Wireless Personal Communications (2006) 37: 139–163
DOI: 10.1007/s11277-006-1574-5

Trust Establishment In Pure Ad-hoc Networks

A.A. PIRZADA and C. McDONALD

School of Computer Science & Software Engineering, The University of Western Australia, 35 Stirling Highway, Crawley, W.A. 6009, Australia
E-mail: pirzada:chris@csse.uwa.edu.au

Abstract. An ad-hoc network is a set of limited range wireless nodes that function in a cooperative manner so as to increase the overall range of the network. Each node in the network pledges to help its neighbours by passing packets to and fro, in return of a similar assurance from them. All is well if all participating nodes uphold such an altruistic behaviour. However, this is not always the case and often nodes are subjected to a variety of attacks by other nodes. These attacks range from naive passive eavesdropping to vicious battery draining attacks. Routing protocols, data, battery power and bandwidth are the common targets of these attacks. In order to overcome such attacks a number of routing protocols have been devised that use cryptographic algorithms to secure the routing mechanism, which in turn protects the other likely targets. A limiting requirement regarding these protocols is the reliance on an omnipresent, and often omniscient, trust authority. In our opinion, this reliance on a central entity is against the very nature of ad-hoc networks, which are supposed to be improvised and spontaneous. We present in this paper, a trust-based model for communication in ad-hoc networks that is based on individual experience rather than on a third party advocating trust levels. The model introduces the notion of belief and provides a dynamic measure of reliability and trustworthiness in pure ad-hoc networks.

Keywords: trust, security, ad-hoc, networks, protocols

> "Trust is the outcome of observations leading to the belief that the actions of another may be relied upon, without explicit guarantee, to achieve a goal in a risky situation"
> . . . Elofson

1. Introduction

Ad-hoc networks can be established without any existing infrastructure and are hence a suitable candidate for military, emergency and relief scenarios. Through mutual cooperation, nodes in an ad-hoc network create a virtual web of connections to direct network traffic. Each node in an ad-hoc networks acts like an independent mobile router with limited resources and so standard inter-router protocols cannot be immediately adapted to ad-hoc networks. A number of routing protocols have been developed for ad-hoc networks and have been classified into two categories by Royer and Toh [1] as Reactive and Proactive. In reactive routing protocols, in order to preserve precious node battery, routes are only discovered on-demand, while in proactive routing protocols routes are established and maintained at all times and hence consume more battery power than reactive routing protocols.

Ad-hoc networks exist and operate, only if the participating nodes demonstrate a cooperative behaviour. However, in real life this is not always true, and there may always be some nodes that try to sabotage or take undue advantage from this sociable environment. As

routing protocols play a major role in the primary communication set-up, it is fundamental that these protocols perform reliably under all circumstances. A number of protocols were thereby developed to secure the routing process against Byzantine [2] and malicious behaviour. The comparison of these protocols by Pirzada and McDonald [3], revealed that all the secure routing protocols were reliant on a central trust authority for implementing conventional cryptographic algorithms. These protocols gave either the assurance of the existence of 100% security or its non-existence, but none of these presented a transitional level of security protection. Authentication, being one of the initial requirements of any secure communication, necessitates the need for pre-shared keys or digital certificates by participating nodes [4]. The requirement of a central trust authority or pre-configuration of nodes does resolve many of the core security issues in wired networks but it is neither practical nor feasible in an ad-hoc network. To differentiate this environment the term "managed ad-hoc network" was introduced in which the nodes could be configured before the network was established. This contradicts the very aim of ad-hoc networks that endeavour to spontaneously establish an improvised network. We distinguish between the two types of network and call the latter a "pure ad-hoc network", which has no required infrastructure and is created on the fly. We also introduce the notion of trust in ad-hoc networks rather than regular cryptographic add-ons that have superfluous requirements. By computing trust levels from the intrinsic knowledge existent in the network, trustworthy routes can be computed that may not be secure in terms of cryptography but do carry an accurate measure of reliability with them.

This paper is focused on introducing a trust model suitable for application to pure ad-hoc networks. The proposed model has been developed keeping in view the improvised nature of ad-hoc networks. It is true that a number of ad-hoc networks are still established in a managed way but operate in a self-organised manner. However, in this paper we are examining networks that are self-organised both during their establishment as well as in their operation. This may seem like restricting the application of our trust model to applications other than the military or law enforcement agencies. However, we emphasise that our model is equally good for all types of environments. It provides the required level of trust in the absence of a trusted third party and would help reinforce existing trust levels in the presence of a trusted third party. Our approach, when used in conjunction with cryptographic mechanisms, elevates the confidence in the underlying security schemes.

The layout of the paper is as follows. In Section 2 we discuss trust and security issues for ad-hoc networks. In Section 3 we examine attacks that are launched against ad-hoc networks. In Section 4 we describe some relevant previous work. In Section 5 we explain our proposed trust model in detail and present its application to three commonly used ad-hoc network routing protocols in Section 6. The simulation environment is presented in Section 7. The results of the simulation and their analysis are presented in Section 8. The rest of this paper consists of an outline of future work in Section 9 and concluding remarks in Section 10.

2. Trust and Security Issues

Trust and security are two tightly coupled concepts that cannot be desegregated. For example, cryptography is a means to implement security but it is highly dependent on trusted key exchange. Similarly, trusted key exchange cannot take place without requisite security services in place. It is because of this inter-reliance that both of these terms are used interchangeably when defining a secure system. Trust in wired networks is usually achieved using indirect trust

mechanisms, including trusted certification agencies and authentication servers. However, establishing this indirect trust still requires some out-of-band mechanism for initial authentication and is usually augmented with physical or location-based authentication schemes. Trust establishment in ad-hoc wireless networks is still an open and challenging field. Ad-hoc networks are based on naive "trust-your-neighbour" relationships. These relationships originate, develop and expire on the fly and usually have a short lifespan. As the overall environment in such a network is cooperative by default, these trust relationships are extremely susceptible to attacks. For a number of reasons, including better service, selfishness, monetary benefits or malicious intent, some nodes can easily mould these relationships to extract desired goals. Also, the absence of fixed trust infrastructure, limited resources, ephemeral connectivity and availability, shared wireless medium and physical vulnerability, make trust establishment virtually impossible. To overcome these problems, trust has been established in ad-hoc networks using a number of assumptions including pre-configuration of nodes with secret keys, or an omnipresent central trust authority. In our opinion, these assumptions are against the very nature of ad-hoc networks, which are supposed to be improvised and spontaneous. We categorise the ones that are based on assumptions as "managed ad-hoc networks" and those without these as "pure ad-hoc networks".

According to Mayer et al. [5] trust is defined as:

> "The willingness of a party to be vulnerable to the actions of another party based on the expectation that the other party will perform a particular action important to the trustor, irrespective of the ability to monitor or control the party".

Josang [6] defines:

> "Trust in a passionate entity (human) as the belief that it will behave without malicious intent and trust in a rational entity (system) as the belief that it will resist malicious manipulation".

Trust in entities is based on the fact that the trusted entity will not act maliciously in a particular situation. As no one can ever be absolutely sure of this fact, trust is solely dependent on the belief of the trustor. The derivation of trust may be due to direct trust based on previous similar experiences with the same party, or indirect trust based on recommendations from other trusted parties. Trust is also time dependent, it grows and decays over a period of time. A pure ad-hoc network closely resembles this human behaviour model, where a number of people or nodes that have never met each other, are able to communicate with each other based on mutual trust levels developed over a period of time. According to Denning [7]:

> "Trust cannot be treated as a property of trusted systems but rather it is an assessment based on experience that is shared through networks of people".

As in real life, trust levels are determined by the particular actions that the trusted party can perform for the trustee. Similarly trust levels can be computed based on the effort that one node is willing to expend for another node. This effort can be in terms of battery consumption, packets forwarded or dropped or any other such parameter that helps to establish a mutual trust level. A trust model that is based on experience alone may not be secluded from attacks in an ad-hoc network but it can identify routes with a certain measure of confidence.

3. Attacks on Wireless Networks

Two kinds of attacks can be launched against ad-hoc networks [8], passive and active. In passive attacks the attacker does not disturb the routing protocol or data packets. It only eavesdrops upon the network traffic in order to extract valuable information like node hierarchy and network topology from it. For example, if a route to a particular node is requested more frequently than to other nodes, the attacker might anticipate that the node is vital for the operation of the network, and putting it out of action could bring down the entire network. Similarly, even when it might not be possible to isolate the precise position of a node, one may be able to determine information about the network topology by analysing the contents of routing packets. This attack is virtually impossible to detect in the wireless environment and hence also extremely difficult to prevent.

