A trust framework for peer-to-peer interaction in ad hoc networks

Boodnah, Javesh

QUEEN MARY
UNIVERSITY OF LONDON

A Trust Framework for Peer-to-Peer Interaction in Ad Hoc Networks

by

Javesh Boodnah

A thesis submitted in partial fulfillment for the
degree of Doctor of Philosophy

in the
School of Electronic Engineering and Computer Science

June 2010

Queen Mary, University of London

Abstract

School of Electronics and Computer Science

Doctor of Philosophy

by Javesh Boodnah

As a wider public is increasingly adopting mobile devices with diverse applications, the idea of who to trust while on the move becomes a crucial one. The need to find dependable partners to interact is further exacerbated in situations where one finds oneself out of the range of backbone structures such as wireless base stations or cellular networks. One solution is to generate self-started networks, a variant of which is the ad hoc network that promotes peer-to-peer networking. The work in this thesis is aimed at defining a framework for such an ad hoc network that provides ways for participants to distinguish and collaborate with their most trustworthy neighbours.

In this framework, entities create the ability to generate trust information by directly observing the behaviour of their peers. Such trust information is also shared in order to assist those entities in situations where prior interactions with their target peers may not have existed.

The key novelty points of the framework focus on aggregating the trust evaluation process around the most trustworthy nodes thereby creating a hierarchy of nodes that are distinguished by the class, defined by cluster heads, to which they belong. Furthermore, the impact of such a framework in generating additional overheads for the network is minimised through the use of clusters. By design, the framework also houses a rule-based mechanism to thwart misbehaving behaviour or non-cooperation.

Key performance indicators are also defined within this work that allow a framework to be quickly analysed through snapshot data, a concept analogous to those used within financial circles when assessing companies. This is also a novel point that may provide the basis for directly comparing models with different underlying technologies.

The end result is a trust framework that fully meets the basic requirements for a sustainable model of trust that can be developed onto an ad hoc network and that provides enhancements in efficiency (using clustering) and trust performance.

Contents

Acknowledgments

No thesis is ever the sole product of one individual's efforts and this work is no exception. I would therefore like to thank the following people, whose contribution, whether direct or indirect, has helped to further the cause in this long journey.

First and foremost, my deep thanks go to my supervisor Dr. Stefan Poslad. Throughout the years, he has consistently and relentlessly pushed me towards achieving my goals and I am extremely grateful for his patience and dedication and the very high standards of his review, which I have persistently tried to meet.

I am also indebted to the School of Electronics and Computer Science and most especially the Department of Electronic Engineering for having offered me a scholarship for my research. I am also particularly appreciative of the flexibility and understanding the Department has shown during some personal hardships. People of note I would like to direct specific thanks to are Prof. Laurie Cuthbert, Dr. John Schormans and Dr. Eric Scharf for their feedback at various stages of completion of this work. A vote of sincere gratitude goes Kok Ho Huen, the department's IT guru and Melissa Yeo, our research students coordinator, for all her assistance and keeping me in check at all times!

I would also like to thank my friends whose support and encouragement have been invaluable during this experience. I am indeed very grateful to Krishnen for his advice on my work, Vivek for his tough talk and Jonathan for his constant encouragement. Special thanks also go to my cousin, Bhai, for being particularly supportive during some of the harder moments of this journey and making sure I was always well fed!

My heartfelt appreciation goes to Dav for her unwavering faith, her *trust,* her emotional support and mostly, for putting her own dreams on hold while I fulfilled mine.

And last but by no means least, I would like to pledge my deepest gratitude to In & Harry, my parents, for all their blessings, prayers, love and moral support. I hope I can make them proud.

To my grandparents

List of Figures

List of Tables

Chapter 1

1 Introduction

In our modern society, every individual depends on others for even the most mundane of tasks. The very basic need to eat in a city for example is dependent on the availability of food stores and eateries and the inherent belief that such outlets will provide nourishment that is safe for consumption. Similarly one relies on the transport network for travelling and on the media for keeping us informed. This dependence is what makes society in general work and underpins what can be characterised as a *human* or *social network.*

There is however no absolute certainty involved when one chooses to place reliance on someone else. Indeed there is a degree of *risk* associated with the *expectation* that the individual we rely on will produce a *beneficial outcome.* Using the examples above, food poisoning can occur in restaurants and fast food outlets, buses and trains often run late or get cancelled and a substantial section of the media regularly publishes biased views that may not reflect a balanced picture of events. The dependence or belief on others brings forward the notion of *trust.*

Trust as a concept is somewhat elusive (Gambetta, 2000). When used colloquially, it is rather easy to understand – a structured definition is not so straightforward. However, based on the observation of the way trust operates, one can deduce that it is brought about by the dependence of an individual on another within a given scenario to produce an outcome that is beneficial. A more precise definition will be provided in Section 1.1.

The domain of computing also sees the evolution of similar scenarios – users place their trust in centralised infrastructures such as servers in order to store their personal data (e.g. email, pictures and files). Network administrators rely on end users to keep

their credentials to access the system safe to minimise the risk of infiltration and allow only authorised users to perform tasks.

These issues are challenging in themselves but become even more problematic when open systems are brought to the fore. The recent trends in computing have been to promote the concept of ubiquitous computing (Poslad, 2009). Users have developed the expectation of round the clock access to computing services outside the conventional boundaries of the home or the office. The relentless development of new mobile devices has increased the momentum of a shift to an increased mobility for modern users. Similarly, there has also been a shift from independent computer systems to large scale and distributed open systems like grid computing. These systems all have properties that vary in time and location and it is expected that any decisions in such systems would have to be *on the fly* and in reaction to changing properties.

The issues of trust arising in such distributed systems are wide-ranging and very complicated to say the least. To that effect, the work in this thesis attempts to address some of the issues encountered within the domain of ubiquitous computing and more specifically in peer-to-peer networking. The work involves the creation of a trust framework with features that reflect the dynamic trust relationships between members of a network. It is necessary however to present the precise definition of trust as taken by this work and the specific set of issues that the framework will seek to address. This is done in the rest of this chapter.

Section 1.1 provides the definition of trust adopted by this thesis. Section 1.2 provides the relevance of trust within computing. Section 1.3 briefly surveys trust within peer-to-peer and ad hoc networking which is the main area from which the majority of the work in this thesis stems. Section 1.4 provides the objectives and 1.5 the contributions of this thesis and Section 1.6 gives an overview of the remainder of the chapters within this work.

1.1 Defining Trust

As stated previously, a universal definition of trust is very hard to achieve. One of many reasons is because trust itself is contextual and may be present in many forms that may or may not be aggregated. For instance, John may trust Marie to win a medal in jogging but would be very wary of allowing her to drive his car. In John's eyes, his trust of Marie is contextual and only applicable to Marie's jogging skills. Furthermore, it is impossible to combine the trust of John in Marie's jogging skills with his distrust in her driving in order to arrive at an overall measure of how much John trusts Marie.

The Oxford Dictionary of Current English (2005) defines trust as the "firm belief in the reliability, truth, ability, or strength of someone or something." This definition is perhaps the most widely accepted notion of trust in society, i.e. as a belief – for instance individuals put their trust in other people, machines, and computers countless number of times over any given day.

However, in the field of computing, several academics have been rather pessimistic about finding a single definition for trust and have termed it as an elusive concept (Gambetta, 2000). Over time, some level of consensus has been reached (Kuhn 1962), primarily on the positive effects of trust. For this work, a modified combined version of the definitions provided by Dasgupta and Gambetta (Dasgupta & Gambetta, 2000) will be consistently applied throughout the thesis.

Definition 1: Trust is the belief that an entity has that the other party will act honestly and reliably in order to produce the outcome expected by the trusting entity, within a given context. Such an outcome will affect the entity's own action.

By using this definition of trust, it is possible to produce a trust framework or model to derive the probability that a particular entity will be behave honestly and reliably. However, it is necessary to define what is meant by *honesty* and *reliability* first.

Definition 2: Honesty is the attribute exhibited by an entity whereby it operates without lying. An entity is also honest when it provides true feedback about its peers. This feedback accurately represents past events.

It may be necessary to provide incentives within a framework so that honesty is encouraged (Ramchurn, 2004).

Definition 3: Reliability is the expectation that an entity will perform to a set standard over and over again.

If this expectation is not met, the requesting entity may need to stipulate a specific standard of service that may result in a penalty if not met (Ramchurn, 2004).

Definition 4: A trustor is the entity monitoring the specific action of another entity as per Definition 1 that may affect its own future behaviour. A trustee is the entity performing said action.

Every time an entity successfully meets the expectations of its trustor, its *trustworthiness* increases. There is therefore a direct link between the action/behaviour of an entity and its *trustworthiness*.

Definition 5: Trustworthiness is the perceived trust exhibited by a trustor in a trustee as a result of the aggregation of one or more successful productions of outcomes expected by the trustor. The crucial point here is that trustworthiness is based directly on the trustor's own interactions with the trustee.

There are however many opportunities, especially in an open environment for entities to interact with one another where they may not have had prior direct experience with the concerned party. In this case, they clearly lack the information required to affect their decision towards any potential future action. Very often, such entities would then consult their known peers in order to verify whether they could provide any information on their assessment of the trustworthiness of the concerned party.

Definition 6: When an entity provides another entity information about the trustworthiness of a third party, it is said to provide a *recommendation* about such party. This is sometimes referred to as *indirect trust* in the literature.

Based on the recommendation of its peer, an entity may then decide whether or not to trust the third party. This decision will also depend on how much the entity trusts its peer to begin with. Often, there will be several peers who will have information about a given third party.

Definition 7: The aggregation of the recommendations of peers of an entity about a given third party is defined as the *reputation* of the third party in the view of the trusting entity. Reputation is affected by the past action/s of an entity and is therefore a good indication of how said entity is likely to behave in the future. Unlike *trustworthiness,* the reputation of a third party in the view of an entity need not necessarily include direct interactions between the trustor and the trustee.

In summary, the following notions are of relevance to the chosen definition of trust:

Two entities: When multiple stakeholders are present within a given scenario, a trust relationship concerns the interaction between two entities at any one time . These can be termed as the *trustor* and the *trustee*. It is important to note that sometimes the entities may not necessarily denote a specific individual but could also represent a collective group of individuals, for example the trust between a teacher and his class (that the class will not copy each other's work). For the purposes of this thesis however, it will be assumed that each entity is an individual.

Context: Trust is contextual – this means that the actions of the trustee are only monitored within a given context only. These actions then determine how the trustor reacts within that context.

Subjectivity: All trust is subjective as it represents the belief of the trustor. While the evidence available to the trustor may be deemed to be objective as it may be based on past actions of the trustee, there are many other factors which could affect the trustor's decision such as personal ties, status in society and so on.

Action: The belief of a trustor in a trustee for a given action has no relevance to any other action, save for the action for which the trustee is being monitored. This is highlighted in the above example whereby John prefers Marie's jogging skills over her driving.

Uncertainty: There is always a degree of uncertainty when considering trust. Actions which are past and therefore confirmed are no longer relevant, except where they provide a basis of expectation for future actions. However, while past history of events can provide an indication of future behaviour, there is inherent uncertainty associated with it. Behaviour can change and it is often not under the system's control. For example, the system's environment can affect future behaviour but this is not under the total control of the system itself – therefore introducing uncertainty. In the above example, one may trust Marie to win another medal based on past performance but her future performance may be affected by active environment factors (such as system instances like other better runners) or passive ones (different running terrains, wet or windy conditions etc.).

1.2 Trust in Computing

With trust and its associated basic concepts now clearly defined, its application to distributed computing scenarios can be considered. Just like trust, distributed computing and interaction are wide-ranging and very diverse. It is beyond the scope of this thesis to be able to comprehensively address all types of networks and their intrinsic trust issues. However, a quick overview of the main types of issues is presented below in order to justify the relevance of this work.

There are several actors to look at with respect to trust establishment within computing. The key ones are the human user, the computing resource (hardware) and the software providing the service required. All three have differing proportions of involvement when it comes to trust depending on the application being surveyed. Typical examples are online shopping where the human user trusts the online software and hardware to keep his details secure, online gambling where the

software is trusted to provide at least a reasonable chance of a win or even software to software autonomous trust for example when a user software (such as an iPhone (Apple, 2009) application) utilises data provided by an information system (such as the weather or train times).

The User: This is akin to social trust where the end user (the human) gathers information in order to derive the trustworthiness of either a particular piece of software or hardware. The trust generated by a human user is understandably subjective and may depend on a host of factors. Examples where human trust is used in the context of computing are feedback systems such as eBay (eBay, 2009) and Amazon (Amazon, 2009) where recommendations are provided in order to assist other users in their purchases.

Software: This is primarily in the domain of service provision in which there is the delegation of tasks to other systems or a reliance placed on information from third party systems. The key aspect of this type of trust is that the delegation or reliance has to take place in such a way that the overall performance of the entity is maximised.

Hardware: In computing, hardware is usually more associated with the concept of security. Unlike trust, which seeks to promote a beneficial outcome and represents the belief towards that outcome, security is mostly concerned with making certain that non-beneficial outcomes do not occur and a secure system is one that provides strong assurances (or beliefs) to that effect.

The main areas of concern in security have traditionally been in authentication (making sure the users are who they say they are), access control (also termed authorisation) and data encryption (modifying the data in a way that renders it useless if intercepted during transit). Security is therefore analogous to safety. However, most definitions of security pose it as being absolute, i.e. an all-or-nothing concept. For instance, a server or a workstation can be either secured or not. If it is not secure, then its use is not warranted. This can prove limiting in several ways. For example, it is amply possible to operate systems with non-critical vulnerabilities in

order to perform non-critical tasks. This is where trust is useful as a concept. Trust provides the ability to rate certain risks gradually rather than by using the notion of the absolute.

1.3 Peer-to-Peer Interaction and Ad Hoc Networking

Peer-to-Peer (P2P) interaction (Verma, 2004) refers to the communication and sharing of data between users independently from service and resource providers. This is done via the use of P2P networks that employ a mechanism which allows every single user to provide content to others in the pool as well as utilise the pool to request specific content. Well-known P2P networks on the Internet are Limewire (Limewire, 2009), KaZaa (KaZaa, 2009) and Gnutella (Gnutella, 2009). P2P is a form of distributed computing but the difference is that there is no centralised entity that manages the network. Instead, every user behaves autonomously and the decision of what and when to share rests solely with that user.

Without such centralised control, trust between peers is a key concept as the potential for abuse is very high within P2P systems. There are several ways in which any potential user may cheat the system, depending on the application. It may be by being selfish and only downloading from the network without sharing the use of any of its own resources. The user may also inject disruptive data (such as viruses, fake files) into the network in order to dissuade other users from sharing. There may also be a Denial of Service (DoS) attack where the network is flooded with bogus requests.

The interest of this work in P2P systems is purely academic and does not focus on the application of P2P systems for sharing what is deemed illegal content. In the view of this thesis, a P2P network is any network that is autonomously set up by its members in order to exchange files or to create a network for access to centralised services by a single gateway node (a gateway node is one that provides access between a P2P network and a backbone infrastructure such as the world wide web or

a central work or education server). This definition although more global deviates somewhat from the general perceived idea of P2P networks.

The interest of this thesis in P2P interaction lies in the fact that resources belonging to many different entities can be pooled together in constructive fashion in order to achieve a common goal. The dynamics of pooling these resources give rise to trust issues that will be dealt with later in this work. The *ad hoc network* is a type of P2P network on which the majority of this work is based.

Definition 8: The ad hoc network is a type of P2P network in which the role of sourcing, transferring and receiving data is done by the users of such data. The users of the network, known as nodes, take on the responsibility of creating routes between the source and the destination of the data. Further, for the benefit of this thesis, the ad hoc network is defined as having only local interactions as the basis of its mobile P2P interaction.

The ad hoc network is therefore analogous to passing information along a chain, the so-called whispering scenario, within a closed boundary. This means that although remote access to an outside network may be possible via a gateway, the core of the ad hoc network remains local, with interactions only taking place within a set geographical area.

Although the actual network topology may appear simple, creating an ad hoc network between strangers presents interesting issues, one of which is the trust that an entity can be reasonably expected to put in another previously unknown entity.

The perceived notion of mobile networking is to find access to a *wireless gateway node* that then provides access to wired services, for example a mobile computer finding a wireless access point or a mobile phone latching on to neighbouring base stations. The trust issue in this case is very simple. As long as there is implicit trust between users and providers, communication can happen seamlessly and can be kept private. The latter notion is facilitated by encryption and authentication, i.e., supported in most mobile networking models.

By removing this trusted conventional *wireless gateway node*, ad hoc networking presents the following challenges:

Lack of existing trusted relationships: There is no established trust between nodes in the network at the initial stage of network formation leading to the need for discovering other neighbours to build new relationships.

Finite resources: Nodes involved generally have finite resources and would have a natural tendency to selfishly preserve those resources.

Low energy transmissions: Finite resources also mean that all transmissions must be low energy, relying solely on multi-hop routing in order to forward information, rather than attempting high power, high range transmissions which may rapidly deteriorate resources.

Reliability issues: Mobile networks, by their nature, have inherent reliability issues normally mitigated by retransmission of dropped data. However, retransmission would equate to lost energy and wasted bandwidth in the case of an ad hoc network. There is therefore the need to always seek the best route through the system.

Risk to network integrity: When decentralising network routing is used by every node on the network, there is an increased risk to the stability of the system should any node be compromised whether by choice (rogue, colluding or selfish nodes) or not (nodes with spent resources or captured/impersonated nodes).

Ad hoc networks therefore have a variety of issues that need to be addressed in order for the system to exist and survive. While each of the above issues may be addressed separately by different solutions (such as the implementation of a robust networking protocol to counter retransmission needs), it is a main hypothesis of this work that a trust framework can successfully aid in mitigating all of the issues described above. While these issues may not be exhaustive (more will be provided in Section 2.2), they represent the most common problems that need to be overcome in order to achieve a viable and sustainable ad hoc network.

1.4 Research Objectives

As has been indicated, trust can be used in order to promote good behaviour in entities. It also provides an indication of the most trustworthy entities thus assuring that informed decisions are made by the concerned stakeholders. These very stakeholders have to work towards a common goal within an ad hoc network even though they may have their own goals and motivations. A trust framework provides the very basis on which this common goal can be achieved by offering the following:

- the right motivation for nodes to work towards their common goal
- adequate punishment for those that do not cooperate, defect or misbehave and attack the network.

A high level aim of the work presented in this thesis is therefore:

To develop a robust and resilient trust framework that promotes peer-to-peer interaction in an ad hoc network with a view to ensuring its sustainability during the whole period that it was initially required to operate for.

The peers considered in such a network are assumed to be autonomous and operate independently of one another. However, they all prescribe to the same algorithms required for the framework to operate. The peers operate as per Definition 8 and a network composed of such peers presents the same characteristics as that of a typical ad hoc network as detailed in Section 1.3.

For the purposes of this work, the types of interactions between peers will be restricted to simple data packets containing only the essential information for routing and trust. The added complexity introduced by varying the sensitivity of transmitted data content is not explored. However, thresholds within the algorithms of the proposed framework could be effectively varied to achieve this aim when required. This thesis focuses mainly on the design of a trust framework with the ability to

generate, disseminate, update and revoke the trust properties of peers within a given ad hoc network.
To achieve this aim, the following low-level objectives need to be met:

Objective 1: The framework must actively seek to derive trust information.

This means that a node must be able to assign an indication of trustworthiness for an entity, with which it interacts or seeks to interact, based upon the history of past interactions with that entity.

One way to achieve this is to assign a value to the entity's trustworthiness that is reflective of its history of past interactions with the querying node. This value, depending on the method used, can be any value on a finite spectrum defined as ranging between full trustworthiness and complete mistrust or may be a discrete value that represents pre-determined levels of trustworthiness. For the purposes of this thesis, the former option is used. This is further explained in Section 2.3 when presenting the definitions of trust.

Objective 2: In line with Objective 1, the framework must also allow a node to derive that trust value of an entity based upon the history of past interactions of a third party. The third party must be either commonly trusted or at least known by both the querying node and the target entity. This objective ensures that trust information flows through the network even between nodes that may not have previously encountered each other.

Objective 3: In order to manage trust information and reputation, an enhanced reputation management system must be introduced that presents clear advantages (such as lower network overheads, reduced computation and fewer storage databases) over having a solely fully distributed system.

This enhanced system is to be based on the notion of super-peers (peers with higher privileges, operating at a higher level than ordinary nodes) but seeks to decentralise that notion down to tiny (comprising of 5-10 nodes at the most) *clusters*. This

introduces the concept of *cluster heads* (Section 3.4) that take on the responsibility of reputation management for other nodes.

Objective 4: The framework must seek to provide motivation for performing nodes that selflessly compute and distribute reliable trust information about their peers. This is not only to be dictated via an enhanced reputation but also by a higher *importance* being afforded to the recommendations of better performing nodes. A system of exemptions and rewards for nodes that take on the role of cluster heads is also introduced.

Objective 5: To introduce resilience to the framework, an appropriate punishment system must be in place to address the likely occurrence of selfish, rogue or misbehaving nodes.

Objective 6: The introduction of the framework must not be accompanied by appreciable increases in workload for the nodes, nor must it introduce crippling amounts of network overheads. Instead, the framework must seek to carefully balance the roles of each of the stakeholders such that a fair system emerges that ensures maximum sustainability for all members.

Objective 7: Aside from the framework, the work in this thesis will also aim to compare and contrast the performance of other trust models without the need for additional design.

To that end, a list of Key Performance Indicators (KPIs) will be drawn up that allow the performance of the framework to be evaluated. Although these KPIs will be mostly low level indicators (primarily at the network layer), an abstraction will be performed that will allow these indicators to provide a good measure of performance at a higher level (application level for the trust framework for example).

1.5 Contribution

This thesis provides an insight into the domain of trust for mobile networking, more specifically in the domain of ad hoc networks. It recognises that ad hoc networks are inherently local interactions that present uniquely complex problems, precisely because of the localised structure. By using notions of trust from the social domain that have been formalised by other researchers (see Section 2.3), the model extrapolates the concept to the ad hoc network. Furthermore, this work was one of the first to introduce clustering within the domain of trust in ad hoc networking. In addition, the following contributions have enhanced existing knowledge in the field.

This works presents a model that meets most of the basic requirements necessary in order to instigate trust within an ad hoc network (see Chapter 2) and thus allows an entity to effectively derive the trustworthiness of its peer both via its own experience and based upon the recommendation of its peers.

The model also has the ability to generate trust without any a priori information (such as an initial common password/key or a "secret bearer"). It therefore does not require a seed in order to kick-start the trust-building process and models initial trust formation on its social counterpart by promoting initial 'break-the-ice' interactions. These initial interactions are usually different from the typical interactions the network may have been formed for as they serve mainly for the purpose of initiating contact between peers.

Aside from the above broad contributions, this work has further advanced the state of the art in the following areas:

a) Provided a means of assessing 'at a glance' the respective performances of trust models within ad hoc networks through the usage of key ratios known as KPIs. While these ratios are based on network and service data, they provide a quick means to calculate how efficient a network is running and whether there is scope for it to be extended.

b) Formally defined different roles for separate classes of node. While this work was among the firsts to introduce the concept of clustering for trust distribution in ad hoc networks, the concept of assigning specific roles dedicated to different classes of node is unique to this work.

c) Created a simple system of weighted averages for reputation that is directly proportional to the trustworthiness of a node. Because the trustworthiness itself is a measure of how reliable the node is within the network (by virtue of assessing the gain, workloads and drop rates), this implies that the best performing nodes have the most influence over the reputation calculation of new nodes. This in itself provides a better backbone for a trust framework by making it more resilient against lying nodes.

d) Evaluated the network overheads created by a trust framework both at a node and at the cluster level. This allowed this research to determine that no appreciable amount of traffic is introduced at the node level and even at cluster level. The performance is substantially higher than that of fully distributed nodes.

e) In using routing data to generate trust, the framework also indirectly provides better routes for ad hoc networks.

1.6 Thesis Structure

Chapter 2 presents the review of relevant literature within the field of ad hoc networking. It provides background to the issues and also provides additional information on the ad hoc network and trust. The scope of the reviewed work is limited to those that have chosen to adopt similar definitions of ad hoc networking. By comparing and contrasting the ways in which ad hoc networking issues have been addressed, a set of requirements is formulated that sets the basis for the proposed trust model in this thesis.

Chapter 3 provides a description of FRANTIC, the trust model proposed by this thesis. Various methods are proposed by the model in response to meeting the Objectives in this Chapter as well as the issues distilled in Section 2.5.

Chapter 4 is a short chapter depicting how the trust framework can be modelled on real life scenarios. Some performance measurement ratios are also depicted in this chapter as a general way of assessing the trustworthiness of a network.

Chapter 5 includes the experimentation performed on the low-level simulator NS-2 (2009). These simulations were performed to demonstrate the response of the framework to data input, and whether or not the trust generation properties function accurately. The resilience of the framework is also tested with the use of misbehaving nodes.

Chapter 6 proposes concluding remarks to this thesis and highlight the main contributions. Ways in which this current work can be extended are also proposed.

Chapter 2

2 Literature Review

In the first chapter, the purpose of this research was stated as being to develop a trust framework with novel features for ad hoc networks in order to enable trustworthy interactions to take place between members of a particular network. Ad hoc networks were isolated as being particularly relevant for the purposes of implementing a trust framework based on their properties and the various challenges they present. Ad hoc networks are also a good representation of the way future communications will be shaped, with more emphasis on P2P interactions, including user-generated inputs, content and appraisals. This chapter therefore presents a review of the literature in the fields of ad hoc networking and trust implementation in ad hoc networking. A concise general survey of trust and its role in computing is also undertaken.

In the first instance, the field of ad hoc networking is analysed (Section 2.1) and its characteristics are explained. The particular type of ad hoc networking suited to this work is also presented, as is its use in everyday life. An analysis of the various issues pertinent to such a network is also undertaken (Section 2.2) as well as the requirements needed in order to implement a prospective solution.

In Section 2.3, a review of trust-related literature is undertaken. Trust, although extensively used as a concept, is hard to define and to put into context. Therefore, it is necessary to understand the origins of trust within computing and how it is relevant to the domain of ad hoc networking.

With the relevance established, Section 2.4 reviews the state of the art within the very specific field of ad hoc network, with special emphasis given to those solutions that adopt a similar type of architecture when proposing their own models.

The chapter concludes with a summary of the requirements distilled from the analysis of the state of the art and the literature review and highlights the issues that are not currently addressed by existing solutions and which will be implemented within the model proposed within this thesis.

2.1 Understanding the Ad Hoc Network

The generally agreed notion of the ad hoc network (following on from the definition in Section 1.3) is that of an open configuration of nodes that undertake to form a network, usually transient, for a particular purpose. Such a network relies most of the time solely on the member nodes to act as hosts and routers in order to forward information, run user applications and share data.

2.1.1 Origins of the Ad Hoc Network

Ad hoc networking, from a historical perspective, can be traced back to as far as 1968, when work on the ALOHA network (Abramson, 1970) was initiated (the objective of which was to connect educational facilities in Hawaii). Inspired by the ALOHA network and early development of fixed network packet switching, the Defence Advanced Research Projects Agency (DARPA) began work in 1973 on the PRnet (Packet Radio Network) – a multi-hop network (Jubin & Tornow, 1987). In this context, multi-hopping means that nodes cooperated to relay traffic on behalf of one another to reach distant stations that would otherwise have been out of range. PRNet provided mechanisms for managing operation centrally as well as on a distributed basis.

Although many experimental packet-radio networks were later developed, they did not really take off in the consumer market until recently. When developing IEEE 802.11 – a standard for wireless Local Area Networks (WLAN) – the Institute of Electrical and Electronic Engineers (IEEE) replaced the term packet-radio network with ad hoc network. Packet-radio networks had come to be associated with

multihop networks of large-scale military or rescue operations and by the use of a new name, the IEEE rightly hoped to indicate an entirely new deployment scenario.

2.1.2 Ad hoc Operation

Generally, ad hoc networks operate wirelessly although wired versions may also exist in certain circumstances. However because ad hoc networks aim to provide a solution towards remote connectivity and given the fact that wired hosts are generally connected to a fixed backbone, most ad hoc networks therefore operate wirelessly. This means that all hosts are equipped with wireless transmitters and receivers. This transmission can be either broadcast, point-to-point, steerable or a combination of all. Hosts communicate with other nodes using an ad hoc network *link.* A link occurs when two or more hosts converge transmission power levels and signal patterns within a common communication channel. This forms a dynamic wireless connection. A series of links that connects two nodes is known as a *path.*

If another host decides to join this network, it goes through the same procedure of detecting a network and if accepted joins it. The same applies if a host decides to leave the network dynamically. After leaving a network, a host can either move to a different ad hoc network or simply not belong to any network at all. This flexibility makes the ad hoc network topology very dynamic and at the same time very unpredictable.

Figure 2.1 shows an example of two ad hoc networks and the change in network topology that occurs by the movement of the hosts. From the network on the left, two hosts leave the network. One of the hosts, User3 simply leaves the network whereas another host, User8 joins the network on the right. This causes a change in the topology of both networks. The network on the left has to reconfigure itself to reflect the configuration of the remaining hosts whereas the network on the right has to reconfigure itself and adjust the arrival of the new host, User8, which is now User 9 in the right-hand network. This type of movement can happen at any time without any restrictions assuming that the hosts can find the right network for them to join.

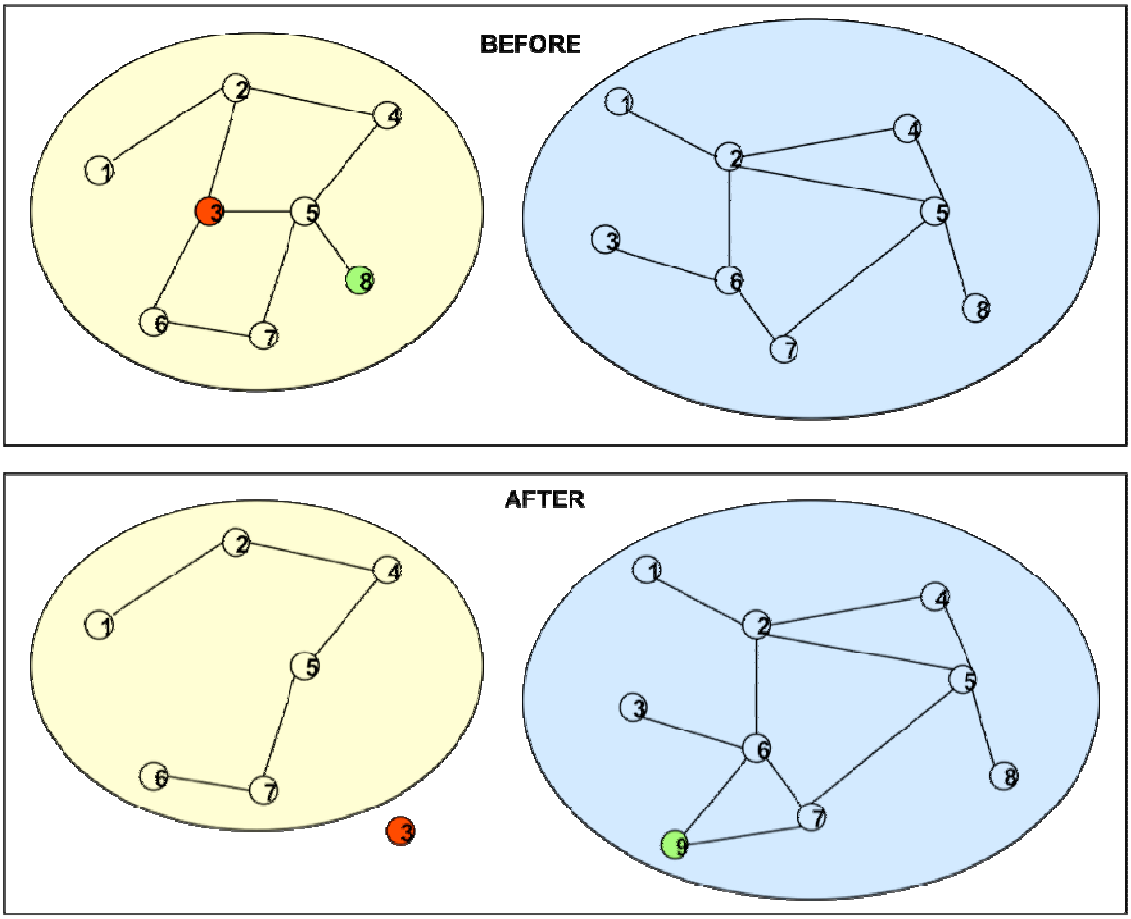

Figure 2.1: Node movement in two different examples (left and right) of an ad hoc network

2.1.3 Routing

Routing in ad hoc networks, a transfer of information from peer to peer, has been well researched. An exhaustive overview of the different routing protocols used in ad hoc networks is provided by Royer & Toh (1999). The current methods of routing use the "ask the neighbour" method recursively, till they reach the destination or another node, which has the route to the destination. Furthermore these routing techniques are classified broadly into several categories. These range from *Table Based Routing*, and *On Demand Routing* to other forms of routing such as *Hierarchal, Geographical, Power Aware and Multicast* (Bakht, 2005). A summary of table based and on demand routing is given below:

Table-based routing

Table-driven protocols are one of the original ways of routing in mobile ad-hoc networks. Each node uses routing tables to store the location information of other nodes in the network. This information is used to transfer data among various nodes of the network.

To ensure that routing tables remain fresh, several mechanisms are adopted such as broadcasting "hello" messages, a special message containing route information, at fixed intervals of time. On receiving this message, each node updates its routing tables. Destination Sequence Distance Vector routing protocol (DSDV) (Perkins & Bhagwat, 1994), Wireless Routing Protocol (WRP) (Murthy & Garcia-Luna-Aceves, 1996) and Cluster-head Gateway Switch Routing (CGSR) (Chiang et al, 1997) are some of the popular table-driven protocols for mobile ad-hoc networks.

Because of the way they operate, table based protocols are not very effective as far as ad hoc networking is concerned. This is due to the fact that nodes in mobile ad hoc networks operate on restricted resources and have limited bandwidth. By maintaining and refreshing routing tables, nodes drain away precious resources (such as battery power and computational capacity) and create unnecessary overheads for the network.

On-demand routing protocols

With on-demand protocols, if a source node requires a route to its destination for which it does not have or has incomplete route information, it initiates a route discovery process which goes from one node to the other until it reaches the destination or an intermediate node and thus has a complete route to the destination.

It is the responsibility of the route request receiver node to reply back to the source node about the possible route to the destination. The source node then uses the reply route for data transmission to the destination. Some of the better-known on-demand protocols are Ad-hoc On-demand Distance Vector routing (AODV) (Perkins & Royer, 1999), Dynamic Source Routing (DSR) (Johnson & Maltz, 1996) and

Temporary Ordered Routing Algorithm (TORA) (Park & Corson, 1997). These protocols have different ways for storing known route information and for using the established route data.

2.1.4 Ad Hoc Configuration and Structure

An ad hoc network can have three types of configuration. This is dependent, to a large degree, on the complexity of the network itself and the number of peers that are included in that particular type of network. The three typical configurations generally encountered are single hop, master-slave and multi-hop.

Single-hop devices describe point-to-point interactions between two entities without them having to rely on third parties to relay their messages. For such a configuration to be viable, this means that all nodes within the network must be connected to one another such that they are accessible directly from one another. This also implies that each time a new node joins the network, it has to ensure it establishes a connection with each and every node within the network. Such configurations are only able to work on very small ad hoc networks with nodes all scattered within operating ranges of one another. Examples of single hop ad hoc networks can be seen in Figure 2.2.

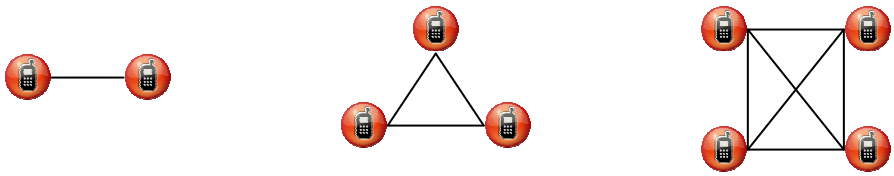

Figure 2.2: Single-hop configurations in small ad hoc networks

The master-slave configuration is a variant of the single-hop model. In this type of ad hoc networks, used especially in Bluetooth networks (Bray & Sturman, 2000), every node within a given cell, called a piconet, is connected to its master node. This results in a hierarchal structure whereby the master node is responsible for up to, in

the case of Bluetooth, seven slave nodes. All communication is routed via the master node as shown in Figure 2.3.

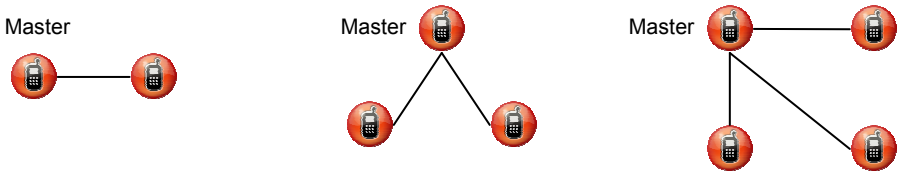

Figure 2.3: Master-slave configurations with all routing taking place via the master node (typical in small secured networks)

The third type of configuration, the multi-hop ad hoc network, is the most commonly used in research simulations. As suggested by its name, the multi-hop network requires its nodes to communicate with one another via a system of routers. In the case of ad hoc networks, such routers are the nodes themselves. This results in a mesh network and offers a certain degree of flexibility within the configuration whereby nodes may form sub-cells within the network or even break off and form separate networks. An example of a multi-hop network has already been seen earlier in Figure 2.1 and in real life may include a wireless home or office network, a temporary conference network or even gaming networks via portable devices.

