Trusted Route Discovery with TORA Protocol

Asad Amir Pirzada, Chris McDonald
School of Computer Science & Software Engineering,
The University of Western Australia,
35 Stirling Highway, Crawley, W.A. 6009, Australia.
{pirzada,chris}@csse.uwa.edu.au

Abstract

An ad-hoc network is formed when a group of wireless nodes pledge to help each other in passing packets based upon a pre-agreed protocol. Temporally Ordered Routing Algorithm (TORA) is one of the commonly used protocols for ad-hoc wireless networks. Route discovery in TORA is done in a cooperative manner with intermediate nodes contributing to the route generation. Precision in route build-up demands that all network nodes portray persistent benevolent behavior. This is however, not always possible to achieve and so a number of malicious nodes participate in the TORA route discovery process only to sabotage the network by violating the protocol. In this paper we present a novel mechanism for establishing trust in ad-hoc networks that execute the Temporally Ordered Routing Algorithm (TORA) Protocol. The routes discovered using our model are not cryptographically secure but each one of them carries a confidence measure regarding its suitability in the current scenario.

1. Introduction

An ad-hoc network is built, operated and controlled by its participating wireless nodes. As each of these nodes requires assistance from other nodes for packet transportation, it pledges to do the same in return for them. These short-term relationships form the basic building block of an ad-hoc network and are mandatory for its successful operation. As these relationships are based on reciprocal benevolent behaviour they can be created and destroyed on the fly. Such a relationship model offers flexibility and diversity, a vital requirement for improvised networks. Routing protocols give a pragmatic representation of these relationships and are hence a critical component of the ad-hoc network. Two types of routing protocols have been developed for this purpose: Proactive and Reactive [1]. Proactive routing protocols also known as Table-driven protocols maintain consistent view of the network at all times and provide instantaneous routes thus avoiding latency delays. Reactive routing protocols, which are also known as On-demand protocols, only find routes when required in order to save precious battery power. Both types of routing protocols are critically dependent on the intra-node relationships for their accurate execution. However, as these relationships are based on naive trust models they can be easily moulded for malevolent purposes by certain nodes. These nodes can be categorized into malicious and compromised. Malicious nodes are external nodes that try to disrupt, divert or disturb the network traffic while compromised nodes are internal nodes that have been captured and configured to perform a similar activity.

To ensure accurate execution of the routing protocols in presence of malicious or compromised nodes, a number of secure routing protocols were developed. A comparison of these protocols [2] revealed that these protocols engaged a variety of cryptographic mechanisms for protecting the operation of routing protocols. These cryptographic mechanisms also required the services of a centralized or distributed trusted third party for authentication and key exchange. To justify the creation of such an entity in an improvised environment the term 'managed ad-hoc network' was introduced where such an entity could be established or the nodes could be pre-configured with encryption keys before joining a network. Employing cryptographic mechanisms is another way of stimulating trust into the network, where the trust is being placed in the trusted third party and the encryption algorithm. However, as the

Proceedings of the Second Annual Conference on Communication Networks and Services Research (CNSR'04)
0-7695-2096-0/04 $20.00 © 2004 **IEEE**

nodes in an ad-hoc network are mobile in an open environment, they are more vulnerable to capture due to limited physical protection. Consequently, the compromise of a key repository or an authentication server [3] could jeopardize the complete security infrastructure.

In this paper we present a unique way of establishing trust in an ad-hoc network without the use of cryptography. We accentuate that trust can be established between unfamiliar nodes based upon a give-and-take mechanism, where the necessity of getting a thing done is vital than its confidentiality. 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 TORA routing 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

According to Jøsang [4], trust and security represent two sides of the same thing. Both these terms are so highly interconnected that they cannot be evaluated independently. To secure any information system, cryptographic mechanisms are generally employed. These mechanisms necessitate trusted key exchange prior to their application. Similarly, for trusted key exchange to take place, a secure channel is required for key transport. Due to this interdependence between trust and security, we need to consider both these issues when defining a secure system. In wired networks, trust is usually realized using indirect trust mechanisms that comprise of trusted certification agencies and authentication servers. However, these mechanisms still necessitate some out-of-band mechanism for initial authentication and are frequently dealt with physical or location-based verification schemes.