In active attacks, the aggressor node has to expend some of its energy in order to carry out the attack. Nodes that perform active attacks with the aim of disrupting other nodes by causing network outage are considered to be malicious, while nodes that perform passive attacks with the aim of saving battery life for their own communications are considered to be selfish. In active attacks, malicious nodes can disrupt the correct functioning of a routing protocol by modifying routing information, by fabricating false routing information, or by impersonating nodes [9].

3.1. Attacks Using Modification

Routing protocols for ad-hoc networks are based on the assumption that intermediate nodes do not maliciously change the protocol fields of messages passed between nodes. This assumed trust permits malicious nodes to easily generate traffic subversion and denial of service (DoS) attacks. Attacks using modification are generally targeted against the integrity of routing computations and so by modifying routing information an attacker can cause network traffic to be dropped, redirected to a different destination, or to take a longer route to the destination increasing communication delays. An example is when an attacker sends fake routing packets to generate a routing loop, causing packets to pass through nodes in a cycle without getting to their actual destinations, consuming energy and bandwidth. Similarly, by sending forged routing packets to other nodes, all traffic can be diverted to the attacker or to some other node. The idea is to create a *black hole* by routing all packets to the attacker and then discarding them. As an extension to the black hole, an attacker could build a *grey hole*, in which it intentionally drops some packets but not others, for example, forwarding routing packets but not data packets. A more subtle type of modification attack is the creation of a tunnel (or *wormhole*) [10] in the network between two colluding malicious nodes linked through a private network connection. This exploit allows a node to short-circuit the normal flow of routing messages by creating a virtual vertex cut in the network that is controlled by the two colluding attackers.

3.2. Attacks Using Fabrication

Fabrication attacks are performed by generating false routing messages. These attacks are difficult to identify as they are received as legitimate routing packets. The rushing attack [11] is a typical example of malicious attacks using fabrication. This attack is carried out against on-demand routing protocols that hold back duplicate packets at every node. An attacker rapidly spreads routing messages all through the network, suppressing legitimate routing messages

when nodes discard them as duplicate copies. Similarly, an attacker can nullify an operational route to a destination by fabricating routing error messages asserting that a neighbour can no longer be contacted.

3.3. Attacks Using Impersonation

A malicious node can initiate many attacks in a network by masquerading as another node (spoofing). Spoofing occurs when a malicious node misrepresents its identity by altering its MAC or IP address in order to alter the view of the network topology that a benign node can gather. As an example, a spoofing attack allows the creation of loops in the routing information collected by a node, with the result of partitioning the network.

4. Previous Work

4.1. Distributed Trust Model

The Distributed Trust Model [12] makes use of a protocol to exchange, revoke and refresh recommendations about other entities. By using a recommendation protocol each entity maintains its own trust database. This ensures that the trust computed is neither absolute nor transitive. The model uses a decentralised approach to trust management and uses trust categories and values for computing different levels of trust. The integral trust values vary from 1 to 4 signifying discrete levels of trust from complete distrust (1) to complete trust (4). Each entity executes the recommendation protocol either as a recommender or a requestor and the trust levels are computed using the recommended trust value of the target and its recommenders. The model has provision for multiple recommendations for a single target and adopts an averaging mechanism to yield a single recommendation value. The model is most suitable for less formal, provisional and temporary trust relationships and does not specifically target ad-hoc networks. Moreover, as it requires that recommendations about other entities be passed, the handling of false or malicious recommendations has to be supported via some out-of-band mechanism.

4.2. Distributed Public-Key Model

The Distributed Public-Key Model [13] makes use of threshold cryptography to distribute the private key of the Certification Authority over a number of servers. An $(n, t + 1)$ scheme allows any $t + 1$ servers out of total of n servers to combine their partial keys to create the complete secret key. Similarly, it requires that at least $t + 1$ servers must be compromised to acquire the secret key. The scheme is quite robust but has a number of factors that limit its application to pure ad-hoc networks. Primarily it requires an extensive pre-configuration of servers and a distributed central authority, secondly the $t + 1$ servers may not be accessible to any node desiring authentication and lastly asymmetric cryptographic operations are known to drain precious node batteries.

4.3. PGP Model

In the Pretty Good Privacy Model [14] all users act as independent certification authorities and have the capability to sign and verify keys of other users. PGP breaks the traditional central trust authority architecture and adopts a decentralised "web of trust" approach. Each individual signs the keys of all other users in order to build a set of virtual interconnecting links of trust. PGP attaches various degrees of confidence levels from "undefined" to "complete trust" to the trustworthiness of public-key certificates and four levels of trustworthiness of introducers from "don't know" to "full trust". Based on these trust levels, the user computes the trust level of the desired party. PGP is suitable for wired networks where a central key server can maintain a database of keys. However, in ad-hoc networks, creation of a central key server creates a single point of failure and also requires uninterrupted access to the nodes. The other option, as in PGP, is where each node stores a subset of the public keys of other users using a subset of the trust graph [15] and merges these graphs with graphs of other users in order to discover trusted routes. This scheme involves extensive computation and memory requirements and is deemed limiting for ad-hoc networks.

4.4. Resurrecting Duckling Model

The Resurrecting Duckling Model [16] is based upon a hierarchical graph of master-slave relationships. The slave (duckling) considers the first node that sends it a secret key through a secure channel as its master (mother duck). The slave always obeys the master and gets all instructions and access control lists from its master. The slave further becomes a master to other devices with whom it can share a secret key through secure means. This master-slave bond can only be broken either by a master, a timeout, or an event, after which the slave is no longer bonded and looks for another master. This model is most suitable for security in large-scale sensor nodes where pre-configuration has to be avoided. As this model demands a hierarchical security chain it is not appropriate for application to ad-hoc networks.

5. The Trust Model

Our trust model is an adaptation of the trust model by Marsh [17] configured for use in pure ad-hoc networks. Marsh's model computes situational trust in agents based upon the general trust in the trustor and in the importance and utility of the situation in which an agent finds itself. General trust is basically the trust that one entity assigns another entity based upon all previous transactions in all situations. Utility is considered similar to knowledge so that an agent can weigh up the costs and benefits that a particular situation holds. Importance caters for the significance of a particular situation to the trustor based upon time. In order to reduce the number of variables in our model, we merge the utility and importance of a situation into a single variable called weight, which may increase or decrease with further experiences over time.

The trust model primarily represents three layers of trust. The finest layer of trust is extracted by monitoring and evaluating the events that the nodes are able to observe. These events are then combined to form the situational trust categories, which in essence represent trust in other nodes in different dimensions. These situational trust categories are then combined to form the aggregate trust, which represents a coarser level of trust in other nodes in the network. The

<table>
<tr><th>OSI</th><th>PROPOSED</th><th>TCP/IP</th></tr>
<tr><td>Application</td><td rowspan="3">Computation
&
Quantification</td><td rowspan="3">Application</td></tr>
<tr><td>Presentation</td></tr>
<tr><td>Session</td></tr>
<tr><td>Transport</td><td rowspan="4">Derivation</td><td>Transport</td></tr>
<tr><td>Network</td><td>Internet</td></tr>
<tr><td>Data link</td><td rowspan="2">Host to Network</td></tr>
<tr><td>Physical</td></tr>
</table>

Figure 1. Structure of trust agent.

aggregate trust layer permits a node to get a general perspective of another node, while the other two layers permit a much refined evaluation and assignment of trust values.

The trust model is executed using trust agents that reside on network nodes. Each agent operates independently and maintains its individual perspective of the trust hierarchy. An agent gathers data from events in all states, filters it, assigns weights to each event and computes different trust levels based upon them [18]. Each trust agent basically performs the following three functions: Trust Derivation, Quantification, and Computation. Approximate partitioning of these functions in comparison with the OSI reference model and the TCP/IP protocol suite is represented in Figure 1.

5.1. Trust Derivation

We compute the trust in our model based upon the information that one node can gather about the other nodes in passive mode, i.e. without requiring any special interrogation packets. Vital information regarding other nodes can be gathered by analysing the received, forwarded and overheard packets if appropriate taps are applied at different protocol layers. Possible events that can be recorded in passive mode are the measure and accuracy of:

1. Frames received,
2. Data packets forwarded,
3. Control packets forwarded,
4. Data packets received,
5. Control packets received,
6. Streams established,
7. Data received, and
8. Data forwarded.

The information from these events is classified into one or more trust categories. Trust categories signify the specific aspect of trust that is relevant to a particular relationship and are used to compute trust in other nodes in specific situations. For example, we might trust a particular node for the category “data forwarding” but not for the category of “accurate routes”.

