Simulation, Analysis and Design of Trust Based Security in MANETs

Thesis submitted in partial fulfillment of the requirements for the award of

Master of Engineering
in
Software Engineering

By
Ghansham Sangar
8043106

Under the supervision of
Mr. Anil Kumar Verma

MAY 2006

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR INSTITUTE OF ENGINEERING AND TECHNOLOGY
(DEEMED UNIVERSITY)
PATIALA – 147004

Certificate

I hereby certify that the work which is being presented in the thesis entitled, **"Simulation, Analysis and Design of Trust Based Security in MANETs"**, in partial fulfillment of the requirements for the award of degree of Master of Engineering in Software Engineering submitted in Computer Science and Engineering Department of Thapar Institute of Engineering and Technology (Deemed University), Patiala, is an authentic record of my own work carried out under the supervision of Mr. Anil Kumar Verma.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.

Ghansham Sangar

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

Mr. Anil Kumar Verma
Thapar Institute of Engineering and Technology
Patiala-147004

Countersigned by

Dr. (Mrs.) Seema Bawa
Head of Department
Computer Science & Engineering Department
Thapar Institute of Engg and Tech.
Patiala.

Dr. T. P. Singh
Dean
Academic Affairs
Thapar Institute of Engg and Tech
Patiala

The M.E. (Thesis) Viva Voca examination of Ghansham Sangar Roll No. 8043106, M.E (Software Engineering), Thapar Institute of Engineering and Technology, Patiala has been held on ……

Supervisor External Examiner

Acknowledgement

No volume of words is enough to express my gratitude towards my guide, Sh. Anil Kumar Verma, System Analyst, Computer Centre, TIET, who has been very concerned and has aided for all the material essential for the preparation of this thesis report. He has helped me explore this vast topic in an organized manner and provided me with all the ideas on how to work towards a research-oriented venture.

I am also thankful to Dr. (Mrs.) Seema Bawa, Head, CSED and Sh. Rajesh Bhatia, P.G. Coordinator, for the motivation and inspiration that triggered me for the thesis work.

I would also like to thank the staff members and my colleagues who were always there at the need of the hour and provided with all the help and facilities, which I required, for the completion of my thesis work.

Most importantly, I would like to thank my parents and the almighty for showing me the right direction out of the blue, to help me stay calm in the oddest of the times and keep moving even at times when there was no hope.

Ghansham Sangar

8043106

Abstract

Ad hoc networks are a new wireless networking paradigm for mobile hosts. In mobile ad hoc networks (MANETs), every node has to act as a router for the network to function. Unlike traditional mobile wireless networks, ad hoc networks do not rely on any fixed infrastructure. Thus, the security threats from malicious attackers take a different shape due to the basic difference in their functionality from their wired counterparts.

Current ad hoc routing protocols assume the networks to be benevolent and cannot cope with misbehavior of nodes. The misbehavior may be due to node being malicious or selfish to save the battery power. Thus, this work is of significant importance because now-a-days, new cryptographic schemes, such as threshold cryptography, are used to build a highly secure key management service that forms the core of security framework presented in this work.

The CONFIDANT (Cooperation Of Nodes: Fairness In Dynamic Ad-hoc Network) on DSR (Dynamic Source Routing Protocol) is simulated in order to evaluate how the network performance changes as dynamic feedback mechanisms are introduced in an ad hoc network to control the node misbehavior. The above results have been used to design a trust based routing protocol that uses reputation data to provide dynamic feedback.

Keywords: MANETs, CONFIDANT, DSR, Security, Trust.

TABLE OF CONTENTS

CONTENTS **PAGE NO.**

LIST OF FIGURES & TABLES

CONTENTS **PAGE NO.**

List of Tables

Chapter 1
Introduction

Mobile Ad Hoc Network (MANET) is a collection of wireless mobile nodes that dynamically communicate amongst themselves without the use of any existing infrastructure and centralized/decentralized administration. The mobile nodes must cooperate at the routing level in order to forward packets to from source to the destination. Current ad hoc routing protocols such as Dynamic Source Routing (DSR) [1] assumes the network is benign and cannot cope with misbehavior, i.e., a misbehaved node may drop packet silently to save battery power, etc.

Till date, a number of secure routing protocols in context of external attacks have been proposed, but very less work has been done to protect ad hoc networks from internal attacks. Dynamic feedback mechanisms [2] have been recently introduced to mitigate the misbehavior of an internal node in a MANET. The idea is to build the trust relationship between the mobile nodes in the MANET and select routes based on the formed trust values. Cooperation Of Nodes: Fairness In Dynamic Ad-hoc Network (CONFIDANT) [3] is a dynamic feedback mechanism in which mobile nodes monitor the behavior of their neighbors and exchange first hand information about other nodes in the MANET.