In comparison, establishing trust in ad-hoc wireless networks is an extremely challenging task. These networks are primarily based upon naive "trust-your-neighbour" relationships with relatively short life spans. This simplistic approach, although workable, makes the trust relationships prone to attacks by malicious nodes present in the network. In addition, the lack of permanent trust infrastructure, restricted resources, ephemeral connectivity and availability, shared wireless medium and physical vulnerability, make trust establishment virtually impossible. A number of assumptions have been made to overcome these problems including pre-configuration of nodes with secret keys and the presence of an omnipresent central trust authority. However, these assumptions are against the very nature of ad-hoc networks that are supposed to be makeshift and impetuous. To differentiate, we classify the ones that are based on assumptions as "managed ad-hoc networks" and those without it as "pure ad-hoc networks".

Mayer et al. [5] defined trust 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 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 one's belief that the trusted entity will not act maliciously in a particular situation. Trust may be achieved directly based on previous similar experiences with the same party, or indirectly based on recommendations from other trusted parties. In addition, trust is also time dependent, it grows and decays over a time.

A pure ad-hoc network strongly resembles the human behaviour model in which people are able to establish trust relationships based upon experiences over time. According to Denning, 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 [6]. We believe that as in real life, trust levels are determined by the particular actions that the trusted party can carry out for the trustee. In the same way trust levels in ad-hoc networks can be computed based upon the effort that one node is prepared to expend for another node. This effort can be in terms of packets forwarded or dropped, battery consumption, or any other such parameter that helps to establish a reciprocal trust level. A trust model that is based on experience alone may not be sheltered from attacks in an ad-hoc network but it can definitely discover routes with an explicit confidence level.

3. Attacks on Wireless Networks

Attacks against ad-hoc networks can be classified into two types: passive and active [7]. In passive attacks the attacker only snoops upon the network traffic and

attempts to extract useful information from it in an inert manner. Passive attacks are imperceptible in the wireless environment and hence also extremely difficult to evade. Whereas, in active attacks, malicious nodes attempt to disrupt the correct functioning of the underlying network protocol by modifying or fabricating packets or by impersonating legitimate nodes [8].

3.1. Attacks Using Modification

Modification attacks are carried out against the integrity of the packets. During these attacks, malicious nodes while forwarding the packets modify the contents so as to achieve a desired outcome. 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 carry out denial of service (DoS) attacks.

3.2. Attacks Using Fabrication

Fabrication attacks are carried out by generating false routing messages. These attacks are difficult to identify as they are received as legitimate routing packets. By generating fallacious control packets, malicious nodes are able to create a topology that is most favourable for accomplishing their desired goals.

3.3. Attacks Using Impersonation

Malicious nodes can instigate several attacks in a network by masquerading as another node (spoofing). Spoofing takes place when a malicious node feigns its identity by varying its MAC or IP address with the intention of deceiving benevolent nodes. Spoofing attacks are frequently used to create routing loops in which packets traverse in localized regions. In worst-case scenarios, routing loops can even cause partitioning of the network.

4. Previous Work

4.1. Distributed Trust Model

The Distributed Trust Model [9] employs a dedicated protocol for exchanging, withdrawing and reviving recommendations. Each entity maintains its own trust database by using this recommendation protocol. This in turn guarantees that the computed trust is neither absolute nor transitive. A decentralized approach to trust management is exercised. The model uses trust categories and trust values to find different levels of trust. The integral trust values in the model vary from –1 to 4 representing distinct levels of trust from absolute distrust (-1) to absolute trust (4). All entities execute the recommendation protocol either as a recommender or a requestor. The trust levels are computed by means of the recommended trust value of the target and its recommenders. The model also supports multiple recommendations for the same target and averages the same to generate a single recommendation value. The Distributed Trust Model is most appropriate for scenarios with casual and short-term relationships and cannot be directly adapted for ad-hoc networks. Furthermore, it does not provide measures for safeguarding against false or malicious recommendations about other entities.