Depending on the context in which they are required to operate, some ad hoc networks may have access to a fixed network backbone via what is known as a boundary node which then acts as an access point. The main advantages of having such an access point are that the ad hoc network as a whole can then have access to more resources such as certificate repositories in order to authenticate their users via the internet.

However as far as the remit of this thesis is concerned, ad hoc networks are defined strictly as pure ad hoc networks – in other words, a collection of nodes that operate independently of one another, without any access to additional resources through an external network such as a central Certificate Authority or database. In other words,

nodes operate solely within the local environment where they are found and all interactions as well as decision-making processes at all levels are strictly local.

The configuration that will be utilised in this thesis uses multi-hop networks. However, in order to enable clustering (a key component of the model proposed), a variant of the master-slave network is also used within the multi-hop architecture. This is only possible because multi-hop networks are very versatile and can be adapted in order to increase efficiency within a particular situation.

2.1.5 Ad Hoc Network Design Considerations

Having ascertained the configuration of the ad hoc network to be utilised, the other factors that would affect the deployment of a trust model onto the network must be considered. These factors, when varied, give rise to what can be termed as *scenarios*. It is somewhat utopian to attempt to provide a trust model that seeks to provide solutions to all permutable configurations and scenarios of an ad hoc network, so the physical constraints that would define a particular type of ad hoc network for a particular model of trust framework must be established.

Some of the main factors affecting the type of deployment of an ad hoc network are detailed below.

Devices in use: The limitations of ad hoc networks will be largely determined by the type of devices in operation. Essentially, any device with a wireless transmitter and receiver can operate as a node. Examples of such devices may include notebook computers, Personal Digital Assistants (PDAs), mobile phones, electronic devices in the home and so on. These devices vary in terms of their computational capacity, battery consumption and other resources. It is necessary while designing a framework or indeed proposing scenarios for a framework that the "weakest" type of device in use be considered as the benchmark by which the framework can operate. Chains are known to be only as strong as their weakest link and this is also true as far as ad hoc networks are concerned. In order to sustain itself, a network has to make

sure that any routing and trust protocols in place are manageable within the constrained operating environments of the lowest powered devices.

Range: The range of the network depicts the area within which a node can send and receive packets from its nearest-hop neighbour. Ranges within ad hoc networks can vary from a few hundred metres (e.g. wireless 802.11g/n) to as low as ten metres (e.g. Bluetooth), depending on the type of technology in use and the power of the device.

Size: The size of the network to be investigated is also paramount in determining appropriate scenarios for depiction. Again, there can be a large variation between small-scale ad hoc networks (for e.g. Bluetooth operating networks, with a few devices) to large-scale networks that can incorporate hundreds or even thousands of devices. The design for any instance of such networks will vary largely and each will thus require its own unique solution.

Purpose: Although the operation of ad hoc networks may be quasi random, they are all set up or activated because of the need to fulfil a goal, usually through mutual collaboration.

This is an important consideration because the purpose of an ad hoc network will often dictate the level of sensitivity of the data being exchanged and hence how crucial the trustworthiness of nodes can be. Goals of member nodes can be common, in which case cooperation can be expected across the board, since all are working to achieve a similar endeavour.

Nevertheless, other ad hoc networks may operate in non-collaborative environments. This is a tricky concept because collaboration is required by default for an ad hoc network to subsist. What a non-collaborative environment implies is that the node requires an incentive in order to undertake its routing duties. This is typical of self-starting networks that involve the sharing of resources. Nodes, like their human counterparts, will generally try to achieve the maximum gain for the work they put in. In fact, given the opportunity, most would leech rather than contribute to a

network. Hence, the goal of an ad hoc network needs to be factored in when determining appropriate scenarios for the application of a given framework since the algorithms determining the operation of the framework need to ensure that a fair system is in place that allows resources to be distributed equally and rewards altruistic nodes with added incentives and conversely punishes selfish ones.

Mobility: This criterion refers to the pause time of the mobile nodes within the network. The pause time is generally referred to as the time between movements for a specific node. In other words, the higher the pause time, the longer a node is stationary and therefore more likely to create trust relationships with its surroundings and further cement that relationship over time. Again, the design requirements for a framework within a highly mobile scenario would differ from that with a high pause time. For example, highly mobile scenarios would prefer refreshing data constantly over bandwidth considerations purely in an attempt to keep its information up to date, whereas scenarios incorporating a high pause time results in fewer data refreshes but with a benefit of less bandwidth usage.

The aim of the framework in this thesis is to encompass as wide a range of the factors as possible. This is very hard to achieve primarily because of the breadth of options available. It is not possible to design one single framework for all scenarios as that would entail serious compromises for e.g. having a trust framework that is able to compute and handle thousands of nodes will lead to the need for heavy processing which may not necessarily be available on low-powered devices. Besides, applications for the framework will vary immeasurably, so it makes sense to depict the specific scenarios in which the framework may be applied to in real life with minimal modification from its generic form, see Chapter 3.

2.2 Issues Within Ad Hoc Networking

There are several issues that need to be addressed in order to set up the infrastructure for an ad hoc network and by extension, a trust model for the ad hoc network. These can be categorised into two types: Physical and Trust Related.

2.2.1 Physical Issues

Member nodes do the routing in the ad hoc network. This means all forwarding of physical packets relies on the performance of nodes. One of the key necessities in an ad hoc network is that the routing protocol used must be robust. This means that the number of dropped packet rates should be low, thereby maintaining a good structural layout of open routes for information to flow between the nodes. Robustness also implies that should a route failure occur, the network is able to react quickly and can provide alternatives.

Wireless links between nodes are open and vulnerable (Cho & Swami, 2009). As such, they can be subject to both internal and external attacks. Internal attacks are caused by actions of the nodes present in the system while external attacks are brought upon either by foreign nodes or the external environment.

Internal attacks generally involve failure of the following safeguards:

- Nodes may not deliberate withhold a packet or generate a duplicate copy in order to dupe other nodes (Hu et al., 2003)
- Nodes may not alter the contents of a packet they have received during transmission
- They may not also intentionally drop packets
- They may not be fair when forwarding packets, preferring some nodes over others (Patwardhan et al., 2005)
- Nodes must also not provide routing information to other users that do not form part of the network
- They must not introduce delays in the routing by not forwarding packets in a timely manner
- They must also only move when required. A high level of mobility can disrupt the topology of the network which could lead to reduced performance or even cause its failure

- Nodes must not masquerade by disguising their identities, i.e., through providing incorrect addresses
- Nodes must also only utilise the network when required. Flooding it with unnecessary requests can result in bandwidth depletion.

External attacks are comprised mostly of link attacks that include passive eavesdropping, interference, impersonation and denial of service. Eavesdropping can allow access to secret information and violate the confidentiality of the network. Ad hoc networks are particularly vulnerable because most interactions between nodes may not include transmission of encrypted data as could be the case for more conventional wireless networks.

Ad hoc networks also do not have centralised monitoring or management points. This responsibility is usually shared among the nodes which all act as hosts. Because of the lack of an established infrastructure, the usual practice of establishing a line of defence distinguishing nodes as trusted and non-trusted, usually through authentication or access control methods, is impeded. There are also no grounds for prior classification of nodes either as the assumption is that no a priori information exists.

The existence of the actual network is also usually temporary and transient. Often this is because nodes are constantly moving and by its very nature, an ad hoc network is designed to be used in such situations. This means that most nodes within the network will often be battery powered. Access to backbone structures such as an electricity grid would also imply that other access could be assumed to exist such as an internet access point, thus making the ad hoc network redundant. Hence, the limited availability of power can be assumed to be almost always present whenever an ad hoc network is in use.

Similarly because of the physical characteristics (memory, processor etc.) of the mobile nodes, the computational power is limited. As transistor designs are improved, computational power in mobile devices will no doubt increase as will the

locally available memory. However, most users will choose to use any extra resources for their personal convenience (such as storing pictures and multimedia messages) and this then entails that resource availability for trust or security evaluation will be very limited.

The dynamic nature of the networks means that the network size can vary considerably. This means that the trust framework needs to be scalable to accommodate this possibility.

Mobile nodes can be physically captured and therefore compromised. There is no scope for locking them up in secure rooms like fixed devices are. Should a host be compromised and go undetected, then attacks can be formed from inside the network as described at the beginning of this section. Hence, the design of any framework should avoid placing a node as an overall central entity so that the compromise of one entity does not risk the whole network.

Compromised hosts will also mean compromised routing. Again, this is very likely to go undetected.

While most of the physical challenges to ad hoc networks are unique in their own right, the biggest hurdle in establishing ad hoc trust communities is the fact that, by the very nature of an ad hoc network, its nodes are completely anonymous. While in some cases, prior interactions may have existed between some nodes, they are not always prevalent and members of an ad hoc community will often find themselves joining an initial network of complete strangers. Without the "luxury" of certificate authorities and peer recommendations, establishing trust becomes an even more challenging matter.

2.2.2 Trust Issues

Trust issues in ad hoc networking are closely linked to the physical issues. This is because most models of trust (including the one in this thesis) derive some input

from the way the network operates. For example, the operation of working links within the network has a direct impact as to how the overall level of trustworthiness is perceived both from an internal and external perspective.

It follows that because the network has to constantly adapt to survive, there is a continuing threat of broken links that may not be within the influence of the network itself. For any given node within the network, this may mean that despite its best efforts, its peers may view it as untrustworthy if the links attached to it are broken by an intermittent connection.

This gives rise to two possibilities: on the one part, it can be argued that because the expectation of the peers have not been met through no fault of the node's actions, its trustworthiness must remain unaffected. This sounds logical since the environment within which the peer operates must not be an agent to its demise. Having said that, if viewed from the perspective of the peers, the outcome remains the same, i.e. a link has been broken and thus an interaction has failed.

The failure of system components within the ad hoc network must also be addressed. This aids in promoting a robust network that can adapt to it. Therefore, the design of the ad hoc network and the framework for trust must accept that failure will be a regular occurrence and necessary measures must be put in place such that the overall efficiency of the network is not affected or affected with minimal disruption. One of the ways to achieve this is to make sure that a fully decentralised system is in operation as this spreads the risk of failure over a much wider remit as opposed to the centralised system where failure of a component at the head of the network would result in affecting the whole network.

Another important consideration that must be had when considering trust within ad hoc networks is the issue in dealing with authorisation and authentication. It should be stressed that these are normally within the capacity of a lower level security system and are not normally addressed by a trust framework. For example, node impersonation or masquerading cannot be prevented as the assumption is that any ad hoc network must not be connected to a centralised infrastructure that would verify

credentials nor could it possess a priori information such as a common password in order to discern genuine entities from fake ones.

However, the key to this issue is to have an appropriate trade-off. While trust models cannot completely eradicate such behaviour, they must be able to mitigate it. Ad hoc networks are transient and not designed to be set up for a long amount of time, already this reduces motivation for any intruder/attacker to put in the amount of effort required to locate and infiltrate such a network.

Indeed given the two scenarios to be considered in this work (Chapter 4), the chances of an opportunistic intruder infiltrating the network are minimal. Even so, while a trust framework cannot fully prevent such intruders, the design for the ad hoc network must be such that it is able to recognise and punish adverse behaviour within the network swiftly and efficiently. This is a good way to restore the network to operating only with the most cooperative nodes. By keeping the safeguards simple and only promoting cooperation and good behaviour, trust within the ad hoc network can be maintained within acceptable levels for the tasks for which the framework was designed and within the context in which it has to operate.

Having put forward some of the main issues in ad hoc networking from the perspective of the overall network, some requirements for a trust model for the ad hoc network can now be formulated:

Requirement 1

Decentralised – The trust model should be decentralised and not be dependent on one entity for either network or trust considerations. This includes access to central repositories of information. Decentralisation aids in achieving a robust network that can operate even when some parts of the network are affected by operational or trust issues (Mühlethaler, 2005; Islam, 2005).

Requirement 2

Scalability – The framework must be scalable and must be able to accommodate increasing amounts of nodes, assuming most are within radio range of one another.

This is in line with the requirement by NIST (NIST, 2006) that ad hoc network topologies should be easily configurable to meet specific needs and thus be scalable from small to large networks.

Requirement 3

Reasoning – The framework must react to changing conditions with respect to perceived trust levels within the network and act accordingly so as to maintain said trust levels within acceptable limits.

The idea of *reasoning* using trust in distributed systems has been explored before. A trust management model giving autonomous entities the ability to reason about trust was depicted by Abdul-Rahman and Hailes (1997). This model however was singularly proactive in that it did not seek to re-evaluate trust decisions once they had been taken.

A more complex form of *reasoning* can be depicted in Multi-Agents Systems (see Section 2.3.2). However, in this thesis, *reasoning* is more of a reactive protocol that only activates only when certain outcomes are achieved (for example reacting to divert traffic away from an overloaded node which may have started to drop packets).

While this may not always be possible, there may also be some benefit in distinguishing environmental network outages from intentional disruption.

2.2.2.1 Issues with the Formation of Trust

With the assumption of the unavailability of both a priori information and centralised repositories, it follows that trust information will have to be generated from within the ad hoc network such that a framework mapping their distribution can be generated (Shand et al, 2003).

Trust needs to be generated between nodes willing to cooperate and therefore the assignment of a trust value to entities will aid in distinguishing those that are most trustworthy than others (Abdul-Rahman & Hailes, 1997). During the formation stage of the network and with all nodes in the open, a lot of those would inevitably interact with one another, thus creating a history of interactions that eventually become relevant the longer it exists. This history attaches itself as part of a node's characteristics and becomes a good indicator of the reliability expected from the node based on its past performance.

In other instances, a node may encounter another with which it may not have had prior interaction (Cahill et al, 2003). Following on from typical human behaviour, the appropriate action would be to seek the reputation with which the interaction is to take place. This reputation (see Definition 7 in Section 1.1) is based on the interactions of others within the same system with that target node.

Using reputation from other nodes introduces another set of issues in itself. For instance, it becomes necessary to identify where the reputation is coming from. This could be either a peer that is trusted by the requester node or a third-party node within the network that may have been recommended by the trusted node. In some cases, there may be localised repositories one may access in order to determine the reputation of a node one may wish to transact with.

The other issue with obtaining a reputation from another entity is the motivation of said entity to provide that reputation (Capra, 2005). Good, reliable information is not usually free in real life and therefore it would make sense that any node providing or storing reputation information may need some form of payment or incentive in order to perform said task.

To that effect, a trust framework may need to cater for several classes of nodes with respect to the functions they perform within the trust framework. The class of node may be affected by several factors ranging from the longevity of its presence within the network to the number of previous interactions it may have had with other nodes within its vicinity.

By creating a class of node one may deduce the reliability of the reputation information provided based on the source of such an opinion.

The above issues thus allow for some additional requirements to be introduced with respect to creating a trust framework.

Requirement 4

Trust information – The framework must be able to generate trust information. This is to be calculated by an entity based on the past interactions of said entity with another entity.

Although this by itself is a most basic requirement of a trust model, the information generated by the framework must be in a mathematical (numerical preferably) format such that it can be easily combined with other similar trust information in order to provide a more balanced view. This notion of *observe* and *infer* was also addressed in the SECURE project (Cahill et al., 2005).

Requirement 5

Trust recommendation – The framework must provide a way for the reputation of a node to be generated and held within a network based on previous interactions within the network. The concept of reputation allows for a node to be able to interact with a hitherto unknown node by requesting this information from a trusted third-party.

Reputation is a common and generally accepted way of reviewing a node's trustworthiness. Models utilising this concept are further reviewed in Section 2.4.

Requirement 6

Reward or payment scheme – In order to incentivise nodes to provide reputation information about third parties to their peers, a system of reward or payment should be implemented within the framework such that nodes are justly compensated for the extra resources they dispense in order to perform these additional tasks.

Quite often, this aspect is neglected in trust frameworks and cooperation is erroneously (in the view of this thesis) assumed (Abdul-Rahman & Hailes, 1997). The importance of having incentive schemes for trust frameworks was also highlighted by Capra (2005).

Requirement 7

Classes of node – Once the network is set up and connections have been formed between nodes, classes of nodes may naturally start to emerge. It would be desirable to make this a feature of any given framework. Nodes that operate selflessly in the network and therefore develop a high work/reward ratio can be given a higher status within the network. Other factors such as longevity of the node's existence within the network may also be taken into account. Nodes which achieve higher status could thus be responsible for more sensitive processes within the network.

While there are other models that operate with super peers or localised network heads (see Section 2.4.5), this work believes the best practice is to assign specific tasks (especially in reputation calculation) to the higher classes of node, while at the same time relieving them from routing networking tasks.

Requirement 8

Repositories – Although not strictly necessary, it may be preferable that higher classes of nodes be allowed to hold reputation information across the network. Such nodes can act as localised repositories allowing quick access to reputation information. This may also aid robustness in the network by reducing the risk of false recommendations being spread throughout the network.

Requirement 8 is in line with the assumption in Requirement 7 that only certain classes of nodes be allowed to compute and store reputation information. Requirement 8 is in fact a by-product of Requirement 7 in that by allowing localised heads only to store the trust information, local repositories are being formed across the network that grant access quick access to reputation information. The geographical proximity of querying nodes also reduces overheads on the network as well as aid robustness as described above.

2.2.2.2 Issues in Trust Management

Once the ad hoc network is up and running with trust distributed throughout the network via relations between nodes, it is important to maintain the data about the level of trust within the network up to date. The assignment of a trust level to a particular node represents a type of contractual agreement that the node is expected to behave in a particular manner. Any deviation from the set of expectations for that node must invariably result in its trust level being updated.

Furthermore, during the initial stage of trust formation, there are a high number of unknown nodes and it is quite possible that some trust relationships would have been formed with so-called rogue or misbehaving nodes. The continuous assessment of nodes while the network is operating ensures that those relationships are rapidly identified as flawed and the network takes appropriate measures to negate their effect.

In other instances, there will be times when some entities will not be able to perform to the set of expectations assigned to it by virtue of its trust level. This may be for several reasons ranging from selfish behaviour to more operational causes such as the scarcity of resources (for example when a node reaches critical power levels). As far as trust is concerned, the framework will need to re-evaluate or even isolate such entities for the benefit of others within the network.

Therefore from a trust maintenance point of view, the following additional requirements would be necessary in a model of trust.

Requirement 9

Constant evaluation – The model must provide a way for an entity's trust value to be varied over time. This variation must be as a direct consequence of the behaviour of the entity towards its peer. Each observation of the peer must therefore result in causing an update to the trust value of the entity such that it is always accurate when requested at any point in time.

Periodic re-evaluation is a feature of some models reviewed in Section 2.4. However, the manner of the reputation refresh tends to vary. The model in this work opts for temporal variation of reputation, including concepts such as the gradual decay of trust reputation (see model in Chapter 3).

Requirement 10

Isolation – In extreme circumstances, it may be necessary for an entity to be revoked from the network membership. This may be to several reasons, some of which may not be malicious in nature. For example, an entity that has run out of battery or whose hardware has malfunctioned will be of little use to the network as a whole. Its removal from the list of member nodes must be a feature of the network as otherwise resources may be lost on attempting to communicate with it based on historical information.

This concept is analogous to the idea of social control mechanisms where malicious peers are revoked from the network based purely on observational methods and without relying on central databases or a given third party (Rasmusson & Janson, 1996).

2.2.2.3 Other Trust Issues

There are other aspects of trust that need to be addressed within the ad hoc network. By creating different classes of node, the network may promote localised groupings of trust within the network as entities become more attuned to the types of neighbours they would like to congregate with.

For example, there may be associations between entities based on the type of content they wish to share (for example a gaming network or individuals sharing music), there may also be associations on the basis of interaction history. Nodes may also aggregate if they have had interactions prior to joining the network. Although this is not a requirement for a trust model, it follows that should the information be

available, the network should make use of it as it would enhance the reliability of information and promote the distribution of trust more quickly around the network.

Another issue that needs to be recognised is that trust is context dependent (Nguyen & Camp, 2008). This means that a trust value is only relevant if future expectations of the node are within the context within which the trust value was generated. Within a trust model for the ad hoc network, such distinctions may at times be necessary. In most cases, the context across the network will remain the same as a spontaneous ad hoc network will most likely be set up for a particular reason and the overall goal of the network would be the same (for instance a network specifically set up for a rescue operation). Nevertheless, a complete trust model must be able to define trust value within its contextual profile.

Finally, with respect to the trust lifecycle, it is important that measures be in place when entities reorganise. This can involve them leaving the network altogether or changing geographical positions within the network due to having higher affinities to a different association.

The above issues would require the following features to be available when considering a trust model.

Requirement 11

Selective grouping – Network associations must be allowed and encouraged within the network. While a fully distributed model is advocated, the presence of groups within the network will enhance a trust framework in two ways. Firstly, members of a group will find it easier to update and exchange information, as they will hold stronger ties. Secondly, the presence of groups may make collusion attacks more detectable as the model will also promote group monitoring as opposed to only node monitoring on an individual basis.

Grouping also known as clustering is further explored among similar models in Section 2.4.5.

Requirement 12

Using external information – Although the assumption is made that no a priori information should be available when an ad hoc network is being formed, the model should have a feature that enables such information to be positively used to enhance the trust formation process. Often, there will be social links between entities prior to joining the network – it will be highly beneficial to include these factors in the determination of trustworthiness.

Requirement 13

Contextual aspect of trust – The trust values calculated and distributed across the network must specify the context in which they were generated. This is especially relevant when members of a network, although having a loose overall goal, may have localised specialised interests. In such scenarios, it becomes crucial that nodes are aware of the type of trust information they are receiving. Any such information would only be relevant for the peer within the context in which the trust information has been generated and stored.

The SECURE model (Cahill et al, 2005) outline this concept that trust is only valid within a given context. Other models using contextual information are Capra (2004) and Nguyen and Camp (2005). The latter in their work seek to use contextual information in order to improve the trust evaluation process. Their definition of context is more complex than the two former works in that they consider context as being determined by environmental factors which influence communications between a client and a server.

Requirement 14

Enabling migration – Nodes may often wish to move to other parts of the network where they may not have had much prior interaction. The network must provide a way for the trust information associated with a node to be appropriately passed across a network as the node migrates. When utilising a fully distributed system, the trust information will not be held centrally. All local repositories would not hold the same information, hence the need for a node's history to be made transferable in

order to distinguish it from a new member node as considerable effort and resources would have been put in for a node to achieve its trust value.

Requirement 14 helps to improve the efficiency of the network in a similar way to the synergies provided by the use of grouping mechanisms.

2.3 General Literature Review

The following section will briefly review two sections of the literature that, although relevant to the main focus of this thesis, are not part of its core scope. However, in order to situate the context of this work, such a review is necessary:

1. in order to understand the origins of the concept of trust (Section 2.3.1).
2. to provide the larger picture of where the work sits by quoting other major bodies of work in trust that have been/are being undertaken alongside that in ad hoc networks (Section 2.3.2).

2.3.1 Trust Literature & Social Models

Having reviewed the ad hoc network as a system and the issues and challenges it presents when seeking to establish trust, the focus of this review will now be on the actual concept of trust. Trust was initially defined within Section 1.1 as were the various concepts associated with trust such as reputation. However, in order to understand its relevance in ad hoc networks, it is necessary to study its origin and the benefits it provides. Trust in computational models stems from trust in social and psychological theory and indeed, the meaning of trust itself has not changed.

To that effect, a review of trust as a concept within the literature follows. This is deliberately kept short as the evolution of trust in social theory is not the focus of this thesis. The review, while not exhaustive, provides an overview of how trust is seen within other fields and its relevance to computational models. The transition from

social to computational trust is taken from previous work completed within the field (Gambetta, 1990).

The concept of trust has been widely studied in the literature (Kramer & Tyler, 1996; Golembiewski & McConkie, 1975) and many sociologists have ventured definitions. According to Marsh (Marsh, 1994), four researchers in trust can be regarded as major contributors in the field: Morton Deutsch (Deutsch, 1962 & 1973), Niklas Luhmann (Luhmann, 1979), Bernard Barber (Barber, 1983) and Diego Gambetta (Gambetta, 2000). Of these four researchers, Gambetta provides the probabilistic definition of trust that is the most formal and therefore the most adaptable to a computational model of trust.

Gambetta presents the following definition of trust:

"Trust (or symmetrically, distrust) is a particular level of the subjective probability with which an agent assesses another agent or group of agents will perform a particular action, both before he can monitor such action (or independently or his capacity ever to be able to monitor it) and in a context which affects his own actions. When we say we trust someone or that someone is trustworthy, we implicitly mean that the probability that he will perform an action that is beneficial or at least not detrimental to us is high enough for us to consider engaging in some form of cooperation with him.

Correspondingly when we say someone is untrustworthy, we imply that that probability is low enough for us to refrain from doing so."

From this definition, Gambetta uses values to depict trust. The use of values allows a more precise interpretation of specific circumstances in trusting behaviour. Gambetta uses values in the range of 0 to 1. He defines trust as a probability (as above), whereby 1 represents implicit trust and 0 complete distrust.

There are however some limiting factors to Gambetta's definitions. For instance, trust relationships are only defined between agents without taking account the

environment in which the agent operates. By referring to just one dimension of trust (predictability), Gambetta ignores the "competence" aspect. When 'A trusts B', along with the actual decision, there is also the act of relying upon B. Social reasoning is complex and Gambetta's subjective probability merges several key parameters and beliefs.

He nevertheless makes a very important comment in his work (Gambetta, 1990). He points out that trust can never do worse than sustained distrust in any given scenario. On the contrary, it might do better, even if it is only marginally so. This means that agents and nodes are likely to relate much better to one another within an unpredictable world, should they be aware of the usefulness of trust. In fact, Gambetta's work is the most relevant to this current research, especially in terms of its conceptual building blocks. For the remainder of this thesis, trust will be *broadly* assumed to be a *subjective* probability with which an agent assesses another agent or group of agents. The term "agent" is loosely used here and *subjective* probability refers to the individual nature of the assessment of trust, in that the probability is derived based on the personal, therefore subjective, observations of one particular node (more in Section 2.3.1.2).

2.3.1.1 Types of Trust

Usually, trust encompasses human personality as well as social formal systems. (Abdul-Rahman, 2000; Luhmann, 1979).

According to McKnight et al. (McKnight & Chervany, 1996), in any social interaction, there are any of the following two or three entities that are involved: an entity, an agent with which it interacts and the environment. They identify three types of trust:

1. System Trust (Structural/Impersonal Trust) – This refers to any trust relation that is not involving the property or state of a trustee, but rather the property of the environment within which it operates. For instance mobile users trust their network

to correctly encrypt and secure their conversations. This is defined as system trust in the network.

2. Dispositional Trust – This refers to the trust of the trustor. It does not depend therefore on any other entity or situation. An entity's disposition to trust is reflected by the initial trust it chooses to give and its reactions to feedback affecting such trust. (Rotter 1967; Brann & Foddy, 1987) in (Abdul-Rahman, 2000). There are further subdivisions of dispositional trust. Type A deals with the trustor's belief that entities are generally trustworthy and should thus be treated accordingly. On the other hand, Type B trust takes the view that a more positive outcome is possible if one acts as if the trustee is really trusted, irrespective of whether the trustee is trustworthy or not.

3. Interpersonal Trust – This type of trust is one that depends on the properties of the trustee. The latter has the power to affect how the trustor views him/her by exhibiting corresponding behaviours.

Abdul-Rahman (2000) also introduces a final type of trust called blind trust, which can fall under each category of trust described above. Blind trust, as its name suggests, implies an extreme form of trust that persists even when there is evidence that discourages such trust from being maintained (Luhmann, 1988).

2.3.1.2 Representations of Trust

According to Abdul-Rahman, there are five different ways of representing trust. These are categorised namely by a relation, a subjective probability, a threshold, dispositional trust and a belief.

Relation

From the very basic definitions of trust, it has been identified to be a three-part relation to the effect of:

X trusts Y about A

In the above example, *X* and *Y* are entities that are involved in the trust process, i.e. the trustor and the trustee while *A* is the action about which *X* trusts *Y* for. This relation is depicted in the following works: (Baier, 1985; Luhmann 1979; Hardin, 1993).

Subjective Probability

In depicting trust as a subjective probability Gambetta assigned certain specific values for his trust semantics. These were 0 to denote complete distrust, 0.5 for an uncertainty and 1 for complete trust. A number of theoretical trust models have used this representation, sometimes including a modification whereby complete distrust is expressed as -1 instead of 0.

However, the meaning of the values assigned by Gambetta are ambiguous to say the least. There can be no comparative analysis between different scenarios. For instance, because of the discrete trust values, two entities exhibiting different levels of trustworthiness may both be classed as 1. This therefore makes this unsuitable for proper implementation in practical computational models and may only operate as an intuitive method of assigning trustworthiness.

Trustworthiness

It is important to note that trustworthiness implies a property that is inherent to an entity. This is different to trust which is more of a varying relationship between a trustor and its trustee. Gambetta highlighted the difference between the two as follows (Gambetta, 2000):

'Trustworthiness' concentrates on a person's overall disposition, his motivation, the extent to which he awards importance to his own honesty. Being able to trust a person to do what he said he would, on the other hand, requires us to know not only something of his disposition, but also something of the circumstances surrounding the occasion at hand. If the incentives are 'right', even a trustworthy person can be relied upon to be untrustworthy.

Threshold

Abdul-Rahman also brings up the notion of trust as threshold, mainly based on the works of Luhman and Gambetta. The threshold is defined as a point beyond which an entity may choose to trust or not. Originally meant as a way to simplify decision-making, the threshold is a minimalist mechanism to define trustworthiness and untrustworthiness.

This may be useful in certain situations in practical life where an entity may only need to make conscious decisions with respect to a variety of possibilities, such as which shop to choose to buy something or which airline to fly. These are decisions where a clear-cut demarcation is appropriate and even desirable.

However, in most computational scenarios being investigated pertaining to ad hoc networks, there is often the need to choose between, philosophically speaking, the better of two goods, or the lesser of two evils. In mathematical terms, it is necessary to have a mechanism in place that can distinguish between two trustworthy entities (for instance both displaying trust values above 0.5 on a continuous scale ranging from 0 to 1).

Dispositional Trust

This model proposed by Hardin (Hardin, 1993) represents levels of dispositional trust that is acquired via a constant re-evaluation of an entity's predisposition to trust, based on experience. It is in direct contrast to trust that is bestowed arbitrarily.

Using his trust model, where he assumes the trust distribution to be linear between 0% and 100%, Hardin proposes a scheme whereby positive pay-offs occur when trust is not betrayed and conversely negative pay-offs occur when there is a betrayal of trust. This results in having two straight curves for high trust and low trust, with a break-even point that denotes the optimum balance usually present in the "objective real world".

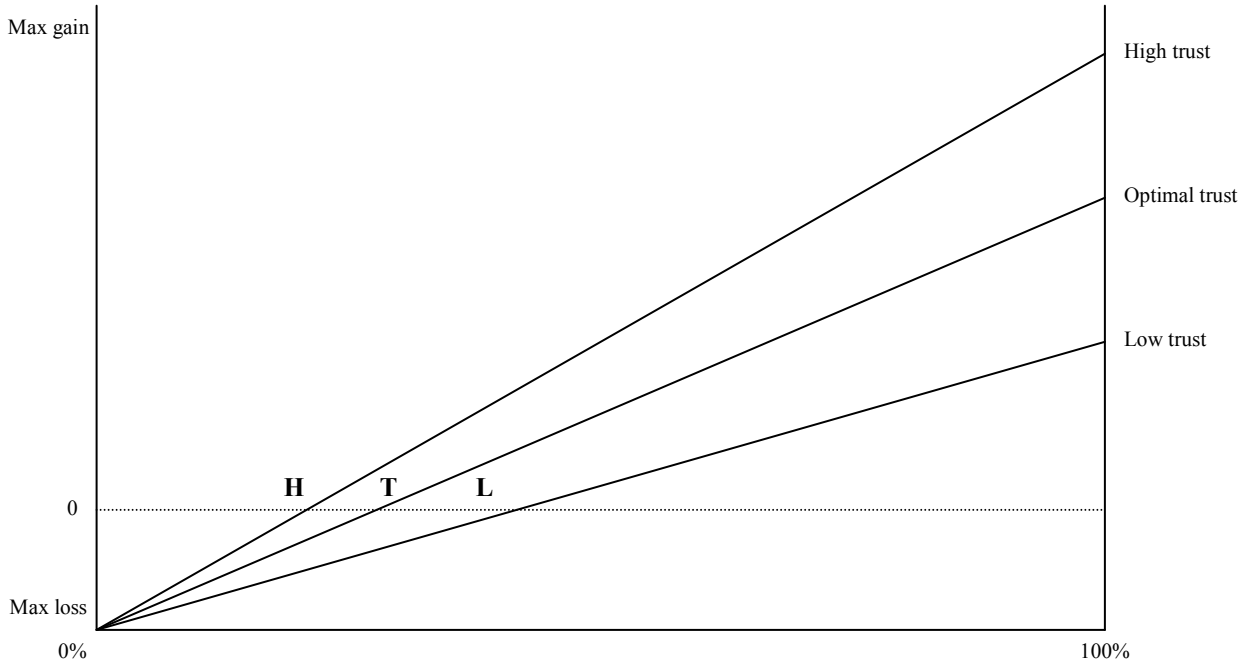

Figure 2.4: Hardin's trust model (adapted from Hardin, 1993)

Hardin acknowledges some limitations of his work. The relative sizes of loss and gain are not defined and there is no mention of situational trust where different levels of trust would have to be applied. He also attributes the model as half-strategic with no "sophistication to the trustee" and ways of learning are also ignored.

Belief

The last representation of trust is the subjective notion of belief and encompasses the previous representations described above. This brings us back to the definition of trust as a subjective belief rather than a property or relationship that is existent between two entities.

Having briefly surveyed the notions of trust within the social literature, the next stage of the review is to focus on other major bodies of work that have implemented trust as a major concept within their research.

2.3.2 Parallel Research in Trust

Trust within computing takes many forms with work being done in various areas across wired and wireless networks using hierarchal or distributed models. This work itself is an application of the trust concept within the wider area of distributed computing. In such systems, the various components of a network are spread in a decentralised fashion and are consistently subject to change throughout the system's lifetime (Ramchurn, 2004).

Examples of environments in which distributed computing is in operation are the Grid (Foster and Kesselman, 1998), the semantic web (Berners-Lee et al., 2001), web services (Seth, 2003), e-business (Kersten and Lo, 2001), m-commerce (Tveit, 2001; Vulkan, 1999), pervasive computing environments (Satyanaranayan, 2001), autonomic computing (Kephart and Chess, 2003) and peer-to-peer computing (Ripeanu et al., 2002) of which this current work is a subset.

A common computation model used in some of the distributed environments detailed above is agent based computing (Jennings, 2001). There is a parallel to be drawn between the concept of an agent as defined by the Artificial Intelligence (AI) community as a whole and that of a network node model. An agent is something that acts – computer agents "operate autonomously, perceive their environment, persist over a prolonged time period, adapt to change and create and pursue goals." (Russell & Norvig, 2010). The network node model can be mostly perceived to be a "rational agent", i.e. one that acts in order to achieve the best outcome, or in the case of uncertainty, the best expected outcome. A common basic design for a rational agent is a reflex agent that fires actions in response to input environment events that are filtered with respect to some condition or rule.

According to Woodridge and Jennings (1995), agents must display the properties of reactivity (the ability to respond to changes), proactiveness (the ability to seek opportunities to satisfy goals) and social ability (the ability to interact with other agents in order to satisfy its goals).

Multi-Agent Systems (MAS) can be used as a formal reasoning model in order to implement trust. Trust in MAS revolves around individual-level trust and system-level trust. According to Ramchurn (2004), individual-level trust exists whereby an agent has some beliefs (for e.g. honesty or reliability) about its interaction partners and system-level trust exists when the participants are forced to be honest by protocols and mechanisms that regulate the system. In fact, while the individual-level trust models allow for an agent to reason about its level of trust in its partners, the system-level mechanisms seek to ensure that the actions of these partners can be actually trusted.

Therefore, formal reasoning at an individual-level can take place in three different ways:

1. Using learning and evolutionary models: this occurs when agents reason about strategies to be used with trustworthy and untrustworthy partners, for instance by reciprocating their trust or being selfish.

2. Using reputation models: this involves reasoning about the trust information gathered by different means either directly or indirectly through reputation models about potential partners.

3. Using socio-cognitive models: this happens when agents reason about the motivations and capabilities of their partners to decide whether to believe in their trustworthiness.

System-level trust on the other hand force agents to act truthfully via the following 3 methods:

1. Using mechanisms that promote trustworthy interaction: this involves creating conditions that would deactivate the operation of agents that do not abide by them.

