Establishing Trust In Pure Ad-hoc Networks

Asad Amir Pirzada and Chris McDonald

School of Computer Science & Software Engineering,
The University of Western Australia
35 Stirling Highway, Crawley, W.A. 6009, Australia.

email: {pirzada,chris}@csse.uwa.edu.au

Abstract

An ad-hoc network of wireless nodes is a temporarily formed network, created, operated and managed by the nodes themselves. It is also often termed an infrastructure-less, self-organized, or spontaneous network. Nodes assist each other by passing data and control packets from one node to another, often beyond the wireless range of the original sender. The execution and survival of an ad-hoc network is solely dependent upon the cooperative and trusting nature of its nodes.

However, this naive dependency on intermediate nodes makes the ad-hoc network vulnerable to passive and active attacks by malicious nodes. A number of protocols have been developed to secure ad-hoc networks using cryptographic schemes, but all rely on the presence of an omnipresent, and often omniscient, trust authority. As this paper describes, dependence on a central trust authority is an impractical requirement for ad-hoc networks. We present a model for trust-based communication in ad-hoc networks that also demonstrates that a central trust authority is a superfluous requirement. The model introduces the notion of belief and provides a dynamic measure of reliability and trustworthiness in an ad hoc network. [1]

Keywords: Trust, Security, Ad-hoc, Networks, Protocols

1 Introduction

Ad-hoc networks are primarily meant for use in military, emergency and relief scenarios where, in spite of nonexistent infrastructure, a network can be established. Nodes help each other in conveying information to and fro and thereby creating a virtual set of connections between each other. Routing protocols play a vital role in the creation and maintenance of these connections. In contrast to wired networks, each node in an ad-hoc networks acts like a router. As these routers are usually on the move, standard intra-router protocols cannot be immediately adapted to ad-hoc networks. Many different types of routing protocols have been developed for ad-hoc networks and have been classified into two categories by Royer and Toh (1999) as Reactive and Proactive. In reactive routing protocols, in order to preserve precious node battery, routes are only discovered when required, while in proactive routing protocols routes are established before usage and hence avoid the latency delays incurred while discovering new routes in the reactive routing protocols. As mentioned earlier, an ad-hoc network can only exist and operate if its nodes demonstrate a cooperative behaviour. However, this is always not true, and there may always exist malicious nodes that aim to eavesdrop on, corrupt, or disrupt the network traffic. As routing protocols play a major role in the communication set-up, it is vital that the protocols have a consistent and accurate performance. A number of such protocols were thereby developed to secure the routing process. A comparison of these protocols was carried out by Pirzada and McDonald (2003) and it revealed that all the secure routing protocols were dependent on a central trust authority for implementing traditional cryptographic algorithms. All the protocols just gave the assurance of either the presence of 100% security or its absence. None of these had an intermediate level of security protection. As authentication is one of the initial requirements of a secure channel, the nodes were required to be in possession of pre-shared keys or digital certificates. This requirement of a central trust authority and pre-configuration is neither practical nor feasible in an ad-hoc network. To distinguish this environment the term “managed ad-hoc networks” was introduced in which the nodes could be initially configured before the network was established. This is in contrast to the actual aim of ad-hoc networks, which targets to establish an improvised network. We call such a network a “Pure ad-hoc network”, which has no assumed infrastructure and is created on the fly. We also introduce the notion of trust in ad-hoc networks rather than inclusion of regular cryptographic schemes. By computing trust levels from the inherent knowledge present in the network, the trustworthiness of routes can be computed. The routes computed through this mechanism may not be secure but certainly have an accurate measure of reliability in them.

This paper is focused on introducing a trust model suitable for application to ad-hoc networks. In Section 2 we discuss trust and security issues for ad-hoc networks. In Section 3 we discuss specific attacks against ad-hoc network routing. In Section 4 we describe some relevant previous work. In Section 5 we describe our proposed trust model in detail and in Section 6 we present its application to the Dynamic Source Routing (DSR) protocol. An analysis of the proposed model is presented in Section 7. The rest of this paper consists of an outline of future work in Section 8 and concluding remarks in Section 9.

2 Trust and Security Issues

Trust and security are two tightly interdependent 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 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 dealt 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 have usually short life spans. 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 presence of 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, Davis and Schoorman (1995) 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”. Jøsang (1996) 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 in-direct 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/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. 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 (Denning 1993).

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 (Hu, Perrig and Johnson 2002), *passive* and *active*. In passive attacks the attacker does not disturb the routing protocol. It only eavesdrops on the routing traffic and endeavours 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 (Dahill, Levine, Royer and Shields 2002).