4.2. Distributed Public-Key Model

The Distributed Public-Key Model [10] employs threshold cryptography to distribute the private key of the Certification Authority over a number of servers. A (n, t+1) scheme permits any t+1 servers out of n servers to join their partial keys to generate the entire secret key. In the same way, it necessitates that a minimum t+1 servers must be overridden to get hold of the secret key. The scheme is fairly robust but due to a variety of reasons its application to ad-hoc networks is severely restricted. First and foremost, it necessitates the pre-configuration of servers and nodes, secondly the t+1 servers may not always be available to nodes seeking authentication and finally asymmetric cryptographic operations have been found to drain valuable node batteries.

4.3. PGP Model

In the Pretty Good Privacy (PGP) Model [11] all users function as autonomous certification authorities. Each user has the ability to sign and verify keys of other users. PGP adopts a decentralized “web of trust” approach rather than the traditional central trust authority architecture. Every user signs the keys of other user’s, which helps to build a set of interconnecting trust links. PGP associates various degrees of confidence levels from “undefined” to “complete trust” to the trustworthiness of public-key certificates. It also makes use of four levels of trustworthiness of introducers from “don’t know” to “full trust”. These confidence and trustworthiness

levels are used to compute the trust in other users. PGP is feasible for environments where a central key server can maintain a database of keys. However, in ad-hoc networks, formation of a central key server creates a single point of failure and also requires uninterrupted access by the nodes. Another option that has been proposed [12] is to store a subset of the public keys of other users using a subset of the trust graph. These graphs are then merged with graphs of other users in order to determine trusted routes. This scheme has high computation and memory requirements and is considered restrictive for ad-hoc networks.

4.4. Resurrecting Duckling Model

The Resurrecting Duckling Model [13] employs a hierarchical graph of master-slave relationships. The slave (duckling) accepts the first node as its master (mother duck), which transmits it a secret key via secure channel. The duckling pledges to obey the mother duck and so obtains all commands and access control lists from it. The duckling further grows to be a mother to any device with which it can share a secret key through secure means. This master-slave bond can only be severed either by a master, a timeout or an event after which the slave looks for a new master. This model is generally appropriate for security in dumb sensor nodes where large-scale pre-configuration has to be avoided. As this model uses a top-down security model it is not suitable for ad-hoc networks that have a flat trust hierarchy.

5. The Trust Model

Our trust model [14] is influenced by Marsh's trust model [15] and has been optimized for application to pure ad-hoc networks. Marsh's model works out the situational trust in agents using the general trust in the trustor. Each agent also assigns importance and utility to the situation in which it finds itself. General trust is defined as the trust that an entity places in another entity based upon all prior transactions in all situations. Importance and utility are considered similar to knowledge so that an agent can weigh up the costs and benefits that a particular situation holds at a particular time. In our model, we use a single variable called weight to represent the utility and importance of a situation. This weight increases or decreases with time and is set according to the hostility of the environment. The trust model is executed by means of agents that reside on network nodes. Each agent operates autonomously and upholds its own point of view as regards to the trust hierarchy. Each agent accumulates data from events that are being experienced by a node in the current environment. These events are filtered and assigned weights so as to compute the trust in other nodes. Every trust agent essentially performs three functions: Trust Derivation, Quantification, and Computation. The estimated division of these functions with respect to the OSI reference model and the TCP/IP protocol suite is shown in Figure 1.