2. Using reputation mechanisms: this involves using reputation to enhance or reduce future interactions of agents depending on how well or not they behave.

3. Using security mechanisms: this involves specifying strict standards of conduct that agents must satisfy and consistently maintain in order to be part of the system.

Individual-level trust and system-level trust can therefore be depicted according to Figure 2.5.

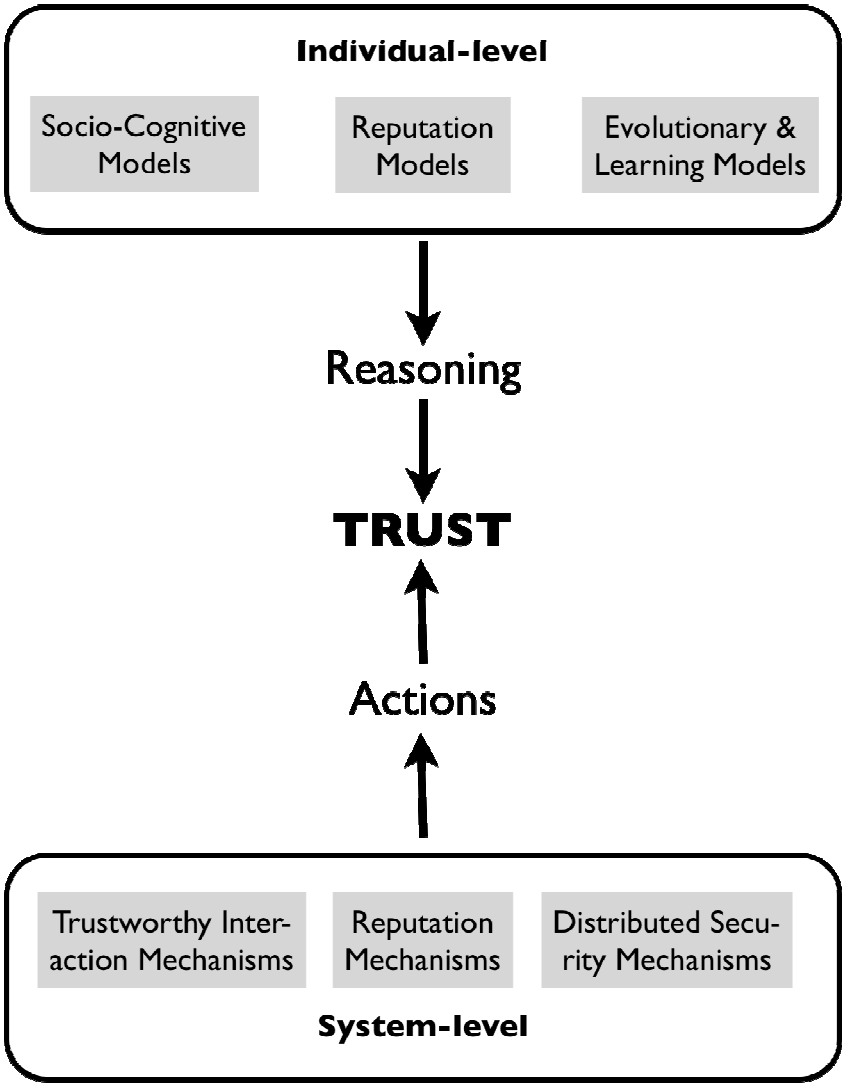

Figure 2.5: Approaches to Trust in Multi-Agent Systems (adapted from Ramchurn, 2004)

These approaches to trust in MAS are relevant to the work in this thesis in that they provide the basis (and justification) for merging social-based reasoning along with the rigours of computational rules and processes. Agents, however, are complex autonomous agents. The peers depicted in this current work are not assumed to be

complex, although the trust framework may be expanded to accommodate additional complexity. However, one of the main aims of this work is as a proof of concept – the scope of the research will therefore be restricted to behaviours that are simple enough to not require complex reasoning, yet, relevant enough that they produce the kind of trust information required for the proof of concept.

The work in this thesis also forms part of a subset of peer-to-peer networks, since it focuses primarily on pure ad hoc networks as defined in Chapter 1. Some aspects of agent trust will therefore be reflected in the properties of the peers depicted within the model since the work finds itself within the wider remit of distributed computing.

However, before defining the trust framework, a review of the state of the art within the specific domain of ad hoc networks is necessary such that the proposed model is able to build on existing knowledge and extend it further.

2.4 State of the Art: Trust Models in Ad Hoc Networks

Trust models in the literature can be classified broadly in to centralised and decentralised models. However, centralised models tend to be within the remit of those models with a backbone infrastructure and access to centralised data. eBay's feedback system (eBay, 2009) is a good example of a centralised trust system that aggregates user opinions on a central database. The information is collated by a central entity, processed by that same central entity and is then public for all to view. This is a feature of most centralised models.

While centralised models offer to meet several requirements of a trust framework (namely Requirements 4 (trust calculation), 5 (reputation calculation), 9 (re-evaluation of trust information) and 13 (some context of trust)), they pose several restrictions with regards to the other Requirements. For example, they are not scablable nor are they able to use additional information in order to enhance their information.

Centralised models are also very inappropriate for ad hoc networks both from a trust perspective and for operational reasons (as explained in Section 2.2). In fact, of all the literature reviewed, only one model used a centralised architecture (Davis, 2004). This was a scheme run by certificates whereby a node is not trusted until it presents a certificate issued by a central authority, with the assumption that said certificate has not been previously revoked nor expired.

The following review of models in ad hoc networks therefore all refer to decentralised systems and it can be assumed that all of them therefore meet Requirement 1 (decentralised model) as will the model proposed by this thesis. They are classified according to the methods they employ. Where there is a mixture of methods employed, the models are classified according to the most relevant one.

2.4.1 Certificate-based Models

These denote models that convey trustworthiness by exchanging what they term as "reliable" certificates to which an entity's identity and history is bound. Trust as a terminology is used rather loosely and sometimes interchangeably with security. The model for certification generally follows a four-part pattern namely:

1. Issuing of Certificate
2. Storage of Certificate
3. Certificate Validation
4. Revocation of Certificate.

While the method proclaims that steps 2-4 are performed locally, that is not the case for step 1 which depends on a Certificate Authority (CA) to issue valid certificates.

This is a very big assumption and a very significant limitation as far as the definition of ad hoc networks goes. For instance, a priori information is not allowed in any transaction that occurs between nodes, let alone the existence of pre-issued certificates. In this case, the usage of such certificates would immediate pose certain

problems whereby all nodes will have to reach a certain agreement about which online CAs are trustworthy or not. This is unlikely to be a trivial matter and furthermore the scheme is not immune from forged certificates that would allow a rogue node to operate at free will within the network, thereby bringing it into jeopardy. "Trust" schemes using certificates could simply be reclassified as an access-control mechanism which allows nodes to enter/leave a network based on credentials issued by a third party. Models using certificates generally only meet one of the requirements of a trust model other than Requirement 1 (Requirement 2 – scalability).

One such model which utilises certificates is that by K. Ren et al (2004) which proposes a distributed trust approach which claims to build well established trust reputation systems without relying on any predefined assumption. Resilience towards nodes' dynamically leaving/joining and scalability are also the aim of this project.

Formalisation of the model is done via a probabilistic method based on a trust digraph model. The authors assume that a certificate graph G(V,E) represents the public keys and certificates of their system, where V and E are the set of vertices and edges respectively. The vertices denote the binding of public keys and their IDs whereas the edges represent the certificates. As shown in the Figure below, a directed edge from a given node *i* (public key of node *i*) to node *j* (public key of node *j*) will exist if there is a certificate signed with the private key of *i* that binds the identifier of node *j* and its public key. A certificate chain from *i* to *j* is represented by a directed path from vertex *i* to vertex *j* in *G*. Trust is therefore established whenever two nodes are connected and the certificate chain represents the trust chain.

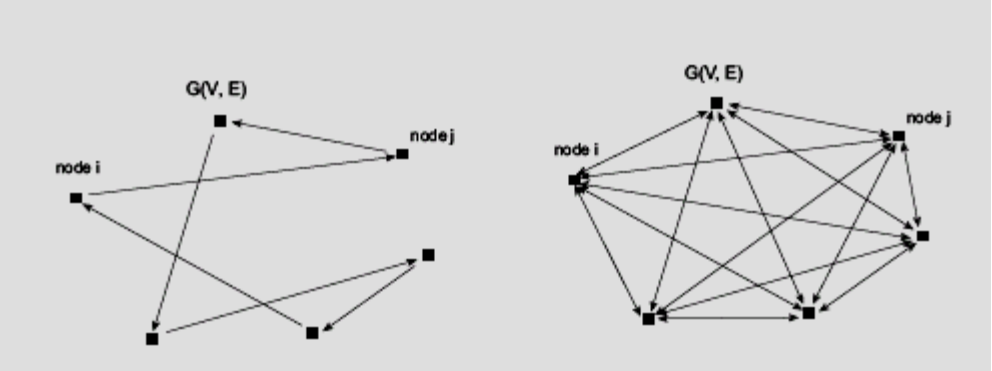

Figure 2.6: Trust chains showing directed paths between vertices indicating a certificate chain (Ren et al., 2004)

However, before such a stage is reached, the authors define what they term as a bootstrapping phase – which represents the initial interaction between nodes. While the initial claim was that this method did not rely on any predefined assumption, the *bootstrapping phase* involves the introduction of a secret dealer to facilitate the initialisation process. The dealer is defined as an external entity that enjoys a long-standing trust relationship with all the nodes that cannot be questioned. An example of a secret dealer would be, according to the authors, a telecommunications provider that is common to all the nodes. The dealer pre-computes an individual secret short list that it then distributes to each member node. This short list contains a node identifier and its subsequent public key. The node then issues its own certificate from its own domain and stores it locally.

The *bootstrapping phase* therefore makes the very crucial assumption of the availability of a priori information from a secret dealer. Again this entails access to some fixed network infrastructure and also readily assumes that all present nodes will necessarily have a long-standing relationship with that provider. This is not only unrealistic but also very unlikely given the number of primary and secondary telecommunications providers available within any given location. Furthermore, the method does not provide for any alternative to the current *bootstrapping phase*. This makes this approach very weak, should a common secret dealer not be available. This model therefore only meets Requirements 1 (decentralised model) and 2 (scalability).

Another model that utilises certificates is by Ren and Boukerche (2008). This model additionally meets Requirements 4 and 5 (the ability to compute trust and reputation) but uses a central node in order to authenticate new nodes. While this central node is group-based, the authentication is via public keys and utilises encryption in order to protect password secrecy. This may enhance security but not necessarily trust as the latter is achieved by promoting interaction between nodes. Furthermore, the use of authentication and encryption poses additional load on the network which can be a critical factor in open environments where limited resources are available.

A variant of the certificate method is a model by employed here by Keoh et al (2003) revolves around a doctrine, which is a specification for trust, defined here as the expectation that nodes within an ad hoc network will enforce rules defined by such a doctrine. The latter is itself a template that is parameterised by participants within the ad hoc community and then made available to anyone to sign up to before joining the network. Information available in the doctrine includes role type specifications, user role assignment policies and policies governing behaviour or entities assigned to the roles. Each role has a defined set of credentials and when a new participant wishes to join the community, they have to ensure that those credential requirements are fulfilled. Once access is granted the new participant can then perform its role based on what is allowed in the doctrine. The use of the doctrine means that this model additionally meets Requirement 12 (additional information).

Credentials are verified here again by means of public key certificates which are issued by Certificate Authorities (CAs) and Attribute Authorities (AAs). It is therefore another requirement that public key information of CAs and AAs has to be included within the doctrine to ensure a seamless verification process. The alternative to such verification as stated by Keoh, for instance in a community where all exchanges are strictly P2P with no access to any fixed network infrastructure, entities have to rely solely on the information they have between themselves. However, such an alternative is not developed nor is a method of achieving it proposed. The assumption that at least one device will have a prevalent intermittent connection to a fixed backbone is justified as being the most realistic scenario.

Again, like most other methods, the access to offline or external data is inherently assumed and relied upon for the successful implementation of the community. In the absence of such data, the community would most likely fail as there is no specific strictly P2P exclusive infrastructure which can kick in as an alternative. Furthermore, there are several definitions such as user and configuration policies that are included in the doctrine but the way in which those are achieved or rather the rationale behind them is not clearly explained.

2.4.2 Reputation-based Methods

Reputation-based methods are those where the focus of the model is on generating reputation information for the nodes in the network and in this respect, they all meet Requirement 5 (calculating reputation).

Schweitzer et al. (2006) describes a mechanism for propagating trust and consolidating it within ad hoc networks. This involves building trust relationships and making the entities autonomous so that they are able to make decisions without referring to a central network. Nodes within the network therefore request trust information about an unknown entity from other nodes that they already trust within the network. Based on the recommendations received from the trustworthy nodes, the requester can then update and consolidate its current trust rating for the new unknown entity. These functions additionally meet Requirements 3 (reasoning), 4 (calculating trust) and partially 9 (constant evaluation).

Any change that occurs within the network for a given trust rating is propagated throughout the network so that tables are updated. There are three basic operations for this to occur which are namely simple, transitive and consensus operations. These operations are largely based on the work of Dempster and Shafer (Smets, 1990), Martucci (Martucci et al., 2004) and Josang (Josang et al., 2003).

The method in this case therefore makes a few assumptions:

1. The recommending trustworthy nodes have some trust information about the new entity. However, despite the fact that for n entities within the network, there will be $(n\text{-}1)^n$ searches for trust information, the latter may not always be available. This limitation particularly highlights the fact that this trust mechanism, although it is able to propagate and consolidate trust, is not able to generate trust information from scratch without having recourse to existing recommendations.

2. Furthermore, the method does not take into account that trust information can decay over time. While trust changes are propagated throughout the network, natural decay is not taken into consideration. Even if this were to be implemented using the same propagation method, it could increase the overhead dramatically within the network, depending on how often trust refresh cycles are performed.

3. To achieve security within the mechanism and increase confidence in the authenticity of a recommendation, each recommendation must be signed using a certificate issued by a recognised and known CA. The authors further recommend that such information be sent over a secure encrypted channel. While the above statements make the mechanism of trust propagation more secure, it nevertheless reduces the importance of such a mechanism to a mere security-enhancing feature. If encryption, CAs and certificates were to be readily available, then the existence of a trust mechanism on top of that is superfluous as relatively robust security can be obtained via existing protocols that have been well defined within the wireless security literature.

The method therefore does not satisfy the requirements of an autonomous trust mechanism as it is unable to operate independently from other existing options such as encryption and CAs. The loose definition of trust as merely a person's knowledge and confidence in another user's behaviour and reputation, also overlaps with the authors' notions of security.

Another trust model that uses trust metrics, this time in terms of predefined trust levels is the work by Liu et al (Liu et al., 2004) where the definition of trust is taken as the "reliability, timeliness and integrity of message delivery to their intended next-hop". The work by Liu et al proposes a trust model that is based on the update of trust levels throughout a given ad hoc network by the use of Intrusion Detection Systems (IDSs) that are installed on all nodes operating in the network. Reports from IDSs, generated after suspicious activity is detected, then flood the rest of the network by means of trust reports, thereby causing all nodes to update their trust levels for given suspicious nodes. The target application here is the calculation of more secure routes when transmitting messages. Source nodes are able to use trust levels as a guide to that effect.

The trust levels used in this work are discrete and are therefore only very broad representations of actual reputations.

Ranging from compromised to highest, the trust levels are not precisely defined in semantic terms. While the authors term their detection systems on the nodes as Intrusion Detection Systems, those systems are also expected to reward nodes for good behaviour by increasing their trust levels. For initialisation, nodes are expected to have been pre-authenticated. Those who have not been authenticated are assigned trust value of "Unknown" while at the same time being allowed to enjoy the same privileges as nodes in a higher trust category. In fact, the discrimination in the various categories is unclear to the extent that in real terms, the system boils down to a two-category trust level scenario: compromised and trustworthy.

A third method that uses reputation is by Rebahi et al. (2005). Reputation is defined simply as "the perception that a node has of another's intention and norms."

Reputation is achieved by means of a module called the Trust Manager that sits on top of every node and monitors packet forwarding. There are four functionalities as detailed in Figure 2.6.

There are several limitations in this model which are detailed as follows:

Reputation is mainly achieved by monitoring packets forward versus packets sent. While previous reputations are catered for, the time factor is not taken into account and the natural decay of trust is not considered. In fact, the reputation concept is very vague and the notion of confidence in a reputation does not arise. Any node can monitor any other of its 1-hop neighbour and distribute reputation information. This makes the system very weak against colluding attacks.

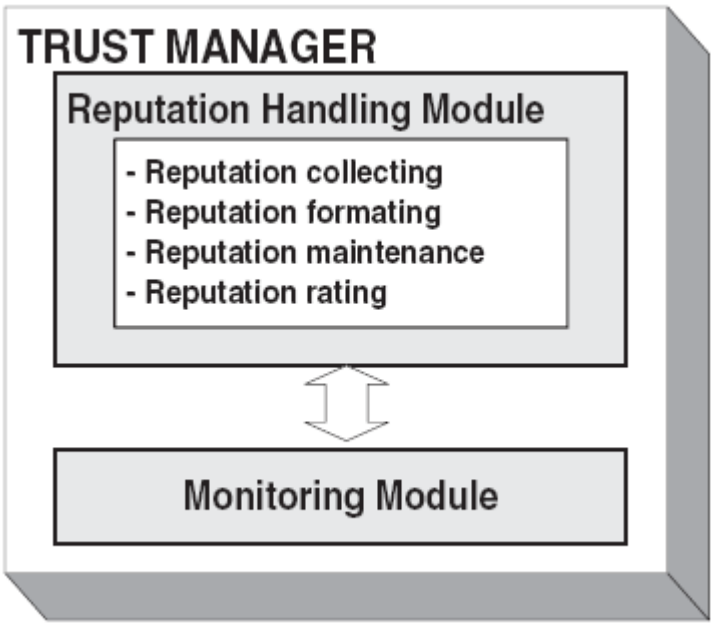

Figure 2.7: The Trust Manager: a trust module including four functionalities monitoring packet forwarding (Rebahi et al., 2005)

Furthermore, reputation is defined as any number between 0 and 1. Yet, when new nodes are introduced to one another, they are assigned a default value of 1, which is the highest possible. In other words, new nodes are being treated as fully trustworthy by the system whereas existing nodes may have to work to earn privileges to operate. Because reputation depends solely on packet data, nodes that do not forward packets are not treated as misbehaving, in other words, uncooperative nodes are not detected or punished. Collaboration is assumed at all stages of the process, from normal routing to malicious packet dropping.

There are other methods within the literature that utilise the concept of reputation. Dewan et al. (2004) utilise reputation to route packets in terms of the most

trustworthy routes as opposed to using the usual shortest path algorithms typically found in most routing mechanisms.

The idea of trust-based routing is also shared by Pirzada (Pirzada & McDonald, 2004). However, in this case, the definitions of trust are rather more explicit. Trust is quantified by assigning weights to specific events being monitored. Weights are assigned by each node based on their own criteria and circumstances. Aggregate trust levels are determined from individual trust values of events for a specific node, based on the different weighting systems. The weights in this case have a continuous range from 0 to +1 to represent the significance of a particular event. The authors do not specify how these weights are assigned, only that they vary based on the type of application, state of the network, time and the node's own criteria and circumstances. There is no predefined standard way in which weights can be reliably assigned to different events. This results in a large number of permutations and unless some consistency is proven across the mechanism, the trust values could fluctuate uncontrollably between its defined bounds of -1 and +1.

Two other methods relying on reputation are by Liu & Issarny (2004) where reputation is represented numerically between -1 and +1 and Yan et al. (2005) who utilise experience statistics.

All these methods meet Requirements 1 to 5 with Rebahi et al.'s model also meeting Requirement 9 (constant evaluation).

2.4.3 Directed Graph Modelling

The authors Theodorakopoulos and Baras (2004) define trust here as a set of relations between entities that participate in various protocols. The focus here is on the evaluation of trust evidence, dealing more specifically with the trust metric itself (Requirement 4).

In particular, collection of evidence from the network, as well as communication and signalling overheads, are not taken into consideration. The evaluation process here is basically a "weighted, directed graph" The definitions are that nodes represent users and edges represent the direct trust relations weighted by the amount of trust a given user A places on user B. All interactions are local. Because each user only has direct relations with other users it has previously interacted with, the process of establishing a trust relation between two users that have not previously interacted is achieved by using the direct trust relations of intermediate nodes.
Nodes form opinions of others and this is represented on a weighted directed graph.

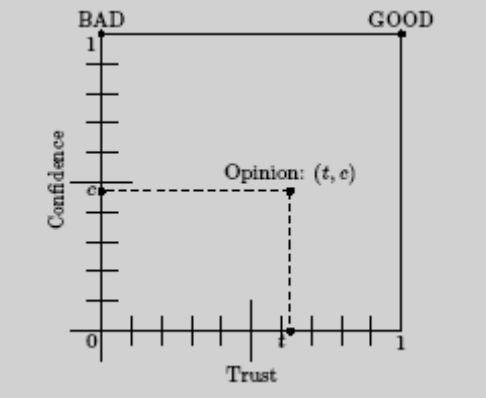

Figure 2.8: Opinion Space showing the formation of opinions from nodes, derived from trust and confidence values (Theodorakopoulos and Baras, 2004)

Each opinion in this case consists of the trust value and the confidence value. The trust value represents the trustworthiness of a given node and the confidence value is defined as the belief that a node's public key belongs to it.

This method is useful in the sense that it does not require any centralised infrastructure in order to work. Users merely base their opinion on second-hand (or third/fourth) evidence in intermediate nodes.

2.4.4 Statistical Models

Statistical models of trust are usually based on a Bayesian approach (Buchegger & Le Boudec, 2004) that proposes the theory for the trust framework. A recent model for the ad hoc network based on such a theory is by Li et al (2007).

The trust management framework is proposed to be more objective than those that merely utilise reputation or trust based methods. They propose to use second-hand information to evaluate trustworthiness as opposed to relying only on direct observations. Along with meeting Requirements 1-5 and 9, this implies that such a method also utilises additional information in order to pursue its goals (Requirement 12).

The formula used for Bayesian approach is the following

$$p(B|O) = [p(O|B) \square p(B)] / Normalization\ Factor \quad (2.1)$$

where *B* means belief and *O* denotes observation. *p(B)* is the prior probability density function for θ, *p(O|B)* is the likelihood function and *p(B|O)* is the posterior distribution function for θ where θ is the probability with which the subject node expects the object node to behave. However, because the framework of this thesis focuses on simple probability functions derived from Gambetta, further analysis of the Bayesian theory is redundant at this stage.

The framework follows a four-step method θ which includes:
1. Providing direct information to a central trust computing module
2. Collect and process second hand information via the central trust module
3. Evaluate a trust and confidence value based on direct and second hand information
4. Evaluate trustworthiness from trust and confidence values.

Although this method uses Bayesian theory as a differentiation from other conventional trust models, the claim that their framework is objective based purely upon second hand information is not warranted. This is because “second hand

information" is implied to be merely recommendation information from other nodes. The term is not clearly defined within the work but the implication is quite clear. This makes this framework no different to others that utilise reputation as a method because by its very definition, reputation is made up of direct observations and recommendations (classed as second hand information here).

The framework nevertheless also includes a way to mitigate attacks by misbehaving nodes and therefore also fulfils Requirement 10 (isolation/banning of nodes).

2.4.5 Group-based Models

This type of model is designed to create sub networks within the overall framework based on different criteria. These sub networks are often known as groups or clusters (Boodnah & Poslad, 2009).

Clustering in ad hoc networks has been proposed before for ad hoc networks and there are several existing ways in which it can be done, including methods to elect cluster heads. They are based on different properties of the nodes at a given time. For example, the connectivity-based algorithm from Gerla et al. (1995) where a cluster head is elected based on the node having the highest degree that is computed from the number of unique identifiers received from the network. Other older methods such as the *Lowest-ID* algorithm (Baker & Ephremides, 1981) are more simplistic whereby the node with the lowest assigned ID is elected cluster head.

More recently, methods have been proposed using criteria ranging from the distance between nodes (Er et al, 2004) and weight-based distributed clustering algorithms (Chatterjee et al, 2002; Kadri et al., 2007) to geographical positioning (Seunghun et al., 2005). However, aside from the model by Seunghun et al. which uses certificate references, none of these methods and models promote the creation and distribution of trust.

Peng et al. (2008) propose a voting-based clustering algorithm that provides subjective trust and stability. The subjective trust from the node is accessed using a Bayesian method and stability is evaluated by monitoring the neighbour change ratio and the remaining battery power of the nodes. Subjective trust and stability are the two criteria via which cluster heads are elected. Although this is not a trust model per se, the clustering algorithm is very relevant and of interest to the work in this thesis.

Aiguo et al. (2008) also propose a cluster-based trust model for ad hoc networks. The aim is to create inter-cluster trust by dividing nodes into clusters. They define one special node (as per Requirement 7 – classes of node) in each cluster to establish trust relationship based on previous transactions. They claim the model is scalable meaning it would meet Requirements 1 to 5, 8 (local repositories) and 11 (selective grouping). What the authors define as inter-cluster trust is really intra-cluster trust as the method relies upon the cluster head computing and storing all reputation information for all the nodes within the cluster. However the method fails to mention how the clusters are formed, nor how the cluster heads are elected. There also does not seem to be any safeguard with respect to cluster head violation. All routing also appears to follow a cluster-based routing pattern that would imply that cluster heads would get overloaded with both trust and normal routing traffic. Unless those cluster heads belong to a different class of device, they are bound to have their limited resources decrease very rapidly. As such, cluster heads could be the very first nodes to have to leave the network purely on operational reasons. Should this happen, then that particular cluster would just introduce appreciable traffic to the network instantly as there will be a group of nodes seeking to establish new trust relationships at the same time.

Park et al. (2008) further extend the work of Seunghun et al (2005) in clustering by proposing a cluster based trust model against attacks in the ad hoc network. The method involves designating the node with the highest trust value as Head (Requirement 7) and this node is then responsible for issuing certificates to the rest of the cluster. When a node moves, the Head transfers the certificate the adjoining cluster. This feature means that the model also covers Requirement 14 (migration of

nodes). The model has the same limitations as Aiguo et al. as the cluster formation is handled in a similar manner.

Furthermore, this method also attempts to detect misbehaving nodes (Requirement 10). A process-based method is proposed where a node has to undergo through a series of steps when they enter the network. Misbehaving nodes are classified as those nodes that do not meet a certain threshold of trust and which do not possess a certificate. However, based on the process proposed, if a node does not have a certificate, there is no way for it to be able to get one issued and as such it remains in the loop of constantly being checked whether or not it is a misbehaving node (even though it may have passed the trust threshold). This is counter-productive and may result in a waste of precious resources. Misbehaving nodes are also only detected at the point of entry. There is no constant evaluation of nodes (Requirement 9) and nodes with a certificate (which may be forged) appear immune from the misbehaving node detection scheme.

2.5 Chapter Summary

This chapter has reviewed the ad hoc network concept and examined the various issues that are encountered within it, more especially with respect to trust formation.

A quick literature review of trust within the social domain was conducted in order to understand the origins of the various facets of trust formation.

The various requirements extracted from the issues in trust in ad hoc networks offer a good basis on which to build the ideal trust framework. No framework is ever ideal as the very engineering concept itself is mired in trade-offs and compromises. However, the thesis of this research is to be able to present a model that is as close to ideal as can be without impacting the practicality of network itself and its proper function.

The models surveyed in Section 2.4 provided a good indication of the state of the art in this research area. A summary of how they fared with respect to the Requirements proposed can be found below. Where relevant, commentary has also been added as to where the framework proposed in this thesis will aim to improve the state of the art.

Requirement 1 (decentralised network) – Decentralised models are a feature of ad hoc network trust models in more cases than not. As such, only 1 of the models reviewed did not conform to a decentralised network. The framework proposed in this thesis will be using a distributed model as well.

Requirement 2 (scalability) – Scalability is a feature of models proposed in Sections 2.4.1, 2.4.2 and 2.4.4. This is a feature that will be retained in this thesis going forward.

Requirement 3 (reasoning) – Trust reasoning is exhibited primarily by models using reputation models and is present in Sections 2.4.2 and 2.4.4. It is a desirable feature that will also be retained.

Requirement 4 (trust information) – This forms the basis of trust computation and as such is present in distributed models surveyed in Sections 2.4.1 - 2.4.5.

Requirement 5 (trust recommendation) – Reputation is another very important measure in trust frameworks and therefore features heavily in Sections 2.4.1 – 2.4.5.

Requirement 6 (reward or payment scheme) – This is a requirement that has not been explicitly noted in any of the models surveyed. It is an important criterion to consider as given the choice, nodes may very well be indifferent as to whether or not they provide a reputation or a trust value. This is especially so for models that exhibit classes of nodes whereby some nodes are required to work more than others. Therefore, this is a requirement that will be included in the model featured in this research.

Requirement 7 (classes of nodes) – Models in Section 2.4.5 using clustering define a second class of node although how they are elected or how the groups are formed is sometimes ambiguous. The model in this thesis also includes classes of nodes as it is viewed as presenting a number of benefits, especially in trust formation and maintenance.

Requirement 8 (repositories) – Ignoring repositories present on all the nodes and central repositories inherent in systems with CAs, only the models in Section 2.4.5 include the storage of data at various parts in the network as this tends to be a task of the head of a cluster by default. It is an important aspect of the network that will be retained going forward.

Requirement 9 (constant evaluation) – Only models in Sections 2.4.2 and 2.4.4 present this feature. While some of the models in Section 2.4.5 have looping mechanisms for misbehaving nodes, there is no explicit evidence that the trust information once calculated is refreshed periodically even though no new interaction occurs. This is an important distinction, as the temporal signature of a trust value must not be ignored.

Requirement 10 (isolation) – Models that take measures towards misbehaving nodes generally include this requirement. They are mostly found in Sections 2.4.4 and 2.4.5. Going forward, this is also a feature to be retained.

Requirement 11 (selective grouping) – Groups are associated with different classes of node and are exhibited by the same models as per Requirement 7. Clusters will be an important feature of the framework in this work too.

Requirement 12 (using external information) – This is partially seen in Section 2.4.1 with Rebahi's doctrine and also to a certain extent in Section 2.4.4 although the external information in question is implied to be merely additional recommendation information, so not external to the network as such.

Requirement 13 (contextual aspect of trust) – This aspect of trust is not strictly defined in any of the models surveyed. However, as explained previously in Section 2.2, the context for trust generation within an ad hoc network tends to be standard. However, it is an ideal feature to have and promotes versatility in the network.

Requirement 14 (enabling migration) – Node migration is seen in Section 2.4.5 where one model stipulates that its Head of a given cluster is able to transfer the information to another Head when a node changes its geographical position. However, the trust relationship between Heads is not explored and it can be limiting to assume that all Heads are inherently above reproach.

The following chapters will now present the model generated by this research. The specifics of the model will be presented in Chapter 3 and the model will be shown to fulfil Requirements 1-11 and 14. Requirement 13 will be addressed theoretically and ways to satisfy Requirement 12 will be put forward as future work in the closing remarks of this thesis.

Chapter 3

3 FRANTIC: A Framework for Ad hoc Networking Trust using Interacting Clusters

One of the key aims in modelling trust in an ad hoc network is to remove some of the uncertainty surrounding interactions among nodes within the network. A framework provides a means to not only model the trust relationships that exist between nodes but also to significantly reduce the level of uncertainty that is inherent when one entity has to make a decision about another.

Therefore, the aim of this chapter is to present a model that meets the broad objectives set out in Chapter 1. The motivation for drawing up these objectives were made clear in Chapter 2 where the issues existent within ad hoc networks, and establishing trust within them, were presented.

FRANTIC is a computational trust model that determines the trust relationships between nodes based on the history of interactions (Objective 1). It also takes into account recommendations from other nodes in order to create a reputation for the entity in question (Objective 2). For operational and trust reasons, the framework architecture adopts a clustered approach (Objective 3). This clustered approach, amongst other advantages, allows for nodes with a good performance and a higher responsibility to be rewarded and for their trust recommendations to be considered ahead of others (Objective 4). Finally, FRANTIC adopts measures in order to introduce resilience in the framework with regard to non-cooperative and misbehaving nodes (Objective 5) while striving to keep the interaction overhead to a

minimum (Objective 6). Aside from these broad objectives, FRANTIC addresses the other issues mentioned in Section 2.5 via its operational measures.

The rest of this chapter provides an overview of how FRANTIC operates. Section 3.1 describes the types of behaviour that will be modelled by FRANTIC and how trust is generated. The calculation of direct trust derived from the observed behaviour will be described in Section 3.2. Following this one to one interaction, Section 3.3 addresses the dependence on others for recommendations and for reputations. The architectural set-up of the framework and clustering are considered in Section 3.4. Finally Section 3.5 proposes how the framework deals with misbehaving and selfish behaviours.

3.1 Modelling from Behaviour

As defined in Chapter 1, trust is the expectation of an outcome from a trustor towards a trustee. Furthermore, the definition provided indicated that to trust an entity meant to expect honesty and reliability from it. Honesty is related to the truthfulness of the entity while reliability is the expectation that the displayed honesty will be repeated over further interactions in time. The expectation of a positive outcome from a party that is honest and reliable can only be predicted if the past behaviour of that party is known. The role of a trust model is therefore to capture both the reliability and the honesty of the agents.

There is no easy way to measure this reliability and honesty other than through direct observation of behaviour and via the sharing of similar information from other peers. In order to a trust model to achieve this, such observations need to be carried out over several interactions. Each additional interaction enhances the precision of the opinion being formulated by the querying peer (the trustor). These interactions build up in order to provide a measure of trust that can then be integrated in the model.

The monitoring of peer behaviour is not a new concept. In fact, it has always been present in the way humans interact and form friends or ignore enemies. However,

human behaviour and social models ultimately represent degrees of complexity that cannot be accurately represented in a computational model. The latter usually operates within a prescribed scope and context.

So that the trust model in this work is able to distill behavioural information and translate it into trust, a level of abstraction is required so that the computational model may mimic a human model. The two concepts that need to be investigated are honesty and reliability. Reliability is easier to monitor because it can be concretely measured based on past performance and by monitoring performance. Honesty on the other hand is harder to capture. The truthfulness of an entity is largely down to its own reasoning faculties. This character is usually representative of more complex agents. For the purposes of this work, it is assumed that honesty accrues over time logically as an entity displays behaviour that is deemed to enhance its trustworthiness. Assuming that the peers in the model are autonomous and have no inputs from their human user, this is a reasonable assumption.

The key aspects to be decided therefore are what the scope of the framework will be and within what context it will operate, that is, what types of behaviour will it monitor. The behaviour being displayed needs to be simple enough that its observation can lead to lean computation of trust information but complex enough that other inferences about the state of the network can be deduced in order to maintain its operation (e.g. whether it is under attack, which are the non-cooperating node amongst others). The behaviour is largely dictated by the scenarios for which a framework is implemented. The scenarios also dictate the level of complexity of the actions performed by peers and the subsequent observation of their behaviour. For a generic framework however, inferring behavioural information from processes taking place at the network layer is a viable alternative. Not only are these processes always present in any type of network, but by utilising network data that is already present, the framework can also present the added advantages of being lean and not introducing appreciable overhead.

In order to judge the behaviour of its peer, a requesting entity needs to define its own measures of what a successful interaction is. This definition needs to be part of the

framework. The successful interaction itself is dependent on the action being performed and therefore the expectation that said action is performed properly to the requesting entity's satisfaction. Other aspects of behaviour that need to be noted for the purposes of a trust model are indications of altruism (the need to do good and be cooperative) and selfishness (non-cooperation). Incentives and disincentives must be put in place via system-level trust (See section 2.3.2) in order to "force", where possible, the peers to behave in a certain way.

The basic data at the network layer is routing data. Route requests and discoveries are always ongoing in an ad hoc network. Using this information in order to infer trust measures, especially at the early stages of the formation of the trust model, bears its advantages. By monitoring routing as a behavioural measure, it could be argued that the system is solely dependent on the performance of the peers and therefore trust is only a measure of reliability which itself is derived directly from the performance of a peer. This is true only at the trust formation stage where the main purpose of the network is route discovery and network formation.

Once the network is established, peers within the network can make choices about the type and amount of data they send, they forward for others and they receive. A feature of the ad hoc network is that in order for a transmission to go from a point A to a point B, there may be several routes involved, each of which may involve a different number of peers. Each of their peers at the route request stage has the choice of sending a route reply acknowledgement when it receives a request for a route. Its reply may be to either accept the route or reject it. The choice this peer has adds a level of complexity. By accepting or refusing a route request, it is exhibiting properties of altruism or selfishness, both of which affect its trust rating in the trust framework used in this work. If the acceptance/refusal of the route was automated and as a result of prevailing network conditions or even the physical state of the peer, then it would simply equate to the notion of performance. However, if there is a "conscious" decision by the peer to accept or reject requests in order to promote its own agenda, then this can no longer be considered as purely performance. It is indicative of the "conscious" state of the peer and would therefore affect its trustworthiness by a measure different to that of simple performance.