5.2. Trust Quantification

Discrete representation of trust is not sufficient to clearly represent trust that normally exhibits a continuous trend. Secure routing protocols represent trust levels by either the presence or absence of security. PGP represents trust using four values ranging from unknown to fully trusted. Discrete values, although easy to represent and classify, are not suitable to represent

trust in ad-hoc networks. Trust in ad-hoc networks is always in a fluid state and is continuously changing due to the mobility of the nodes. As the period of interaction with any node may be brief, it is imperative that trust be represented as a continual range to differentiate between nodes with comparable trust levels. In our trust model we represent trust from -1 to $+1$ signifying a continuous range from complete distrust to complete trust.

5.3. Trust Computation

Trust computation involves an assignment of weights (representing utility or importance factor) to the events that were monitored and quantified. The assignment is totally dependent on the type of application demanding the trust level and varies with state and time. All nodes dynamically assign these weights based upon their own criteria and circumstances. These weights have a continuous range from 0 to $+1$ representing the significance of a particular event, from unimportant to most important. The trust values for all the events from a node can then be combined using individual weights to determine the aggregate trust level for another node. We define this trust T, in node y, by node x, as T_{xy} and is given by the following equation:

$$T_{xy} = \sum_{i=1}^{n} [W_{xy}(i) \times T_{xy}(i)]$$

where $W_{xy}(i)$ is the weight of the ith trust category of node y to node x and $T_{xy}(i)$ is the situational trust of node x in the ith trust category of node y. The total number of trust categories n is dependent on the protocol and scenario to which the trust model is being applied.

6. Extension to Ad-hoc Network Routing Protocols

To demonstrate the utility of our Trust Model, we present in this section, its applicability to three routing protocols commonly used in ad-hoc networks. As these protocols are currently under their development phase, and are being constantly improved, we have applied our model to the latest known working versions. We state how trust can be extracted from each of these protocols using different trust categories. A number of other trust categories can also be extracted from each of these protocols, but we have described here only those that have maximum impact on trust development.

6.1. DSR Protocol

The Dynamic Source Routing (DSR) protocol by Johnson et al. [19] is an on-demand routing protocol. Its most interesting feature is that all data packets sent using the DSR protocol have absolutely no dependency on intermediate nodes regarding routing decisions, as each carries the complete route it traverses. When a node requires a route to a particular destination, it broadcasts a `ROUTE REQUEST` packet. Each recipient node that has not seen this specific `ROUTE REQUEST` and has no knowledge about the required destination rebroadcasts this `ROUTE REQUEST` after appending its own address to it. If this `ROUTE REQUEST` reaches the destination or an intermediate node that has a route to the destination in its cache of routes, it sends a `ROUTE REPLY` packet containing the complete route from the source to the destination. The source

node may receive a number of such ROUTE REPLY packets and may decide to select a particular route based upon the number of hops, delay or other such criteria. All nodes forwarding or overhearing any packets must add all usable routing information from that packet to their own cache of routes. For route maintenance, intermediate nodes that find any route broken, return a ROUTE ERROR packet to each node that had sent a packet over that particular route.

6.2. AODV Protocol

The Ad-hoc On-Demand Distance Vector (AODV) by Perkins et al. [20] is inherently a distance vector routing protocol that has been optimised for ad-hoc wireless networks. It is an on demand protocol as it finds the routes only when required and is hence also reactive in nature. AODV borrows basic route establishment and maintenance mechanisms from the DSR protocol and hop-to-hop routing vectors from the DSDV protocol [21]. To avoid the problem of routing loops, AODV makes extensive use of sequence numbers in control packets. When a source node intends communicating with a destination node whose route is not known, it broadcasts a ROUTE REQUEST packet. Each ROUTE REQUEST packet contains an ID, source and the destination node IP addresses and sequence numbers together with a hop count and control flags. The ID field uniquely identifies the ROUTE REQUEST packet; the sequence numbers provide the freshness of control packets and the hop-count maintains the number of nodes between the source and the destination. Each recipient of the ROUTE REQUEST packet that has not seen the Source IP and ID pair or doesn't maintain a fresher (indicated by a larger sequence number) route to the destination rebroadcasts the same packet after incrementing the hop-count. Such intermediate nodes also create and preserve a REVERSE ROUTE to the source node for a certain interval of time. When the ROUTE REQUEST packet reaches the destination node or any node that has a fresher route to the destination a ROUTE REPLY packet is generated and transmitted back to the source of the ROUTE REQUEST packet. Each ROUTE REPLY packet contains the destination sequence number, the source and the destination IP addresses, route lifetime together with a hop count and control flags. Each intermediate node that receives the ROUTE REPLY packet, increments the hop-count, establishes a FORWARD ROUTE to the source of the packet and transmits the packet on the REVERSE ROUTE. To preserve connectivity information, AODV makes use of periodic HELLO messages to detect link breakages to nodes that it considers as its immediate neighbours. If a link break is detected for a next hop of an active route, a ROUTE ERROR message is sent to its active neighbours that were using that particular route.

6.3. TORA Protocol

The Temporally Ordered Routing Algorithm (TORA) by Park and Corson [22] is a distributed routing protocol for multi-hop networks. The unique feature of this protocol is that it endeavours to localize the spread of routing control packets. The protocol is basically an optimised hybrid of the Gafni Bertsekas (GB) protocol [23] and the Lightweight Mobile Routing (LMR) [24] protocol. It guarantees loop freedom, multiple routes and minimal communication overhead even in highly dynamic environments. The protocol attempts to minimise routing discovery overhead and in so doing prefers instant routes to optimal routes. The protocol supports source-initiated on-demand routing for networks with a high rate of mobility as well as destination oriented proactive routing for networks with lesser mobility. TORA maintains state on a per-destination basis and runs a logically separate instance of the algorithm for each destination.

TORA assigns directional heights to links so as to direct the flow of traffic from a higher source node to a lower destination. The significance of these heights, which are assigned based on the direction of a link towards the destination, is that a node may only forward packets downstream but not upstream, i.e. to another node that has a higher, undefined or unknown height. The height is represented by a quintuple (τ, oid, r, δ, i) where the first three values represent a reference level and the last two represent the change with respect to the reference level. Each time a node loses its downstream link due to a link failure, a new reference level is computed using either a partial or full link reversal mechanism. The values in the height quintuple indicate the following:

τ Logical time of a link failure
oid Unique ID of the router that defined the reference level
r Reflection indicator bit
δ Propagation ordering parameter
i Unique ID of the router

In the on-demand mode, TORA algorithm performs three routing functions: Route Creation, Route Maintenance and Route Erasure. To accomplish these functions it uses three distinct control packets: Query (`QRY`), Update (`UPD`) and Clear (`CLR`). During route discovery, a source node requiring a route to a destination, broadcasts a `QRY` packet containing the destination address. The `QRY` packet is propagated through the network until it reaches the destination or any intermediate node possessing a route to the intended destination. The recipient of the `QRY` packet broadcasts an `UPD` packet that lists its height with respect to the destination. If the destination itself replies to a `QRY` packet it sets the height to zero in the `UPD` packet. Each node that receives the `UPD` packet sets its own height greater than that in the `UPD` packet. This results in creation of a directed acyclic graph (DAG) with all links pointing in the direction of the destination as the root. In the proactive mode, routes are created using the Optimisation (`OPT`) packet that is sent out by the destination. The `OPT` packet, which is similar to the `UPD` packet, also consists of a sequence number for duplication avoidance. Each recipient nodes adjusts its height data structure and sends out a `OPT` packet to neighbouring nodes.

When a node discovers that it has no downstream links, either due to a link failure or to a link reversal, it modifies its height based upon five predefined cases. The first case (`GENERATE1`) is applicable when there are no downstream links due to a link failure, in which the node defines a new reference level. All other cases (`PROPAGATE`, `REFLECT`, `DETECT` and `GENERATE2`) are executed by nodes having no downstream links due to a link reversal. These cases define and propagate new reference levels upon reception of `UPD` packets based on different criteria. Whenever a partition is spotted through a `DETECT` case, the node sets its own and the height of all its neighbours to `NULL` and broadcasts a `CLR` packet. The neighbouring nodes that receive the `CLR` packet, based upon the reference level, also set the heights in a similar manner and rebroadcast the `CLR` packet. In this way the height of each node in the portion of the network that is partitioned is set to `NULL` and all invalid routes are erased. As each node in TORA maintains multiple DAGs to the destination so in any network with an average n number of nodes each with $\frac{n}{2}$ downstream neighbours, a node could still effectively communicate with the destination node upon link failure of $\frac{n}{2} - 1$ nodes. However, to sustain this redundancy, each node maintains a height data structure, link status along with a number of state and auxiliary variables for each destination node.