This thesis investigates dynamic feedback mechanisms as a security solution for MANETs, implements CONFIDANT protocol using ns2 as simulation environment, and evaluates the performance of CONFIDANT fortified DSR in the MANET where misbehaved nodes are present. The simulation results are analyzed and compared with that of standard DSR.

We propose a reputation-based trust management scheme to detect misbehaving nodes and to exclude them from the network. This is done by monitoring the neighbors, which results in a direct trust value, and by asking others for reputation. Direct trust and reputation together form total trust, which then is used for routing decisions. To be able

to control routing decisions from source to destination, a source routing protocol is needed. Therefore, we considered the DSR protocol and extended it on our trust management. Finally, we conclude with the future scope.

Chapter 2
Background Information

2.1 Ad hoc Networks

Computer networks were originally developed to operate by connecting computers together with wires and transmitting data over these wires. Network sizes and occurrences increased creating a requirement for inter-network communication. This led to the development of the Internet and its suite of protocols. The use of the Internet and its applications became ubiquitous. A need for providing network access to entities while not physically attached to the wired network arose. To enable this wireless networking was developed, providing devices with methods to connect to a wired network using radio wave technologies through wireless access points. Simultaneously, telephone networks were undergoing a similar transformation. Cellular network technologies [4] were developed to allow mobile phones to connect via base stations and communicate in a circuit switched environment. The area of mobile ad-hoc networking deals with devices equipped to perform wireless communication and networking, but without any existing infrastructure such as base stations or access points. Wireless devices form a network as they become aware of each other's presence. They communicate directly with devices inside their radio range in a peer-to-peer nature and to communicate with a device outside their range, they can use an intermediate device or devices within their radio range to relay or forward communications to the device outside their range.

An ad-hoc network is self-organizing and adaptive. Networks are formed on-the-fly; devices can leave and join the network during its lifetime, devices can be mobile within the network, the network as a whole may be mobile and the network can be deformed on-the-fly. All this needs to be done without any system administration and without the requirement for any permanent devices within the network. Devices in mobile ad-hoc networks should be able to detect the presence of other devices and perform the necessary set-up to facilitate communications and the sharing of data and services.

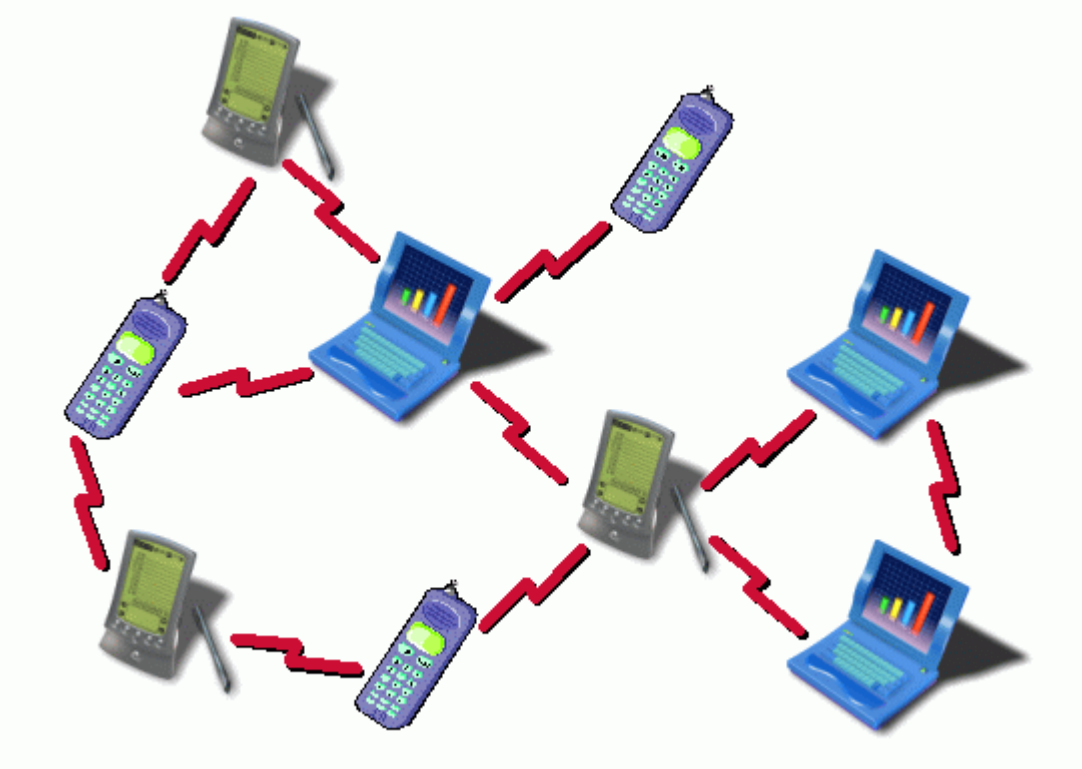

Figure 2-1 A Sample ad hoc network [5]