<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

5.1. Trust Derivation

Trust derivation is carried out in passive mode i.e. without entailing the use of special interrogation packets. Essential information regarding other nodes can be gathered by examining the local network traffic. Taps are inserted at different points in the protocol stack so as to analyse events that are currently in progress. Likely events that can be recorded in passive mode are the measure and precision of:

- Frames received
- Data packets forwarded
- Control packets forwarded
- Data packets received
- Control packets received
- Streams established
- Data forwarded
- Data received

The information obtained from these events is split into one or more trust categories. These categories specify the explicit feature of trust that is pertinent to a particular relationship. This categorization of events facilitates the computation of situational trust in other nodes.

5.2. Trust Quantification

Trust generally portrays a continuous trend and hence discrete representation is insufficient to clearly represent trust. Secure routing protocols represent

trust in a binary form by either the presence of security or its absence. As discussed earlier, PGP symbolizes trust using four values ranging from unknown to fully trusted. Discrete values, although straightforward to represent and categorize trust, are not suitable for application to ad-hoc networks due to dynamism of the topology. The period of interaction with other nodes is also succinct, which necessitates that trust be represented as an incessant range in order to differentiate between nodes with comparable trust levels. We signify trust from –1 to +1, representing an unremitting range from complete distrust to absolute trust.

5.3. Trust Computation

During trust computation, weights are assigned to the previously monitored and quantified events. All nodes dynamically allocate these weights depending upon their own criteria and situation. These weights are represented in a continuous range from 0 to +1, where 0 signifies unimportant and +1 signifies most important. The aggregate trust level is then computed by combining the situational trusts according to their importance and utility. We define the aggregate trust T in node y, by node x as $T_x(y)$, which is given by the following equation:

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

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 number of trust categories n is reliant upon the protocol and situation in which the trust model has been employed.

6. Extension to TORA

6.1. TORA Protocol

Temporally Ordered Routing Algorithm by Park et al. [16] 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 optimized hybrid of the Gafni Bertsekas [17] and Lightweight Mobile Routing [18] protocols. It guarantees loop freedom, multiple routes and minimal communication overhead even in highly dynamic environments. The protocol targets at minimizing routing discovery overhead and in doing so 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 delta 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 four routing functions: Route Creation, Route Maintenance, Route Erasure and Route Optimization. To accomplish these functions it uses four distinct control packets: Query (QRY), Update (UPD), Clear (CLR) and Optimization (OPT). 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 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 (GENERATE) 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 GENERATE) 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 as shown in Figure 2.

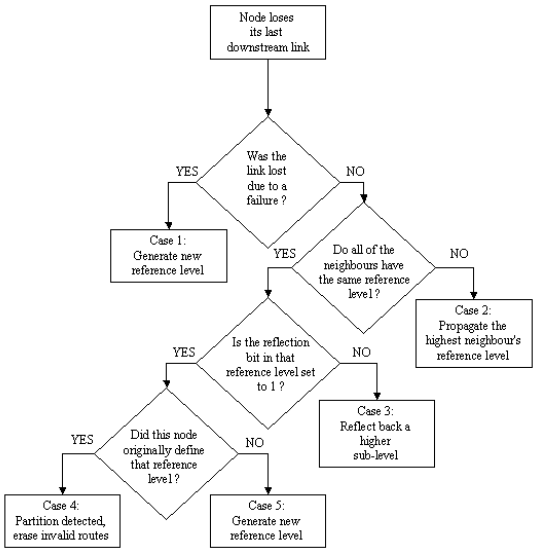

Figure 2. TORA Route Maintenance Tree

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 DAG's to the destination so in any network with an average n number of nodes each with n/2 downstream neighbours, a node could still effectively communicate with the destination node upon link failure of (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 Internet MANET Encapsulation Protocol (IMEP). IMEP [19] 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).

6.2. Trust Derivation

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

6.2.1. Acknowledgments. Any node can obtain information about the successful transmission of a packet through the following three methods:

- Link-Layer Acknowledgements
  By means of Link-Layer acknowledgments the core MAC protocol provides feedback of the successful delivery of the transmitted data packets.

- Passive Acknowledgements
  When passive acknowledgements are used, the sending node places itself into promiscuous mode after the transmission of a packet so as to eavesdrop upon the subsequent retransmission by the next node.