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 take a longer route to the destination increasing communication delays. An example is for an attacker to send 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 it. 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*) 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 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* 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 vision 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 (Rahman and Hailes 1997) 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 decentralized 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 (Zhou and Haas 1999) 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 (Garfinkel 1995) 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 decentralized "web of trust" approach. Each individual signs each other's keys that help 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 (Hubaux, Buttyan and Capkun 2001) 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 (Stajano and Anderson 1999) 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 dumb sensor nodes where pre-configuration has to be avoided. As this model uses 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 (1994) 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 in turn increases or decreases with time. In our model we make use of 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. 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[2] protocol suite is represented in Figure 5.1.

OSI	PROPOSED	TCP/IP
Application	Computation & Quantification	Application
Presentation		
Session		
Transport	Derivation	Transport
Network		Internet
Data link		Host to Network
Physical		

Figure 5.1: Structure of Trust Agent

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 forwarded
8. Data received

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 has a continuous trend. Secure routing protocols represent trust levels by either the presence of security or its absence. 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 the 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 (utility/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 $T_x(y)$ and is given by the following equation:

$$T_x(y) = \sum_{i=1}^{n} [\ W_x(i) \text{ x } T_x(i)\ ]$$

where $W_x(i)$ is the weight of the i^{th} trust category to x and $T_x(i)$ is the situational trust of x in the i^{th} trust category. The total number of trust categories n is dependent on the protocol and scenario to which the trust model is being applied.

[2] In the TCP/IP suite, both Computation and Quantification functions reside in the single application layer

6 Extension to DSR

6.1 DSR Protocol

The Dynamic Source Routing (DSR) protocol (Johnson, Maltz and Hu 2003) 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 ROUTE CACHE, 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 replies 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 ROUTE CACHE. 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.

We have augmented the DSR Protocol with our proposed trust model in order to find trustworthy routes. As DSR uses source routing, each routing or data packet received contains a complete list of nodes through which it has passed.

6.2 Trust Derivation

In DSR, we use the following inherent features to build up trust categories for our model:

6.2.1 Acknowledgments

A node can get information about the successful transmission of any packet that it sent, through the following three methods:

6.2.1.1 Link-Layer Acknowledgements

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

6.2.1.2 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 node.

6.2.1.3 Network Layer Acknowledgements

This method permits the sender to explicitly request a network layer acknowledgement from the next hop using the DSR options header.

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 6.21. 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.

Node Acknowledgement	Route Request (R_q)		Route Reply (R_p)		Route Error (R_e)		Data (D)	
	Success R_{qs}	Fail R_{qf}	Success R_{ps}	Fail R_{pf}	Success R_{es}	Fail R_{ef}	Success D_s	Fail D_f

Table 6.21: Trust table based on Passive Acknowledgments

6.2.2 Packet Precision

The accuracy of received data and routing packets offers a measure to compute trust levels. 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 6.22. Counters are maintained for every received packet that are incremented based upon the accuracy or inaccuracy of the packet.

Node Packet Precision	Route Request (R_q)		Route Reply (R_p)		Route Error (R_e)		Data (D)	
	Success R_{qs}	Fail R_{qf}	Success R_{ps}	Fail R_{pf}	Success R_{es}	Fail R_{ef}	Success D_s	Fail D_f

Table 6.22: Trust table based on Packet Precision

6.2.3 Gratuitous Route Replies

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 Gratuitous route replies can be considered as a trust category as they provide the following information about the sender of the Gratuitous route reply:

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 6.23. All Gratuitous route reply packets that are found to be correct or incorrect are recorded using appropriate counters.

Node	Gratuitous Route Replies (G)	
	Success G_s	Fail G_f

Table 6.23: Trust table based on Gratuitous Route Replies

6.2.4 Blacklists

DSR maintains 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 6.24.

Node	Present in Blacklist (B)

Table 6.24: Trust table based on Blacklists

6.2.5 Salvaging

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 be considered as a trust category. All salvaged route errors found to be correct or incorrect are recorded using counters, as shown in Table 6.25.

Node	Salvage Route Error (S)	
	Success S_s	Fail S_f

Table 6.25 : Trust table based on Salvaging

6.3 Trust Quantification

The events recorded in the tables during the trust derivation process are quantised and assigned weights so as to compute the situational trust values for different nodes.