TORA is not a standalone routing protocol but requires the services of the Internet MANET Encapsulation Protocol (IMEP) proposed by Corson et al. [25]. IMEP has been designed as a

network layer protocol that provides link status, neighbour connectivity information, address resolution and other services to Upper Layer Protocols (ULP).

6.4. TRUST DERIVATION

In all three routing protocols, we use one or more of their inherent features to build up the following trust categories:

6.4.1. *Acknowledgments (P_A)*

Applicability: DSR, AODV and TORA. A node can get information about the successful transmission of any packet that it sent, through the following three methods:

6.4.1.1. *Link-Layer Acknowledgements.* Using Link-Layer acknowledgments the underlying MAC protocol provides feedback of the successful delivery of the transmitted data packets.

6.4.1.2. *Passive Acknowledgements.* In this method the sender node places itself in promiscuous mode after the transmission of any packet so as to overhear the retransmission by the recipient nodes.

6.4.1.3. *Network Layer Acknowledgements.* In AODV, this method permits the sender of a `ROUTE REPLY` packet to explicitly request a `ROUTE REPLY ACKNOWLEDGEMENT` from the recipient of the `ROUTE REPLY` packet. In DSR, the sender is permitted to explicitly request a network layer acknowledgement from the next hop using the DSR options header. In TORA, network layer acknowledgements are sent in response to object block receptions in IMEP when either reliable delivery is required or a receiver is implicitly or explicitly included in the response list. Once a neighbouring node has acknowledged a given block of data, it is removed from the response list and is not required to acknowledge future retransmissions.

All of the above methods provide information about the successful transmission of a packet. However, the passive acknowledgment method also provides us with the following information about the next hop, including:

1. It is acting like a black hole if the packet is dumped and not retransmitted,
2. It is carrying out a modification attack if the contents have been fallaciously modified,
3. It is carrying out a fabrication attack if a self generated fallacious packet is transmitted,
4. It is carrying out an impersonation attack if the MAC or IP addresses have been spoofed,
5. It is showing selfish behaviour by not retransmitting a packet, and
6. It is inducing latency delays by delaying the retransmission of the packet.

The method of passive acknowledgment can be further classified into acknowledgements for data packets and acknowledgements for control packets. The number of these acknowledgements occurring with respect to every node are maintained and tabulated as shown in Table 1. For every packet transmitted, the appropriate counter in the table for success or failure is incremented, depending if the neighbouring node has correctly forwarded it or not.

6.4.2. *Packet Precision (P_P)*

Applicability: DSR, AODV and TORA. The category Packet Precision ensures the integrity of the data and control packets that are either received or forwarded by other nodes in the network.

Table 1. Trust table for Category P_A

Node acknowledgement									
Route request/QRY (R_q)		Route reply/UPD (R_p)		Route error/CLR (R_e)		Route optimisation/OPT (R_o)		Data (D)	
Success R_{qs}	Fail R_{qf}	Success R_{ps}	Fail R_{pf}	Success R_{es}	Fail R_{ef}	Success D_{os}	Fail D_{of}	Success D_s	Fail D_f

Table 2. Trust table for Category P_P

Node packet precision									
Route request/QRY (R_q)		Route reply/UPD (R_p)		Route error/CLR (R_e)		Route optimisation/OPT (R_o)		Data (D)	
Success R_{qs}	Fail R_{qf}	Success R_{ps}	Fail R_{pf}	Success R_{es}	Fail R_{ef}	Success D_{os}	Fail D_{of}	Success D_s	Fail D_f

For intermediate nodes, which execute a distance vector routing protocol, the received `ROUTE REQUEST` packets are used to create the `REVERSE ROUTES` to the source. These routes are later used to send data packets to the source nodes. In the same way, the `ROUTE REPLY` packets help in forming `FORWARD ROUTES` that lead to the destination. The correctness of these control packets plays a vital role in the establishment of accurate routes through the network. Similarly, for source routing protocols the accuracy of control packets is equally imperative.

The precision of the control packets is determined when the routes are actually utilised while that of the data packets is verified during forwarding. For instance, if routing packets are received that are found to be correct and efficient, then the originator can be allotted a higher trust value along with the set of nodes provided in that packet. The above method can be further categorised into data and control packet types and allocated different trust values as shown in Table 2. Counters are maintained for every received packet and are incremented based upon the accuracy or inaccuracy of the packet.

6.4.3. *Gratuitous Route Replies (G_R)*

Applicability: DSR and AODV. The DSR protocol provides the facility of "route shortening" to avoid unnecessary intermediate nodes. For example, if a node overhears a data packet that is supposed to traverse a number of nodes before passing through it, then this node creates a shorter route known as `GRATUITOUS ROUTE REPLY` and sends it to the original sender. The AODV protocol permits intermediate nodes with fresher routes (larger destination sequence numbers) to respond to `ROUTE REQUEST` queries if the `DESTINATION ONLY` flag is not set in the `ROUTE REQUEST` packet. This reduces the overall latency delays and resource utilisation in contrast to the case where the `ROUTE REPLY` packets are generated only by the destination node. Although beneficial, these intermediate `ROUTE REPLY` packets don't inform the destination node regarding a route to the source node. Thus for connection-oriented sessions, the destination node has to initiate another route discovery process for the intended communication. In order to avoid such repetitive route discoveries the source node can set a `GRATUITOUS ROUTE REPLY` flag in the `ROUTE REQUEST` packet. If this flag is set then the intermediary responder to a `ROUTE REQUEST` is also required to create and unicast an appropriate `ROUTE REPLY` to the final destination. The `GRATUITOUS ROUTE REPLY` packets can be considered as a trust category as they provide the following information about the sender:

Table 3. Trust table for Category G_R

Node	
Gratuitous route replies (G)	
Success G_s	Fail G_f

Table 4. Trust table for Category B_L

Node	Present in Blacklist (B)

1. It is displaying either malicious or benevolent behaviour, and
2. It is not showing selfish behaviour.

If the Gratuitous Route is found to be accurate, then the originator can be allotted a higher trust value along with the set of nodes provided in that route. The above method can be used to allocate different trust values to different nodes, as shown in Table 3. All `GRATUITOUS ROUTE REPLY` packets that are found to be correct or incorrect are recorded using appropriate counters.

6.4.4. *Blacklists (B_L)*
Applicability: DSR and AODV. DSR and AODV maintain blacklists for nodes displaying uni-directional behaviour, i.e. if a neighbour node has received a packet and either due to a unidirectional link or selfish behaviour the sender cannot hear it retransmitting. If the MAC protocol is expected to provide feedback (like IEEE 802.11) then this implies that the links must be bi-directional and the neighbour node is acting selfishly. The blacklists can be used to provide trust values for nodes while computing route confidence levels. The format of the trust table based on blacklists is shown in Table 4.

6.4.5. *BEACON/HELLO Packets (H_M)*
Applicability: AODV and TORA. AODV uses `HELLO` packets to maintain local neighbourhood connectivity information. Each active node that has not sent any broadcast in a certain period, broadcasts a `HELLO` packet (`ROUTE REPLY` Packet with Hop-Count = 0) with time-to-live set to 1. These packets ensure that all neighbours maintain active routes between each other at all times. All recipient nodes create forward routes to the transmitting node. The absence of a `HELLO` packet from a neighbour for a certain duration makes the route to that node invalid. `HELLO` packets indicate that a node is actively participating in the routing mechanism and not acting as either a passive eavesdropper or a selfish node. TORA, uses `BEACON` packets to ascertain the connection status between adjacent nodes. Each node periodically broadcasts a `BEACON` that contains a Router Identification number. In response to the `BEACON`, every recipient node broadcasts an `ECHO` packet that contains its IP address. These packets ensure that all nodes maintain local neighbourhood connectivity information at all times.

This feature can be used to access the reliability of the neighbours based upon the precision of these messages. The format of the trust table based on these messages is shown in Table 5.