- IMEP Acknowledgements
  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 three methods inform the sender regarding the successful receipt of a packet. However, the passive acknowledgment method also makes available the following information about the next hop:

- It is not acting like a black hole,
- It is not carrying out a modification attack,
- It is not carrying out fabrication attacks,
- It is not carrying out an impersonation attack,
- It is not showing selfish behaviour, and
- It is not inducing latency delays.

The quantity of these acknowledgements taking place with respect to each node are preserved and tabularized as shown in Table 1. For every transmitted packet, the relevant counter in the table is incremented based upon the success or failure of the event.

Table 1. Trust table based on Passive Acknowledgments

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 R_{os}	Fail R_{of}	Success D_s	Fail D_f

6.2.2. Packet Precision. The precision of received data and routing packets permits computation of trust levels. For example, if routing packets are received that are found to be accurate and effective, then the originator of the packets can be assigned an elevated trust value. The above technique can also be grouped into data and control packet types and allotted trust values as shown in Table 2. Success and failure counters are maintained for every received packet, which are incremented based upon the accuracy or inaccuracy of the packet.

Table 2. Trust table based on Packet Precision

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 R_{os}	Fail R_{of}	Success D_s	Fail D_f

6.2.3. Beacon Packets. IMEP 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 assess the reliability of the neighbours based upon the frequency and accuracy of these packets. The format of the trust table based on BEACON packets is shown in Table 3.

Table 3. Trust table based on Beacon Packets

Node	Beacon Packet (B)	
	Success (B_s)	Fail (B_f)

6.2.4. Authentication Objects. IMEP enabled nodes are able to support multiple types of authentication from simple to complex verification schemes. The IMEP messages between any two nodes are verified using IMEP Authentication Objects [20], 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. These Authentication Objects are used to define a situational trust category. The trust table based upon Authentication Objects is shown in Table 4.

Table 4. Trust table based on Authentication Objects

Node	Authentication Objects (A)	
	Success (A_s)	Fail (A_f)

6.3. Trust Quantification

During trust quantification the recorded events from the trust derivation phase are measured and allotted weights so as to calculate the situational trust values for other nodes.

6.3.1. Trust Category P_A. The trust category that is derived using Passive Acknowledgements is represented by P_A. The events recorded in Table 1 are quantized as per the following equations:

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

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

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

$$R_o = \frac{R_{os} - R_{of}}{R_{os} + R_{of}} \text{ for } R_{os} + R_{of} \neq 0 \text{ else } R_o = 0$$

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

In order to limit the trust values between –1 to +1, we normalize the values of R_p, R_q, R_e, R_o and D. Complete distrust is represented by –1, zero stands for a non-contributing event and a value of +1 implies absolute trust in a particular event. If the number of

failures is more than the number of successes then a negative trust value can also occur. These normalized trust levels are then assigned weights in a static or dynamic manner depending upon their utility and importance. The situational trust $T_n(P_A)$ in node *n* for trust category P_A is calculated by means of the following equation:

$$T_n(P_A) = W(R_p) \times R_p + W(R_q) \times R_q + W(R_e) \times R_e + W(R_o) \times R_o + W(D) \times D$$

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

6.3.2. Trust Category P_P. Trust category P_P, which is based on Packet Precision is derived from the events recorded in Table 2. The following equations are used to quantize the events:

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

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

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

$$R_o = \frac{R_{os} - R_{of}}{R_{os} + R_{of}} \text{ for } R_{os}+R_{of} \neq 0 \text{ else } R_o = 0$$

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

The normalized events are then assigned weights according to their respective priority. The situational trust in category P_P for node *n* is computed using the following equation.

$$T_n(P_P) = W(R_p) \times R_p + W(R_q) \times R_q + W(R_e) \times R_e + W(R_o) \times R_o + W(D) \times D$$

