. *Mitigating Routing Misbehavior in Mobile Ad Hoc Networks*. Mobicom 2000, August 2000, pp. 255 265.
- [14] Y. Huang and W. Lee. *A Cooperative Intrusion Detection System for Ad Hoc Networks*. Proceedings of the ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN '03). Fairfax VA, October 2003.
- [15] M. Virendra, M. Jadhwal, M. Chandrasekaran, and S. Upadhyaya. *Quantifying trust in mobile ad-hoc networks*, In Proceedings of IEEE International Conference on Integration of Knowledge Intensive Multi-Agent Systems (KIMAS'05), (Waltham, USA), Apr. 2005.
- [16] D. Johnson and D. Maltz. *Dynamic Source Routing in Ad Hoc Wireless Networks*, Mobile Computing, T. Imielinski and H. Korth, Ed., Kluwer, 1996.