6.4.6. *Destination Unreachable Messages (D_U)*
Applicability: AODV. In AODV, data packets that are waiting for a route to be established are usually buffered at the node requesting the route. This buffering is accomplished in a "First-in

Table 5. Trust table for Category H_M

Node	
Beacon/Hello packet (H)	
Success H_s	Fail H_f

Table 6. Trust table for Category D_U

Node	
Destination unreachable message (U)	
Success U_s	Fail U_f

First-out" manner. However, if the route discovery exceeds the maximum number of `ROUTE REQUEST` retries then the data packets are dropped from the buffer. However, a `DESTINATION UNREACHABLE MESSAGE` is returned to the sending application. These messages inform the recipient regarding the following:

1. It is not acting like a black hole,
2. It is displaying either malicious or benevolent behaviour, or
3. It is not showing selfish behaviour.

These messages can be used to provide trust values for nodes while computing route confidence levels. The format of the trust table based on `DESTINATION UNREACHABLE MESSAGE` is shown in Table 6.

6.4.7. *Salvaging (S_G)*
Applicability: DSR. In DSR, if an intermediate node receives a packet for which its next hop is not available, it may drop the packet and inform the sender. However, if it has a route to the final recipient it can salvage that route from its cache, send the packet on the new route and inform the sender about the failed link. If the salvaged route is found to be correct then it reveals that the sender of the route error is displaying a benevolent and altruistic behaviour. Hence, this information can be used to build up trust levels and considered as a trust category. All salvaged route errors found to be correct or incorrect are recorded using counters, as shown in Table 7.

6.4.8. *Authentication Objects (A_O)*
Applicability: TORA. In TORA, IMEP enabled nodes are able to support multiple types of authentication from simple to complex verification schemes. The IMEP messages between

Table 7. Trust table for Category S_G

Node	
Salvage route error (S)	
Success S_s	Fail S_f

Table 8. Trust table for Category A_O

Node	
Authentication objects (A)	
Success A_s	Fail A_f

any two nodes are verified using IMEP Authentication Objects [26], which are passed with all IMEP messages. The Authentication Objects contains a digital signature of the contents of the IMEP message. Based on the type of Public Key Infrastructure (PKI) the nodes have the option to pass the certificate along with every IMEP message. The recipient node uses the Public Key from the certificate and the digital signature to verify the contents of the IMEP message. The Authentication Objects can be used to define a trust category. The trust table based upon Authentication Objects is shown in Table 8.

6.5. Trust Quantification

The events recorded during the trust derivation process are quantised and assigned weights so as to compute the situational trust values for different nodes. For example, in order to compute the situational trust in category Passive Acknowledgements (P_A), the events recorded in Table 1 are first quantized using the following equations:

$$R_p = \frac{R_{ps} - R_{pf}}{R_{ps} + R_{pf}} \text{ for } R_{ps} + R_{pf} \neq 0 \text{ else } R_p = 0$$
$$R_q = \frac{R_{qs} - R_{qf}}{R_{qs} + R_{qf}} \text{ for } R_{qs} + R_{qf} \neq 0 \text{ else } R_q = 0$$
$$R_e = \frac{R_{es} - R_{ef}}{R_{es} + R_{ef}} \text{ for } R_{es} + R_{ef} \neq 0 \text{ else } R_e = 0$$
$$R_o = \frac{R_{os} - R_{of}}{R_{os} + R_{of}} \text{ for } R_{os} + R_{of} \neq 0 \text{ else } R_o = 0$$
$$D = \frac{D_s - D_f}{D_s + D_f} \text{ for } D_s + D_f \neq 0 \text{ else } D = 0$$

By normalising the values of R_p, R_q, R_e, R_o and D we limit the trust values between -1 to $+1$. Negative values for trust can occur as a result of more failures than successes for an event. Hence, a trust value of -1 represents complete distrust, a value of 0 implies a non-contributing event and a value of $+1$ means absolute trust in a particular event. These trust levels are then assigned weights in a static or dynamic manner depending on their utility and importance. The situational trust $T_{xy}(P_A)$ in node y for trust category P_A is computed by node x using the following equations:
For DSR/AODV

$$T_{xy}(P_A) = W_{xy}(R_p) \times R_p + W_{xy}(R_q) \times R_q + W_{xy}(R_e) \times R_e + W_{xy}(D) \times D$$

For TORA

$$T_{xy}(P_A) = W_{xy}(R_p) \times R_p + W_{xy}(R_q) \times R_q + W_{xy}(R_e) \times R_e + W_{xy}(R_o) \times R_o + W_{xy}(D) \times D$$

where W_{xy} is the weight assigned by node x to the event that took place with node y. Similarly, all other events recorded in Tables 2–8 are quantised and weighed, in order to determine the situational trust in categories P_P, G_R, B_L, H_M, D_U, S_G and A_O respectively.

6.6. Trust Computation

The situational trust values from all trust categories (P_A, P_P, G_R, B_L, H_M, D_U, S_G and A_O) are then combined according to their assigned weights, to compute an aggregate trust level for a particular node. The aggregate trust T in node y by node x is represented as T_{xy} and given by the following equations:
DSR

$$T_{xy} = W_{xy}(P_A) \times T_{xy}(P_A) + W_{xy}(P_P) \times T_{xy}(P_P) + W_{xy}(G_R) \times T_{xy}(G_R) + W_{xy}(B_L) \times T_{xy}(B_L) + W_{xy}(S_G) \times T_{xy}(S_G)$$

AODV

$$T_{xy} = W_{xy}(P_A) \times T_{xy}(P_A) + W_{xy}(P_P) \times T_{xy}(P_P) + W_{xy}(G_R) \times T_{xy}(G_R) + W_{xy}(B_L) \times T_{xy}(B_L) + W_{xy}(H_M) \times T_{xy}(H_M) + W_{xy}(D_U) \times T_{xy}(D_U)$$

TORA

$$T_{xy} = W_{xy}(P_A) \times T_{xy}(P_A) + W_{xy}(P_P) \times T_{xy}(P_P) + W_{xy}(H_M) \times T_{xy}(H_M) + W_{xy}(A_O) \times T_{xy}(A_O)$$

where W_{xy} represents the weight assigned to a situational trust category of node y by node x.

The aggregate trust tables for the DSR, AODV and TORA protocols are shown in Table 9. The aggregate and situational trust values are maintained and updated for each node based upon the frequency of events and severity of the situation.

Two kinds of initial trust values can be set in the aggregate trust tables: Neutral or Trustworthy. In the neutral category the trust value of the node is set to 50%. This implies that all nodes are initially considered as neutral until the time they have interactions with other nodes in the network. Based upon these interactions the trust values can either improve or diminish. However, this approach causes a problem when dealing with nodes portraying grey holes. As these nodes vary their packet drop pattern, it is possible that the trust level of such nodes may increase than that of a neutral node. Such an occurrence would prevent neutral nodes from getting analysed by the trust model in the presence of any grey hole in their near vicinity. The other mechanism, which we have used, is to set each node initially as trustworthy or with 100% trust level. Nodes that execute the protocol in a benevolent manner thus maintain a higher trust level than those which are detected as selfish or malicious by the trust model. This mechanism

Table 9. Aggregate trust tables

DSR

Node	Passive acknowledgement	Packet precision	Gratuitous route replies	Black lists	Salvage route errors	Aggregate trust level
y	$T_{xy}(P_A)$	$T_{xy}(P_P)$	$T_{xy}(G_R)$	$T_{xy}(B_L)$	$T_{xy}(S_G)$	T_{xy}

AODV

Node	Passive acknowledgement	Packet precision	Gratuitous route replies	Black lists	Beacon/ Hello packets	Destination unreachable messages	Aggregate trust level
y	$T_{xy}(P_A)$	$T_{xy}(P_P)$	$T_{xy}(G_R)$	$T_{xy}(B_L)$	$T_{xy}(H_M)$	$T_{xy}(D_U)$	T_{xy}

TORA

Node	Passive acknowledgement	Packet precision	Authentication objects	Beacon/ hello packets	Aggregate trust level
y	$T_{xy}(P_A)$	$T_{xy}(P_P)$	$T_{xy}(A_O)$	$T_{xy}(H_M)$	T_{xy}

is resilient to the grey hole attack, since any malicious activity by a node, drops its trust level further down in comparison to that of the neighbouring nodes.

6.7. Trust Application

In DSR, AODV and TORA protocols, before initiating a new route discovery, the routing table or cache is first scanned for a working route to the destination. In the event of unavailability of a route from the routing table or cache, the `ROUTE REQUEST` or `QRY` packet is propagated. Accordingly, when the search is made for a route in the table or cache, the least cost path in terms of number of hops is always returned. Similarly, each forwarding node scans its local routing table or cache to find the least hop path leading to the packet's destination. We modify this rule and associate the aggregate trust values as the cost of nodes. Each time a packet is sent or forwarded, the sending or forwarding node first scans the routing table or cache for all alternate paths leading to the same destination. It then compares the aggregate trust levels of all next hops in these paths and selects the one with the highest trust level. In case, the next hop is not previously known to the sender or forwarder, then the path with the least distance to the destination is selected.