Although routing data is an initial source of data for observing the behaviour of peers in the framework, other data being transferred can be just as easily adapted within the framework. From the system-level trust, there are certain rules that can be modified in order to adapt to the data being transmitted. When operating in a real world scenario, the concept of context also becomes relevant as the needs of various peers will vary and the type of data and content requested will also vary. Hence the trust values generated will need to be in line with the context in which they happened. Because the formalisation of the model is based on a single context for simulation purposes, multi-context scenarios are not addressed here. However, the scope to add context information to the trust data is present. This is addressed further in Chapter 5.

When judging success, peers in the framework are presented with two outcomes: positive and negative. Each outcome is added incrementally to the series of outcomes an entity stores for another. These outcomes and their success/failure are then converted into a measure of trustworthiness by an algorithm in the framework. Assuming completely untrustworthiness to equate to 0 and full trustworthiness to equate to 1, then it can be assumed that the trustworthiness of any peer in a network will lie on the spectrum of values ranging between these two extremes.

The outcome of every interaction in the network is recorded. Depending on the type of interaction and the action performed, they each will impact the trust rating of the peer differently. For example, by showing altruism, a peer can expect a positive impact on its trustworthiness. Conversely, refusing to cooperate within the network will result in a negative impact. Algorithms within the framework then combine these different impacts in order to deduce that measure of trustworthiness that is then passed on to higher nodes such that an overall reputation value can be computed.

Therefore, it can be seen that the analysis of behaviour by the framework can be very flexible and it can be easily tweaked in order to adapt to a given scenario. The measure of behaviour ultimately depicts a measure of trustworthiness.

3.2 Modelling Trust

There are three stages in the calculation of trustworthiness by the framework. They are:

1. Gathering trust information: this involves member nodes storing behavioural outcomes for their immediate 1-hop neighbours after each successful/failed interaction

2. Conversion: This behavioural data is then converted into a trust value by means of an algorithm within the framework. This trust value is known as direct trust.

3. Calculating reputation: Having acquired a set of direct trust values for a given peer, the framework can combine these in order to produce a more balanced representation of a node by calculating a reputation value (See Section 3.3).

3.2.1 Calculating Direct Trust

In the model, in order to calculate trust, several new concepts are introduced: *gain, workload* and *drop rate*. Using routing information as the basis for generating trust information is not overly complex. However, more complex inferences can be made based on just this simple behaviour (see Section 3.1) and trust values can be adjusted accordingly to reflect those inferences. For example, balancing network traffic across an ad hoc network is a desirable feature to have, if not crucial at times. It does not have a direct impact on trust but a well-balanced traffic load will enable all peers to perform to the ability they choose to, rather than being congested and having to refuse or drop packets. The ideal situation is for all decisions made by peers to be fully intended and not as a result of network outages (as even these affect trustworthiness). This would present the most accurate depiction of trust.

Consequently, a node that creates traffic for the network via route requests can have a negative connotation attached to it whereas another that selflessly forwards the

routing packets can be deemed to have a positive connotation. A node receiving a packet can also have a negative connotation because it benefits from the work done by other nodes within the network. While these "connotations" are not the sole representations of behaviour, taking them into account just makes the trust value calculated for an entity that much more relevant and fair.

Connotations are therefore there to aid the fairness of the framework. In the example of the receiver node, it can be inferred that is accepting a packet from the network. In order to receive this packet, other nodes would have had to utilise their own resources so as to create a route for its transmission. The receiver node on the other hand only receives a benefit (the packet) from the network without having to input any resource. Therefore, the framework will deem the acceptance of this packet as a negative connotation, and this will have a slight negative impact on its trust rating. The goal here is to make sure that nodes do not ask the network to perform tasks that may not be necessary as any work that they ask of their peers will result in having an impact on their own trustworthiness. Similarly a positive connotation also introduces an aspect of a reward for a selfless node within the network. These connotations will become clearer with the following calculation of direct trust.

Figure 3.1 shows a typical routing path as adopted during a transaction.

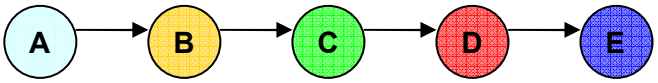

Figure 3.1: Routing through nodes with information transfer running from A (source) to E (sink) and B, C and D acting as intermediaries (routers)

Node *A* is the source and initiates a packet transfer to node *E* (sink) with *B*, *C* and *D* acting as intermediary nodes to route the packet.

The following behaviours are observed:

1. A node that initiates a packet transfer or receives a packet, i.e. nodes which are either sources or sinks, gain from such a transaction, the former by transmitting information, and the latter by receiving such information. In other words, they form the network work for their own benefit. This *gain*, denoted as *G*, is calculated by the following equation:

G_A = number of packets initiated by *A*.

This information is obtained by all nodes within the chain each time a route request is generated by *A*.

Similarly,

G_E = number of packets received by *E*.

This information is obtained during the route reply process.

2. Each time a node that is not a source or a sink forwards a packet by being a router node it is performing work for the network and therefore has a *workload* termed *W*.

W_B = number of packets forwarded by *B* (same principles apply for *C* and *D*).

3. When a node drops a packet for any reason, this is known as the *drop rate*, *D*.

D_B = number of packets dropped by *B* (or *C, D* or *E* as per the above example).

At any point in time, depending on the routes being requested, there will be several permutations of the above example with the roles of sources, routers and sinks changing intermittently such that all nodes will eventually have a combination of *gains*, *workloads* and *drop rates*.

So, taking an example of node *B*'s view of node *A*:

$$P_{AB} = \{[W_{AB} - (0.5*G_{AB} + D_{AB})] / 2*S_{AB}\} \quad (3.1)$$

where P_{AB} is the overall *Performance* of node A as seen by node B, G_{AB} is the number of packets received by B from A, D_{AB} is the number of packets dropped by node A that were received by B and W_{AB} is the number of packets forwarded by B for A.

This is an indication of how node A has fared with respect to its overall input to the network. By being altruistic and performing work for the network, A will have a higher W_{AB} factor than its G_{AB} or D_{AB} factors (assuming it is not overloaded and drops packets purely due to congestion – even if this were the case, then this will have a self-regulatory effect by having packets routed away from A temporarily during any decline of its *performance*). S_{AB} simply denotes the overall number of packets that transited through A (i.e. when operating as a source, router and sink inclusive) as measured by B. S_{AB} will be determined by the time over which the observation takes place. This time period is configurable in the framework and set depending on the level of activity of the network itself. It cannot be set within the above formula as this time period will be often determined by the higher nodes' request for trust information (see cycles in Chapter 5). Networks with a high latency and low interaction may require a greater amount of time in order to generate enough packets so that an accurate representation of *performance* can be determined.

Because the gain is only a negative connotation, the effect of the *gain* is deemed to be less serious in terms of its effect on the network than the *drop rate.* Therefore its effect is halved (*gain* factor of 0.5) in the above equation. This halving is arbitrary and can be modified to any other value depending on how relevant it is at a particular moment in time. Where a network is heavily congested, the *gain* factor may be raised to a higher value in order to dissuade nodes from introducing non-essential traffic to the network. Similarly, at periods of low utilisation, the *gain* factor could be lowered to encourage participation. Such traffic shaping, although not an objective of the trust framework, is a welcome by-product of its implementation as it aids in promoting a fairer network.

The *workloads, gains* and *drop rates* are divided by twice the total number of packets so as to provide a maximum possible *performance* of 0.5. This is important because the *performance* of a node is also proportional to the increment by which a node's initial trust value is adjusted. Given that a node's maximum trust value is 1 and its minimum value is zero (with 0.5 as the middle initial level for unknown nodes), it is important that its trust value does not converge to either extreme too quickly. This convergence will depend on the amount of interactions taking place between nodes. In a network where S is high, then the effect of the *performance* must be adjusted accordingly.

The direct trust value of a node is therefore calculated from the following equations.

$$\text{If } 1 \leq S_{AB} < 10, \text{ then } T'_{AB} = \{[(P_{AB})^5 * T_{AB}] + T_{AB}\} \quad (3.2)$$

$$\text{If } 10 \leq S_{AB} < 25, \text{ then } T'_{AB} = \{[(P_{AB})^4 * T_{AB}] + T_{AB}\} \quad (3.3)$$

$$\text{If } 25 \leq S_{AB} < 50, \text{ then } T'_{AB} = \{[(P_{AB})^3 * T_{AB}] + T_{AB}\} \quad (3.4)$$

$$\text{If } S_{AB} \geq 50, \text{ then } T'_{AB} = \{[(P_{AB})^2 * T_{AB}] + T_{AB}\} \quad (3.5)$$

where T'_{AB} is the new value of trust of B in A and T_{AB} is the initial value.

The basis for these formulae is in order to achieve suitable convergence based on the numbers involved in the typical scenarios depicted in Chapter 4 and the simulations performed in Chapter 5. Understandably, this is just one way of achieving convergence and the values above can be modified in order to increase or drecrease the amount of time convergence happens and thus the sensitivity of the network in reacting to behavioural inputs. Convergence is made to be gradual so that the trust build up process also takes into account the longevity of the node in the network – it takes a minimum amount of time for the trust values of nodes to reach certain target levels (for example to be promoted to a different class of node).

If a generic model were used, then convergence could be achieved by using n as a constant and using equation 3.6 to achieve a similar solution.

$$T'_{AB} = \{[(P_{AB})^n * T_{AB}] + T_{AB}] \quad (3.6)$$

where n is a constant derived from the number of packets and the packet rate. If S_{AB} is high due to a large number of packets flowing through in a small time interval, then n can be adjusted to take this into account. This can be fully configurable depending on the application in use and how fast convergence is required. One way to derive n would be as per equation 3.7.

$$n = S_{AB} / z * Q_{AB} \quad (3.7)$$

where z is a constant to be chosen by the designer and Q_{AB} is the packet rate.

By using the above rules, the model also makes sure that the trust value of a node is not incremented artificially by demonstrating an excellent *performance* without having performed an adequate *workload* within the time frame, since *performance* is, by definition, a ratio. It also encourages nodes to maximise their *workloads* by forwarding a maximum number of packets because of a higher improvement in measures of *performance*. These rules are nevertheless customisable should the need arise, depending on how busy the network is, as described earlier.

It should be noted that the maximum value of T is rounded off at 1 and the minimum value is rounded off at zero. This means that any trust value that happens to be ≥ 1 is then rebased to 1 just like any trust value that is ≤ 0 is rebased to 0. Accumulated "credit" is not stored. For example, a node that finds itself with a trust value of 1.45 that is rebased to 1 does not have the extra 0.45 stored on the system to be offset against any future reduction in trust. This is to ensure that all nodes perform adequately throughout the existence of the network as opposed to being very productive at one instance and latent at another.

Once a node has obtained its new value of Trust, *T'*, the value is stored with the following information:

{node_id of trustor, node_id of trustee, T', t}
In the above example, the data that would be stored would be as follows:

node_id of trustor	***node_id of trustee***	***T'***	***t***
B	***A***	T'_{AB}	***t'***

Table 3.1: Data set for trust showing the different types of information collected within the node

t is a time-stamp that allows the user of the table to distinguish recent values from old ones. In the above example, if no traffic flowed though *A* and its trust value remained unchanged, then the above information would be sent with a time stamp of *t*, not *t'*, thereby defining the time at which said trust value was current.

3.3 Modelling Reputation

Having laid down the basis for a calculation trust on a one-to-one basis, the focus of the next section will be on how to deal with the concept of reputation. The need for reputation information arises when an entity has no experience of another entity it is about to interact with.

Using the reputation of the entity provides an appropriate solution. A reputation is simply the aggregation of opinions about a target entity from a variety of sources. This reputation is calculated by the requester based on the trust information it receives from other nodes in the network. This makes reputation a more subjective notion since it will be up to the requesting node to determine whose recommendation it chooses to peruse and how it assesses each of those recommendations in order to reach a final reputation score.

3.3.1 Calculating Reputation

For every recommendation a node receives from another node, it keeps the following information:

{node_id of trustor, node_id of trustee, trust value of trustee, time-stamp t}
A node may elect to keep a table for each entity it interacts with. Every time a recommendation is received for that entity, the recommendation is added as a row to that entity's table. Using the same notations as in Section 3.2.1, a reputation R for a given node X is usually the statistical average of the various recommendations provided by other nodes. This is denoted by equation 3.6.

$$R_X = (\Sigma T_{XY}) / \delta \tag{3.8}$$

where Y is any node which produces a recommendation for X. δ is the number of nodes that provide a recommendation.

This statistical average provides the reputation for X. However, this is an over-simplistic way to provide an accurate reputation and moreover would fail. Objective 4 as defined in Chapter 1 states that better performing and more trustworthy nodes must be considered more important than their peers. This gives rise to the concept of *importance,* denoted by I. Because importance would be directly proportional to the actual reputation of the recommending node itself, it is logical to use the reputation value of the trustworthiness as the *importance* factor of its recommendation. Thus, nodes with a higher reputation would see their recommendation affect the reputation of the target node to a higher degree.

This *weighted* average can be calculated as follows.

$$R'_X = [\Sigma(T'_{XY1} * I_{Y1} + T'_{XY2} * I_{Y2} + \ldots + T'_{XY\delta} * I_{Y\delta})]/\beta \tag{3.9}$$

where I_Y is the *importance* of Y and is simply defined as the actual reputation of Y at that point in time and β is the sum of all "*importances"* from Y_1 *to* Y_δ.

This weighted average is one way of mitigating the effect of false recommendations from misbehaving nodes as they would normally have low reputation scores themselves. The amount of time, effort and resources required for a misbehaving node to build up its reputation sufficiently such that it would be able to distribute false recommendations that would have marked impacts, acts as a deterrent. Misbehaving nodes are often opportunistic and the likelihood is that they would potentially move on to softer targets, thus reducing the likelihood of attacks on the network

In the unlikely event of a concerted attack on the network (where other softer targets may not exist, thus increasing the incentive for misbehaving node to attack only the available targets), the system includes a rule-based mechanism in order to eliminate non-cooperative and misbehaving nodes (See Section 3.5.1). While it is acknowledged that a misbehaving node may not be detected while it is operating covertly by behaving as required in order to build its reputation, it can be argued that the work done by misbehaving nodes in building up reputations offers some benefit to the network for instance by offering more available paths. It is only once the paths are broken that the system reacts in order to exclude any malice.

3.4 Clustering

One of the basic features of FRANTIC is that it implements a cluster-based architecture. The benefits of clusters are well-known from a networking point of view as they are analogous to the concept of super-peers. A trust framework can also benefit from using such an architecture.

Clustering generally involves creating small aggregations of nodes within a particular geographical area. These small aggregations are then “held together” by a membership that is reflected in a single node called the cluster head (CH). The CH acts as an authority figure for the cluster and is generally the node that has existed in the cluster for the most amount of time and therefore also has the highest reputation of all the nodes.

3.4.1 Initial Motivation

Related traffic simulations using uniformly-sized public key certificates in order to monitor additional overhead highlighted the advantages of using a network created by using a clustered approach versus a fully distributed one. The following graphs display the results obtained (Boodnah & Scharf , 2004).

In this experiment, the main aim was solely to establish the total traffic, therefore an indication the total bandwidth consumed by the trust mechanism on its own. Public key certificates were used purely as a measure of simplification as this study was for overhead measurement only and not for the generation or distribution of trust or reputation.

The system was not brought into any form of congestion to prevent dropped packets and the duplex links were set at an above-threshold bandwidth. Nevertheless, the system was also set to exhibit the worst-case scenario in order to determine the maximum possible bandwidth consumption.

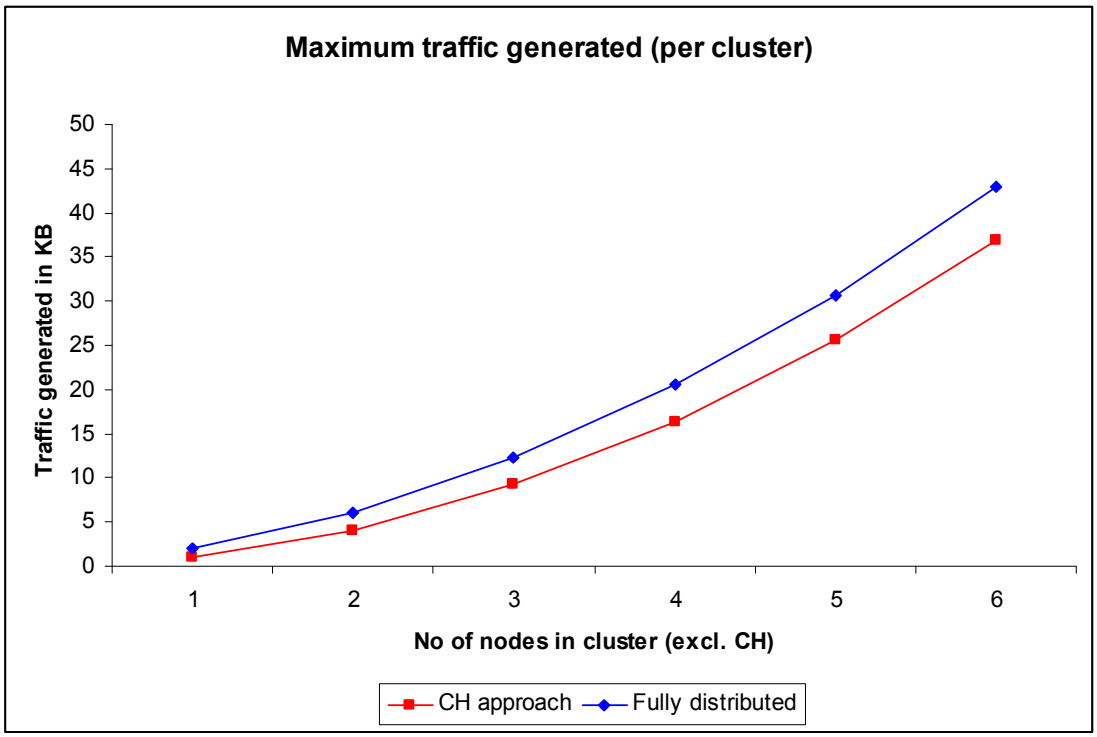

Figure 3.2: Traffic generated per cluster

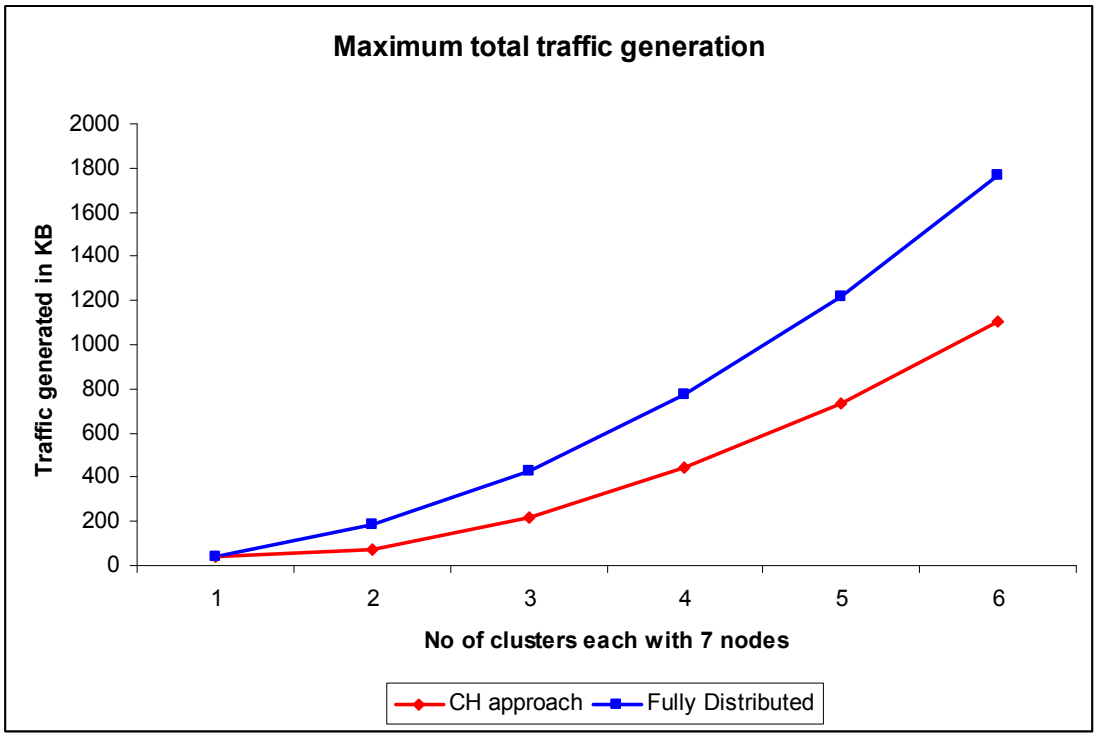

Figure 3.3: Traffic generated within the network

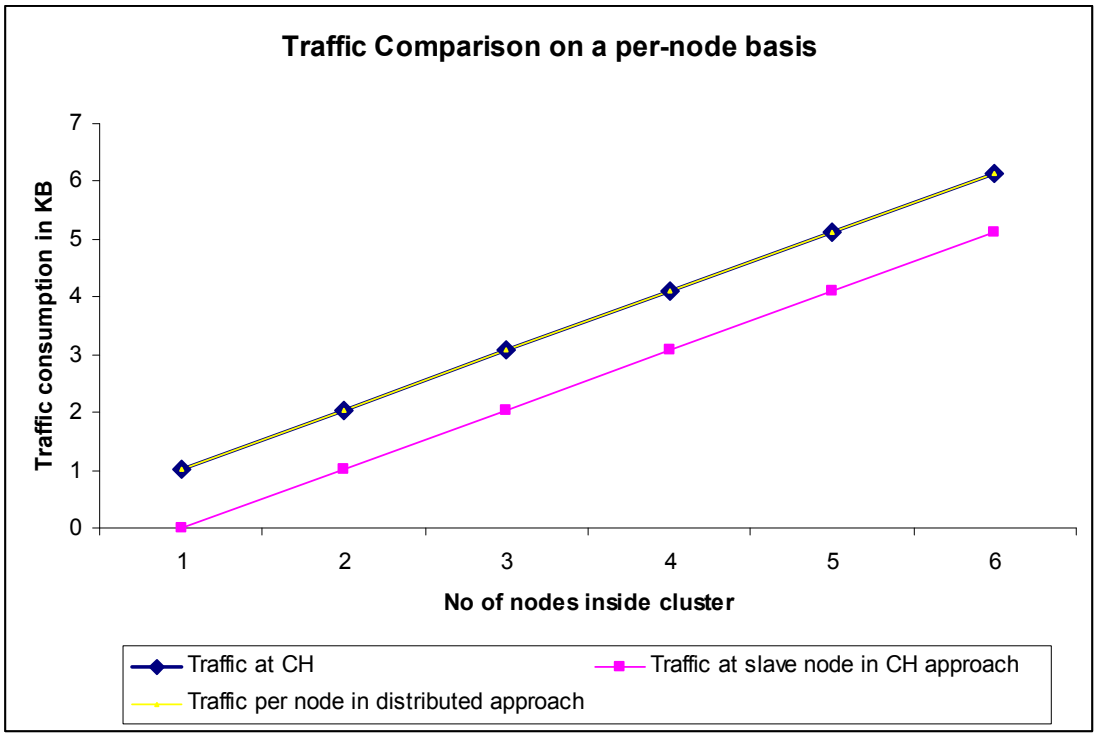

Figure 3.4: Traffic generated according to node class

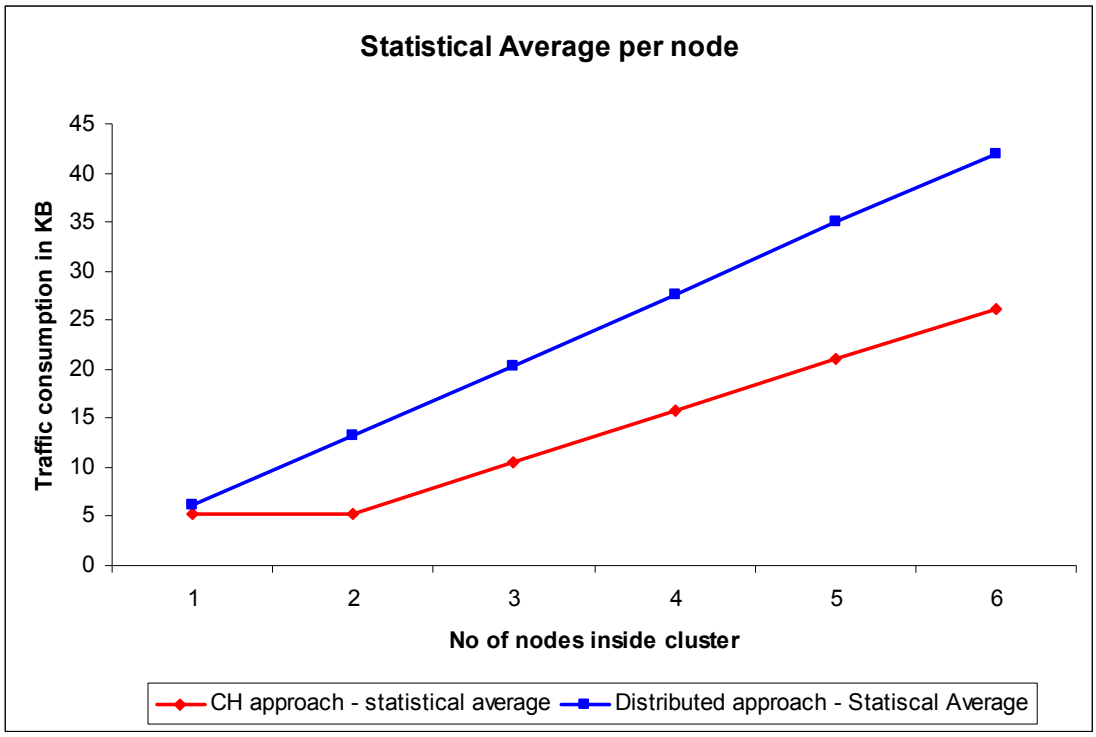

Figure 3.5: Traffic averages as calculated for the CH approach and for the fully distributed approach

These included:

- Simultaneous sending of certificates from individual nodes to their CHs.

- All nodes request certificates from one another (in the fully distributed mode). This is extreme because it would be highly unlikely that this should happen in real life. However, it provides a useful upper bound for the bandwidth consumption.

- The system is considered trustworthy, when all nodes within the cluster or network have access to the certificate of every other node within the cluster to which they belong.

The following outcomes are detailed in Figures 3.2 – 3.5.

- In Figure 3.2, it can be clearly inferred that the CH approach generates less overall traffic within the cluster. This advantage increases as the number of nodes increases. However, this can only go up to the congestion limit of the CH, that is the maximum number of certificate requests it can accommodate without starting to drop packets.

- In Figure 3.3, the same experiment is repeated as in (a) but this time increasing numbers of clusters with fixed sizes (6 slave nodes) are compared in both cases. Again, the CH approach generates less traffic and this time, the larger the network, the better the CH approach fares.

- Figure 3.4 compares the amount of traffic within a cluster at the CH and the slave node (for the CH approach) and at each node (for the distributed traffic). This indicates that while the CH has to accommodate more traffic than its slave counterpart, that traffic is no more than what any node would have to accommodate in a fully distributed scenario. Hence the CH method allows the slave nodes to have more resources for other purposes, unrelated to trust propagation.

- Finally, as a statistical confirmation, the average traffic per node is calculated for each method, confirming the superiority of the CH approach (Figure 3.5).

This experiment was a very simple demonstration of the bandwidth usage of the clustering method. It uses assumed certificates as a measure of simplicity purely so that the traffic flow may be monitored between the two different set-ups.

Besides networking issues, there are also several reasons why clustering is an attractive proposition for FRANTIC:

1. Clustering can redistribute the tasks involved in trust formation and distribution. Normal nodes can take up the responsibility for generating trust information based on direct experiences, whereas cluster heads merely collect

such evidence and in turn generate an overall reputation for each node within its cluster. For distributed approaches, this would mean that all nodes have to generate their own reputation about all the other nodes they are involved with and have to compute the recommendations from their trusted peers as well. The clustering approach therefore decreases the amount of computation involved by task redistribution. Furthermore the reputation of any node within a cluster is more representative since it will be deduced based on recommendations from all other member nodes with which it will invariably have more interactions.

2. New nodes entering the network are able to immediately have access to reputation information about member nodes. They do not have to wait and perform calculations over a given period of time in order to find out which nodes they can trust within the network. In distributed approaches, every time a transaction was initiated, this would have involved querying recommendation information from other peers. Even so, this can be very difficult for a new joiner who may not have had time to know enough peers in order to trust their recommendation about a given node it may wish to execute a transaction with.

3. Assuming some cluster heads are within range of one another, it is much easier to exchange inter-cluster trust information for those nodes that may wish to perform transactions with nodes in another cluster. Recommendations between clusters are quicker than individual nodes having to find a long recommender chain in order to obtain similar information, which may not be as accurate. Cluster heads are also better placed to "hand over" the reputation information of a node as it moves from cluster to cluster, thus ensuring it does not have to re-initialise its reputation once it moves (this meets Requirement 14 from Section 2.5).

4. In conjunction with Point 1, less computation overall will mean less drain on the resources of nodes within the network, such as battery levels. Another consideration is that the amount of space required for storing reputation

information is reduced although this may only be applicable in the case of very large networks.

3.4.2 The Cluster Model Topology

The architecture of the trust model is different to that of the network model. Within the network model, nodes may be connected to the cluster head as well as boundary nodes that may connect to other nodes in different clusters. This is essential as it allows the network to be continuous – otherwise the whole network would only be a disjointed collection of clusters. However, in the case of the trust mode, the topology is different. This is because the CH is responsible for tasks that the nodes are not. It is therefore necessary for all nodes of a specific cluster to constantly report to their respective cluster head and provide all information requested. The difference in topologies is illustrated in Figure 3.6.

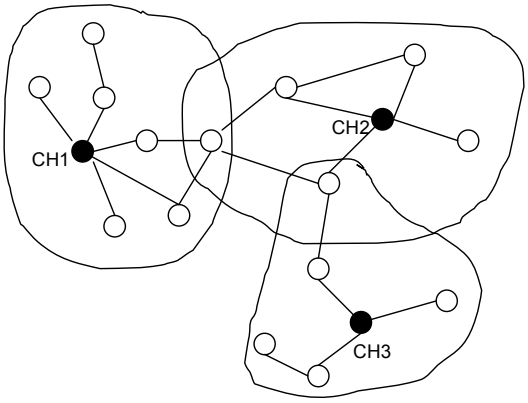

Figure 3.6a: Network Topology

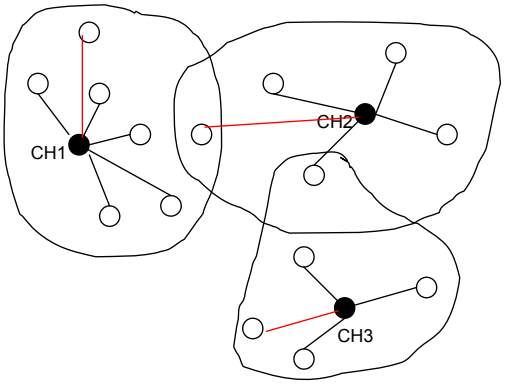

Figure 3.6b: Trust Topology

As can be seen from the above schematic, a network is divided into different clusters, each of which contains a cluster head (CH) that oversees the trust operations. A node is defined as belonging to a cluster when it has a bi-directional link to the CH. It follows from the diagrams above that in terms of the trust topology, nodes follow a hierarchal structure within their respective clusters, with the CH at the top of the

hierarchy. Boundary nodes (that fall mid-way or are within ranges of two CHs) may choose either CH in order to belong to its respective cluster. CHs themselves form a super mesh network in which they may communicate to one another, whenever they are in range in order to exchange trust information for transactions between clusters. All nodes within the model are equal in terms of computational power and other resources (range of radio transmission, software etc.). This is a reasonable assumption as the framework is designed to work between similar devices e.g. between palmtops or between laptops.

All nodes also include the trust component that allows the trust information to be generated, distributed, updated and stored. Additionally, all nodes are also able to operate as cluster heads if required to do so, thereby expanding the possibility of promotion to a cluster head to all nodes equally within the network. This trust component is displayed in Figure 3.7. The concepts of trust maintenance and audit will be explained in Section 3.5.

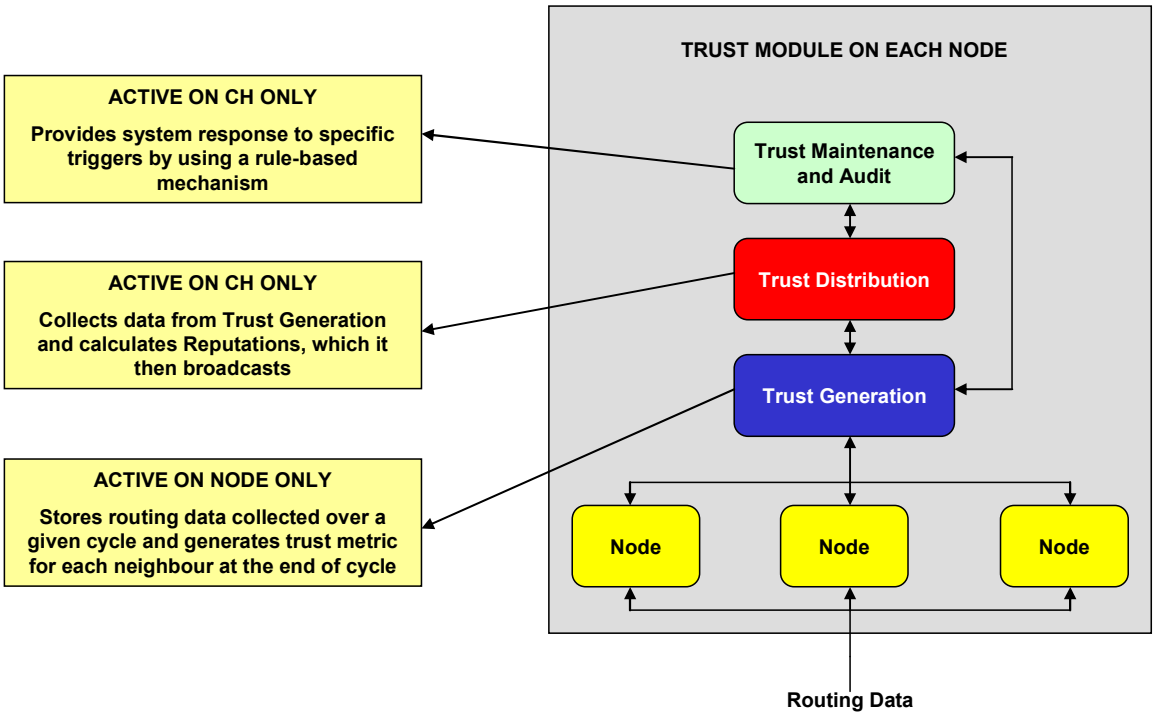

Figure 3.7: Trust module present on each node slowing the calculation and storage required for reputation information

3.4.3 Bootstrapping the Network

So far, the clustered framework has been described as if it were fully functioning. However, this is not always so – there are several states/phases in which the network may find itself, three of which are not stable. For the purpose of this work, four states have been identified: discovery, stable, unstable and decay. The stable state is the normal operating state of the network and the framework. The discovery stage is the one where bootstrapping of the network takes place; this is one of the most crucial states as it is effectively the beginning of the formation of the ad hoc network and the initial trust relationships. Before the discovery state is described in detail, an overview of the four states follows.

Discovery State: This is the state of the network at the beginning of the monitored period where no trust relationships exist between any given nodes. It is also called the bootstrapping phase. However, in this case, no trusted third party is involved within the discovery phase.

Stable State: This refers to the stage after the discovery phase has completed. In this phase, cluster heads have been elected and clusters have been formed with members clearly aware of their identities. The process of generating and maintaining trust evidence can therefore begin. This stage may include the exchange of sensitive information if required. It should be worth noting that any new nodes joining during the stable phase have to go through the same initiation process as others during the discovery phase.

Unstable State: This relates to the state of the network when it is being put under stress. This may be either due to network issues such as acute congestion or a high turnover of nodes necessitating a high number of transactions when initiating new nodes or reallocating cluster heads. It may also be due to passive or active attacks from within or from outside the network. There are two options for a network in an unstable phase: it can either return to its previous stable point or it may move to what is termed the Decay phase.

Decay State: This is the stage at which the network disintegrates. This means that clusters break down and nodes become individual components again. This may happen when a network has been in an unstable phase for a substantial amount of time without being able to stabilise. At the decay phase, the nodes may choose to move away and join a different network or try and create a new network by moving back into a discovery phase. The latter can only be an option however if the decay phase was due to a network overload or some similar resource constraint. It is futile to re-organise a network that has been targeted by misbehaving nodes and has failed to weed them out, thereby taking down the whole framework.

Bootstrapping

In order to bootstrap a network from the ground up (that is assuming no prior interaction or knowledge at all), the network must follow the social model of "breaking the ice". Nodes may elect to get to know one another by exchanging "trivial" information such as playing a game, or exchanging news in order for them to have an early assessment of the reliability and therefore the trustworthiness of their neighbours. This also aids the early election of cluster heads. Honesty is more difficult to assess at this early stage, so this means that the network is more vulnerable. However, any network that fails at the discovery state in all likelihood includes a high proportion of misbehaving nodes and would not have been able to survive to form a stable state anyway.