2.2 Major Security Attacks on MANETs [6]

Current ad hoc routing protocols are basically exposed to two different types of attacks:

- Active attacks
- Passive attacks

An attack is considered to be active when the misbehaving node has to bear some energy costs in order to perform the threat while passive attacks are mainly due to lack of cooperation with the purpose of saving energy selfishly. Nodes that perform active attacks with the aim of damaging other nodes by causing network outage are considered to be *malicious* while nodes that perform passive attacks with the aim of saving battery life for their own communications are considered to be selfish. Malicious nodes can disrupt the correct functioning of a routing protocol by modifying routing information, by fabricating false routing information and by impersonating other nodes. Recent research studies brought up also a new type of attack that goes under the name of wormhole attack [7] [8] [9].

On the other side, selfish nodes can severely degrade network performances and eventually partition the network by simply not participating to the network operation.

- *Eavesdropping*: This attack is used to gain knowledge of the transmitted data. This is a passive attack, which is easily performed, in many networking environments. However using an encryption scheme to protect the transmitted data can prevent this attack.
- *Impersonation*: Since current ad hoc routing protocols do not *authenticate* routing packets a malicious node can launch many attacks in a network by masquerading as another node (spoofing). Spoofing occurs when a malicious node misrepresents its identity in order to alter the vision of the network topology that a benign node can gather. As an example, a spoofing attack allows to create loops in routing information collected by a node with the result of partitioning the network.
- *Modification*: Existing routing protocols assume that nodes do not alter the protocol fields of messages passed among nodes. Malicious nodes can easily cause traffic subversion and denial of service (DoS) by simply altering these fields: such attacks compromise the integrity of routing computations. By modifying routing information an attacker can cause network traffic to be dropped, redirected to a different destination or take a longer route to the destination increasing communication delays.
- *Fabrication*: The notation "fabrication" is used when referring to attacks performed by generating false routing messages. Such kind of attacks can be difficult to identify as they come as valid routing constructs, especially in the case of fabricated routing error messages claiming that a neighbor can no longer be contacted.
- *Wormhole Attack* [7] [8] [9]: A more subtle type of active attack is the creation of a tunnel (or wormhole) in the network between two colluding malicious nodes linked through a private network connection. This exploit allows a node to short-circuit the normal flow of routing messages creating a virtual vertex cut in the network that is controlled by the two colluding attackers.
- *Lack of cooperation*: A selfish node that wants to save battery life for its own communication can endanger the correct network operation by simply not participating to the routing protocol or by not executing the packet forwarding (this attack is also known as the black hole attack). Current ad hoc routing protocols

cannot cope with the selfishness problem and network performances severely degrade.

- *Denial of Service (DoS)*: This active attack aims at obstructing or limiting access to a certain resource. This resource could be a specific node or service or the whole network. This will affect the availability security service mentioned above. The nature of ad-hoc networks where several routes exist between nodes and routes are very dynamic gives ad-hoc a built-in resistance to DoS attacks, compared to fixed networks.

According to another classification, attacks on ad hoc network routing protocols generally fall into one of two categories:

- *Routing Disruption attacks*

In a routing disruption attack, the attacker attempts to cause legitimate data packets to be routed in dysfunctional ways. Some of the examples of the Routing Disruption attacks are described below:

  - An example of a routing disruption attack is for an attacker to send forged routing packets to create a routing loop [10], causing packets to traverse nodes in a cycle without reaching their destinations, consuming energy and available bandwidth.
  - An attacker may similarly create a routing black hole [11], in which all packets are dropped: by sending forged routing packets, the attacker could cause all packets for some destination to be routed to itself and could then discard them, or the attacker could cause the route at all nodes in an area of the network to point "into" that area when in fact the destination is outside the area.
  - As a special case of a black hole, an attacker could create a gray hole [11], in which it selectively drops some packets but not others, for example, forwarding routing packets but not data packets.
  - An attacker may also attempt to cause a node to use Detours (sub-optimal routes) [12] or may attempt to *partition* the network by injecting forged routing packets to prevent one set of nodes from reaching another.
  - An attacker may attempt to make a route through itself appear longer by adding virtual nodes to the route; we call this attack gratuitous detour [12], as a shorter route exists and would otherwise have been used.