6.3.1 Trust Category P_A

The trust category derived using Passive Acknowledgements is denoted by P_A. The events recorded in Table 6.21 are quantised as per the following equations to provide trust levels:

Where $$R_q = \frac{R_{qs} - R_{qf}}{R_{qs} + R_{qf}} \text{ for } R_{qs}+R_{qf} \neq 0 \text{ else } R_q = 0$$

$$R_p = \frac{R_{ps} - R_{pf}}{R_{ps} + R_{pf}} \text{ for } R_{ps}+R_{pf} \neq 0 \text{ else } R_p = 0$$

$$R_e = \frac{R_{es} - R_{ef}}{R_{es} + R_{ef}} \text{ for } R_{es}+R_{ef} \neq 0 \text{ else } R_e = 0$$

$$D = \frac{D_s - D_f}{D_s + D_f} \text{ for } D_s + D_f \neq 0 \text{ else } D = 0$$

By normalizing the values of R_p, R_q, R_e 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 than assigned weights in a static or dynamic manner depending on their utility and importance. The situational trust $T_n(P_A)$ in node *n* for trust category P_A is computed using the following equation:

$$T_n(P_A) = W(R_q) \text{ x } R_q + W(R_p) \text{ x } R_p + W(R_e) \text{ x } R_e + W(D) \text{ x } D$$

Where W is the weight assigned to the event that took place with node *n*.

6.3.2 Trust Category P_P

We derive trust category P_P from the events recorded in Table 6.22 based upon the Packet Precision. The following equations are used to compute the trust levels:

Where $$R_q = \frac{R_{qs} - R_{qf}}{R_{qs} + R_{qf}} \text{ for } R_{qs}+R_{qf} \neq 0 \text{ else } R_q = 0$$

$$R_p = \frac{R_{ps} - R_{pf}}{R_{ps} + R_{pf}} \text{ for } R_{ps}+R_{pf} \neq 0 \text{ else } R_p = 0$$

$$R_e = \frac{R_{es} - R_{ef}}{R_{es} + R_{ef}} \text{ for } R_{es}+R_{ef} \neq 0 \text{ else } R_e = 0$$

$$D = \frac{D_s - D_f}{D_s + D_f} \text{ for } D_s + D_f \neq 0 \text{ else } D = 0$$

All these events are than assigned weights in a similar manner and the situational trust in category P_P for node *n* is computed using the following equation.

$$T_n(P_P) = W(R_q) \text{ x } R_q + W(R_p) \text{ x } R_p + W(R_e) \text{ x } R_e + W(D) \text{ x } D$$

6.3.3 Trust Category G_R

The trust category based upon Gratuitous Route Replies is derived using Table 6.23 and is denoted as G_R. The trust levels are quantised using the following equation:

$$G = \frac{G_s - G_f}{G_s + G_f} \text{ for } G_s+G_f \neq 0 \text{ else } G = 0$$

The situational trust in category G_R for a node *n* is computed using the equation.

$$T_n(G_R) = W(G) \text{ x } G$$

6.3.4 Trust Category B_L

Table 6.24 is used to derive trust category B_L, which is based on Blacklists. The value of B is Boolean reflecting the presence or absence of a node in the trust table. The situational trust for category B_L in node *n* is computed using the following equation:

$$T_n(B_L) = W(B) \text{ x } B$$

6.3.5 Trust Category S_G

The trust category derived using the Salvaging information in Table 6.25 is denoted as S_G. The trust levels are quantised using the following equation:

$$S = \frac{S_s - S_f}{S_s + S_f} \text{ for } S_s+S_f \neq 0 \text{ else } S = 0$$

The situational trust in category S_G for a node *n* is computed using the equation.

$$T_n(S_G) = W(S) \text{ x } S$$

6.4 Trust Computation

The situational trust values from all trust categories (P_A, P_P, G_R, B_L, S_G) are then combined according to assigned weights, to determine an aggregate trust level for a particular node. Trust T in node *y* by node *x* is represented as $T_x(y)$ and given by the following equation:

$$T_x(y)=W_x(P_A) \text{ x } T_x(P_A) + W_x(P_P) \text{ x } T_x(P_P) + W_x(G_R) \text{ x } T_x(G_R) + W_x(B_L) \text{ x } T_x(B_L) + W_x(S_G) \text{ x } T_x(S_G)$$

where W_x represents the weight assigned to a trust category by *x* and T_x is the situational trust of *x* in that trust category. The aggregate trust table is shown in Table 6.4.

Node	Passive Ack	Packet Prec	Grat Route Replies	Black Lists	Salvage Route Replies	Agg Trust Level
y	$T_x(P_A)$	$T_x(P_P)$	$T_x(G_R)$	$T_x(B_L)$	$T_x(S_G)$	$T_x(y)$

Table 6.4 : Aggregate Trust Table

The aggregate and situational trust values are then maintained and updated for each node based upon the frequency of events and severity of the situation. In DSR, by analysing the routes using a link cache organization scheme like Link-MaxLife (Johnson, Maltz and Hu 2003), a node can create its private view of the current network topology. If the computed trust levels are associated as weights to these links, the sending node can use a shortest-path algorithm to find the most trustworthy path to the destination. The routes thus found using this method may not be safe in terms of security but they all carry along an associated level of trustworthiness with them.