Once nodes have exchanged trivial services between one another (this is arbitrarily decided based on the framework being implemented), there should be direct trust and recommendation data for most of the nodes within the network. It should be noted that at this stage, the network is fully distributed and all nodes perform their own calculations for the aggregation of recommendations. There is no involvement from any third-party (cluster head).

The election of cluster heads can take place once sufficient trust information has been propagated throughout the network. The voting rules for the cluster heads are based on the work of Guo and Li in sensor networks (Guo and Li, 2007).

1. Each node can only vote for another node for cluster head if that specific node is the most trustworthy of all the nodes within its neighbours.
2. The radius of interaction is only one-hop. This means that all nodes only vote for their one-hope neighbour nodes and all votes are not forwarded by other nodes.
3. The node with the highest number of votes is elected as the cluster head.
4. If two nodes have a similar number of votes, then the node with the lowest node ID is elected as the cluster head.

By introducing a voting system, the framework becomes more democratic and more stable. This means that only the nodes with the highest trust ratings become cluster heads. Furthermore, only the better connected nodes become cluster heads as they will be the ones with the highest number of votes. Nodes that are at the periphery of the network would receive fewer votes although they may have been just as trustworthy. Such a set-up ascertains that the cluster head is well-placed within the cluster and has good radio contact with all nodes that elected it as the head of their cluster.

Once a cluster head is elected, it becomes responsible for collecting reputation ratings about all nodes within its clusters. Direct trust calculation is still done by each member node but the cluster head does the computation of reputation. This has the advantage of giving the reputation a more objective value, as it becomes a single value for each of the nodes within a cluster. A node's reputation thus becomes an attribute of the node, as opposed to being constantly and subjectively redefined by each node it encounters.

The cluster head calculates the reputation a node based on the recommendations on every other node within the cluster that has previously interacted with that node.

Just like member nodes, the cluster head calculates reputation values at the end of a given time period. There are two ways to achieve this. The first option is to have defined periods at which reputation is computed. While this may seem like an orderly way of getting reputations, it has one serious flaw. By allowing all nodes to send their data at the same instant to the cluster head, the risk of congestion is

increased and may result in packets being dropped, thereby resulting in the cluster head receiving inadequate or incomplete information. There is also the possibility of a second wave of congestion as the cluster head then proceeds to flood the cluster with the reputation tables. Not only is this a waste of resources, but it may also involve nodes receiving redundant information about nodes they may not have a one-hop relationship with.

Therefore, for this model, a second option is better where the time frame for collating reputation information is staggered. This means that the cluster head can compute reputation information as they come in and does not broadcast it across the cluster. Instead, the cluster head stores the information and only replies to nodes that provide fresh data or that request such information. For example a node that provides an updated recommendation on a node may then be issued with the new computed value of the overall reputation of that node and may then use that new reputation figure as the basis of its initial trust value for calculating further direct trust values while interacting with that target node.

Once the CH has calculated the reputation values, they are then stored. The following information is recorded (example below assuming node *X* interacting with nodes *A, B, C, D* and *E*). *Comp* refers to the reputation computation process.

Comp t	**node_id**	**trust values received**					**reputation**	**time**
	X	T_{XA}	T_{XB}	T_{XC}	T_{XD}	T_{XE}	R_X	*t seconds*
Comp t + n	X	T'_{XA}	T'_{XB}	T'_{XC}	T'_{XD}	T'_{XE}	R'_X	*t + n seconds*

Table 3.2: Data table showing the information stored by the Cluster Head for a particular node when receiving trust data from other members

If no trust values are received for *X* during a given time period *n*, then the cluster head updates the table with a value of *X* that reflects the temporal decay of trust. In FRANTIC, this temporal decay is defined as being half of the difference of the trend previously noted for node *X* (this is arbitrary and can be adjusted to suit the sensitivity of the framework to temporal decay).

Therefore, taking the example shown above, if no trust information is received from node X at $Comp$ $(t + 2n)$, then the cluster head decreases the reputation of X by $|(R'_X - R_X)|/2$. The modulus of the value is always taken to account for a drop in reputation which would have produced a negative value and thus actually increased the reputation of X. This is not possible as the temporal decay in trust always results in a decrease in reputation by definition.

If no information is obtained from its peers at $Comp$ $(t + 3n)$, then the same process is carried out and the reputation of X is further reduced. This process carries on until either the cluster head receives new trust values for X. This temporal decay of trust is useful because it caters for two situations which are that either the node has left the network or it has adopted a latent role. Either way, it is imperative that the node is weeded out gradually as a latent node may also mean that it is a misbehaving node that is observing the network for a possible attack. The benefit of the doubt falls to the latent node – hence why it is weeded out gradually rather than swiftly. The framework promotes false negatives over false positives when detecting misbehaving nodes. This is because nodes are always given a chance to redeem themselves (should the latency have been without specific intent). If the system is set up so that misbehaving nodes are swiftly removed, this may also create the situation where a large number of "good" nodes are excluded due to network latency issues caused by factors outside their control.

3.4.4 Incentives and Dangers

This is an aspect of trust frameworks that is generally overlooked. For a framework to be viable, nodes need to have the predisposition to be altruistic, i.e. behave like rational agents (See Section 2.3). Similarly traffic from each node should be therefore continuously monitored in order for a sustainable trust framework to develop and for trust to propagate around the network. But for this to happen, nodes need to have some form of incentive in order for them to invest their resources into achieving these tasks.

All nodes need some sort of trustworthy framework within which they can operate in relative safety. Relative being the operative word, because safety, like security can almost always be improved – hence the need to find a compromise and stick to it.

If nodes were to operate in a fully distributed mode, they would all need to be responsible to obtain direct evidence from their peers, to query their neighbours for recommendations, to assign weights to those recommendations, to gauge newcomers and to use computing resources to calculate trust metrics intermittently.

FRANTIC solves part of the problem already through segregating tasks. Member nodes are responsible for acquiring direct experience, while cluster heads take responsibility for collecting recommendations, creating reputations and distributing these around the cluster as well as maintaining and auditing trust over their cluster. The cluster heads are only able to perform these tasks because they have been freed from other tasks such as the responsibility of having to query nodes to obtain recommendations or to monitor the network to obtain direct evidence. These are incentives in themselves. Furthermore, unless they misbehave, cluster heads are immune from non-cooperation accusations. In mathematical terms, this means that the values of G (the gain) and D (drop rate) for a cluster head are always nil. This means that other nodes cannot accuse a cluster head of non-cooperation as the very measures by which non-cooperation is determined have been disabled. This is yet another incentive for the added workload that they perform in order to keep the trust framework running. The performance of cluster heads must always remain optimal as far as possible, hence most routing requests for instance may bypass it if there is the option of an alternative route.

Member nodes on the other hand have a much simplified task in terms of generating trust evidence. They only need to compute a single metric and send this to the cluster heads, based on their observation over a given time-period.

The way the performance is measured is also an incentive for nodes to contribute more to the network. The algorithm has been defined in such a way that nodes which route the most packets generate the highest performance values and may therefore

increase their reputations in less time. The advantages of an increased reputation are: firstly the possibility of being promoted to cluster head once the current one retires or is revoked and secondly to have the luxury of making the network work for it (increasing its *gain*) without having to worry about the consequences.

Dangers

Because the cluster head is a major player in FRANTIC, it may also prove to be its Achilles' heel should the network be attacked. That is one of the reasons why cluster heads are not required to perform routing tasks for other nodes. They will only participate in routing when they are the sources (i.e. broadcasting information to other nodes) or sinks (receiving trust information from other nodes). By limiting its participation in the network (other than for personal and trust management services), the probability of the cluster head being involved in a DoS attack, for instance, is thereby decreased.

If the cluster head is compromised, then a mechanism needs to exist for it to be detected. It is one assumption that all nodes are equal in terms of computational power and other resources and that they are all equipped with a trust mechanism. This means that, for member nodes, although a lot of the functionality is not always used, they still have access to it when needed. This is how cluster heads are monitored. For example, every time a node issues its trust value to the cluster head and in turn gets returned a reputation table, it should always compare its previous table with the new one before overwriting it. Because there is only one new set of data, it is not possible to perform standard deviations, however, any big discrepancy can easily be detected through simple thresholding.

If such a discrepancy is noted, then the node should not overwrite its first cycle. Instead, it should verify whether or not this happens again at the following cycle by comparing the new reputation table supplied by the cluster head, much in the same way that the cluster head monitors variations. If the discrepancy repeats itself, then the node may trigger a *vote* by broadcasting a *vote-table* to the rest of the network. This table contains only the *node_id* of the requester and a *vote* column in which

replying nodes may respond by a 0 or a 1. A 1 indicates confidence within the cluster head, a 0 indicates the opposite. If there is a majority of 0s, then the cluster head's authority is revoked and the node with the next highest reputation is elected. Remaining member nodes then send their trust data to the new cluster head.

This method is devised in order to make sure that cluster heads perform as they should. Another situation that may induce a vote of no confidence is when the cluster head does not respond to reputation requests or not broadcast or reply to reputation tables/requests in a timely fashion.

A measure which may also increase the robustness of the cluster head is to elect a deputy cluster head which operates like any other member node, except that it receives the full list of reputation tables, black lists and variation lists (see Section 3.5) at every cycle as a back-up in case the cluster head ceases to operate either due to mechanical faults or because of an attack/capture. It may also make the transition between cluster heads more seamless, once the existing cluster head decides to leave of its own accord.

3.5 Trust Maintenance

Although part of trust maintenance involves updating reputation values for nodes as per Section 3.4, in this section is concerned more with specific situations that may arise that may require specific trust maintenance to be performed. Generally, trust maintenance refers to the processes that are performed in order to update reputations across the network during its stable phase. In contrast, a trust audit (Section 3.5.1) refers to processes that take place when the network is either in the unstable phase or is being threatened into the unstable phase by current events.

First, a few trust maintenance scenarios that commonly arise are described and then how the model deals with them will be explained.

1. *A new node wishes to join the network during its stable phase.* Once the discovery phase is over, the network may still allow newcomers to join the cluster – these join the one they are geographically closest to. When a new node presents itself to a cluster by broadcasting a *hello* message to all cluster members (a *hello* message generally only contains the *node_id*), the cluster head registers this new node on its table and assigns it a mid-level trust rating of 0.5 and broadcasts this information to the rest of the cluster. This means that the node is neither trustworthy nor untrustworthy. This is to give the node the maximum possible chance to integrate itself into the cluster and to increase its trust value and overall reputation by contributing to the functions of the network.

A new joiner is the only event when the cluster head will broadcast reputation information without first being queried. This is understandably to alert its cluster members of the presence of the new node and its reputation so they can create a slot within their tables for this new node. It is important for the cluster head to send out this broadcast message confirming the acceptance of the new node, instead of all nodes just adding the new node once they receive the *hello* message. This is because a cluster head may also choose to reject a certain node, based on several reasons. One reason may be because of possible congestion issues within the cluster or because the cluster may have reached its maximum target size.

If the node is accepted within the cluster, it then starts to behave just like other nodes during the trust generation exercise. If it is refused entry, it will then have to move, in order to find a cluster willing to accept it. Another option may be to elect itself as a cluster head and to wait for other nodes to join it. However, this may be a more difficult option since it will have a low rating compared to other cluster heads and may not be an attractive option for new nodes.

2. *A node wishes to move to an adjacent cluster.* If a node wishes to change clusters, it needs to send a *hello* message to its new cluster head but this time with the *node_id* of the cluster head whose cluster it has just left. The new cluster head then sends an R_{rep} message (which is a reputation request) to the adjacent cluster head. However this can only happen if the two cluster heads are within radio range of one another,

i.e. forming part of their own super mesh network. The former cluster head may reply with the reputation value of the transferring node. The reason why direct radio contact is preferable is to prevent any reputation requests from being intercepted and forged along a transmitted route between cluster heads. If the two cluster heads are not within range of each other, then the node has to follow the exact procedure as a new node and build its reputation from scratch. Such an occurrence however is very unlikely because nodes will not move to adjacent clusters that are within radio range of one another. Instead it is very likely that they will move out of the network. They will then be treated as new nodes in whichever network they choose to join.

3. *A node leaves a network.* The node does not need to do anything when it leaves the network. It will be automatically phased out by a gradual decay of its reputation.

4. *A cluster head wishes to leave the network.* Once a cluster head decides it wants to move away from the network, it sends a transfer request to the node within the cluster with the highest reputation. This transfer request will contain the following data:

{node_id of outgoing CH, dead value, time stamp}

The *dead value* is seen as an indication that the cluster head no longer wishes to be one. It sets a value of 0 within the table for the *dead value*. Only a prospective replacement receives this message. It is not broadcast to the whole cluster. If the replacement decides to accept the position, it replies with a similar message, but this time with its own *node_id* and a *dead value* of 1. Once the outgoing cluster head receives this information, it broadcasts it to the rest of the cluster and the new cluster head then takes over the operation of the cluster. The outgoing cluster also transfers across the reputation tables for the cluster as well as other maintenance tables such as variation and black list tables (see Section 3.5.1).

These are a few of the common events that may happen within FRANTIC with an indication of the various rules set in place in order to accommodate such events. The

next section deals with the Trust Audit which is a more serious form of maintenance, especially when the network comes under attack or faces non-cooperation.

3.5.1 Trust Audit

There are various instances in which a trust audit will need to be performed. Put simply, a trust audit is trust maintenance in a state other than the stable state (usually the unstable state). This means taking corrective action in order to prevent a stable network from becoming unstable or to try and stabilise a network which has already been made unstable. A trust audit is important because it is what characterises the resilience of the model and defines how it can face known trust issues within ad hoc networks.

FRANTIC uses a set of rule-based mechanisms to deal with audits. These may be customised depending on how severe a threat appears to specific networks. The threshold values within rules may be either arbitrarily defined or set as a result of a study on various threats to the network.

It can be notoriously difficult to identify an attack, especially when it is not particularly vicious in nature. Hence, it is better to err on the side of caution by taking an early corrective action for any node that seems suspicious. While this may create a higher than normal incidence of false negatives, it will also help in the early detection of attacks or non-cooperation and thereby help preserve the network.

The following are actions that are taken by the model, based on a pre-defined set of rules. It should be noted that some of the rules have threshold values that are set on an ad hoc basis but are configurable based on the specific requirements of the scenario within which the framework has been deployed. Generally, the more risk there is to the network, the more stringent the rules will be.

1. *The reputation of a node falls below 0.2.* This is the set threshold for early corrective action. It has been set by default at 0.2 as this level is just below mid-way

between a node that is completely untrustworthy (0) and one that is neither trustworthy nor untrustworthy (0.5). What failing to stay at or over a reputation of 0.2 involves in FRANTIC is a temporary ban for the node. In practical terms, this means that the *node_id* of said node is dropped from the reputation table that is sent out in reply to member nodes. The effect this has is that this node will then be barred from using the network as it will not be considered as being part of the cluster and therefore part of the network. This is a temporary ban from the cluster head and it only lasts for a given time period *n*. Once the period is over, then the cluster head broadcasts the node's reputation within all its tables at the next request. The reputation assigned to the node is the one which it had prior to its reputation falling below 0.2.

2. *The reputation of a node falls below 0.2 for a second time.* This time, the *node_id* is deleted from all tables and the node is therefore ejected from the network and its *node_id* is broadcast to all adjacent cluster heads to be added to a special table for barred nodes.

3. *The reputation of a node falls below 0.1 within a single cycle.* This is very rare and either depicts a very selfish node or a node that has been dropping packets voluntarily out of malice. The same procedure as point 2 is employed, with no chance for the node to redeem itself.

4. *A node issues false recommendations, whether highly favourable or highly negative for another node.* In order to cater for such eventualities, the cluster head keeps a record of the variation, λ, of actual trust values from individual member nodes from overall reputation. The table below shows an example. In this case, the reputation of node *X* is used to monitor the variance of the trust values supplied by nodes A to E. There will be a similar table for all reputations that are calculated.

comp n	**node_id**	**trust values supplied**					**reputation**	**Stand. Dev.**
	X	T_{XA}	T_{XB}	T_{XC}	T_{XD}	T_{XE}	R_X	σ_x
variation		λ_{XA}	λ_{XB}	λ_{XC}	λ_{XD}	λ_{XE}		

Table 3.3: Variation of trust values during a trust audit

The standard deviation is also calculated as σ_x using the following equation:

$$\sigma_x = \text{sqrt.}[(1/N)*\{(\lambda_{XA})^2 + (\lambda_{XB})^2 + (\lambda_{XC})^2 + (\lambda_{XD})^2 + (\lambda_{XE})^2\}] \quad (3.10)$$

where N is the number of recommendations obtained for X (in this case: 5) and sqrt. denotes the square root.

If any λ_X value is found to be more than 1 standard deviation away from the mean value $R_{X,}$ then the node is marked on the black list table. This is valid for all recommendations that that particular node may make. If the difference in value exceeds the standard deviation again, whether in the same cycle or not, then the node is barred from the network. It does not matter whether there were artificial increments or decreases to the actual value. It should also be noted that the black list table, which also includes a list of barred nodes is permanent and is passed down to future cluster heads as mentioned previously.

These are therefore the main situations in which trust audit may be deemed necessary in order to ensure survival of the network. Although some of the rules may appear too stringent, it is necessary to err on the side of caution when operating in open networks as the risk to the network is much enhanced. Besides, any wrongly barred node has a chance to redeem itself such that there is no danger of the network being depleted of good nodes due to a too stringent trust audit mechanism. There is an issue with barred nodes in that they may spoof their identities in order to attempt to rejoin the same or a different cluster. Because the environment is open and identities are not authenticated with central repositories, the threat for this to happen is significant. This may not necessarily be an issue as the framework processes would automatically result in the node being ejected again, much as it was the first time. It is all based on the principle that it does not matter who the node is but rather what they do that bears more significance.

3.6 Chapter Summary

This chapter has detailed how FRANTIC, a framework for decentralised trust modelling, operates. This model allows an entity to determine how much it can trust another entity it wishes to interact with.

FRANTIC, as presented in this chapter, meets Objectives 1 to 5 from Chapter 1, with Objective 6 to be presented in Chapter 5 and Objective 7 in Chapter 4.

The model provides nodes with a way to calculate direct trust that encourages fairness and promotes reliability and honesty. It also provides a way to calculate reputation.

For reasons depicted in Section 3.4, the model also utilises clustering as part of its architecture and provides for the election of cluster heads democratically via a voting method. Cluster heads allow for the segregation of duties that offers significant operational advantages, see Section 3.4.4.

Finally, in order to meet Objective 5, the framework proposes a rule-based mechanism in order to introduce resilience by punishing nodes that prove to be selfish or misbehaving.

Chapter 4

4 Model Evaluation

In Chapter 3, FRANTIC was presented as a novel model for trust within ad hoc networks that allows nodes to reduce the uncertainty of interacting with other nodes within the network. With the bulk of the theory proposed, it is necessary to verify the validity of this approach.

In order to verify the validity of the framework through empirical methods (Chapter 5), the scenarios for which the tests are undertaken must be defined. In this short chapter, these scenarios are presented along with the specific issues they present.

4.1 Model Configuration

In order to be viable for use in an ad hoc network, the framework needs to exhibit the following general properties, from a practical viewpoint:

- Lightweight. There must be no significant addition to the existing network overhead. In other words, the framework must be "lean" through using existing 'in-band' data as far as possible and minimising the use of additional traffic, 'out-of-band' data that loads the network more and can even provoke congestion. There should also be no significant additional demand on storage capabilities. In this case, the framework minimises the need for additional data by using routing information as part of the trust generating process. Furthermore, the segregation of duties between the cluster head and the member nodes address the issues posed by the storage of trust data.

- Easy and rapid deployment. The framework application should be easily accessible to all devices and be able to work on the proposed target devices. The trust module added to nodes is simple and can be quickly adapted in order to suit specific platforms.

- Scalable. The framework should be usable in both small (fewer than 20) and larger (more than 100) networks without needing any modifications. Very large networks (300 and above) are not necessarily included in this particular requirement since those would usually be found in instances where security architectures attached to backbone structures would be available (for example wireless computers on a university campus). The operation of the framework is restricted to self-starting ad hoc networks only, i.e. those whose nodes have had no prior interactions.

4.2 Scenarios

FRANTIC scenarios are designed to implement the following range of properties (see Chapters 1 and 2):

- A network consisting primarily of mobile devices such as mobile phones, PDAs or notebooks. The framework needs to be able to cater for devices with the least operating capacity and resource constraints, e.g., a typical mobile phone. Although homogeneous rather than heterogeneous device capacities can make a network easier to manage, this framework should also be able to operate with different types of devices.

- Device networks range from a few metres to at least a few hundred metres using 802.11x technology in the upper band. This means that next-hop neighbours will be able to locate one another relatively easily within a restricted space such as the waiting room of a train station or airport.

- A medium-sized network that can range from fewer nodes (more than 20) to around 100 or more nodes.

- Nodes may either have a collaborative or non-collaborative purpose. Based on the scenarios being described, users may elect to behave either way with the framework having to maintain fairness in the system.

- A network with low to medium mobility, i.e. nodes with high or medium pause times. This is a necessary requirement since for any viable trust formation to take place and propagate through the network, the level of transience has to be low. In other words, the framework proposed here will prioritise resource-saving over information refreshing.

With these properties in mind, the following two scenarios are proposed for the implementation of FRANTIC.

These scenarios are extracted from real life and as such pose a level of complexity that is outside the scope of this work as discussed in Section 3.1. The purpose of these scenarios is to propose ways in which the framework may be implemented in real life such that it is able to utilise some behavioural aspects of these scenarios in order to generate trust. The behaviours that make these scenarios relevant for the framework are listed at the end of each section.

4.2.1 Disaster Recovery

This scenario is one that has contributed much in spurring the development of ad hoc networks. A typical example could include a situation where several emergency units from various departments (fire, ambulance and police) have to communicate how best to deal with a disaster scenario.

Often emergency response teams enter areas of poor wireless coverage from central carriers. This may be because the emergency is underground or it is in a remote

location or the sheer demand on the network actually renders the existing infrastructure close to useless. Furthermore, the majority of communications required to coodinate a rescue effort is bound to be local to the area and thus not really require in the input of centralised networks. In this respect, it is of crucial importance for devices to develop mutual trust in an effort to promote peer groups that may then network in terms of how best to approach a situation. The risk factors in this case are likely to be the very perpetrators of said disasters. This case study is examined in the following paragraphs.

Brief

A self-starting ad hoc network is required at a disaster site. For discussion purposes, this is assumed to be in the form of a terror attack within a city centre. We assume conventional methods of communication such as mobile phones are not operational due to over-demand or because the infrastructure has been severely damaged. In terms of a trust scenario, this presents us with the following.

Stakeholders

1. Victims of the terror attack (survivors)
2. Opportunists. These form part of scavenger groups that are there to extract maximum personal benefit from a crisis situation irrespective of what may be happening around them. Activities include looting and pilfering.
3. Disruptors. These are perpetrators of the said attack themselves or their accomplices. Often, tactics involve luring in more people by a first attack, then carrying out a second attack for maximum damage to life and property.
4. Helpers. This group includes all emergency services and voluntary organisations that are there solely for the benefit of victims.

Clearly, there are four major categories of stakeholders within this trust scenario, each with very different motives or needs. In this particular example, victims are likely to be passive and therefore act as the object of the operation, rather than take

any active part within it, which leaves three very specific groups of people with conflicting interests.

The overall site can be depicted as per Figure 4.1.

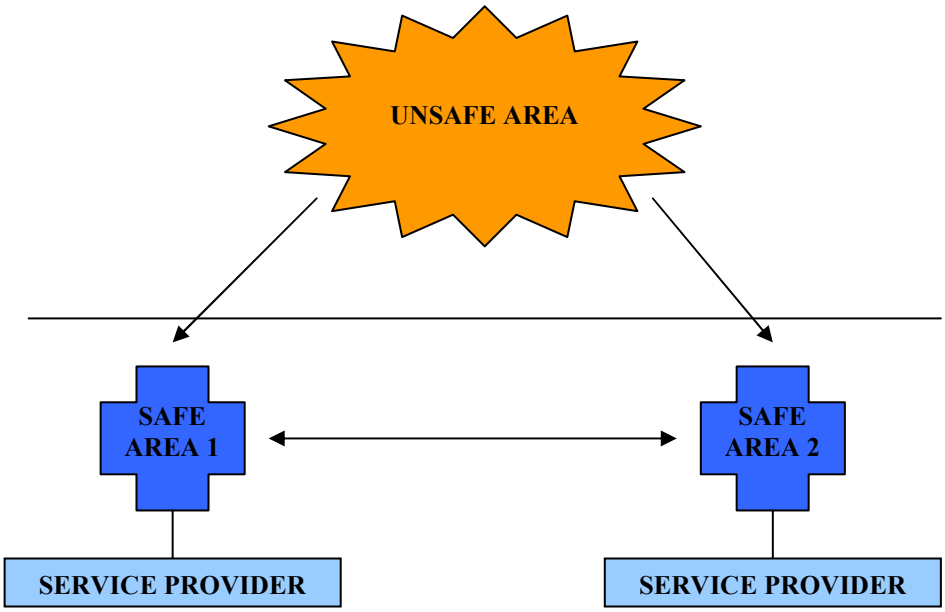

Figure 4.1: Schema for the disaster site showing the location of two service providers electing base areas (determined as safe)

Assuming that two safe areas have been set up, service providers will normally be operating within those areas to reduce the risk to their personal life. It is the duty of emergency services to then bring elements of the disaster (victims or rescued vital property) to the safe zones. Service providers range from on-site first aid medical teams to voluntary organisations providing shelters and so on.

In this context, safe areas can be easily identified as trustworthy clusters. This is possible because safe areas will be usually designated as such and hence easily identifiable from visual cues. Models that work within trust management are those that can be effectively bootstrapped, otherwise it becomes necessary to use trust established out of band. For service providers operating within the disaster zone, suitable networking and being able to trust fellow members of the "Helpers" group is vital to the success of the overall operation, particular if there is a threat from the

"Disruptors" group. One of the major issues within this frantic scenario is that it is very difficult to verify people as who they say they are. Because of limited network communication, off-site identification confirmations are not possible. Therefore, it is imperative to build on the local information available through more localised networking between the multiple clusters of aid people.

Initial recognition between clusters in this scenario is mostly visual, for instance via logos on vans or uniforms and may be used as a bootstrapping mechanism for later exchanging information about rescue operations. The existence of masquerades is possible – however, post initial recognition, misbehaving individuals would be recognised by their negative actions thus triggering remedial responses as appropriate to safeguard the rest of the organisation. By virtue of the hierarchal structure of most organisations, each cluster will very likely have a commanding figure, and a deputy, holding senior positions, thereby negating the need to have cluster head election. Instantiation of the ad hoc "community" therefore is relatively trivial. The focus shifts to how to maintain relationships between the different clusters to ensure that vital or confidential information is not being passed on to members of groups with motives other than search and rescue.

A typical play-out of the scenario would be as follows: a new voluntary organisation turns up at the site. Assuming the organisation is not a well-known one, its intentions may appear obscure to others. In order to ascertain the integrity of the members of that organisation present on site (forming a cluster), other clusters will have to assess its trustworthiness. There is a range of non-networking tasks that need to be performed at a rescue site. These can be assigned to new joiners in order to evaluate their integrity. Once they reach a certain trust threshold, they can then be integrated into the existing ad hoc network operating between older clusters. By making sure that new joiners have to validate themselves before they are privy to information private to the cluster, the risk of the network being corrupted by non-aid groups is relatively small. Opportunists especially will not lose time in trying to collude and earn trust in order to earn side benefits from the disaster site. By their very nature, they thrive on making the most out of a confusing situation on the spur of the moment.

A more severe threat would be from Disruptors who have a motive to break down any rescue effort. However, even their motives would be highly inhibited if they were required to have to earn trust before being allowed into the unsafe zone or networking with clusters within the safe zone.

A variant of this type of scenario is the military or battlefield scenario where soldiers may communicate with their battalions via self-starting networks within enemy territory.

This particular scenario is one that is medium in size (from several dozens to the low hundreds), with a common goal, utilising sensitive data, high range (200-400m), medium-powered devices. Mobility would vary dependent on the scenario being enacted. Battlefield scenarios may expect to be more mobile than disaster recovery ones as the latter are often focused around the disaster area and hence likely to operate within a specific area, of limited range.

The application of a trust framework to a military context however is more of the domain of low-tech militia as opposed to the high-end military operations such as those commandeered by the USA. It is assumed that fighters on the ground do not have access to satellite information, nor are they equipped with GPS devices. Data transmission is not encrypted either as each end user does not possess equipment adequate enough to deal with it. The focus of trust management therefore relies heavily on peer evaluation. In a militia scenario, there may not be visible cues as to the identification of friend or foe and combatants may need to rely on recommendations from peers in order to evaluate the level to which they should trust whomever they network with on the field.

This scenario presents the following advantages: clustering has in a sense already happened due to the different types of stakeholders available. Bootstrapping of the network can also be assumed to be fairly complete as the emergency response team will be coordinating with their known bases and it is highly likely that smaller rescue teams would be headed by a trusted peer, who can then naturally act as a cluster head.

Therefore in this scenario, the focus would be more on detecting selfish nodes and attempting to detect and promote interactions only with collaborative nodes. This can happen using the rule-based mechanisms present in the framework. In order to assess how quickly they work, nodes in the framework can be made to drop packets quickly and the response times noted. This behaviour is typical of Disruptors who would seek to bring down any rescue effort. The ability of the framework to deal with them can thus be measured.

In terms of context, this scenario proposes a common goal for the participants of the ad hoc network. The use of context information is not as important here because all Helpers would be seeking to trust other Helpers based on their ability to aid their terror victims. The major goal here would be coordination of the rescue effort and the type of messages being relayed would also be voice messages or text messages signalling the location where resources are required. The success or failure of each request for transmission of such messages is analogous to the request of routing messages involved in this framework and they can be inferred to provide similar trust information. This therefore makes the experiments performed in Chapter 5 at a low level relevant for a higher level via abstraction.

4.2.2 Travellers' Web

The expectation from ubiquitous computing is that many people now expect to be able to communicate with one another at all times, especially when travelling and having to wait in airport lounges, cafes, train stations and cabins and so forth.

While most will now have the ability to do so thanks to the different types of networks available to them (ranging from wireless LANs to cellular networks), they may also want to interact locally with fellow passengers. With the amount of multimedia content now available on typical cellular phones and tablet computers, such interactions may involve anything from sharing music files to playing games with one another.

The rationale behind promoting this type of network is that although most users will have access to online resources such as web gaming or downloading, a high percentage of them will prefer networking with their local counterparts as this type of interaction is often free, as opposed to significant charges that may be levied on value-added services by telecommunications providers. Indeed, downloading music or even playing games online from your mobile can prove costly to the average user. Faced with the increasing capabilities of cellular phones (which are rapidly turning into mobile multimedia devices), networks now have no option but to limit the amount of data that can be transferred over the air. Along with this restriction, in many situations, no networking coverage is available at all (as is currently the case for underground travel in the UK).

Nonetheless in order to be able to interact and share resources with their peers, nodes will need to be made trustworthy (at least to some extent) by a trust framework. Again, using the basic factors detailed previously, such networks will be medium sized (typically in the dozens of participants or even lower) using low-powered devices (generally PDAs or cellular phones) with a medium range and low mobility (passengers in a train for instance, although on the move will be stationary with respect to one another).

In this case, users may be able to select a certain service level they wish to provide or receive. For example, some may wish to receive news from fellow passengers' PDAs which may have previously synchronised with an online source. Others may elect to play games, or indeed share files or music. Each of these service levels has different trust requirements because the level of involvement of each user, and therefore the level of risk, is different. Clusters of users can be formed between those that fall within a particular level, which for the sake of argument, can be termed "privacy" levels. In other words, newsreaders can be pooled into a relatively low risk category and share their resources that way. Once they have managed to exchange information, their trust reputation will go up vis-à-vis each other. They may then elect to move to a more sensitive privacy level and opt to play games with each other. It should be noted that users that allow interactions at more private levels

automatically include those within lesser levels. For instance, a gamer can also interact within a news reading cluster.

Such a scenario therefore constitutes one more example where a trust framework can be effectively applied in order to manage a number of users who wish to cooperate or pool resources for common usage. The focus in this type of scenario is on service provision, whereby different participants will have varying needs for several services and the trust requirements for these services will vary too. In effect, this models the concept of giving contextual information to trust.

A variant of the travellers' web scenario would be the exhibition scenario (Boodnah & Poslad, 2009). The scenario depicted by the model here is that of a large exhibition centre where users are spread out over a relatively large area but at the same time are bound within a confined space (the perimeter of the exhibition centre). Because of the way exhibition centres are laid out, there is the natural tendency for clusters to be formed based on people's locations. These clusters will generally be either specific stands where people congregate or areas of the exhibition centre relative to the interest of a certain section of the visiting population (e.g. car technology for male visitors and beauty products for female visitors – these can be subdivided into further segments).

This particular scenario is interesting in that it presents two options for bootstrapping: a cluster-head can be naturally assumed for a given cluster (in the scenario above, this can be the exhibition manager of a particular stand, a large trustworthy client or even exhibition centre hosts), or an election can take place in order to generate a cluster head as per Chapter 3.

Brief

An outline of the scenario that can be depicted in the travellers' web can be as follows. One can consider that the formation of a temporary network aboard a train to be an effective way to manage devices that may wish to interact with one another. As mentioned before, although a train may be mobile, its passengers are often not

with respect to one another, so clusters will tend to form naturally within and between carriages. Newcomers and leavers are depicted by passengers getting on and off the train at stations. In this scenario (unlike that of the disaster recovery), stakeholders can be simplified as belonging to two main groups: willing participants (good nodes) and disruptors (bad nodes – these include those unwilling to cooperate as well as those seeking to actively disrupt).

Playout

In this scenario, the key stakeholders would interact in the following way.

Upon departure from the originating station, participants may start to interact with one another by seeking others that may share the need for the same or similar services.

The interaction of nodes looking for a particular type of service will naturally result in natural aggregation. In this case, clustering of nodes may form not only based on geographical positions (such as carriages or seats etc.) but also based on their service needs.

The existence of different service levels with varying trust requirements presents a good opportunity to allow entities to interact on a low-risk basis (by using non-sensitive services) in order to establish initial trust relationships. Once the latter are in place and relationships as well as reputations evolve within the network, entities would naturally move on to the natural election of cluster heads and from thereon diversifying the portfolio of services they feel ready to request and share. In this particular scenario, it may be worthwhile to note that instantly recognisable entities such as on-board train staff may operate passive devices that help in the establishment of the trust communities within compartments. However, for the purposes of this work, this assumption, although probable in real life, is not made.

Disruptors in this scenario would primarily come in two forms: entities wishing to leech resources from the network (thus being non-cooperative) and those wishing to

inflict intentional harm to the network with a view of either stopping its usage or gathering any form of sensitive data such as the service usage patterns of participants with a view to using such information commercially (for instance in targeted advertising).

Although there are two distinct types of disruptors here, the framework will treat both in the same manner, ultimately resulting in both types being evicted from the network. It may be argued that actively disruptive nodes should be removed more pressingly but the resources required to distinguish an actively disruptive node would be mostly found on centralised trust models with online access (for instance to only allow authorised nodes), so this is not possible here.

Nodes within this scenario have limited mobility with respect to one another. There is the need for the network to acknowledge new entrants and leavers as well as those nodes that may seek to migrate to different parts of the train.

Contrarily to the Disaster Recovery scenario, in this instance, one cannot assume that clusters have been predetermined or that the users may have previously interacted. One could argue that some passengers may know each other by sharing the same commute but this is not always the case.

There is the possibility to mimic the data transfer in this case by seeing how long a hitherto unknown network creates a trust framework by allowing interaction between peers. Although the actual content being shared will vary, and with it the context within which trust is acquired, the model can still provide a good indication of how trust can aid in selecting appropriate partners for sharing and pooling resources. By using low level data to create the trust relationship, the framework can then allow users to share more complex content once the relationship has been formed. This makes the framework relevant for use even in such a complex scenario. Even though the individual contexts within which the trust has been acquired will vary, one can argue that it remains sufficiently similar for collaboration to take place. For example, one may trust a user to share cached news on the same level that one may trust said

user to play an online game or share non-sensitive data such as user playlists or even music files.

With these two scenarios defined (with some of their trust aspects to be verified in Chapter 5), Section 4.3 now focuses on defining some performance metrics in order to produce meaningful data that can provide the potential user of a framework relevant information with regards to its effectiveness and efficiency.

4.3 Key Performance Indicators

Having depicted two general scenarios under which FRANTIC may be called to operate, the following section proposes a few novel ways in which the trustworthiness of a network may be assessed. These metrics are general and have not been produced uniquely to assess FRANTIC. The very existence of these metrics is to allow direct comparison of frameworks by very general attributes. This work can be considered laterally to the main argument of this thesis and offers an additional independent contribution to the state of the art in this area.

There are two main types of metrics that can be generated. One is a service environment metric and the other a service specific metric. Service environment metrics generally depict resource availability whereas service specific metrics further define the particular characteristics for an individual service.