- In ad hoc network routing protocols that attempt to keep track of perceived malicious nodes in a "blacklist" at each node, such as is done in WATCHDOG and PATHRATER, an attacker may blackmail [13] a good node, causing other good nodes to add that node to their blacklists, thus avoiding that node in routes.
- A more subtle type of routing disruption attack is the creation of a Wormhole in the network, using a pair of attacker nodes A and B linked via a private network connection. Every packet (or selected packets) that A receives from the ad hoc network, A forwards through the wormhole to B, to then be rebroadcast by B; similarly, B may send all ad hoc network packets to A. Such an attack potentially disrupts routing by short-circuiting the normal flow of routing packets, and the attackers may also create a virtual vertex cut that they control.
- The rushing attack [9] is a malicious attack that is targeted against on-demand routing protocols that find routes through a Route Discovery protocol and use duplicate suppression of the ROUTE REQUEST messages in that protocol at each node. An attacker disseminates ROUTE REQUESTs quickly throughout the network, suppressing any later legitimate ROUTE REQUESTs when nodes drop them due to the duplicate suppression.

- *Resource Consumption attacks*

In a resource consumption attack, the attacker injects packets into the network in an attempt to consume valuable network resources such as bandwidth, or to consume node resources such as memory (storage) or computation power.

An example of a resource consumption attack is for an attacker to inject extra data packets into the network, which will consume bandwidth resources when forwarded, especially over detours or routing loops. Similarly, an attacker can inject extra control packets into the network, which may consume even more bandwidth or computational resources as other nodes process and forward such packets. With either of these attacks, an Active-VC attacker can try to extract maximum resources from the nodes on both sides of the vertex cut; for example, it might forward only routing packets and not data

packets, such that the nodes waste energy forwarding packets to the vertex cut, only to have them dropped.

2.3 MANET Attack Tree

Having discussed the security issues in MANETs, the attacks can be classified as:

Active attacks

- Incorrect forwarding
  - No forwarding
    - Data packets
    - Routing packets
      - Error packets
      - Route request packets
      - Route reply packets
  - Too slow
  - Replay
  - Changing of packet (before forwarding)
    - Route change
      - Silent route change
      - Route salvaging although no error has been observed (DSR specific)
    - Data manipulation
  - Forwarding messages to partners for analysis
- Denial of service
  - Bogus routing information
    - Replay of old routing information
    - 'Black hole routes'
  - Distorting routing information
  - Cause overload
    - Sending route updates at short intervals
    - Sending route requests at short intervals
- Lack of error messages, although an error has been observed

- Gathering information
  - Unusual traffic attraction
    - ❖ Advertising many very good routes
    - ❖ Choose a very short reply time, so the route will be prioritized

Passive attacks (eavesdropping)

- Gathering information
  - Use promiscuous mode to listen to traffic destined for other nodes

2.4 Security Mechanisms

Security mechanisms for wireless ad-hoc networks should aim to provide all the security services listed above and prevent any of the attacks mentioned. However, due to the lack of infrastructure in an ad-hoc wireless network, typical wired-network implementations of the methods mentioned above may not be possible. Along with the general issues listed above, there are also other specific key issues and challenges for providing security in ad-hoc.

- *Link Level Security:* In wireless environment the links are susceptible to attacks where eavesdropper can intercept data packets. Physical barriers such as walls, rooms, etc. provide no barrier to wireless radio packets.
- *Routing/Network layer Security:* The routing within ad hoc networks is more vulnerable to attack as each device itself acts as a router. An attacker can pose as a member node and incorrectly route packets to achieve an attack. Denials of service attacks are particularly easy doing this. Thus implementation of secure routing protocol is one of the challenges within ad hoc network. The use of IPSec to provide authentication, confidentiality and integrity is discussed in this report. By securing all IP traffic (or whatever network layer protocol is used), you are also securing routing.
- *Key Management:* General network security implementation of keys involves a trusted authority. Given the lack of infrastructure in ad-hoc, it is generally not possible to have a fixed trusted authority; therefore an alternative to this is required.

Chapter 3

Review of the State of Art

3.1 Overview of Network Security in MANETs

When discussing network security in general, two aspects need to be considered: the services required and the potential attacks. The security services aspect includes the functionality that is required to provide a secure networking environment while the security attacks cover the methods that could be employed to break these security services.