7 Analysis

The precise amount of trust established using the proposed model is currently being investigated and simulated, but inherently the model is simple, flexible and pragmatic for use in pure ad-hoc networks. 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, Giuli, Lai and Baker (2000). 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. In DSR each node maintains a table of next-hop destination nodes. If latency values are also maintained for these nodes then it provides us with a measure to compute if the neighbour node is moving toward or away from us. This measure could also be used to differentiate between malicious and benevolent behaviour of nodes.

8 Future Work

In this paper we have presented a framework for trust establishment in an ad-hoc network without the existence of a central trust authority. The proposed trust model is most suitable for such networks as it operate passively and has minimal energy and computation requirements. Currently we are implementing this model in the Network Simulator (NS-2 1989) to develop realistic feedback on the model's scalability, cost/benefit ratio and overhead. However, we intend integrating an effort-based mechanism like HashCash (Back 2002) into our trust model, to also 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 plan extending our model to other ad-hoc network routing protocols like TORA, AODV and DSDV. We will also look at further issues that have not been addressed in this paper, including trust decay over time, trust acquirement through malicious behaviour, malicious colluding nodes, and a security analysis of the proposed model against attacks.

9 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 (Rahman and Hailes 1997). Instead it aims at building confidence measures regarding route trustworthiness in nodes that are dynamically computed and modified based on effort expended and passively observed by other nodes. 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 will be most suited to pure ad-hoc networks where there is no trust infrastructure and the trust relationships are less formal, temporary or short-term.

10 References

Back, A. (2002): Hashcash: A Denial of Service Counter-Measure, http://citeseer.nj.nec.com/back02hashcash.html Accessed 25 Aug 2003.

Dahill, B., Levine, B. N., Royer, E. and Shields, C. (2002): A secure Routing Protocol for Ad Hoc Networks, *Proc. of International Conference on Network Protocols (ICNP)*, 78- 87.

Denning, D. (1993): A new paradigm for trusted systems. *Proc. ACM New Security Paradigms Workshop*, 36-41.

Garfinkel, S. (1995): *PGP : Pretty Good Privacy*, O'Reilly & Associates, Inc.

Hu, Y-C, Perrig, A. and Johnson, D. B (2002): Ariadne : A secure On-Demand Routing Protocol for Ad Hoc Networks, *Proc of the eighth Annual International Conference on Mobile Computing and Networking*, 12-23.

Hubaux, J. P., Buttyan, L. and Capkun, S. (2001): The Quest for Security in Mobile Ad Hoc Networks. *Proc. of ACM Symposium on Mobile Ad Hoc Networking and Computing*, 146-155.

Johnson, D. B., Maltz, D. A. and Hu, Y. (2003): The Dynamic Source Routing Protocol for Mobile Ad hoc Networks (DSR), IETF MANET, Internet Draft.

Jøsang, A. (1996): The right type of trust for distributed systems. *Proc. of the ACM New Security Paradigms Workshop*, 119-131.

Marsh, S. P. (1994): Formalizing Trust as a Computational Concept. Ph.D. Thesis. Department of Mathematics and Computer Science, University of Stirling.

Marti, S., Giuli, T., Lai, k. and Baker, M. (2000): Mitigating routing misbehavior in mobile ad hoc networks, *Proc. of the Sixth annual ACM/IEEE International Conference on Mobile Computing and Networking*, 255-265.

Mayer, R. C., J. H. Davis and F. D. Schoorman (1995): An Integrative Model of Organizational Trust, *Academy of Management Executive*, 20(3):709-73.

NS-2 (1989): The Network Simulator, http://www.isi.edu/nsnam/ns/. Accessed 25 Aug 2003.

Pirzada, A. A. and McDonald, C. (2003): A Review of Secure Routing Protocols for Ad hoc Mobile Wireless Networks, (*To be published in) Proc. of 2nd Workshop on the Internet, Telecommunications and Signal Processing (DSPCS'03 & WITSP'03).*

Rahman, A. A. and Hailes, S. (1997): A Distributed Trust Model, *Proc. of the ACM New Security Paradigms Workshop*, 48-60.

Royer, E. M. and Toh, C. -K. (1999): A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks. *IEEE Personal Communications Magazine*, 6(2):46-55.

Stajano, F. and Anderson, R. (1999): The Resurrecting Duckling: Security Issues for Ad hoc Wireless Networks, *Proc. of the 7th International Workshop on Security Protocols*, 1796:172-194.

Zhou, L. and Haas, Z. J. (1999): Securing Ad Hoc Networks, *IEEE Network Magazine*, 13(6).