4.3.1 Service Environment Metrics

One of the key measures of a service environment (Kalasapur et al., 2006) is service density which is an indication of the ability of the environment to support user tasks. This is given by the ratio of the total number of services to the total number of requests.

$$Service\ density = \{n_t / n_r\} \tag{4.1}$$

In order to meet all user requests, the density should be at least 1.

In FRANTIC, the number of requests for reputation a cluster head can process within a given time divided by the actual number of requests from member nodes can be defined as the cluster density. Using the concept of cycles as depicted in Section 5.2.1, the cluster density, C_n, for a given cycle n is given by:

$$C_n = \{R_t / R_a\} \tag{4.2}$$

Where R_t is the total number of requests that can be accommodated by the cluster head and R_a is the actual number of requests received.

When C_n falls below 1, this means that the cluster head is over-subscribed. This may be because it has exceeded the maximum number of nodes it can accommodate within its cluster or it has received an inordinate amount of requests for reputation due to heavy network traffic.

Another metric which provides an overview of how clustered the framework is is network density, N_d. This is simply the ratio of the total number of nodes, T_n, to the total number of clusters, T_c.

$$N_d = \{T_n / T_c\} \tag{4.3}$$

The network density is helpful at the discovery state. A low network density is likely to translate into faster trust establishment and thus faster cluster head election, with the stable state being reached sooner. This is because smaller clusters are able to compute their interactions within a shorter period of time. Also, the fewer nodes in a cluster, the more likely they are to interact with one another.

4.3.2 Service Specific Metrics

One of the services in which a measurable impact of trust can be determined is in route discovery. The success rate of route requests can be determined by S_r.

$$S_r = \{Q_s / Q_r\} \tag{4.4}$$

Where Q_s is the number of successful route requests and Q_r is the total number of route requests.

With the trust framework in operation in the stable state, a new success rate, S_{rt}, can be calculated in much the same way as above.

$$S_{rt} = \{Q_{st} / Q_{rt}\} \tag{4.5}$$

The impact of the trust framework can then be calculated from the following

$$I_{mr} = S_{rt} - S_r \tag{4.6}$$

Where I_{mr} is the impact of the framework on routing requests.
The impact of a framework can be an objective way in which to measure the efficiency of that framework for a given service, in this case route discovery. While route discovery is not the prime motivation of a trust framework, it is an important consideration in an ad hoc network and good routes are central to the survival of the network.

Another instantaneous metric that can provide an indication as to how the framework is faring in terms of its trust service availability is the quick ratio.

$$\text{Quick ratio} = A_n / A_t \tag{4.7}$$

This is the ratio of the number of good nodes, $A_{n,}$ to the total number of nodes, A_t. The higher the ratio, the better equipped the network is to perform constructive tasks pertinent to the existence of the network. The quick ratio does not differentiate between non-cooperative and misbehaving nodes. This is because the end product is the same: non-availability of nodes to perform routine tasks and because this is a measure of service availability, this differentiation is not required.

Trust audit is a much needed feature of trust frameworks and measurement of audit capacity is useful. To this end, the audit impact, I_{ma}, can be measured as per equation 4.8.

$$I_{ma} = [(A_{b1} + A_{l1}) - (A_{b2} + A_{l2})] / (t_2 - t_1) \quad (4.8)$$

A_{b1} is the number of bad/misbehaving nodes (trying to harm the network with intent) at time t_1 and A_{l1} is the number of latent nodes at the same time frame. The same notation is used respectively for time $t_{2.}$ It can be seen that the impact of the framework on trust audit is measured by the reduction (or increase) in the number of non-performing nodes within a given time period. It is different to the quick ratio which measures trust service availability at a given time (snapshot). The impact is measured over a given period and focuses exclusively on the ability of the framework to weed out non-performing nodes.

There are other ratios that could similarly be developed for a host of services that may be expected to run on an ad hoc network. However, route discovery, trust service availability as well as audit capacity are the key metrics to consider in a trust framework. These metrics correspond to the network being established, trust being established, and maintaining trust. These ratios are also non-specific and could be applied across a range of trust frameworks built on different technologies and using different techniques. These ratios make such frameworks directly comparable in an objective way. Such assessment of frameworks is novel. It is hoped that these ratios may offer the building ground to develop standard ratios that could be used across the trust design field in order to compare methods and assess efficiency.

4.4 Chapter Summary

This chapter proposes ways in which trust frameworks can be adequately assessed. Two different real-life scenarios are proposed on which the application of a trust framework would greatly enhance the delivery of services.

In the first instance, a disaster recovery scenario is envisaged. This is the more dynamic of the two in that nodes are expected to move relatively frequently in and out of clusters. However, the initial establishment of clusters proves to be more straightforward purely due to the range of services on offer and the ability to seek out trusted individuals easily. The increased mobility of nodes is therefore tempered by the strong bases clusters and their cluster heads provide.

In the second scenario, nodes tend to be less mobile. Interaction within the cluster is expected to take precedence as nodes aggregate around those with which they share common themes. They are therefore unlikely to move on until acted upon by external factors (such as the need to get off at a train station or moving on to the next exhibit at a trade fair). The more challenging aspect of the second scenario is in effect the bootstrapping mechanism whereby nodes have to interact "blindly" at first in order to establish a rapport.

The third part of this chapter moves on to a more general way of assessing trust frameworks. It proposes a series of key metrics, analogous to those in the financial services, in order to enhance comparison between frameworks. The development of key metrics also allows potential users of a trust framework to assess its efficiency quickly. While these metrics are not foolproof, they provide an adequate snapshot of the state of the network at any point in time and can also help assess the performance of the framework over a given period.

Chapter 5 will now deal with some empirical assessments of the framework by looking at performance both within and between clusters. An assessment of the way in which FRANTIC meets the objectives initially set out in this thesis will then be made.

Chapter 5

5 Performance Assessment

5.1 Introduction

In order to determine whether the model depicted in Chapter 3 is viable, it needs to be verified and tested. This can be done in several ways, ranging from simple experiments to confirm whether or not its operations are as expected, to more elaborate scenarios designed to confirm that the framework is able to perform within more realistic situations. To do this, a set of experiments has been designed in order to verify the several key aspects of FRANTIC and to ascertain that the process of generating, distributing and maintaining trust within an ad hoc network is successfully implemented and that a display of resilience against malevolent behaviour is also present. The experimentation is laid out in Section 5.3.

These experiments are then validated against the set objectives and requirements from Chapters 1 and 2 respectively. This can be found in Section 5.4.

Most of the simulations have been performed on a Network Simulator, NS-2 (NS-2, 2009), with some modifications brought in where needed. The detailed specifications of each implemented scenario precede each experiment.

In order to facilitate the understanding of the experimental layout, a tabular roadmap of the experiments and their aims is presented in Section 5.2. Further to that, an evaluation summary is also given prior to the discussion in Section 5.4 to aid the mapping of objectives and requirements to the experimentation.

5.2 Model Walkthrough & Experimental Roadmap

5.2.1 Walkthrough

This particular section of the experimentation involves purely testing the model and its algorithms to make sure they work as intended and that the outputs are as expected. The aim is to make sure they are no anomalies present within the design, and if there are, that they may be investigated and corrected. Once this is ascertained, the remainder of the experiments will show that the trust model, when applied, is effective.

In order to calculate trust values and reputation values in a timely manner so that they can be accurately represented graphically, the cluster heads in FRANTIC will be programmed to calculate the reputation in cycles. This means that at the end of every cycle, all nodes send their trust data to the cluster head that then computes the individual reputations. However, safeguards must be put in place because by allowing all nodes to send their data at the same instant to the cluster head, the risk of congestion is increased and may result in packets being dropped, thereby resulting in the cluster head receiving inadequate or incomplete information. There is also the possibility of a second wave of congestion as the cluster head then proceeds to flood the cluster with the reputation tables. Not only is this a waste of resources, but it may also involve nodes receiving redundant information about nodes they do not have a one-hop relationship with.

The cycles are therefore staggered. This means that the cycles for the cluster head and member nodes are out of sync with one another. At the end of every cycle, the cluster head calculates reputations. However, it does not proactively broadcast those in order to minimise bandwidth usage. Instead, the cluster head stores the information and only replies to nodes that provide fresh data during the next current cycle.

Cluster heads only keep a maximum of four cycles' worth of data with the fifth one replacing the first and so on. This is shown in Figure 5.1.

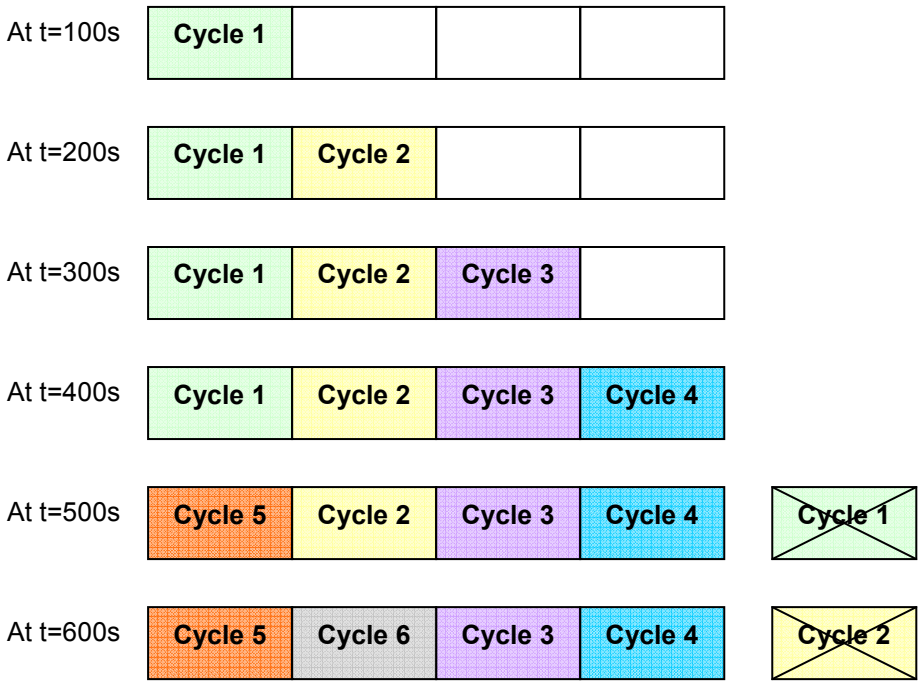

Figure 5.1: Cycles stored by the Cluster Head up to a maximum of four. The oldest set of data is then removed as new data comes in

For each recommendation it receives from a cluster member, the cluster head keeps the following information:

{node_id of trustor, node_id of trustee, trust value of trustee, time-stamp t}

Each time it receives a recommendation about a node, the cluster head stores it in the table for that cycle, within a given row for that node. Any other incoming recommendations that fall within the same cycle are then added to the same row.

At the end of the cycle, the cluster head calculates the average of the recommendations it has received for a given node and produces the reputation value.

5.2.2 Testing Roadmap & Experimental Plan

Experimental testing is key in order to determine the validity of a framework in performing assigned objectives and requirements. To that end, a set of experiments has been devised that aids this purpose. There are several stages in a trust framework and these relate namely to the trust formation within the network, its distribution to the other nodes and hosts and its maintenance over time. Also relevant to the trust framework are the issues of efficiency and resilience. As an aside to testing, some comparative metrics are also calculated within Section 5.3.6. The list of experiments, along with their high level aims, is given in Table 5.1.

Experiment Performed	**Aims of Experiment**	**Related Trust Aspect**
Section 5.3.1 Initial Algorithm Testing Experiment A	The aim of this particular experiment is to firstly check the functioning of the trust algorithm defined in Chapter 3. The experiment also serves to show the trust formation process within a single node, interacting within its cluster. It is possible to see the evolution of trust in response to the behaviour of the node as it deals with requests from its neighbours	Trust formation
Section 5.3.1 Initial Algorithm Testing Experiment B	This second experiment is designed to test a different feature of the trust algorithm within the same experimental set-up of Experiment A. The aim is to confirm that the framework can adequately reverse the trust build up process in response to negative feedback from a node's behaviour, i.e. that is able to respond to good as well as bad feedback on varying levels (e.g. selfishness or disruptive misbehaviour).	
Section 5.3.2 Distribution Testing Experiment A	Experiment A looks at the trust distribution process within a cluster. Having observed the impact of the framework on a single node's reputation, this experiment is to verify whether this impact is replicated across the cluster, i.e. whether reputation is being accurately propagated. In order to do this, all members of the cluster are monitored based on random inputs.	Trust distribution
Section 5.3.2	The aim of Experiment B is the next step up to determine	

Experiment Performed	Aims of Experiment	Related Trust Aspect
Distribution Testing Experiment B	the distribution of trust within the whole network. With the previous experiments depicting behaviours in a single node followed by that in a single cluster, Experiment B extends the concept to the whole network. The network chosen here is one represented by a travellers' web scenario with 4 interacting clusters. To aid graphical representation, the varying reputations of each of the 6 nodes in each cluster are averaged out such that the average cluster reputation is presented. The aim here is to show distribution and evolution of reputation across the network (this has already been shown for nodes and individual clusters, so the data is not shown again here for clarity).	
Section 5.3.3 Maintenance Testing	This experiment offers different data inputs at various cycles in order to show the trust maintenance process at work within a normal trust lifecycle. Part of this experiment is similar to what was implemented in Experiment B of Section 5.3.1, however, in this case, other inputs are also addressed such as the need for the framework to respond to non-cooperation and data loss as well as displaying the concept of trust natural decay. This experiment is performed on a single node and scalability of similar behaviour across the network is assumed from results obtained in Section 5.3.2.	Trust maintenance
Section 5.3.4.1 Node Analysis (Network Overhead Simulation)	Having investigated the trust lifecycle, the aim is now to test whether the framework operates efficiently. This is achieved by monitoring the network overhead generated from the trust processes. In this experiment, the impact is assessed at a node level.	Trust framework efficiency
Section 5.3.4.2 Cluster Head Analysis (Network Overhead Simulation)	The aims for this experiment are the same. However, in this case, the impact on the cluster head is noted as it is a very important element in the trust formation process and is likely to handle more traffic than its lower peers.	
Section 5.3.5.2 Threat Testing (Trust Auditing)	This experiment measures the response of the framework to misbehaviour – a term used encompassing various types of attacks (whether intentional or not) that would	Trust threat audit

Experiment Performed	Aims of Experiment	Related Trust Aspect
	result in detrimental impact to accurate reputational representation. The isolation of nodes indulging in misbehaviour is one expectation of the framework. Also of interest is the investigation of a node when faced with a temporary Denial of Service attack. The aim here is to show that the framework allows such nodes to rebuild their reputation, especially those that are active within the network.	
Section 5.3.6 Analysing Service Metrics	While these are not experiments as such, the aim is to show that it is possible to have an idea of the state of the network by using pre-defined metrics. Assuming similar data is available from other networks, these metrics can enhance comparability between models running on different types of architectures, where direct comparisons may not have been previously possible.	Trust framework comparability

Table 5.1: Experimental roadmap

With the experimental roadmap clearly defined, the actual testing and the results are shown in Section 5.3.

However, along with a list of the experiments and their aims, it is important to decide on an experimental plan, i.e. which sets out the list of Objectives or Requirements the testing refers to. In this case, the categories of experimentation are more relevant than the experiments themselves. Table 5.2 shows these categories and the Objectives and Requirements they seek to fulfil.

Category of Experiments	Objectives	Requirements
Trust formation	These experiments should meet Objective 1 by showing that a node can effectively derive an indication of trustworthiness based on interactions. Objective 4 can also be partially met if	Trust formation must show reasoning (Requirement 3) as well as trust information (Requirement 4).

Category of Experiments	Objectives	Requirements
	good behaviour is shown to be rewarded.	
Trust distribution	These experiments which denote the propagation of trust within the network must meet Objective 2, thus allowing for reputation information to flow in the network even for nodes that may not have previously met one another. Based on the way the model operates, these experiments must also fulfil part of Objective 3 – a set number of databases are used to store reputation information which are smaller and fewer. Objective 4 can also be partially met if good behaviour is shown to be rewarded.	This must meet Requirement 2 showing that the features of the model are scalable from clusters to the network and potentially larger networks. Distribution of trust also involves meeting Requirement 5 on trust recommendation and reputation. Requirement 11 on selective grouping can also be demonstrated here.
Trust maintenance	The aim here is to meet Objectives 1-4 as trust maintenance encompasses all 4 of them, including reputation and trust management as well as a system of rewards and exemption in the case of the cluster head.	Requirement 6 must be met by these experiments as it must show nodes being rewarded by performing well for the network. Requirement 8 on local repositories can be shown to be present here as the data moves between nodes and clusters. Can also show Requirement 11 as for trust distribution.
Network overhead simulations	These experiments are designed to meet Objectives 4 and 6. The efficiency of the system must be shown as well as the advantages over a fully distributed system.	This must show that the cluster head receives limited additional overhead if at all because of the incentives available to it (Requirement 6). The experiment must also meet Requirement 7, with a special experiment highlighting the cluster head and thus showing different classes of node. Requirement 8 must also be fulfilled as local repositories are key to better efficiency.
Trust auditing	Trust auditing relates to Objective 5 and	These experiments are key to

Category of Experiments	Objectives	Requirements
	the experiments are designed to show some resilience from the framework along with the existence of an appropriate punishment system.	highlight Requirement 10 on isolation.
Service metrics analysis	This analysis is designed to meet Objective 7, which is to enhance comparability of trust models.	This analysis is designed to meet an Objective relating to general models and therefore is not relevant to a specific requirement for this particular framework.

Table 5.2: Experimental plan

5.3 Experimental Testing

5.3.1 Initial Algorithm Testing & Trust Formation

Experiment A

Aim of Experiment: This experiment is conducted to mimic the trust formation process within a given cluster in the Travellers' Web Scenario. The trust information is distributed throughout the cluster.

Strategy: Participants in the Travellers' Web scenario would upon joining exchange unimportant data in order to create some interactions that would allow trust build up to start. Routing data at the network layer is just as useful in that respect and this is what is used in the simulation with NS-2. In order to monitor the trust formation process, the evolution of trust within one node is monitored.

Data Set Selection: Routing data is perfectly adequate for this use as the framework only needs to analyse the behaviour of the node in response to the traffic inputs. As a single node is being observed, static CBR over fixed links is the best way to control the experiment and provide targeted inputs to the node. Bursts of data flow are a good approximation of how data would be expected to flow within the scenario as

random requests are received from users of the network (for example a file download would result in a data burst across the node, followed by no data, if no other user requests are noted).

Methodology: The network is set up, with one cluster (Cluster A) as per Figure 5.2 below. Cluster A is defined as the set of participants within the Travellers' Web scenario that are looking to exchange information within 1 carriage. The experiment is performed within only one cluster at this point in time in order to ascertain how the trust evolves within a member node within the cluster. This is purely to verify that algorithms work as expected. Nodes 1-6 are the member nodes of the cluster with CH, designated as cluster head. At this point in time, cluster heads are assumed to have been designated and are a trusted entity in the network as the focus is on the trust distribution rather than bootstrapping.

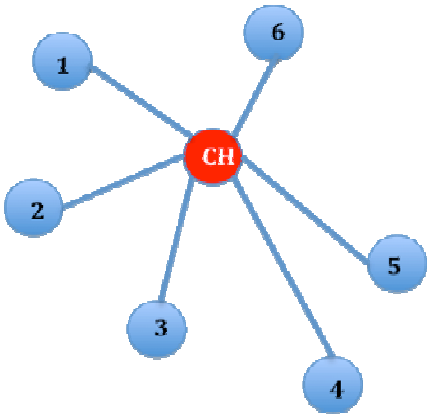

Figure 5.2: Intra-cluster testing set-up – a Cluster Head is designated with 6 nodes being monitored at any one time

As per previous specifications, the assumption that the model operates in cycles of 100s will be maintained. This assumption is valid for the rest of the experiments within this chapter, unless specified otherwise. The cycle of 100s is chosen for operational reasons here but cycle durations can vary depending on the level of mobility within the network and how fast trust needs to be established and distributed.

Nodes are expected to collect data on their one-hop neighbours as expected with the data being fed to the CH at the end of every individual cycle. In this particular experiment, it will also be assumed that all bi-directional links between nodes are fully functional and that no packets are dropped by any node. The effects of trust decaying over time are also ignored in this experiment such that the reputation of a node will remain constant if no trust data is collected about it within a given cycle.

This experiment is designed to investigate the effects of traffic flow on node 1 as viewed by the CH. The experiment will be run for 1200s or 12 cycles. During this period data flow through node 1 will be manually controlled by initiating the flows at specific periods in time. However, the amount of actual data flowing though the node within those specific periods will remain random. All packets sent are of fixed size, with UDP (User Datagram Protocol) being used to simulate CBR (Constant Bit Rate) traffic. Because these simulations are run on a low-level simulator, no actual service is depicted. Within the real life scenario however, the information exchange would be relevant to the service being requested and distributed.

Based on the design of FRANTIC, in this case, because the node under investigation is node 1, only nodes 2 and 6 will be involved in the trust rating process since they are the only 2 nodes being on a one-hop link to node 1 (excluding the CH which has one-hop links to all nodes within its cluster).

Using concepts defined within Chapter 3, the following traffic (Figure 5.1) was generated across Node 1.

Cycle (seconds)	**Data transfer**	**Flow start (time stamp in seconds)**	**Flow end (time stamp in seconds)**	**Role of node**
0-100	0	0	0	none
100-200	fid_1	125	150	router
200-300	0	0	0	none
300-400	fid_2	350	375	router

Cycle (seconds)	Data transfer	Flow start (time stamp in seconds)	Flow end (time stamp in seconds)	Role of node
400-500	fid_3a	410	430	router
400-500	fid_3b	450	470	source
500-600	fid_4	530	570	router
600-700	fid_5	620	660	sink
700-800	fid_6	710	730	router
800-900	fid_7a	815	840	source
800-900	fid_7b	850	870	sink
900-1000	fid_8	930	970	router
1000-1100	fid_9	1000	1040	source
1100-1200	0	0	0	none

Table 5.3: Input to trust framework for node 1 showing the effect of various types of traffic flows on the reputation of the node

Fid are the traffic patterns generated within the network. For example, the generation of packets between 125 and 150 seconds after the simulation start is known as fid_1. The role of the node is simply defined as its purpose within the selected fid. For example in Table 5.3, all the traffic data relates to Node 1. Different traffic patterns are sent through the node under investigation.

One can therefore see that during the course of the simulation, Node 1 behaves as source, router and sink altogether. When behaving as a source, the traffic originates from Node 1 and conversely when behaving as a sink, the traffic ends at Node 1. When Node 1 merely routes data for other nodes, its role is denoted as being a router.

Having run the experiment, the values outputted by the CH in its table for node 1 are then plotted against time cycles as per the graph in Figure 5.3.

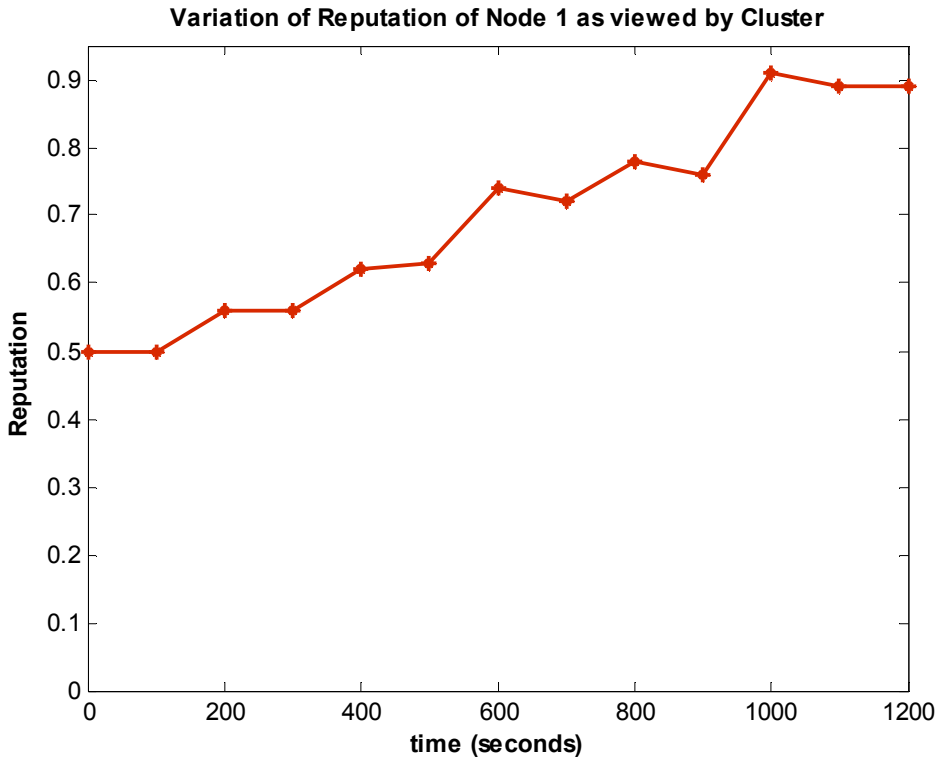

Figure 5.3: The variation of reputation of Node 1 following the inputs of various flows of traffic at each cycle

Findings: The above graph is a clear display of how node 1 is able to build its reputation in the eyes of its peers by behaving correctly within the network. It can be seen that the reputation of the node increases as it accepts traffic requests for others and falls slightly when it asks the network to perform work for it (this only happens in the absence of work being done for the network concurrently). For example when the node operates only as a sink or as a source during a given time period and does not perform any relaying duties that would have counteracted the negative impacts of the source and sink roles.

Therefore, in the case of new node that would have joined the Travellers' Web scenario, one can see how its reputation would have evolved if it had performed the mundane tasks associated with network routing. Having acquired a reputation and therefore a history within the network, it would then be able to interact more freely and increase the complexity of the information being exchanged with its peers and get rated accordingly.

There are a few points that should be noted:

- The reputation of the node does not change when there is no traffic passing through it. Although in FRANTIC no traffic would have been equivalent to its reputation undergoing *decay,* this feature will not be implemented in this particular experiment to highlight the effects of how the node's actions have repercussions on their reputation.

- Further analysis of the graph confirms that the cycles where the node gained the maximum in terms of its reputation were where it routed more traffic. Two long CBR-data packed cycles between 500-600s and 900-1000s confirm this hypothesis to be true as the graph shows the steepest gradients at those points.

- Furthermore, the effect of the node imposing its own load on the network (by acting as a source or sink) is noticeable, but because the node has a *drop rate* of zero and has otherwise behaved altruistically, such effects are minimal as can be evidenced from the low gradients of the drops in reputation. This is because the negative impact due to the *gain* of the node is less significant than the positive impact received by the network due to its *workload*. This was incorporated in the design so as to provide an incentive for nodes to cooperate to the network.

However, contrary to the *gain,* the *drop rate* has just as aggravating an effect on reputation as the *workload* has a positive one. This is illustrated in the following experiment with data being fed as per the table below. All assumptions remain the same at this point.

Experiment B

Aim: This is to investigate the effect of having a high drop rate on the reputation of a node. This behaviour mimics that of Disruptor nodes within the Travellers' Web and even the Disaster Recovery scenarios.

Strategy: The same set-up is used as previously, except that the nature of the traffic passing through the node under investigation and its subsequent behaviour towards that traffic are altered in order to meet the aim of the experiment.

Data Set Selection: This is the same set-up as in Experiment A.

Cycle (seconds)	Data transfer	Flow start (time stamp in seconds)	Flow end (time stamp in seconds)	Role of node
0-100s	0	0	0	none
100-200s	fid_1	130s	180s	sink
200-300s	0	0	0	none
300-400s	fid_2	310	350	sink
400-500s	fid_3	410	450	router (drop all)
500-600s	fid_4	510	560	router (drop all)
600-1200s (7 cycles)	0	0	0	none

Table 5.4: The traffic input to the trust framework in order to show the significance of nodes dropping traffic whether by intent or not

Methodology: Again, this is similar to the previous experiment. The difference is in the type of traffic being sent and in the behaviour of the node.

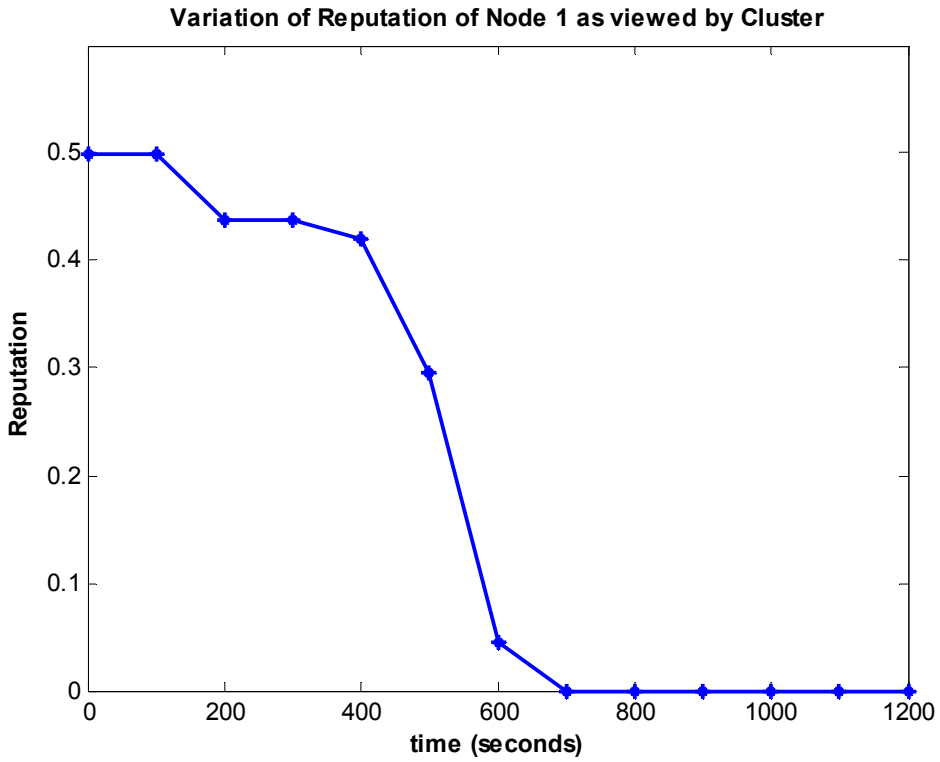

Figure 5.4: Variation of reputation of Node 1 after traffic input in order to show the effect of a node dropping packets

The overall result from this flow is displayed in Figure 5.4.

Findings: This experiment illustrates the characteristics of a typical Disruptor node in the Traveller's Web scenario. In the first two cycles where data is exchanged, the node only behaves a sink and does not forward any packets at all. This results in its reputation dropping as consistent with the algorithm punishing selfishness. However the drop is not steep because the selfishness of the node is not severe and also selfishness is not as bad as losing data although both would eventually result in reputation loss. The first two cycles therefore exemplify the Disruptor node when it is operating non-cooperatively (one of the two types of Disruptor nodes identified in the scenario).

The intentional Disruptor is modelled in the second set of cycles between 400 to 600 seconds. Here it can be seen that the node has been made to expressly drop all the data it has been sent. This is a worst-case scenario that is not likely to happen, unless

in the most severe of attacks or if a node has a technical failure. However, this is displayed here to show that the model can respond to nodes exhibiting high *drop rates* by lowering their reputations accordingly.

After 600s, the node's reputation falls below 0.1 within a single cycle, and it is banned from the network with its reputation thereby registering as from 700s onwards and no traffic being routed through or by it since it is no longer allowed to participate in the network. This is an insight of trust auditing at work (more on this in latter sections of this chapter).

These two experiments clearly demonstrate that the two types of stakeholders identified within the Travellers' Web scenario can be appropriately modelled. By analysing how reputation varies at the node level and the varying degrees of feedback the framework provides in response to altering behaviour, the parallel between individuals' behaviour within the scenario and the system response can be drawn.

5.3.2 Distribution Testing

Experiment A

Aim: Further to the experiments performed in 5.3.1, the aim here is to see whether the model appropriately scales the algorithm such that the reputation of all the nodes are accurately depicted as was the case with Node 1 previously.

Strategy: The same strategy is employed, except that this time the monitoring is extended to cover all 6 nodes within the network.

Data Set Selection: This is again the same as for Section 5.3.1. However, with regards to the number of nodes in the cluster, 6 nodes have been picked. This is because this is potentially the maximum number of people that would be in close proximity for data exchange and is derived from the Bluetooth piconet design where a master can only have a maximum of 7 slaves and therefore 6 users were chosen as being just under the maximum.

Methodology: In order to do this, the same cluster of 6 nodes (Figure 5.2) is set up with a cluster head. The experiment is then started as per section 5.3.1 using the same assumptions. However, instead of focusing on only one node, traffic is generated across the whole cluster.

In this case, random flows of traffic are generated across all nodes so they may act as routers, sources and sinks during various cycles. The experiment is run for 1200s as previously and the reputation of all nodes as calculated by the cluster head is then plotted as a function of time.

Findings: Figure 5.5 shows the variation of reputation within the cluster. This shows that in a particular cluster, trust relationships exist and evolve based on the behaviour of nodes previously explored in 5.3.1.

Different nodes within the cluster see their reputations evolve based on the random flow of traffic they experienced.

In this experiment, 4 nodes were classed as good nodes within the scenario (as per the 1st experiment in 5.3.1) and 2 nodes set up as Disruptors (as per the 2nd experiment in 5.3.1). Of the two, one was a clear non-cooperative node (Node 4) and the other was an intentional Disruptor that also attempted to masquerade as a good node by temporarily contributing to the network (Node 3). However, as can be seen from Figure 5.5, the framework effectively distinguishes the Disruptors from the good nodes, with a clear disparity between the reputation values at the end.

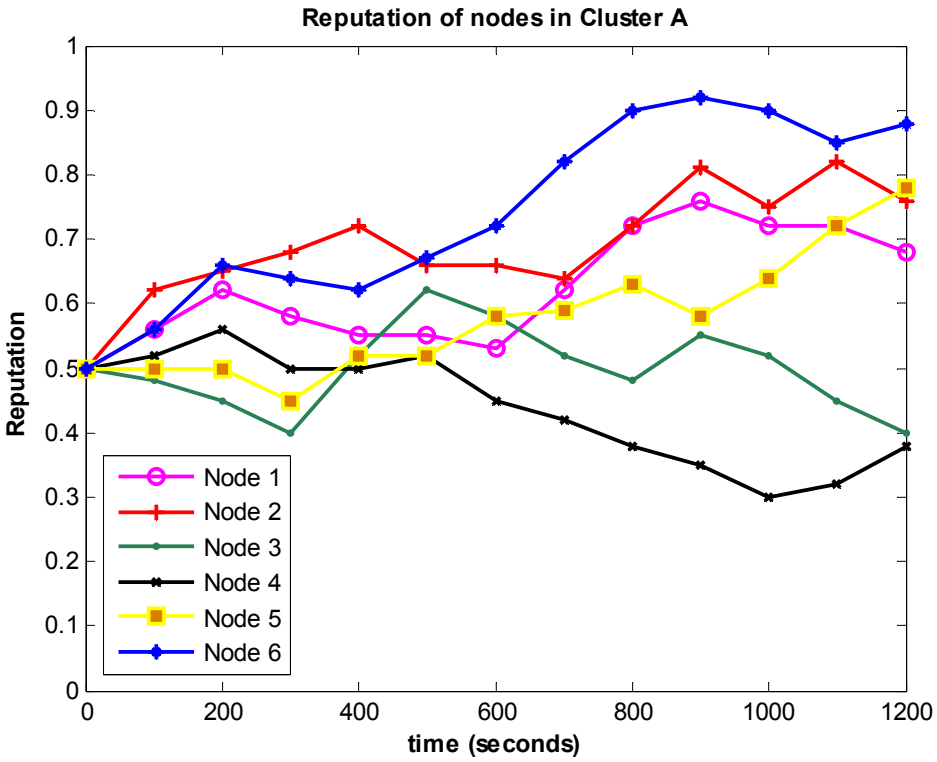

Figure 5.5: Reputation of nodes within cluster A with 6 nodes depicted in the cluster being fed very differing random traffic flows

This experiment therefore not only shows that a cluster can evolve into its own mini trust network but also that ultimately the framework is able to recognise Disruptors based on their actions. This ability is highly relevant in both scenarios explained in Chapter 4. However, it should be noted that here, only the evolution of the trust has been explored. The rule-based mechanisms that punish misbehaving nodes is not switched on within this set of experiments.

Experiment B

Aim: Having ascertained trust distribution within a cluster, the natural progression is to investigate the reputation model at work within a plausible actual scenario. This would involve looking at the propagation of trust within several clusters concurrently.

Strategy: In this case, sticking with the Travellers' Web scenario, a simulation of a 2-carriage train is undertaken with the formation of four clusters each containing exactly 6 nodes. A 2-carriage train is assumed so that inter-cluster communication is possible and within range. The number of participants equates to 24 which would be an acceptable number in a real life situation for an off peak suburban train. Again, the aim is to use network routing information in order to deduce the trust information that can then be superimposed on the network in order to create trust relationships.