All sending and forwarding nodes also try to minimise the cost by selecting adjacent nodes, which have an aggregate trust level equal to or greater than the specified trust threshold (T_T). In case there is no next hop available, with a trust level greater than the trust threshold, then a local link repair is initiated at the intermediate nodes. Any received data packet, with an unavailable trustworthy next hop, is buffered for a certain interval in the Interface Queue so as to facilitate the discovery of an alternate trustworthy route leading to the same destination. In case an alternate route is found, the packet is sent onto that route. A `ROUTE ERROR` packet is also propagated to the source node informing it of the link severing.

The trust threshold can be specified in a variety of ways depending on the application. A higher threshold enforces a rigid forwarding criterion that is to be met by all network nodes. Such a threshold is helpful in cases where precise throughput and integrity verification is

required. However, there is high probability that a traffic saturated node may be incorrectly diagnosed as malicious by the trust model. On the other hand, a lower threshold provides this requisite leverage and is hence used to detect nodes that portray sustained malicious behaviour.

Under high mobility conditions, packet drop may be experienced by benevolent nodes due to saturation of Interface Queues. Similarly, packet drop can also occur as a consequence of MAC layer collisions occurring due to congestion in the network. These legitimate packet drops are influenced by the mobility pattern and traffic load of the network and accordingly influence the different performance metrics of the network. This is confirmed by the fact that the throughput of the protocols always remains lower than 100% even when no malicious nodes are present in the network. However, the ratio of legitimate packet drop to deliberate drop is negligible even under high mobility, as will be shown in the results. This disparate behavioural pattern thus permits selection of a suitable trust threshold value, which can successfully differentiate between malevolent and benevolent behaviour under diverse mobility conditions.

The size of the trust tables is limited to the number of nodes involved during the definition of any category. However, this number can vary according to the mobility, density and traffic load of the network. Therefore, any node, depending upon its memory and computational constraints, may adopt a purging policy, where the trust entries for nodes portraying sustained benevolent behaviour ($T_{xy} \geq T_T$) may be periodically deleted.

7. Simulation

7.1. Set-Up

To evaluate the effectiveness of the proposed scheme, we simulated the trust based DSR and AODV routing protocols in NS-2 [27]. We implement the random waypoint movement model for the simulation, in which a node starts at a random position, waits for the pause time, and then moves to another random position with a velocity chosen between 0 m/s to the maximum simulation speed. All benign nodes execute the trust model for the duration of the simulation. In order to reduce the computation complexity and limit the number of recorded events, we have used the accuracy and quantity of packets forwarded by a neighbouring node as the measure of its Direct Trust. The category P_P and P_A are employed in combination to protect the protocol against deceptive alteration of vital protocol fields and for identifying selfish node behaviour respectively. The simulation parameters are listed in Table 10.

7.2. Attack Patterns

Malicious nodes simulate the following types of attacks against data and control packets:

Modification Attack. These attacks are carried out by adding, altering or deleting IP addresses from the `ROUTE REQUEST`, `ROUTE REPLY`, `ROUTE ERROR` and Data packets that pass through the malicious nodes.

Black Hole Attack. In this attack the malicious node drops all packets, which it is supposed to forward.

Grey Hole Attack. In the grey hole attack the malicious node selectively dumps data and control packets at random intervals.

Table 10. Simulation Parameters

Simulator	NS-2
Examined Protocol	DSR & AODV
Simulation time	900 seconds
Simulation area	1000 × 1000 m
Number of nodes	50
Transmission range	250 m
Movement model	Random waypoint
Maximum speed	20 m/s
Pause time	10 seconds
Traffic type	CBR (UDP)
Maximum connections	30
Payload size	512 bytes
Packet size	4 pkt/sec
Maximum malicious nodes	20
Types of attacks	Modification, Black and Grey hole
Black/Gray hole attacks $W(P_A)$	0.25
Modification attacks $W(P_P)$	0.75

7.3. METRICS

To evaluate the performance of the proposed scheme, we use the following metrics:

Packet Loss Percentage. It is the fraction of packets that were dumped by malicious nodes without any notification.

Throughput. It is the ratio between the number of packets received by the application layer of destination nodes to the number of packets sent by the application layer of source nodes.

Packet Overhead. This is the ratio between the total number of control packets generated to the total number of data packets received during the simulation time.

Byte Overhead. This is the ratio between the total number of control bytes generated to the total number of data bytes received during the simulation time.

Average Latency. Gives the mean time (in seconds) taken by the packets to reach their respective destinations.

Path Optimality. It is the ratio between the number of hops in the optimal path to the number of hops in the path taken by the packets.

8. Results and Analysis

Figure 2 presents the performance results for the trust based scheme compared with that of the standard DSR protocol, in the presence of a varying number of malicious nodes. The packet loss in the standard DSR protocol is more than 10% higher than that of the trusted DSR protocol. This can be attributed to the fact that the former does not take into account the benevolence levels of the nodes and prefers shorter routes by default. All listed attacks generate no form of notification that informs the other nodes of their malevolent activities. So

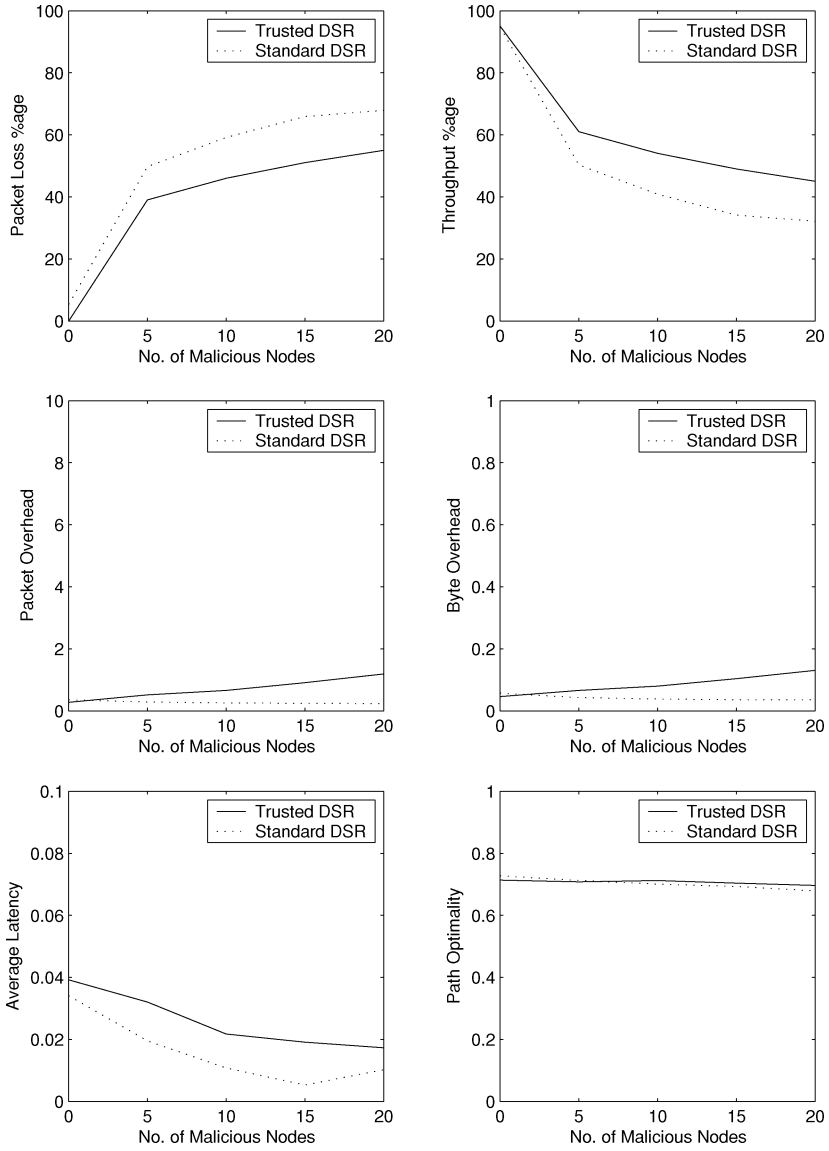

Figure 2. Simulation of the trust-based DSR protocol in the presence of malicious nodes.

the malicious nodes are constantly selected in the routing process which leads to an overall lower throughput of the network.