6.3.3. Trust Category B. Table 3 is used to derive trust category B, which is based on the frequency and accuracy of Beacon packets. The recorded events are quantized using the following equation:

$$B = \frac{B_s - B_f}{B_s + B_f} \text{ for } B_s+B_f \neq 0 \text{ else } B = 0$$

The situational trust for category B in node *n* is computed using the following equation:

$$T_n(B) = W(B) \times B$$

6.3.4. Trust Category A. Table 4 is used to derive trust category A, which is based on the frequency and accuracy of Authentication Objects. The recorded events are quantized using the following equation:

$$A = \frac{A_s - A_f}{A_s + A_f} \text{ for } A_s+A_f \neq 0 \text{ else } A = 0$$

The situational trust for category A in node *n* is computed using the following equation:

$$T_n(A) = W(A) \times A$$

6.4. Trust Computation

To compute the aggregate trust level for a particular node, the situational trust values from all trust categories (P_A, P_P, B, A) are then assigned weights and combined. Trust T in node *y* by node *x* is represented as $T_x(y)$ and is given by the following equation:

$$T_x(y) = W_x(P_A) \times T_x(P_A) + W_x(P_P) \times T_x(P_P) + W_x(B) \times T_x(B) + W_x(A) \times T_x(A)$$

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 5.

Table 5. Aggregate Trust Table

Node	Passive Acknowledgement	Packet Precision	Beacon Packets	Authentication Objects	Aggregate Trust Level
y	$T_x(P_A)$	$T_x(P_P)$	$T_x(B)$	$T_x(A)$	$T_x(y)$

Each agent dynamically updates the aggregate and situational trust values depending upon the occurrence of events and severity of the situation. These trust levels are associated as weights to the locally maintained link heights in the TORA routing tables. The sending node can thus figure out whether a particular route suffices the minimum trust level essential for communication with a particular destination.

7. Analysis

The trust model for the Temporally Ordered Routing Algorithm Protocol is based upon proof-of-effort. The trusting node analyses this evidence passively without any superfluous computations and transmissions, making the model flexible and pragmatic for use in pure ad-hoc networks. Any node can passively receive vital information about the network by placing its interface into promiscuous

mode. This information can be utilized to build trust levels for other nodes in the network. This technique has however certain drawbacks, which have been highlighted by Marti et al. [21]. One of them 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. The other is the receiver *collision problem*, in which a node A overhears node B broadcast a packet to C but cannot perceive the collision which takes place at node C. In the same way, if the nodes have different transmission power ranges, the mechanism of passive acknowledgments might not function correctly. To overcome these problems, the weights that are assigned to different events and situational trust categories need to be selected very critically and dynamically updated based upon the antagonism of the environment.

8. Future Work

The proposed trust model is appropriate for application to ad-hoc networks as it operates in a passive manner with negligible energy and computational requirements. Presently, we are implementing the model in Network Simulator [22] so as to obtain a realistic feedback on the model's scalability, cost/benefit ratio and overhead. We are also in the process of integrating effort-based mechanisms like HashCash [23] into our trust model to reinforce sharing of node reputations. To carry out an analytical evaluation of the trust model, we are investigating the use of Zero-Knowledge and Game Theory concepts for trust establishment. We envisage extending our model to other ad-hoc network routing protocols like AODV and DSDV. Some of the issues that have not been addressed in this paper, including trust decay over time, malicious colluding nodes and security analysis of the proposed model against attacks are also currently being investigated.

9. Conclusion

In this paper we have presented a model for trust establishment in ad-hoc networks that execute the Temporally Ordered Routing Algorithm Protocol without the requirement of a central trust authority. The trust model dynamically generates and preserves trust levels based upon an effort/return mechanism. The model is not a new type of hard-security cryptographic or certification mechanism. Rather it aspires to provide confidence in nodes and subsequently routes using direct trust mechanisms. The routes that are found out in this way are not secure cryptographically but each route is linked with a definite confidence measure. The autonomy of the trust model from trusted third parties makes it applicable to both pure and managed ad-hoc networks. We believe that our model will be most appropriate for realizing reliable routing in ad-hoc networks that can be created on the fly without a trust infrastructure.