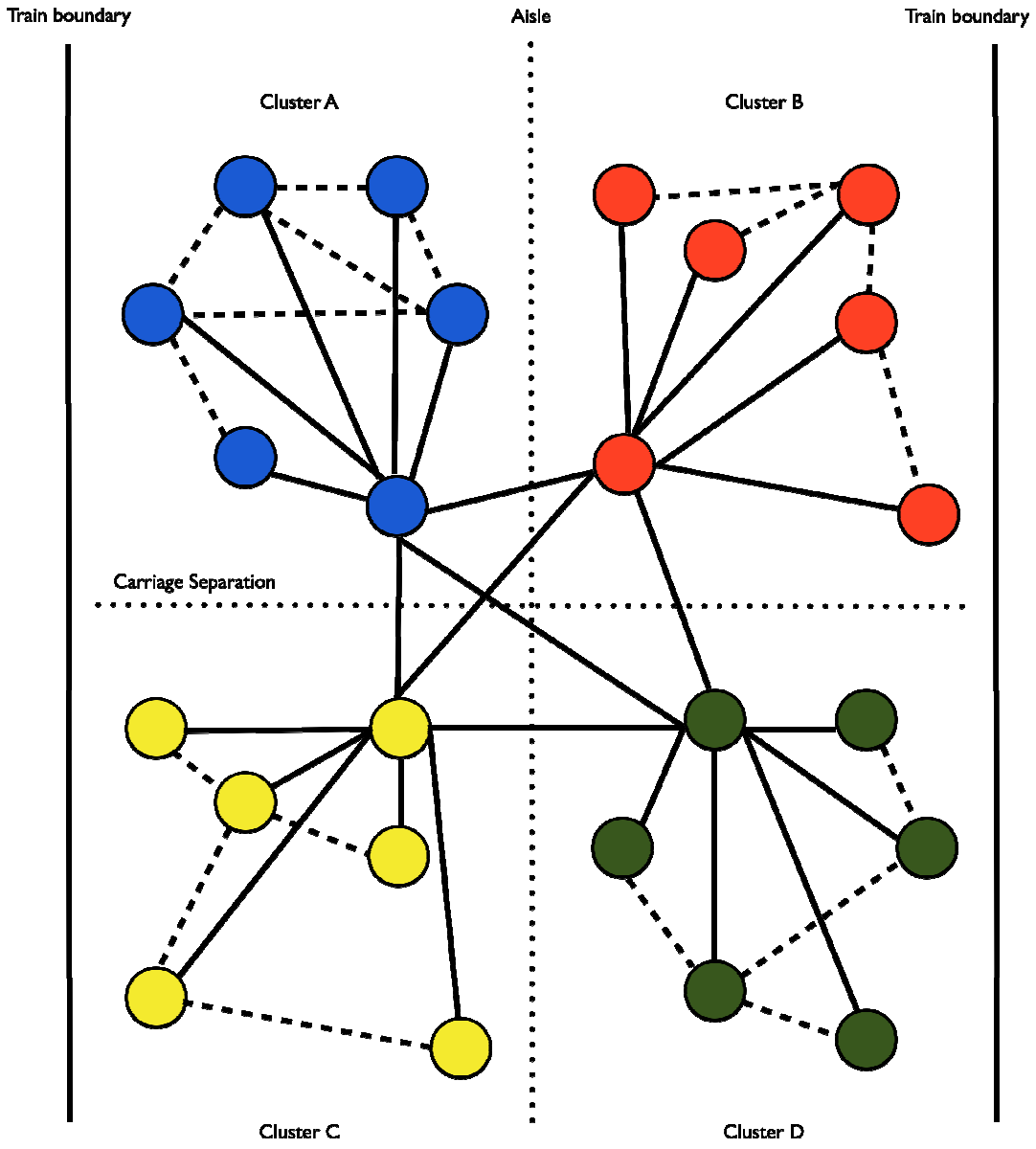

Figure 5.6: Representing the layout of clusters in the Travellers' Web scenario. For clarity, the traffic patterns are only shown at a particular moment in time. Non-CH traffic is dotted

Data Set Selection: This is largely the same as for Section A. However, the number of nodes being used here is designed to represent a typical scenario in a train carriage. While there is no expectation of there being only 24 passengers in 2 carriages of a train, it is more plausible to suggest that only around 24 passengers may be making use of the localised networking abilities of their devices.

Methodology: Again, the same methodology is employed as previously with each node in each cluster having varying types of traffic running through it and the cluster head calculating the subsequent reputation for each node. For convenience, these types of traffic are not depicted here as it would involve the reproduction of 24 separate tables. What is under investigation here is the variation of reputation over time within the clusters. In order to have a quick preview of the reputation of a given cluster, statistical averages of the reputations of all the nodes at given points in time are performed and these are then recorded on a graph.

Findings: Figure 5.7 shows the average reputation of nodes per cluster. While specific inferences are not possible from statistical averages, the plot does show that reputation varies across all clusters. All flows have been manually initiated between nodes using random CBR and some nodes have been purposely designated to have high *drop rates* in some cases and high *workloads* in others so as to have fairly defined statistical representations.

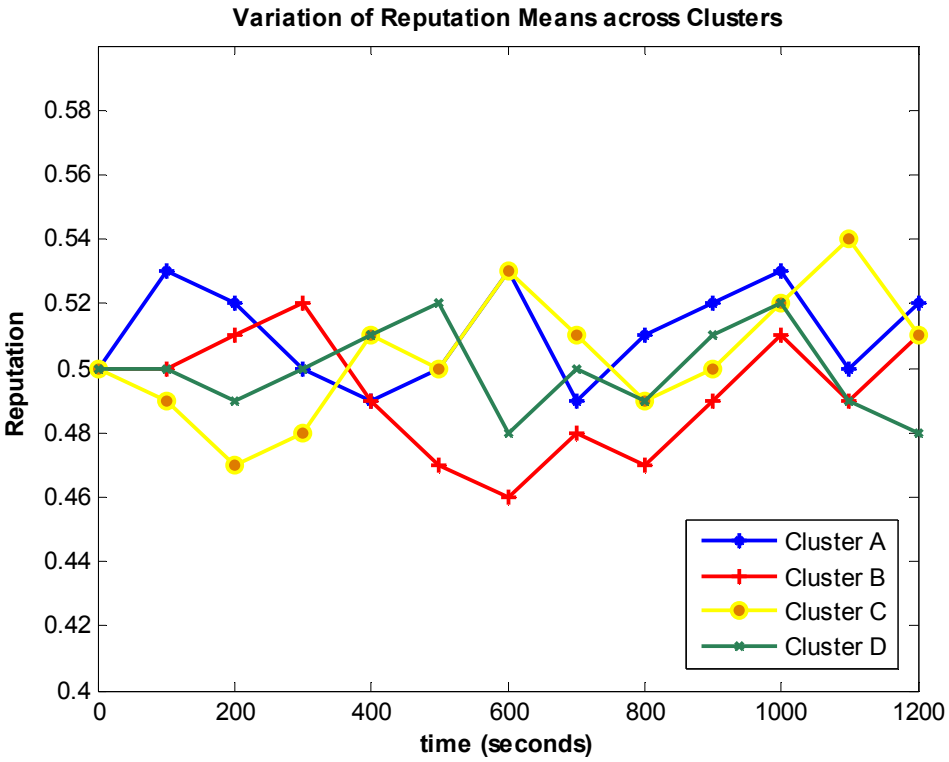

Figure 5.7: Statistical Averages of Cluster Reputations based on the simulations of the Travellers' Web scenario

The clusters all have different statistical averages purely due to the random nature of the traffic flows and the different conditions imposed on some of the nodes (to act as a particular type of stakeholder). This results in sometimes large variations between cycles. Going back to the Travellers' web scenario, one could infer that Cluster A for instance would be able to exchange more sensitive information between the nodes than Cluster B, especially within the middle part of the simulation.

5.3.3 Maintenance Testing

FRANTIC needs to be able to perform trust maintenance, in that trust values need to be updated accordingly over time and it should also be able to respond to threats or non-cooperating nodes. There has already been one instance where the cluster head bans a node whose reputation happens to drop to less than 0.1 within a single cycle (Section 5.3.1). However, there are also other instances of maintenance where the

mechanism will have to react to reputation variations in nodes without necessarily banning them.

Aim: It is a vital factor of FRANTIC that it should be able to emulate, to a certain degree, the concept of humanised trust, i.e. nodes that have behaved "badly", i.e. by overloading the network, losing data or being uncooperative should be allowed to redeem themselves. In this particular set-up the concept of trust decay over time is also factored in.

Strategy: A single node in the network is investigated while the mechanisms of the network that deal with trust decay as well as allow nodes to make up their reputation after they have been damaged are switched on. For this experiment, the Disaster recovery scenario is used with the node being targeted as a member of the Helper stakeholder group. Helpers are generally nodes that are on site to help Victims and therefore deemed to seek to behave selflessly at all times.

Data Set Selection: This is the same as for Section 5.3.1 as the experimental set-up is similar.

Methodology: This is as before and the various inputs to the network are detailed in Table 5.5.

Cycle (seconds)	Data transfer	Flow start (time stamp in seconds)	Flow end (time stamp in seconds)	Role of node
0-100s	fid_1	30	60	source
100-200s	fid_2	125s	150s	router
200-300s	fid_3	260	300	sink
300-400s	fid_4	350	375	router
400-500s	fid_5	410	460	router(drop all)
500-600s	fid_6	530	590	router(drop all)
600-700s	banned	banned	banned	banned
700-800s	0	0	0	none
800-900s	fid_7	815	840	router

Cycle (seconds)	Data transfer	Flow start (time stamp in seconds)	Flow end (time stamp in seconds)	Role of node
900-1000s	fid_8	930	970	router
1000-1100s	0	0	0	none
1100-1200s	0	0	0	none

Table 5.5: Input to trust framework for node 1 in order to determine the response of the framework when aiming to maintain trust

Findings: The Helper node is initially made to behave as a source, router and sink in order to make sure it is working properly and at the 5th cycle is made to start dropping packets in order to lower its reputation. This happens up until the point the node passes the 0.25 threshold value for the cluster head, at which point it receives a ban. As per the model, this is only a temporary ban that lasts for only one cycle. This drop in performance from the node (by dropping packets) which lowers its reputation needs to be modelled because frequently in disaster recovery scenarios, Helper nodes will find themselves in situations where they are overloaded, whether it be at a human level (having to attend to victims) or at a networking level (receiving too many requests for information transit). If there is no chance for a node to redeem itself, then the framework will quickly isolate said node from the network permanently and this can seriously hamper any recovery operations.

Instead, as can be seen in Figure 5.8, the ban results in a break in the graph whereby no values are recorded for the reputation of Node 1. By banning the node temporarily, the system provides it with an opportunity to address any internal or external pressure that may have been affecting its performance. These temporary "breathers" may well be necessary in a high-pressure environment as a disaster recovery scenario is likely to be.

Once the node completes its ban cycle, it is reintegrated into the network and its value is reset to 0.25. In its first cycle after being reintegrated, this node experiences no traffic. However, this may not necessarily mean that it is a sign of non-cooperation. It could well be because of its low reputation value, other nodes may not be keen on using it as a router. To cater for this is why in this is a special case,

the cluster head does not apply the decay principle to the reputation calculation as the node has only just come back from a ban. Besides, in order to apply a trust decay decrease, the cluster head would require the difference in reputations from the previous interaction which it does not have as the said node did not have any interactions during the time it was banned.

As it goes through the next two cycles (between 800-1000s), the Helper node redeems itself by increasing its workload and therefore increasing its own reputation. During the last two cycles however, it does not experience any traffic and this time the cluster head ages its reputation by factoring in the decay principle and its reputation value is seen to fall.

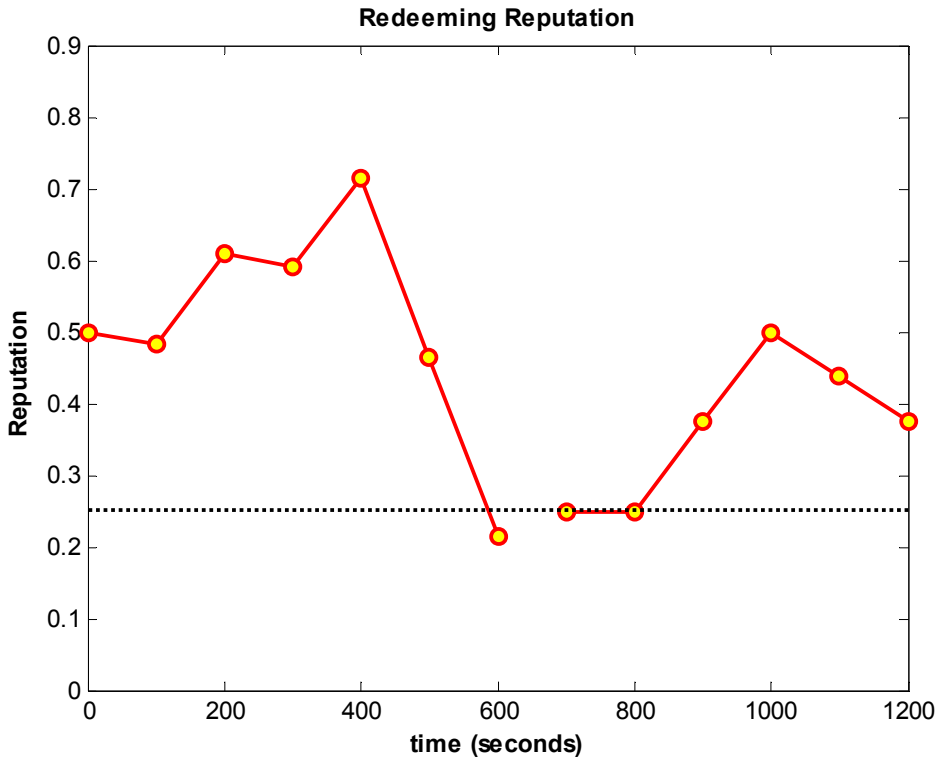

Figure 5.8: Redeeming reputation – Node 1

N.B. The break in reputation starting at 600s indicates that the node was banned.

Having thus ascertained that the model responds as required to a host of different data inputs administered under controlled conditions in both sets of scenarios, the impact of other parameters can now be studied such as verifying how accurate the

model is when detecting rogue nodes, or how effective it is in using network resources.

5.3.4 Network Overhead Simulations

One of the primary considerations of adding a framework onto an existing network is the effect, positive or otherwise, that addition has. Often, by adding functionality or features, there is a certain compromise to be made in terms of reduced resources available to do other tasks, thereby resulting in some networking tasks not being performed (such as packet drops). This then affects the overall performance of the network and therefore would render the addition of any feature highly questionable.

5.3.4.1 Node Analysis

Aim: In this case, it is proposed that the effect FRANTIC's architecture has on the network should be verified in 2 folds.

Strategy: First the impact of any additional overheads created at the node level, then at the cluster level, is studied. In this way, it is possible to isolate any congestion points and possibly refine the model to address such issues. The impact on the overall network will simply be a scaled up version of the average impacts on a cluster. Because of the topology of the design the largest concentration points will be around the cluster head, if at all. As stipulated before, this is mitigated by staggering node trust reports at various stages of a particular cycle.

Data Set Selection: The arguments for data selection are the same as for Section 5.3.2, Experiment B. The other criteria noted here are the pause times, which denote the amount of time a mobile user is expected to stay stationary – just under 2 minutes was deemed a good approximation, especially for someone moving in the pattern of the Random Waypoint Model. The node ranges are 250m, allowing some nodes to get out of range if required. The node speed, while high for a human user, is chosen

here for experimental purposes only. Slower speeds, coupled with the nodal range, may not have accurately reflected mobility, given the defined cycle and pause times.

Methodology: For this part of the experimentation, the following parameters are assumed within NS-2.

Simulation time: 1200s
Cycle time: 100s
Number of nodes: 42 (6-7 clusters depending on prevalent configuration)
Simulation area: 1000m x 1000m
Pause time: 100s
Movement: Random Waypoint Model
Nodal range: 250m
Capacity: 2Mbps
Application: CBR
Speed: 10 m/s

These values are typical of currently available hand-held devices and the simulation area has been set wide enough so they can reflect both the Travellers' Web and Disaster Recovery scenarios.

The overhead is also defined as the number of routing packets (received/transmitted or issued) through a node. The corresponding reputation overhead is then the sum of the routing overhead and the number of all reputation-only packets.

The simulation was carried out as required above for 1200s with varying loads of traffic through the 12 cycles that comprise this experiment. This was a generic simulation that would be equally applicable to all stakeholders in the two scenarios depicted in Chapter 4. The graph below pertains to the traffic through one node only within a cluster. Data was analysed at the end of each cycle as opposed to on the fly.

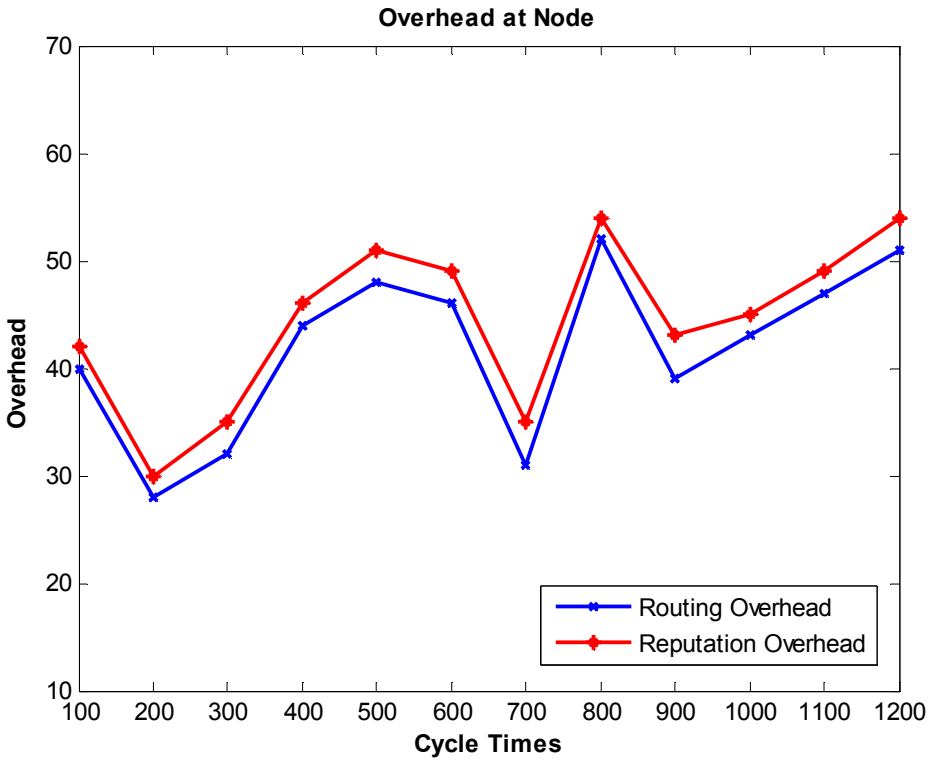

Figure 5.9: Overheads generated by the framework when determining reputation at the node level. The reputation overhead includes the routing overhead

Findings: As far as the node is concerned, it can be seen that the overhead added by the trust mode by virtue of its reputation mechanism is not appreciably higher than the routing overhead. The reputation overhead values include the routing overhead, so it can be seen that the additional overhead added by the reputation mechanism is only a fraction more than that added by the routing procedures alone. This means that the node generates minimal additional overhead when operating within the trust framework. Furthermore, these simulations were carried out with test packet sizes. In real life scenarios, especially in the Travellers' Web scenario, the size of the service packets being transferred is likely to be much higher than the size of the control packets for the trust mechanism anyway, so the fractional increase the framework introduces will be even lower. Hence, this particular simulation should be seen as a worst-case scenario as far as the normal node class is concerned.

It should be noted that the overhead is not uniformly higher at all cycles because some cycles have varying traffic patterns with nodes coming in and leaving the

network. *Hello* messages are also included within the overhead calculation for the reputation model. It should also be noted that this is for the overhead at the normal node only, not the cluster head class of nodes. This will now be dealt with in the next section.

5.3.4.2 Cluster Head Analysis

Having looked at the traffic flowing through one single node, it follows that the rest of the nodes within a cluster will follow the same pattern; in fact, this has been confirmed via further simulations, but in view of the results being very similar to Section 5.3.4.1, these have not been displayed here. The cluster head however is different because it operates on different terms in contrast to its member nodes.

Aim: To investigate the effect of the framework on the cluster head with regards to overheads generated.

Strategy and methodology: As for normal node, but with the algorithm modified since as mentioned previously, cluster heads do not need to perform routing for other nodes, unless absolutely necessary. They are not penalised with the *gain* system in place for other nodes since they are required to do other duties for the cluster, namely storing reputation tables, calculating and updating new reputation values, distributing updated values to its members and monitoring and taking corrective action against non-cooperative or misbehaving members.

Data Set Selection: This follows the same argument as for Section 5.3.4.1 for the nodal analysis.

Findings: The following graph therefore gives an indication of the routing vs. reputation overhead for the cluster head. Again this is a generic simulation applicable to both scenarios in Chapter 4.

As it can be seen, the reputation overhead in this case highly outweighs the routing overhead, at least in a much more pronounced manner than was the case for the normal node class. This may seem like a gross disadvantage of being a cluster head. However, in order to set things in perspective, the average reputation overhead of all the member nodes is then plotted in comparison. The overall reputation overhead (which includes the routing overhead) of the cluster head is lower than the overhead of its member nodes. This is to be expected as the cluster head has other duties to perform and having to operate in lesser traffic is an incentive on one hand and also makes sure the cluster head is not congested. It is important that cluster heads, once established are relieved of routine routing issues wherever possible in order to preserve their resources towards the computation and storage of the trust data within their respective clusters.

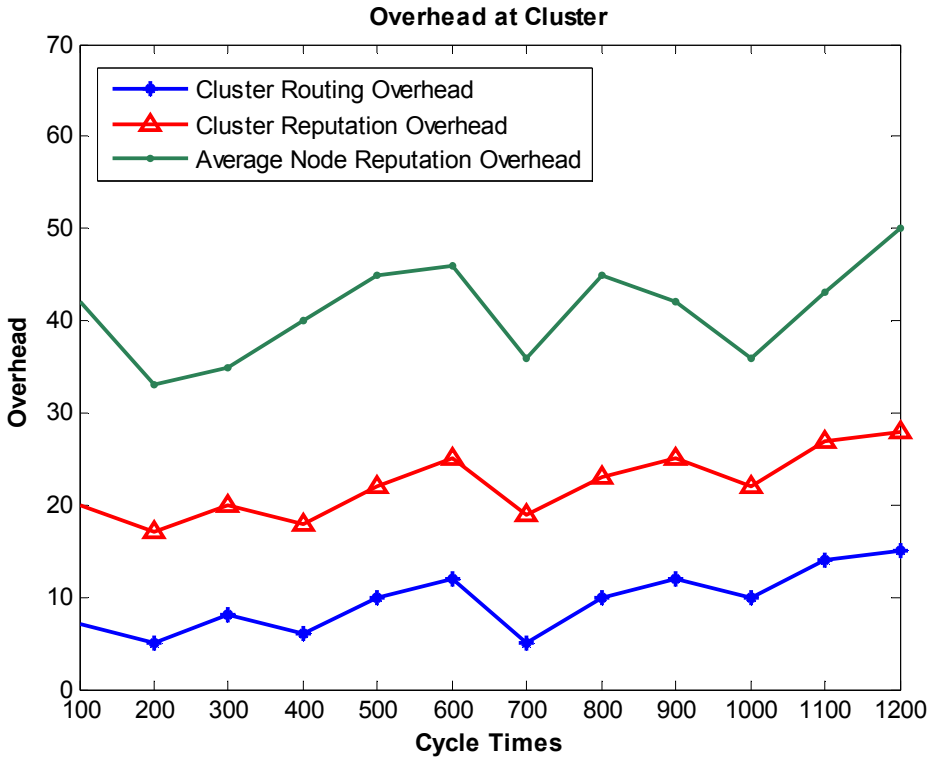

Figure 5.10: Overheads within clusters when seeking to determine reputation information. The reputation overhead includes the routing overhead for the Cluster Head

5.3.5 Trust Auditing

As described within Chapter 3, trust audit involves the actions taken by the framework in order to counteract any actions by nodes that may be seen to be detrimental to the proper operation of the network. Such actions may not always be misbehaving in nature but most may result in taking the network from its stable phase to an unstable one, which is why it is essential for it to develop some safeguards.

FRANTIC was designed with node cooperation in mind and aims to emulate the natural tendency to trust which is normally inherent among most human beings. It is therefore not claimed that the model is immune from all forms of attacks – in fact it is very hard for any model to be able to make such a claim, even those using hardcore security authentication and encryption mechanisms, aided by trusted third-party authorities. Security and trust can almost always be breached if the attacker is relentless enough, so the aim here is not to deflect all forms of attacks, but merely to ascertain that the model can identify early symptoms of impending node, cluster or network failure and take measures to remedy the situation.

5.3.5.1 Threat Analysis

There are several scenarios which wireless networks may come across that may be detrimental to its operation. Some of these are not intentional (such as congestion), but should nevertheless be remedied. The main threats to an ad hoc network are briefly described below.

5.3.5.1.1 Lying Nodes/Black Hole Attacks

This occurs when misbehaving nodes propagate false information within the network. For instance, during a route discovery process, they may choose to send route reply acknowledgements for routes they do not have. What does happen is that

once the packets are routed through that node, it cannot forward them on and simply drops them.

It is expected that FRANTIC will be able to deal with such situations because the one-hope neighbours of the misbehaving nodes will be able to detect that packets have been dropped en-route and they will lower their reputation ratings accordingly. Within the next cycle, depending on its *performance* the misbehaving node will then either be temporarily or even permanently banned. Because a misbehaving node will most likely also not forward any packets, this will compound the effect of its *drop rate* on its *performance* resulting in quicker punishments.

5.3.5.1.2 Intentional Packet Drops

In this case, the node continuously drops all the packets it receives. There is no attempt at lying in order to induce fake trust, the node just drops all packets and does not send route error messages when it does so. This type of behaviour is again easy to detect in FRANTIC based on the close monitoring of its one-hop neighbours. This will then result in the node being punished via a ban.

5.3.5.1.3 Denial of Service Attacks

In this instance, a node or a series of nodes may target a particular victim within the network and flood it with many packets in order to generate a denial of service attack. While such attacks can be hard to detect, the way in which the framework's algorithm is designed caters for the fact that nodes may node flood a network for a sustained period of time.

There are two reasons why the motivation for flooding is reduced. The first is that there are equal benefits to be had in the attacked node doing an accelerated workload. In fact, by flooding the node, they will be increasing its trust rating. There will of course come to a point where the link capacity may be exceeded and the node will start to drop packets, but this will be outweighed by the number of packets it has

forwarded. So, the likelihood of the attacked node being excluded at the next cycle is rather small. Furthermore, denial of service attacks will need to originate from nodes and will artificially increase their *gain* without adding appreciable *workload,* assuming the majority of their time is spent operating the attacks. This will result in the reputation of the attacking nodes being reduced. A prime candidate for denial of service attacks are cluster heads purely because of the central role they occupy within the cluster's reputation hierarchy. However, cluster heads are not meant to be included in natural routing paths unless absolutely necessary, so a lot of the requests to use the cluster head as a forwarding node will actually be deflected to alternative, possibly longer, routes. Even if the routes are longer, and may affect effectiveness of the network to a certain extent, it is necessary to ensure that the cluster head does not become the weak point of the framework.

Another effect of a node being under attack is that the surge in forwarding requests it receives will eventually start to reflect upon its limited resources. In other words, its battery level may start to drop or it may not be able to perform internal tasks due to excessive processing power being consumed by the increased routing. In this case, the node has a choice of shutting itself down as a router temporarily. While this may reflect selfishness and may result in the cluster head decaying its reputation over the next cycle, the attacked node will be relatively immune to this if it resumes normal operation after the maximum number of four cycles has passed and it is weeded out of the network. This "pause" time in its routing operations will cease all denial of service attacks and it may then resume normal operation, safe that the attacker may have then moved on, up until the point where the latter's reputation is driven down to the point where is then banned or ejected from the network.

5.3.5.1.4 Collusion Attacks

These types of attacks are performed by several nodes operating together in order to fulfil a common goal. An example of a collusion attack is when several nodes provide fake recommendations about each other, in order to artificially raise their reputation and then proceed to launch an attack on the network once they are trusted.

Another form of collusion is when rogue nodes work together in singling out a victim node and generating fake recommendations about the latter and reducing its reputation value such that it is then ejected from the network or shunned by its peers. They then proceed to do the same to another node until they then control a particular part of the network.

Collusion attacks are the hardest form of attacks for the model to deal with. In fact, because of the nature of ad hoc networks, any attempt at being effective in counteracting a certain type of attack invariably renders one vulnerable to a different type of attack. This is the case here. While FRANTIC is expected to fare relatively well in the previous types of attacks described, it may not be able to repel a very strong collusion attack made of several nodes. However, this is not only representative of this model. Many types of ad hoc networks and trust frameworks would fail if faced with a certain amount of colluding nodes. This is because colluding nodes are very hard to detect and their effects are further amplified in nodes that base their operation on recommendations from trusted peers. While the network is able to weed out singular outbursts of rogue nodes, a coordinated attack may find a cluster overwhelmed in terms of numbers. In any case, if the number of rogue nodes closely matches the number of altruistic nodes, then there is a higher chance for the network to fail purely because of the specific coordination of colluding nodes.

There are other types of attacks that do happen in ad hoc networks but the above are the most common types that will be perpetrated within self-starting networks such as FRANTIC. This is because the lack of rigid authentication protocols in networks without access to a certified central infrastructure allows such forms of attacks to happen.

5.3.5.2 Threat Testing

Aim: This is to measure how effective the model is when faced with attacks detailed in the previous section and what its accuracy is when trying to detect misbehaving nodes from honest ones.

Data Set Selection: This is again the same as for Section 5.3.4.1, with the exception of the variable pause time, explained within the Strategy section. A larger number of nodes are used this time with more clusters as there is the need for some of them to be misbehaving nodes and thus to be eliminated from the network.

Strategy and Methodology: In order to simulate threat scenarios, the same simulation set-up is maintained as in Section 5.3.4. The difference now is that a variable pause time is set and nodes are allowed to move according to the random waypoint model, i.e. at a constant speed determined randomly using a uniform distribution between 10 to 20 m/s. This is more suited to the Disaster Recovery scenario where node movement in and around the disaster area and to and from the safe zones are to be expected, unlike the Travellers' Web where similar movement is restricted by the physical settings of the scenario.

The number of nodes is set to 60 this time allowing for the formation of about 10 clusters depending on the configuration of the network at any particular point in time. CBR is still used, with packets being sent at a rate of 1 packet every 0.25s and a constant bandwidth of 2Mbps.

In the first scenario, a varying number of Disruptor nodes will be implemented which drop most of the data packets they are sent, typically anywhere between 80% (lower band) to 100% (higher band). The routing protocol being used is DSR, with no routing packets dropped at this point. This is so as not to make detection of the misbehaving nodes too obvious. If the misbehaving nodes drop all packets they are sent (routing and data packets included), then they will be weeded out of the network by the cluster head very quickly as their reputation will rapidly fall below a threshold level within a cycle or two.

Findings: The following experiment shows the results obtained from the average of 4 series of 10 simulations performed, with a pause time of 50s for the nodes, and increasing the number of misbehaving nodes in order to verify the amount of time it took the network to detect and isolate them. Figure 5.11 shows the results of those simulations. They represent the totality of the nodes in the network and are not depicted on a per cluster basis.

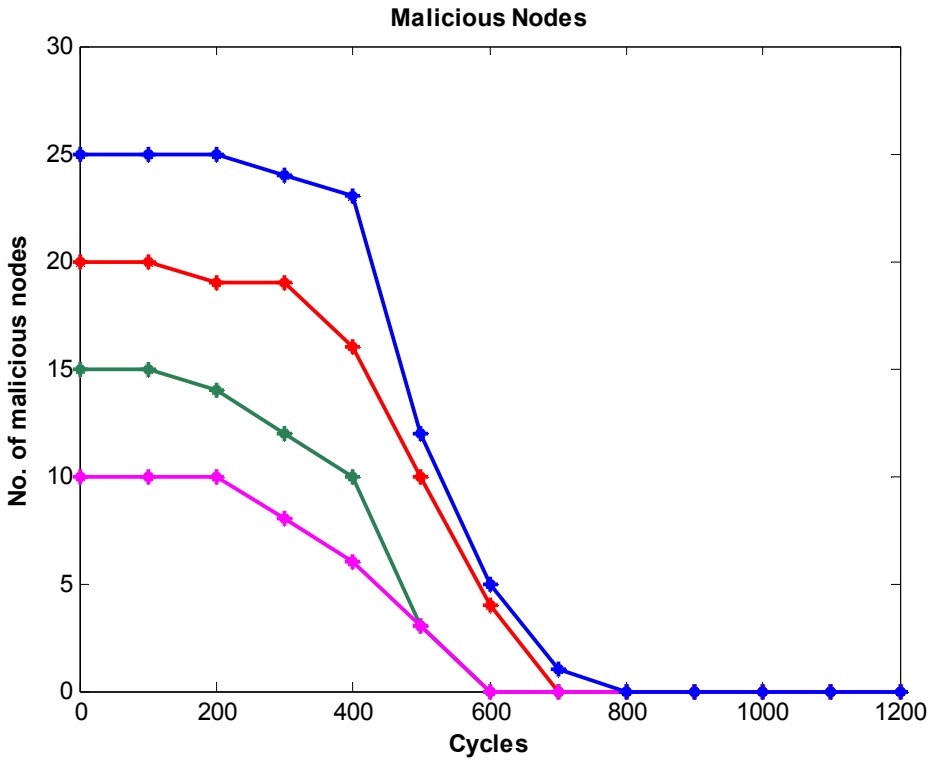

Figure 5.11: Banning of misbehaving nodes by the framework after their reputation is determined to fall too drastically thereby triggering the rule-based mechanisms

This graph shows that it takes roughly about 6 cycles for all nodes to be properly eliminated from the network. It should be noted that it has been assumed that there is no collusion between nodes and that the Disruptor nodes do not drop all packets. Because their dropping rate varies and because they perform some *workload* in terms of routing for the network, they are able to sustain a good reputation for longer. However, these are extreme situations; misbehaving nodes will often drop all packets indiscriminately resulting in a quicker ban from the network.

The next set up is that of a single Helper node within the network undergoing a denial of service attack. The graph shows that although the reputation of the node decreases at times and it even undergoes a period of reputation decay for around 200s to protect itself from a surge of forward requests, it is still able to strive in the network. The only drawback would have been a drain on its resources which FRANTIC cannot dictate. It is the user's choice and an indication of their freewill as to how long they wish to tolerate heightened data rates before refusing to forward any more packets.

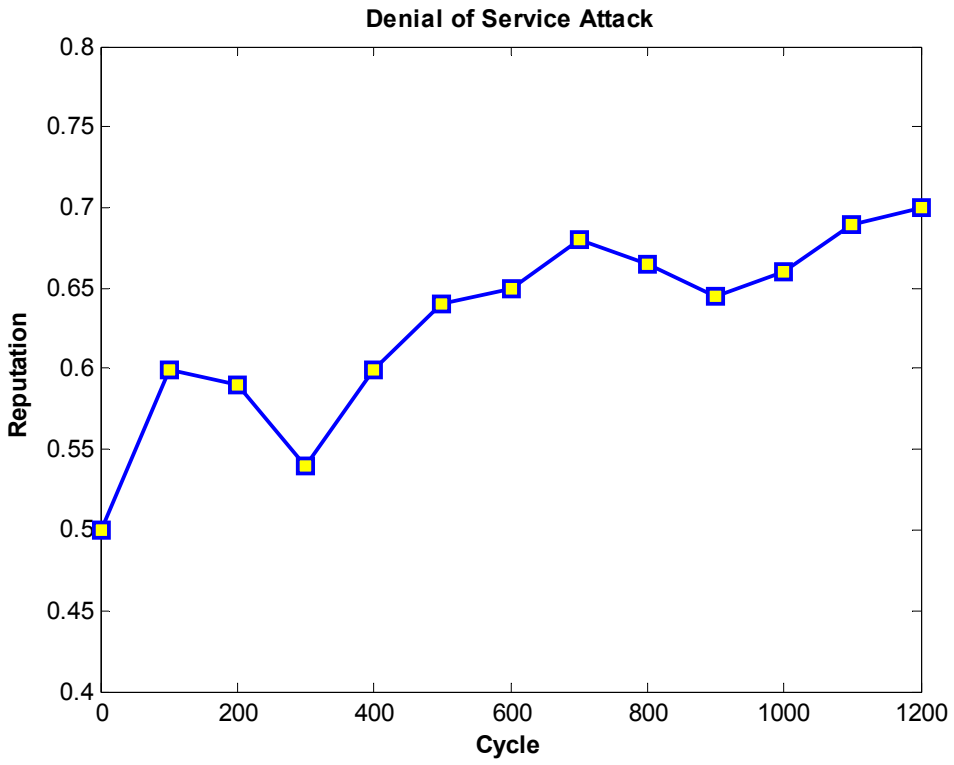

Figure 5.12: A Denial of Service Attack showing that the node undergoes a fall in reputation as the attacks force it to drop the traffic flowing through it

It should be noted that although the Denial of Service attack appears to have a positive effect on the reputation of the node, this has come at a cost. It has lost its resources in maintaining a high data rate, such that packet drops did not have a big influence on it. Between cycles 100 to 300s, the node dropped more packets than it forwarded, resulting in its reputation falling. At the end of cycle 700s, the node has refused any more packet forwarding. This means that no trust information is reported

to the CH during the next two cycles, resulting in a natural decay of its reputation. When it resumes normal operation at 900s, the node is able to increase its reputation by behaving altruistically.