The trusted DSR protocol, on the other hand, constantly monitors the ongoing behaviour of its neighbouring nodes. It selects or deselects en route nodes based upon their trust levels and thus attempts to avoid any malicious nodes. This deviation from the optimal paths leads to an increase in the packet and byte overhead. An increase in the packet latency and deviation from the optimal paths has also been observed. This can be attributed to the fact that the routes obtained from the cache are not optimal in terms of hops but instead consist of nodes that have been found to be more trustworthy than the others.

The results of the trust based scheme for AODV are shown in Figure 3. The results indicate that the total number of packets lost with the trusted AODV protocol are always lower than that

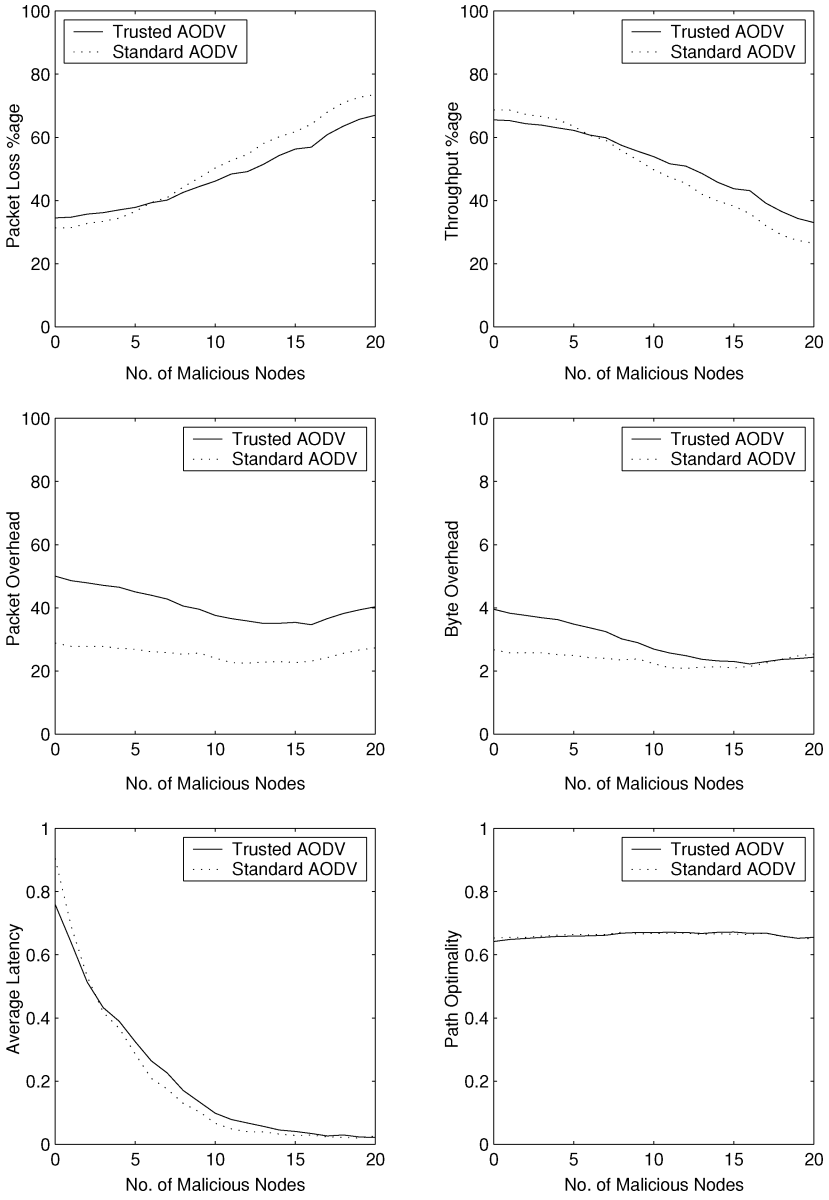

Figure 3. Simulation of the trust-based AODV protocol in the presence of malicious nodes.

of the standard AODV protocol despite the increase in malicious nodes. This is due to the fact that the trust model is based upon respective trust levels and so malicious nodes are bypassed during any subsequent route discoveries. The lower packet loss also helps to maintain a better throughput of the network in the presence of malicious nodes. The trusted AODV protocol has a higher packet overhead due to the deviation from shortest paths. The average latency of the packets also increases with the number of malicious nodes, as the trusted paths are not always the shortest in terms of number of hops. This increase in the path lengths leads to higher latency delays than those occurring in the standard AODV protocol.

The weights that are assigned to different events and situational trust categories, play a critical role in determining the efficacy of the trust model. We accentuate that these weights

are in no way bound to particular values and vary with the type and environment of trust application. These weights may be varied in real time according to the modus operandi and may differ from node to node. However, in prior simulations [28] we have determined the values of these weights by targeting for maximal throughput and minimal packet loss. The results of these simulations indicate that Passive Acknowledgements (P_A) and the Packet Precision (P_P) situational trust categories have maximum impact on the derivation of direct trust in another node.

Any node that can place its interface into promiscuous mode, can passively receive a lot of information about the network. This information can be further used to build trust levels for different nodes. However, this method has certain drawbacks that have been highlighted by Marti et al. [29]. The foremost is the *ambiguous collision* problem in which a node A cannot hear the broadcast from neighbouring node B to node C, due to a local collision at A. In the *receiver collision* problem node A overhears node B broadcast a packet to C but cannot hear the collision which occurs at node C. Similarly, if nodes have varying transmission power ranges the mechanism of passive acknowledgments might not work properly. To avoid these problems the weights assigned to different trust levels in our proposed model need to be selected critically, possibly set to zero, and be dynamically updated to reflect the current scenarios.

It is possible that a node may spoof its IP address in order to steal a data connection, either by advertising a small distance in a `ROUTE REPLY` packet or responding on behalf of the destination. Such an activity, which may not be directly perceivable to the neighbouring nodes, is still detectable if a MAC to IP address binding is maintained at each node. Each time an IP address corresponding to a MAC address is changed, it is considered as a modification attack, and so the spoofing node is graded untrustworthy by its adjacent nodes.

All legitimate and spoofed `ROUTE REPLY` packets received by the source will cause multiple paths to be created for the same destination. It is possible that the initial data connection is released on one of these spoofed paths, however, the dynamism of the topology would help to switch between the spoofed and legitimate routes upon link severing. In case the spoofing acts have been monitored by the malicious node's immediate neighbours, the `ROUTE REPLY` packets being forwarded or originated by the malicious nodes would not be propagated any further.

Similarly, it may be argued that a node may spoof both its MAC and IP addresses. Such an alteration is not directly perceivable to any of its adjacent nodes, which may consider the spoofed node to be just another network node. However, such a scenario is likely to cause IP address collisions, where interactions with the spoofed and legitimate node may break the MAC and IP bindings. Such spoofing attacks can be avoided either by engaging an IP address duplication avoidance scheme [30] or through some other suitable identity theft protection scheme [31].

9. Future Work

In this paper we have presented a framework for trust establishment in an ad-hoc network without the presence of a trusted third party in the network. The proposed trust model is most suitable for such networks as it operates passively and has minimal energy and computation requirements. The model has been used to extract only the direct trust in the network and then applying this to the routing mechanism. However, we intend using an appropriate reputation exchange mechanism like OCEAN [32], CORE [33] or CONFIDANT [34] to reinforce the

trust model through indirect trust values. Currently we are evaluating the trust model for the TORA routing protocol, in order to develop realistic feedback on the model's cost/benefit ratio and scalability to a link reversal protocol. We also intend integrating an effort-based mechanism like HashCash [35] into our trust model to provide active challenge-response based trust values. For analytical evaluation we are investigating the use of Zero-Knowledge and Game Theory concepts in ad-hoc networks for trust establishment. We will also look at further issues that have not been addressed in this paper, including trust dispersal, trust decay over time, trust acquirement through malicious behaviour, malicious colluding nodes, and a security analysis of the proposed model against common known attacks.

10. Conclusion

We have presented here an approach for establishing and managing trust in ad-hoc networks. This is not another type of hard-security cryptographic or certification mechanism [12]. Instead it aims at building confidence measures regarding route trustworthiness that is computed and modified based on effort expended and passively observed. The trust measures described in the paper may also be developed in combination with many existing cryptographic techniques. In an ad-hoc network where doubt and uncertainty are inherent, our trust model creates and maintains trust levels based on an effort/return mechanism. The routes selected using our model may not be cryptographically secure but they do establish relative levels of trustworthiness with them. The trust model is applicable to both pure and managed ad-hoc networks as it provides confidence measures regarding the reliability of routes computed using direct trust mechanisms instead of recommendations from trusted third parties. We believe that our model is most suited to pure ad-hoc networks where there is no trust infrastructure and the trust relationships are less formal, temporary or short-term.