10. 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*, 6(2), 1999, pp. 46-55.

[2] A. A. Pirzada, and C. McDonald, "A Review of Secure Routing Protocols for Ad hoc Mobile Wireless Networks", *Proc. of 7th International Symposium on DSP for Communication Systems (DSPCS03) and 2nd Workshop on the Internet, Telecommunications and Signal Processing (WITSP03)*, 2003, pp. 118-123.

[3] A. A. Pirzada, and C. McDonald, "Kerberos Assisted Authentication in Mobile Ad-hoc Networks", *Proc. of 27th Australasian Computer Science Conference (ACSC'04)*, 26(1), 2004, pp. 41-46.

[4] A. Jøsang, "The right type of trust for distributed systems", *Proc. of the ACM New Security Paradigms Workshop*, 1996, pp. 119-131.

[5] R. C. Mayer, J. H. Davis and F. D. Schoorman, "An Integrative Model of Organizational Trust", *Academy of Management Executive*, 20(3), 1995, pp. 709-734.

[6] D. Denning, "A new paradigm for trusted systems", *Proc. ACM New Security Paradigms Workshop*, 1993, pp. 36-41.

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

[8] Dahill, B. N. Levine, E. Royer and C. Shields, "A Secure Routing Protocol for Ad Hoc Networks", Proc. of International Conference on Network Protocols (ICNP), 2002, pp. 78- 87.

[9] A. A. Rahman, and S. Hailes, "A Distributed Trust Model", *Proc. of the ACM New Security Paradigms Workshop*, 1997, pp. 48-60.

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

[11] Garfinkel, S., *PGP : Pretty Good Privacy*, O'Reilly & Associates, Inc, 1995.

[12] J. P.Hubaux, L. Buttyan and S. Capkun, "The Quest for Security in Mobile Ad Hoc Networks", *Proc. of ACM Symposium on Mobile Ad Hoc Networking and Computing*, 2001, pp. 146-155.

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

[14] A. A. Pirzada and C. McDonald, "Establishing Trust In Pure Ad-hoc Networks", *Proc. of 27th Australasian Computer Science Conference (ACSC'04)*, 26(1), 2004, pp. 47-54.

[15] S. P. Marsh, "Formalizing Trust as a Computational Concept", Ph.D. Thesis. Department of Mathematics and Computer Science, University of Stirling, 1994.

[16] V. D. Park, and M. S. Corson, "Temporally-ordered routing algorithm (TORA) version 1 functional specification", IETF Draft: draft-ietf-manet-tora-spec-04.txt, 2001.

[17] E. Gafni, and D. Bertsekas, "Distributed algorithms for generating loop-free routes in networks with frequently changing topology", *IEEE Transactions on Communications*, 29(1), 1981, pp. 11-18.

[18] M. S. Corson, and A. Ephremides, " Lightweight Mobile Routing protocol (LMR) :A distributed routing algorithm for mobile wireless networks", Wireless Networks, 1995.

[19] S. Corson, S. Papademetriou, P. Papadopoulos, V. Park, and A. Qayyum, "An Internet MANET encapsulation protocol (IMEP) specification", IETF Draft: draft-ietf-manet-imep-spec02.txt, 1999.

[20] S. Jacobs, and M. S. Corson, "MANET authentication architecture", IETF Draft: draft-jacobs-imep-auth-arch-01.txt, 1999.

[21] S. Marti, T. Giuli, K. Lai and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks", *Proc. of the Sixth annual ACM/IEEE International Conference on Mobile Computing and Networking*, 2000, pp. 255-265.

[22] NS-2, "The Network Simulator", http://www.isi.edu/nsnam/ns/, 1989 (last accessed 21 Dec 2003).

[23] A. Back, "Hashcash : A Denial of Service Counter-Measure", http://citeseer.nj.nec.com/back02hashcash.html, 2002 (last accessed 21 Dec 2003).