The above increase in reputation however would not have been possible without honest nodes reporting about the correct packet forwarding taking place. This then assumes that at least the recipients of the forwarded packets (the node's one-hop neighbours, other than the attacker/s) are honest in reporting trust ratings to the cluster head. If the neighbours are also part of the network of nodes perpetrating the denial of service attack, then it is highly likely that the node will not survive in the network since this would be akin to a collusion attack. Therefore, another important factor to consider is the relative ratio of Helper nodes to Opportunists and Disruptors. Generally, this ratio is favourable to the Helper nodes but in those situations where this is reversed, then because of collusion, it is very likely that the likelihood of reaching good nodes decreases.

5.3.6 Analysing Service Metrics

In Section 4.3, several service metrics were identified that allow for a snapshot evaluation of trust frameworks to be performed. These were both service environment and service specific centred.

In order to verify whether the metrics indeed paint an accurate picture of the state of the network at a particular time, the KPIs were calculated for a few of the experiments previously performed at specific time periods and the relevance of the KPI to the state of the network was determined. Because the experiments were controlled, the service environment metrics are not required, so the focus will be on the service specific metrics. The observations are summarised in Table 5.6.

Service Metric	**Value**	**Time Stamp (seconds)**	**Network State**	**Observations**
Node Analysis Scenario from Figure 5.5 (Node 3 only)				
S_r	0.82	300	Discovery to Stable	In this initial stage, it can be seen that the success rate of route requests is relatively high as nodes seek to discover one another.
S_{rt}	0.98	500	Stable	At this point, the network is stable with established trustworthy routes being chosen.
I_{mr}	0.16	N/A	Stable	In the absence of other pressure from its environment, the impact on the framework on the measurable improved efficiency of routing can be determined.
Cluster Analysis Scenario from Figure 5.6				
QR (Quick Ratio)	0.95	300	Stable	This is the QR for Cluster B. It is a very high ratio and this is reflected in the higher average mean reputation values. The QR does not distinguish between different types of "bad" nodes, so any non-performing nodes are automatically classed as "bad".
QR	0.83	500	Stable	Again for Cluster B. This shows the ratio has dipped. This could be an indication of a network issue (overload) or the presence of non-cooperative or misbehaving node. A lower reputation value confirms this to be the case.
QR	0.91	700	Stable	This is the QR for Cluster A. With a high QR, the average reputation may be predicted to be higher – however, actual results show that it is lower than expected. This is an important distinction to make – the QR predicts the trust service availability of the network, not instantaneous reputation values, although often, both track each other. In this case however, although the trust service ability

Service Metric	Value	Time Stamp (seconds)	Network State	Observations
				of the network is predicted by the QR as favourable, the mean reputation is lower than expected because the network has just come out of a cycle where "bad" nodes may just have been weeded out. Nevertheless, the QR accurately predicts the trust serviceability of the network as the mean reputation values immediately recover past the 700s time stamp.
Maintenance Testing from Figure 5.10				
I_{ma}	0.005 (for initial value of 25 misbehaving nodes)	200-400	Unstable	The audit impact is not a snapshot measurement but rather a measurement of performance over time, hence the range over which it is measured. This demonstrates that the network's audit capacity is reduced and takes longer to operate when there are a large number of misbehaving nodes (QR of 0.58)
I_{ma}	0.020 (for initial value of 10 misbehaving nodes)	200-400	Unstable	This is twice the audit impact of the framework when operating with 25 misbehaving nodes. It is a lot more effective when the QR is in its favour. (QR of 0.83)
I_{ma}	0.095 (for initial value of misbehaving nodes of 25)	400-600	Unstable to Stable	Once the framework identifies the good nodes from the bad, the QR rapidly improves from 0.58 to 0.92 and the network returns to stability.
I_{ma}	0.03 (for initial value of misbehaving nodes of 10)	400-600	Unstable to Stable	The same resolution is seen in the framework for the more favourable scenario of 10 misbehaving nodes with the QR improving from 0.83 to 1.00. This is brought about by the permanent banning of the nodes due to highly erratic performances identifying them as misbehaving.

Table 5.6: Applying Metrics to Evaluation Scenarios

As can be seen from the above table, the trust metrics (KPIs) provide good guidance as to how the network is likely to behave (trust service availability) or how it has behaved (audit capacity). Interpreted properly, they can offer sound indications of whether a network is exhibiting signs of distress, whether due to network issues or trust issues.

In real life scenarios, there will be also scope to utilise service environment metrics to also aid in reaching inferences about a target framework or the state of the network.

The applications of such KPIs, once adopted as convention, are widespread and may be used at least initially to classify and review trust models as they are developed.

5.4 Discussion

The purpose of the experiments in this chapter is to ensure that the algorithms of the model work as expected and as per the specifications in Chapter 3. Furthermore, based on the simulation data supplied, the model should also reflect possible instances of the scenarios depicted in Chapter 4. It is not possible to address the complexity of all the scenarios but only certain instances. While implementing a full-scale physical set-up was outside the scope of this work, parallels from the simulation data should highlight the relevance of the trust framework in the scenarios proposed. For example, clusters in disaster scenarios could easily depict a specific service (Fire, Ambulance etc.), similarly within the Travellers' web, natural clusters could form within the carriage of an intercity train, or as previously suggested among participants of similar interests at trade fairs.

In order to determine the success of the framework, it is important to revisit the Objectives of the research as mentioned in Chapter 1 and determining whether those Objectives have been met as well as the Requirements from Chapter 2.

5.4.1 Meeting the Objectives

Objectives 1 & 2 - The very first objective is that the framework must be able to accurately derive trust and reputation information. In so far as the experimentation is concerned, this holds true as well for all the experiments in this current chapter. As an example, looking at Figure 5.3 from Section 5.3.1 on the variation of the reputation values of Node 1, it is clear that based on the observation data being parsed (in this case the routing data at the network layer), the model reacts as expected. The reputation of the node is shown to increase at the points where it acts as a router for other nodes. This effect is more pronounced the higher the number of packets that it successfully transmits. Furthermore, where the node acts as a source or a sink with no routing, the overall gain it achieves from the network results in its reputation being flat or decreasing slightly. This again accurately depicts the relationship between gain, workload and reputation thus showing that the algorithm within the framework is operating as required with the desired effect on the reputation of the node as seen from the cluster head, and by extension, the remainder of the cluster.

Objective 3 – The advantages of cluster heads from a networking point of view are well documented in the literature and have furthermore been proved in Section 3.4.1 in initial experimentations. However, further to this networking benefit, the overheads associated with the implementation of trust models such as FRANTIC had not been adequately proven. To that end, the experiments in Section 5.3.4.1 and 5.3.4.2 with regards to Node and Cluster analysis should strengthen the argument that clustering also provides additional benefits within a trust framework. For example, looking at the overhead generated by the reputation mechanism in Figure 5.8 from the point of view of a single node, it can be quickly inferred that no appreciable overhead is generated by the trust framework. This is partly due to the lean nature of the framework itself and partly due to the fact that the framework is able to utilise existing routing data in order to form an opinion on trust.

Furthermore, with respect to the overhead generated within a whole cluster, it is also noticeable in Figure 5.9 that although the trust framework introduces some additional

traffic mainly due to the various requests for recommendation that have to go to and from the cluster head. This would still be less than what would have been generated if the architecture had been fully distributed. This is because in the absence of the cluster heads, all nodes would be required to compute and store reputation information for their neighbour nodes, potentially increasing the overhead generated many folds.

The lack of crippling overheads therefore also enables the framework to meet **Objective 6** and the role segregation introduced by the cluster head versus node member relationship allows the various stakeholders to perform their respective roles and a fair system to emerge.

This fair system leads into the fulfilment of **Objective 4.** The key concepts of gain, workload and drop rate in order to determine the performance of a node are crucial in introducing fairness to the model. They act as a safeguard to prevent nodes from exploiting the network for their own selfish purpose and also propose an appropriate trade-off between work and rewards. By incorporating the concept of importance within the determination of reputation by the cluster head, better performing nodes are further rewarded as their recommendations are classed above those of non-performing or new nodes. This means that the longstanding well-performing nodes have a better say in determining how the trust profile of a node is determined. In using this experience from its most longstanding nodes, the framework also reduces its exposure to attacks and selfishness from new or misbehaving nodes.

However, reducing the exposure may not be solely enough to negate the effects that may be brought on by misbehaving or selfish nodes. In order to meet **Objective 5**, the framework has to show resilience to such behaviour. By operating a rule-based mechanism, it aims to do just that as presented in Chapter 3. This mechanism is further evaluated in threat scenarios in Section 5.4.5.2 where the network is flooded with misbehaving nodes. The framework reduces the incidence of misbehaving nodes to zero within 6 cycles. While 6 cycles may appear a lot, it should be noted that the behaviour of the misbehaving nodes in this instance was not made overly obvious. For example, the misbehaving nodes were not instructed to indiscriminately

reject all route requests or drop all packets received. They were programmed to perform some work for the network in order to hinder detection. It should be noted that most misbehaving nodes tend not to incur a high computational expense in order to disrupt a network and are therefore more likely to be banned within a shorter space of time.

Objective 7 is addressed in a theoretical evaluation of the performance of trust frameworks, see Section 4.3. This is because this objective was to create a list of KPIs that allows direct comparisons of trust models and this has been done in Section 4.3 in theory. There can be no empirical validation of Objective 7 as by its very nature, framework assessment via KPIs is theoretical.

5.4.2 Addressing the Requirements

As stipulated in Section 2.5, FRANTIC attempts to address most of the requirements present in the domain of ad hoc networking and trust. It should be noted that the high level objectives specified in Section 5.4.1 were mostly derived from an analysis of the issues inherently present when attempting to create a trust framework for the ad hoc network, therefore a high degree of overlap in meeting the respective aims is expected.

Requirements 1 & 2 – The basic requirements for the model to be decentralised and scalable are fully met by design. FRANTIC operates on a fully decentralised basis. The notion of cluster heads, although operating with a hierarchal architecture as far as intra-cluster trust traffic is concerned, still remain fully distributed inter-cluster wise (for trust) and both intra- and inter-cluster wise for networking.

There is no upper limit on the number of nodes that can be accommodated by the framework although from a practical point of view, based on the scenarios depicted, anything higher than 100 nodes may prove improbable in real life, but certainly not impossible.

Requirements 3 – 5 – These are addressed by Objectives 1 & 2 as described in Section 5.4.1.

Requirement 6 – This is the basis of Objective 4 and as such is appropriately met by the framework.

Requirements 7 & 8, 11 – These form part of the functioning of the cluster head and are met by design. Cluster heads form a separate class of node from ordinary nodes as far as the trust framework is concerned. They are also able to store local information about reputation from all the nodes in their cluster thus creating local repositories across the network.

Requirement 9 – This is a feature of the cyclic way in which reputation is calculated by the cluster head in each cluster. These cycles provide the assurance that reputation information for all nodes are up to date by default and not just for resilience or defence purposes.

Requirement 10 – This is met partially for misbehaving nodes in that there is provision within the network for isolating nodes that misbehaving. As described previously, actual malicious behaviour involves a more complex degree of reasoning not explored in this work.

Requirements 12 & 13 – These are not met by this framework. However, 12 forms part of suggested future work in which this current work could be extended and 13 has been addressed theoretically in the literature and background. There is no motivation for implementing 13 as a feature of the network as the emphasis for the behaviour leading to trust information was to be derived from simple mechanics of the target node. As such, it is assumed that all trust information falls within the same context, which is true in so far as network traffic is concerned.

Even in cases where the context in which trust is generated varies (as is often the case in the case of the Travellers' Web scenario), there is still the possibility to create a trust network as the context within which the peers operate remain similar at a high

level, even though they may in essence be different. As explained before, if trust is built up within the context of sharing music, it is expected that such trustworthiness may be extended to other contexts such as sharing cached news without any adverse impact on the validity of the trust information.

Having said that, context within trust is indeed an important concept and extending the scope of the research to address more complex computation of trust is part of the future direction this research can take within latter publications.

Requirement 14 – The clustered framework allows for a node to migrate from one cluster to another and measures are also in place for its reputation to be transferred to its new cluster as long as both cluster heads are within direct radio range of each other. In effect, cluster heads transfer the node's reputation information to their new geographical cluster so they don't have to start building a reputation profile from scratch. It can only happen when there is direct radio contact between cluster heads as this is to ensure that no interception takes place while such sensitive information is in transit. However, it was also noted that it is rare for nodes to merely move to adjacent nodes without valid reasons (unless it was purely for operational efficiency) – if nodes were to move, they would most likely continuously travel across several clusters or leave the network altogether.

The framework therefore appropriately addresses the issues presented in Chapter 2 with one recommendation for further study.

FRANTIC is also adequately modelled and proves the basics of its operation, both from a node and cluster perspective. The implemented algorithms behave as expected.

Chapter 6

6 Conclusion and Further Work

6.1 Conclusion

This chapter presents a summary of the research carried out and highlights the contributions of this work as well as the possible avenues for further research.

As computational systems become more ubiquitous and P2P interaction increases, the demand for self-starting networks will also see a rise as people look towards other means of communicating and sharing without having to incur additional costs. An ad hoc network is one of the ways to achieve that. Aside from catering for the general public interaction, an ad hoc network could also adequately improve communication in disaster scenarios where traditional methods of communications fail. This can happen by the use of a trust framework that allows peers to find the most helpful routes in order to disseminate recovery information.

With this in mind, the need for entities to find reliable and honest partners with which they can interact becomes increasingly pressing. Trust is inherent in everyday life and it is a concept that, though hard to grasp, is responsible for most of the tasks performed in today's society. Its transition from the social domain to the computing domain has nonetheless not been so straightforward purely because computers are yet to reach the level of complexity typically associated with human behaviour. However, by building upon this very lack of complexity, it is possible to install trust frameworks onto ad hoc networks such that it aids the decision making process of the entities within it.

The chief benefit of FRANTIC is that it can provide an entity with the means to assess behavioural evidence in order to form an opinion about the trustworthiness of another entity. Furthermore, when an entity's previous interactions with its target are few, the framework can draw on the recommendations of others with more interactions in order to calculate the target entity's reputation.

Along with being able to provide resilience by isolating misbehaving nodes, FRANTIC also presents an adequate way of organising the trust information generated within the network that presents some synergies regarding overheads generated. This is done by clustering the framework such that classes of node are formed which are able to segregate their duties in order to better perform on their own specific tasks. By distributing this trust information in local repositories across the whole network, FRANTIC also enhances its robustness against attacks. Any part of its infrastructure being taken over in an attack will not necessarily result in the whole network failing as the partitioning into clusters means that some parts of the network still remain dependable and trustworthy and can continue to function as before.

6.2 Research Contribution

With these advantages in mind, the following points are a summary of the main contributions of this thesis.

1. As described in the state of the art, no current trust model is able to meet all the requirements for a trust model for an ad hoc network. While this model does not claim to meet all the requirements either, it is nevertheless a novel framework that meets most of the basic requirements. While it can be argued that some models which fulfil fewer requirements may do those better than one which fulfills most of them, it would be impractical for one particular network to be loaded with several trust models when one could do the job adequately. The amount of additional resources consumed in loading additional frameworks would more than negate any benefit accrued in having some requirements met in

a superior manner. The model allows an entity to effectively derive the trust value of its peer through two key stages:

- via personal experience
- through recommendations from its peers when it has no prior experience of its trustee. In particular, this experience has been gained objectively from its peers within its vicinity and is the sum of all of their recommendations.

2. FRANTIC generates trust without any a priori information. While there are several models in the literature that claim to bootstrap trust information, most still require a seed in order to bootstrap the process. This seed may be a common password or it could even assume a priori information or even existing trust relationships between nodes. The FRANTIC method is to initially start off the network with trivial routing in order to kick-start the trust formation process. Only when this is set up and running is it proposed that cluster formation and cluster head election take place.

3. FRANTIC is also an autonomous system and is able to work independently of any form of backbone structure. It requires no access to centralised servers or repositories in order to function.

4. The framework also operates a rule-based mechanism in order to implement resilience against misbehaving nodes. Aside from the above broad contributions, the framework has further advanced the state of the art as follows. It has:

a) provided a means of assessing ‘at a glance’ the respective performances of trust models within ad hoc networks through the use of key ratios known as KPIs. While these ratios are based on network and service data, they provide a quick means to calculate how efficient a network is running and whether there is scope for it to be extended

b) formally defined different roles for separate classes of node. While this work was the first to introduce the concept of clustering for trust distribution in ad

hoc networks, subsequently adopted by other researchers, the concept of assigning specific roles to different classes of node is unique to this work.

c) created a simple system of weighted averages for reputation that is directly proportional to the trustworthiness of a node. Because the trustworthiness itself is a measure of how efficient the node is within the network (by virtue of assessing the gain, workloads and drop rates), this implies that the best performing nodes have the most influence into the reputation calculation of new nodes. This in itself provides a better backbone for the trust framework by making it more resilient against lying nodes.

d) evaluated the network overheads created by a trust framework both at a node and at the cluster level. This allowed this research to determine that no appreciable amount of traffic is introduced at a node level and even at a cluster level, the performance is substantially higher than that of fully distributed nodes.

e) in using routing data to generate trust, indirectly provided better routes for ad hoc networks.

6.3 Further Work

Most of the work contained in this thesis has focused on implementing a framework via managing trust. Like all research, this is not completive and there are several other avenues that can be pursued in order to extend this work.

As part of validating the framework, simulations were carried out. Further work could be done in extending the implementation to the physical domain by using actual handsets and PDAs in order to mimic the implementation of the model. One practical way of doing that would be to install specific software onto each mobile terminal that allows trust evaluation of peers and allows it to maintain trust information which it can use at various levels.

In so far as trust generation is concerned, simulations (both software-based and physical) can be carried out on using various other bootstrapping mechanisms in order to find a most suitable one. Currently the method has its own mechanism which involves gradual trust build-up through experience, followed by a voting process for the election of cluster heads. However, this may not be totally suited to all networks, where faster start-up times may be necessary. The crucial and determining factor of all self-starting networks is the bootstrapping mechanism and it is vital that this part of the process happens with optimal results as the success or failure of the network during the latter stages is dependent on it to a very large extent.

In determining trustworthiness, the framework utilises network data that is essentially part of all the internal data to which the nodes have access. However, as per Requirement 12, it may be highly beneficial for the model to be able to assimilate external data in order to enhance the accuracy of the reputation being computed. More often than not, this external data would require access to a backbone structure (such as the web) or a similar type of repository (such as a credit rating agency). However, there may also be local external data that may be fed into the framework that may enhance the quality of the trust information being produced. For example, in the exhibition scenario mentioned in Chapter 4, the size of the stall or the number of people visiting the stall may be a factor that aids the reputation of a particular merchant. Similarly, the displays could be rated according to their extravagance and this may have a direct impact as to how big the company is and therefore how reliable.

Another improvement to the current framework is to add contextual data such that different classes of trustworthiness can be stored based on how they were obtained and in what context they are relevant in. Although there is an argument in this thesis for not using contextual information, more complex systems using high end devices that operate a larger variety of applications on an ad hoc network may well require it.

With these improvements in mind, it is quite likely that trust models that work autonomously will soon find their way by getting adopted into real applications. Already, they exist on backbone structures, but with the advent of new device types

(such as the Apple iPad), ubiquity in computing is going on a whole new level that will make the need for localised resources very pressing.

Bibliography

Abdul-Rahman, A. & Hailes, S. (2000), 'Supporting Trust in Virtual Communities'. In Proc. of the 33rd IEEE Hawaii International Conference on System Sciences, Washington, USA, volume 6, p. 6007.

Abdul-Rahman, A. & Hailes, S. (1997), 'Using Recommendations for Managing Trust in Distributed Systems.' In Proc. of IEEE Malaysia International Conference on Communication (MICC'97), Kuala Lumpur, Malaysia.

Abramson, N. (1970), 'The ALOHA system – another alternative for computer communications'. In Proceedings of the American Federation of Information Processing Societies (AFIPS) Fall Joint Computer Conferences, Houston, Texas, pp. 695-702.

Amazon (2009), Internet resource: http://www.amazon.com/gp/help/customer/display.html?nodeId=1161258, last accessed 15 June 2009.

Apple (2009), Internet resource: http://www.apple.com/uk/iphone/, last accessed 15 June 2009.

Baier, A. (1985), 'Trust and Antitrust'. Ethics, Vol. 96, pp. 231-260.

Baker D.J. & Ephremides A. (1981), 'The Architectural Organization of a Mobile Radio Network via a Distributed Algorithm'. IEEE Transactions on Communications, Vol. 29, No. 11, pp. 1694-1701.

Bakht, H. (2005), 'Routing protocols for mobile ad hoc networks'. In GERI Annual Research Symposium (GARS2005), Liverpool, UK.

Barber, B. (1983), 'Logic and Limits of Trust'. Rutgers University Press, New Jersey, USA.

Berners-Lee, T., Hendler, J. & Lassila, O. (2001), 'The semantic web. Scientific American, 284(5), pp. 34-43.

Boodnah J. & Scharf E.M. (2004), 'Trust in Ad Hoc Networks: A Novel Approach based on Clustering'. In Proceedings of the London Communications Symposium, University College London, London, UK, pp. 257-261.

Boodnah J. & Scharf E.M. (2005), 'Applying Clustering to a Framework for Generating Trust'. In Proceedings of the International Workshop on Wireless Ad Hoc Networks, London, UK.

Boodnah, J., Poslad, S. (2009), 'A Trust Framework for Peer-to-Peer Interaction in Ad Hoc Networks'. 1st Int. Conf. on Adaptive and Self-adaptive Systems and Applications, SELFTRUST, Athens/Glyfada, Greece, pp. 707-712.

Bram, P. & Foddy, M. (1987), 'Trust and the consumption of a deteriorating common resource'. Journal of Conflict Resolution 31, pp. 615–63.

Bray, J. & Sturman, C.F. (2000), 'Bluetooth: Connect Without Cables'.Prentice-Hall PTR Publishers.

Buchegger, S. & Le Boudec, J-Y. (2004), 'A Robust Reputation System for Peer-to-Peer and Mobile Ad-hoc Networks'. In Proceedings of P2PEcon, Harvard University, Cambridge MA, USA.

Cahill, V. et. al. (2003), 'Using Trust for Secure Collaboration in Uncertain Environments.' In IEEE Pervasive Computing Mobile And Ubiquitous Computing, 2(3), pp. 52-61.

Capra, L. (2004), ‘Engineering human trust in mobile system collaboration.’ In SIGSOFT-12, Newport Beach, USA.

Capra, L. (2005), ‘Reasoning about Trust Groups to Coordinate Mobile Ad-Hoc Systems.’ In Proc. of the 1st IEEE Workshop on the Value of Security Through Collaboration, in conjunction with IEEE/CreateNet, Athens, Greece.

Chatterjee, M.; Das, S.K; Turgut, D. (2002), 'WCA: A Weighted Clustering Algorithm for Mobile Ad Hoc Networks'. Journal of Clustering Computing, Vol. 5, No. 2, pp. 193-204.

Chiang, C-C.; Wu, H-K.; Lui, W.; Gerla, M. (1997), 'Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel'. In Proceedings of the IEEE Singapore International Conference on Networks (SICON '97), Singapore, pp. 197-211.

Cho, J. H. & Swami, A. (1999), ‘Towards Trust-based Cognitive Networks: A Survey of Trust Management for Mobile Ad Hoc Networks.’ In 14th Int’l Command and Control Research and Technology Symposium, Washington D.C.

Davis, C.R. (2004), 'A localized trust management scheme for ad hoc networks'. In Proceedings of 3rd International Conference on Networking (ICN'04), pp. 671-675.

Deutsch, M. (1962), 'Cooperation and Trust: Some Theoretical Notes'. In Jones, M. R. (ed.), Nebraska Symposium on Motivation, Nebraska University Press.

Deutsch, M. (1973), 'The Resolution of Conflict'. Yale University Press, New Haven, USA.

Dewan P.; Dasgupta, P. & Bhattacharya, A. (2004), 'On Using Reputations in Ad Hoc Networks to Counter Misbehaving Nodes’. In Proceedings of the 10th

International Conference on Parallel and Distributed Systems (ICPADS), pp. 665-672.

eBay (2009), Internet resource: http://pages.ebay.co.uk/help/feedback/about-feedback.html, last accessed 15 June 2009.

Er, I.I. & and Seah W.K.G. (2004), 'Mobility-Based D-Hop Clustering Algorithm for Mobile Ad Hoc Networks'. In Proceedings of 2004 IEEE Wireless Communications & Networking Conference (WCNC 2004), Atlanta, USA, Vol. 4, pp. 2359-2364.

Foster, N. & Kesselman, C. editors (1998), 'The Grid, Blueprint for a New Computing Infrastructure.' Morgan Kaufmann Inc.

Gambetta, D. (2000) 'Can We Trust Trust?', in Gambetta, Diego (ed.) Trust: Making and Breaking Cooperative Relations, electronic edition. Department of Sociology, University of Oxford, chapter 13, pp. 213-237.

Gerla, M. & Tsai, J.T.C. (1995), 'Multicluster, Mobile, Multimedia Radio Network'. ACM Wireless Networks, Vol. 1, No. 3, pp. 255-265.

Gnutella (2009), Internet resource: http://rfc-gnutella.sourceforge.net/, last accessed 30 November 2009.

Golembiewski, R.T. & McConkie, M. (1975), 'The centrality of interpersonal trust in group processes'. In Theories of Group Processes, L.G Cooper, ed., John Wiley and sons, London, pp. 131-185.

Guo, B. & and Li, Z. (2007), 'United Voting Dynamic Cluster Routing Algorithm Based on Residual-Energy in Wireless Sensor Networks'. Journal of Electronics & Information Technology, Vol. 29, No. 12, pp. 3006-3010.

Hardin, R. (1993), 'The street-level epistemology of trust'. Politics and Society, vol. 21(4), pp. 505-529.

Hu, Y.C; Perrig, A; Johnson D.B. (2003), 'Rushing Attacks and Defense in Wireless Ad Hoc Network Routing Protocols'. In Proceedings of the ACM Workshop on Wireless (WISE2003).

Islam, S. (2005), ‘Efficient Key Management Scheme for Mobile Ad Hoc Networks.’ Master of Science. Royal Institute of Technology (KTH) SecLab Department of Computer and System Sciences, Stockholm, Sweden.

Jennings, N. R. (2001), ‘An agent-based approach for building complex software systems.’ Communications of the ACM, 44(4), pp. 35-41.

Johnson, D.B. & Maltz, D.A. (1996), 'Dynamic Source Routing in Ad Hoc Wireless Networks'. In Mobile Computing, edited by Tomasz Imielinski and Hank Korth, Kluwer Academic Publishers, pp. 153-181.

Jøsang, A.; Gray, E.; & Kinateder, M. (2003), 'Analysing Topologies of Transitive Trust'. In Proceedings of the Workshop of Formal Aspects of Security and Trust (FAST), Pisa, Italy.

Jubin, J. & Tornow, J.D. (1987), 'The DARPA packet radio network protocol',.Proceedings of the IEEE, 1st Edition, pp. 21-32.

Kadri, B.; M’hamed, A.; Feham, M. (2007), 'Secured Clustering Algorithm for Mobile Ad Hoc Networks'. International Journal of Computer Science and Network Security, Vol. 7, No. 3, pp. 27-34.

Kalasapur, S.; Kumar, M.; Shirazi, B. (2006), 'Evaluating service oriented architectures (SOA) in pervasive computing'. In Proceedings of the 4th Annual IEEE Int. Conf. on Pervasive Computing and Communications (PerCom 2006), pp. 276-285.

Kazaa (2009), Internet resource: http://www.kazaa.com, last accessed 30 November 2009.

Keoh, S.L. & Lupu, E. (2003), 'Trust and the Establishment of Ad-hoc Communities'. In 2nd Internal iTrust Workshop on Trust Management in Dynamic Open Systems, London, UK.

Kephart, J. O. & Chess, D.M. (2003), 'The vision of autonomic computing.' IEEE Computer, 36(1), pp. 41-50.

Kersten, E. & Lo, G. (2001), 'Negotiation support systems and software agents in e-business negotiations.' In Proceedings of the 1st International Conference on E-Business.

Kramer, R.M. (1996), 'Divergent realities and convergent disappointments in the hierarchical relation, trust and the intuitive auditor at work', In Kramer, R.M., Tyler, T.R. (Eds), Trust in Organizations: Frontiers of Theory and Research, Sage, London, pp. 216-46.

Kuhn, T.S (1962), The Structure of Scientific Revolutions. University of Chicago Press.

Li, R.; Li, J.; Liu, P. & Chen, H-H. (2007), 'An Objective Trust Management Framework for Mobile Ad Hoc Networks'. In Proceedings of the IEEE 64th Vehicular Technology Conference, Springer, pp. 56-60.

Limewire (2009), Internet resource: http://www.limewire.org, last accessed 30 November 2009.

Liu, J. & Issarny, V. (2004), 'Enhanced Reputation Mechanism for Mobile Ad Hoc Networks'. In Proceedings of 2nd International Conference on Trust Management, iTrust, Oxford, UK, pp. 48-62.

Liu, Z.; Joy, A.W.; & Thompson, A. (2004), 'A Dynamic Trust Model for Mobile Ad Hoc Networks'. In Proceedings of the 10th IEEE International Workshop on Future Trends of Distributed Computing Systems (FTDCS'04).

Luhmann, N. (1979), 'Trust and Power'. John Wiley & sons, Chichester, UK.

Luhmann, N. (1988), 'Familiarity, Confidence, Trust: Problems and alternatives'. In Trust, Diego Gambetta, ed., Basil Blackwell publishers.

Marsh, S. (1994), 'Formalising Trust as a Computational Concept'. PhD Thesis, University of Stirling, Scotland.

Martucci L.A.;, Schweitzer, C.; Venturini, Y.R., Carvalho, T.C.M.B., & Ruggiero, W. (2004), 'A Trust-Based Security Architecture for Small and Medium-Sized Ad Hoc Networks'. In The Annual Mediterranean Ad Hoc Networking Workshop, Turkey.

McKnight, D. Harrison & Chervany, Norman L. (1996), 'The Meanings of Trust'. Technical Report 94-04, Carlson School of Management, University of Minnesota, USA.

Mühlethaler, P. (2005), 'Security Schemes for the OLSR Protocol for Ad Hoc Networks.' Doctoral thesis, Universite Paris 6, Pierre et Marie Curie, France.

Murthy, S. & Garcia-Luna-Aceves, J.J. (1996), 'An efficient routing protocol for wireless networks'. In Mobile Networks and Applications (Volume 1, Issue 2): Special Issue: Routing in Mobile Communication Networks, Kluwer Academic Publishers, USA, pp. 183-197.

Nguyen, C. T. & Camp, O. (2008), 'Using context information to improve computation of trust in ad hoc networks.' In Proceedings of the International Workshop on Security and Privacy in Wireless and Mobile Computing, Networking and Communications (SecPri-WiMob'2008).

NIST (National Institute of Security and Technologies), (2006), 'Wireless Network Security 802.11, Bluetooth and Handheld Devices.' Technical Report, SP 800-48

NS-2 (2009), 'Network Simulator'. Internet resource: http://nsnam.isi.edu/nsnam/index.php/User_Information, last accessed March 2010.

Park, Vincent D. & Corson, M. Scott (1997), 'A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks'. In IEEE Conference on Computer Communications (INFOCOM'97), IEEE, pp. 1404-1413.

Patwardhan, A.; Parker, J.; Joshi, A.; Iorga, M.; Karygiannis T. (2005), 'Secure Routing and Intrusion Detection in Ad Hoc Networks'. In Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications, Hawaii, USA.

Peng, S.; Jia, W.; Wang, G. (2008), 'Voting-Based Clustering Algorithm with Subjective Trust and Stability in Mobile Ad Hoc Networks'. In Proceedings of the 2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing, Volume 2, IEEE Computer Society, pp. 3-9.

Perkins, C.E. & Bhagwat, P. (1994), 'Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers'. Computer Communications Review 34(1), pp. 234-244.

Perkins, C.E. & Royer, E. (1999), 'Ad-hoc On-Demand Distance Vector Routing'. In Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'99), IEEE, pp. 90-100.

Poslad, Stefan (2009) 'Ubiquitous Computing: Smart Devices, Environments and Interactions, Wiley, ISBN: 978-0-470-03560-3.

Pirzada, A.A. & McDonald, C. (2004), 'Establishing Trust in Pure Ad Hoc Networks'. In Proceedings of the 27th Australasian conference on Computer science, pp. 47-54.

Ramchurn, S. D. (2004), 'Multi-Agent Negotiation using Trust and Persuasion'. PhD Thesis, Electronics and Computer Science, University of Southampton.

Rasmusson, L. & Janson, S. (1996), 'Simulated Social Control for Secure Internet Commerce.' In New Security Paradigms Workshop, Lake Arrowhead, CA, ACM Press, pp. 18-26.

Rebahi, Y.; Mujica, V.E. & Sisalem, D. (2005), 'A Reputation-Based Trust Mechanism for Ad Hoc Networks'. In Proceedings of the 10th IEEE Symposium on Computers and Communications (ISCC 2005), Murcia, Cartagena, Spain, pp. 37-42.

Ren, K.; Li, T; Wan, Z.; Bao, F.; Deng R.; Kim, K. (2004), 'A Highly Reliable Trust Establishment Scheme in Ad Hoc Networks'. Computer Networks: The International Journal of Computer and Telecommunications Networking, vol. 45, no. 6, pp. 687-699.

Ren, Y. & Boukerche, A. (2008), 'Modeling and managing the trust for wireless and mobile ad hoc networks'. In Proceedings of the IEEE International Conference on Communications, pp. 2129–2133.

Ripeanu, M., Foster, I. & Iamnitchi, A. (2002), 'Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design.' IEEE Internet Computing special issue on Peer-to-Peer Networking, 6(1), pp. 50-57.

Rotter, J.B. (1967), 'A new scale for the measurement of interpersonal trust', Journal of Philosophy no. 35, pp. 651-665.

Royer, E. M. & Toh, C-K. (1999), 'A review of current routing protocols for ad-hoc mobile wireless networks'. IEEE Personal Communications Magazine, Vol. 6, No.2, pp. 46-55.

Russell, S.J. & Norvig, P. (2010), 'Artificial Intelligence: A Modern Approach, Third Edition,' Prentice-Hall, ISBN: 978-0-13-604259-4.
Satyanarayanan, M. (2001), 'Pervasive computing: Vision and challenges.' IEEE Personal Communications.

Schweitzer, C.M.; Carvalho, T.C. & Ruggiero, W. (2006), 'A Distributed Mechanism for Trust Propagation and Consolidation in Ad Hoc Networks'. In The International Conference on Information Networking (ICOIN).

Seth, M. (2003), Internet Resource: 'Introduction to web services management.' http://www.developer.com/services/article.php/1583511, Developer.com, last accessed: January 2010.

Seunghun, J.; Chanil, P.; Daeseon, C. (2005), 'Cluster-Based Trust Evaluation Scheme in an Ad Hoc Network'. ETRI Journal, Vol. 27, No. 4, pp. 465-468.

Shand, B., Dimmock, N. & Bacon, J. (2003), 'Trust for Ubiquitous, Transparent Collaboration.' In First International Conference on Pervasive Computing, Dallas-Fort Worth, Texas, pp. 153–160.

Smets, P. (1990), 'What is Dempster-Shafer's model?'. In: R. Yager, M. Fedrizzi, J. Kacprzyk (Eds.), Advances in the Dempster-Shafer Theory of Evidence, Wiley, pp. 5–34.

The New Oxford Dictionary of Current English (2005), 2nd ed. Oxford: Oxford University Press.

Theodorakopoulos, G. & Baras, J.S., 'Trust evaluation in ad-hoc networks'. In Proceedings of the ACM Workshop on Wireless Security (WiSE'04), Philadelphia, USA, pp. 1-10.

Tveit, A. (2001), 'Peer-to-peer based recommendations for mobile commerce.' In Proceedings of the 1st international workshop on Mobile commerce, ACM Press, pp. 26-29.

Verma, D.C. (2004), Legitimate Applications of Peer to Peer Networks, John Wiley and sons Inc.

Vulkan, N. (1999), 'Economic implications of agent technology and e-commerce.' The Economic Journal, 109(453), pp. 67-90.

Wooldridge, M. & Jennings, N. R. (1995), 'Intelligent agents: Theory and practice.' Knowledge Engineering Review, 10(2), pp. 115-152.

Yan, Z.; Zhang P. & Virtanen, T. (2005), 'Trust Evaluation Based Security Solution in Ad Hoc Networks', in Nokia Research Publication.