Acknowledgements

This work was supported by the Australian International Postgraduate Research Scholarship and the University of Western Australia Postgraduate Award.

References

1. E.M. Royer and C.K. Toh, "A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks", *IEEE Personal Communications Magazine*, Vol. 6, No. 2, pp. 46–55, 1999.
2. B. Awerbuch, D. Holmer, C. Nita-Rotaru, and H. Rubens, "An On-Demand Secure Routing Protocol Resilient to Byzantine Failures", *Proceedings of the ACM Workshop on Wireless Security (WiSe)*, 2002.
3. A.A. Pirzada and C. McDonald, "A Review of Secure Routing Protocols for Ad-hoc Mobile Wireless Networks", *Proceedings of the 7th International Symposium on DSP for Communication Systems (DSPCS) and 2nd Workshop on the Internet, Telecommunications and Signal Processing (WITSP)*, pp. 118–123, 2003.
4. A.A. Pirzada, A. Datta, and C. McDonald, "Propagating Trust in Ad-hoc Networks for Reliable Routing", *Proceedings of the International Workshop on Wireless Ad-hoc Networks (IWWAN)*, 2004a.
5. R.C. Mayer, J.H. Davis, and F.D. Schoorman, "An Integrative Model of Organizational Trust", *Proceedings of the Sixth annual ACM/IEEE International Conference on Mobile Computing and Networking*, Vol. 20, No. 3, pp. 709–734, 1995.

6. A. Josang, "The Right Type of Trust for Distributed Systems", *Proceedings of the ACM New Security Paradigms Workshop*, pp. 119–131, 1996.
7. D. Denning, "A new Paradigm for Trusted Systems", *Proceedings of the ACM New Security Paradigms Workshop* pp. 36–41, 1993.
8. Y.C. Hu, A. Perrig, and D.B. Johnson, "Ariadne: A Secure On-Demand Routing Protocol for Ad-Hoc Networks", *Proceedings of the Eighth Annual International Conference on Mobile Computing and Networking (MobiCom)*, pp. 12–23, 2002.
9. B. Dahill, B.N. Levine, E. Royer, and C. Shields, "A Secure Routing Protocol for Ad-Hoc Networks", *Proceedings of the International Conference on Network Protocols (ICNP)* pp. 78–87, 2002.
10. A. Perrig, Y.C. Hu, and D.B. Johnson, "Wormhole Protection in Wireless Ad-Hoc Networks", Technical Report TR01-384, Department of Computer Science, Rice University, 2001.
11. Y.-C. Hu, A. Perrig, and D.B. Johnson, "Rushing attacks and defense in wireless Ad-Hoc network routing protocols", *Proceedings of the 2003 ACM workshop on Wireless Security*, pp. 30–40, 2003.
12. A.A. Rahman and S. Hailes, "A Distributed Trust Model", *Proceedings of the ACM New Security Paradigms Workshop*, pp. 48–60, 1997.
13. L. Zhou and Z.J. Haas, "Securing Ad-Hoc Networks", *IEEE Network Magazine*, Vol. 13, No. 6, pp. 24–30, 1999.
14. S. Garfinkel, *PGP: Pretty Good Privacy*, O'Reilly and Associates, Inc., 1995.
15. J.P. Hubaux, L. Buttyan, and S. Capkun, "The Quest for Security in Mobile Ad-Hoc Networks", *Proceedings of the ACM Symposium on Mobile Ad-Hoc Networking and Computing (MobiHoc)*, pp. 146–155, 2001.
16. F. Stajano and R. Anderson, "The Resurrecting Duckling: Security Issues for Ad-Hoc Wireless Networks", *Proceedings of the 7th International Workshop on Security Protocols*, pp. 172–194, 1999.
17. S.P. Marsh, "Formalizing Trust as a Computational Concept", Ph.D. Thesis, University of Stirling, 1994.
18. A.A. Pirzada, A. Datta, and C. McDonald, "Trustworthy Routing with the AODV Protocol", *Proceedings of the International Networking and Communications Conference (INCC)*, pp. 19–24, 2004.
19. D.B. Johnson, D.A. Maltz, and Y. Hu, "The Dynamic Source Routing Protocol for Mobile Ad-Hoc Networks (DSR)", *IETF MANET, Internet Draft (Work in Progress)*, 2003.
20. C. Perkins, E. Belding-Royer, and S. Das, "Ad-Hoc On-Demand Distance Vector (AODV) Routing", *IETF RFC 3591*, 2003.
21. C.E. Perkins and P. Bhagwat, "Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers", *Proceedings of the SIGCOMM Conference on Communications, Architectures, Protocols and Applications*, pp. 234–244, 1994.
22. V. Park and S. Corson, "Temporally Ordered Routing Algorithm (TORA) Version 1 Functional Specification", *IETF MANET, Internet Draft (work in progress)*, 2001.
23. E. Gafni and D. Bertsekas, "Distributed Algorithms for Generating Loop-Free Routes in Networks with Frequently Changing Topology", *IEEE Transactions on Communications*, Vol. 29, No. 1, pp. 11–18, 1981.
24. M.S. Corson and A. Ephremides: "Lightweight Mobile Routing Protocol (LMR), A Distributed Routing Algorithm for Mobile Wireless Networks", *Wireless Networks*, 1995.
25. S. Corson, S. Papademetriou, P. Papadopoulos, V. Park, and A. Qayyum, "Internet MANET Encapsulation Protocol (IMEP) specification", *IETF MANET, Internet Draft (work in progress)*, 1999.
26. S. Jacobs and M.S. Corson, "MANET authentication architecture", *IETF MANET, Internet Draft (Work in Progress)*, 1999.
27. NS, "The Network Simulator", *http://www.isi.edu/nsnam/ns/*, 1989.
28. A.A. Pirzada and C. McDonald, "Reliable Routing in Ad-hoc Networks Using Direct Trust Mechanisms (to appear)", In: D.Z. Du and G. Xue (eds.), *Advances in Wireless Networks and Mobile Computing*, Springer, 2005.
29. S. Marti, T. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile Ad-Hoc Networks", *Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking (MobiCom)*, pp. 255–265, 2000.
30. A.P. Tayal and L.M. Patnaik, "An Address Assignment for the Automatic Configuration of Mobile Ad-Hoc Networks", *Personal and Ubiquitous Computing*, Vol. 8, No. 1, pp. 47–54, 2004.
31. M. Cremonini, E. Damiani, S.D.C.D. Vimercati, and P. Samarati, "Security, Privacy and Trust in Mobile Systems and Applications", In: M. Pagani (ed.): *Mobile and Wireless Systems beyond 3G: Managing New Business Opportunities*. IRM Press, 2005.

32. S. Bansal and M. Baker, "Observation-based Cooperation Enforcement in Ad-Hoc Networks", Technical report, Stanford University, 2003.
33. P. Michiardi and R. Molva, "CORE: A Collaborative Reputation Mechanism to Enforce Node Cooperation in Mobile Ad-Hoc Networks", *Proceedings of the IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security*, Vol. 228, pp. 107–121, 2002.
34. S. Buchegger and J. Boudec, "Performance Analysis of the CONFIDANT Protocol: Cooperation of Nodes – Fairness In Distributed Ad-hoc NeTworks", *Proceedings of the IEEE/ACM Workshop on Mobile Ad-Hoc Networking and Computing (MobiHOC)* pp. 226–236, 2002.
35. A. Back: 2002, "A Denial of Service Counter-Measure", http://www.hashcash.org/.

Asad Amir Pirzada is presently doing his Ph.D. on trust and security issues in ad-hoc wireless networks at The University of Western Australia. His current research interests include wireless communications, networking, cryptography, real-time programming and data acquisition systems. He holds a BE Avionics from NED University Pakistan, a MSc Computer Science from Preston University USA and a MS Information Security from the National University of Sciences and Technology Pakistan.

Chris McDonald holds a B.Sc(Hons) and Ph.D. in Computer Science from The University of Western Australia, and currently holds the appointments of senior lecturer in the School of Computer Science & Software Engineering at UWA and adjunct professor in the Department of Computer Science at Dartmouth College, New Hampshire.

Chris has recently taught in the areas of computer networking, operating systems, computer & network security, computer architecture, distributed systems programming and, together with these areas, his research interests include network simulation, ad-hoc & mobile networking, programming language implementation, open-source software.