The salient features of ad hoc networks pose both challenges and opportunities in achieving these security goals:

- Firstly, use of wireless links renders an ad hoc network susceptible to link attacks ranging from passive eavesdropping to active impersonation, message replay, and message distortion. Eavesdropping might give an adversary access to secret information, violating confidentiality. Active attacks might allow the adversary to delete messages, to inject erroneous messages, to modify messages, and to impersonate a node, thus violating availability, integrity, authentication, and non-repudiation.
- Secondly, nodes, roaming in a hostile environment (e.g., a battlefield) with relatively poor physical protection, have non-negligible probability of being compromised. Therefore, we should not only consider malicious attacks from outside a network, but also take into account the attacks launched from within the network by compromised nodes. Therefore, to achieve high survivability, ad hoc networks should have a distributed architecture with no central entities. Introducing any central entity into our security solution could lead to significant vulnerability; that is, if this centralized entity is compromised, and then the entire network is subverted.
- Thirdly, an ad hoc network is dynamic because of frequent changes in both its topology and its membership (i.e., nodes frequently join and leave the network),

and as a result the trust relationship among nodes also changes, for example, when certain nodes are detected as being compromised. Unlike other wireless mobile networks, such as mobile IP, nodes in an ad hoc network may dynamically become affiliated with administrative domains. Any security solution with a static configuration would not suffice. It is desirable for our security mechanisms to adapt to these changes on the fly.

Finally, an ad hoc network may consist of hundreds or even thousands of nodes. Security mechanisms should be scalable to handle such a large network.

3.2 Network Security Goals

In providing a secure networking environment some or all of the following services may be required:

- *Confidentiality:* It ensures that the intended receivers can only access the transmitted data. This is generally provided by encryption. Two types of encryption are commonly used (a detailed description of these is outside the scope of this report).
  - *Symmetric Encryption*, where 2 nodes share a key (e.g. - DES, AES). Any data transmitted between the nodes is encrypted using this key. This key must be provided to the nodes over a secure channel. Symmetric encryption generally requires less computational resources than public key encryption.
  - *Public Key Encryption*, where all nodes participating generate a public\private key pair *puck/ proven*. The node makes its public key *puck* available to all nodes. If other nodes wish to send data to node *n*, they encrypt their data using *puck,* safe in the knowledge that it can only be decrypted by *n*'s private key *proven,* which only node n knows.

- *Integrity*: Ensures that the data has not been altered during transmission. The integrity service can be provided using cryptographic hash functions along with some form of encryption. When dealing with network security the integrity service is often provided implicitly by the authentication service.

- *Authentication*: Both sender and receiver of data need to be sure of each other's identity. Authentication can be provided using encryption along with cryptographic hash functions, digital signatures and certificates. Details of the construction and operation of digital signatures can be found in RFC2560 [14].

- *Non-repudiation*: Ensures that parties can prove the transmission or reception of information by another party, i.e. a party cannot falsely deny having received or sent certain data. Non-repudiation requires the use of public key cryptography to provide digital signatures. A trusted third party is required to provide a digital signature, such as Thawte [15] used on the Internet, nowadays.

Availability: Ensures that the intended network security services listed above are available to the intended parties when required. The availability is typically ensured by redundancy, physical protection and other non-cryptographic means, e.g. use of robust protocols. This is a vague metric and is provided in varying degrees by all security protocols.

3.3 Payment Systems

Payment systems provide economic incentives for the cooperation in MANET. They consider that each node in MANET is its own authority and tries to maximize the benefits it gets from the network. Thus each node tends to be selfish, dropping packets not destined to them but make use of other nodes to forward their own packets. The purpose of payment systems is to encourage the cooperation within the MANET by economic incentives. There are several variations of payment systems proposed, discussed below:

3.3.1 Nuglets

Nuglets [16] is a virtual currency mechanism for charging (rewarding) server usage (provision). Nodes that use a service must pay for it (in nuglets) to nodes that provide the service. A typical service is packet forwarding that is provided by intermediate nodes to the source and the destination of the packet. Therefore either the source or the destination

should pay for it. There are two models for charging for the packet forwarding service: the Packet Purse Model (PPM)[17] and the Packet Trade Model (PTM)[17].

In the PPM, the sender pays for the packet. It loads the packet with a number of nuglets when sending the packet. Each intermediate forwarding node acquires some nuglets from the packet that covers its forwarding costs. If a packet does not have enough nuglets to be forwarded, then it is discarded. If there are nuglets left in the packet once it reaches destination, the nuglets are lost.

In the PTM, the destination pays for the packet. Each intermediate node "buys" the packet from previous one for some nuglets and "sells" it to the next one for more nuglets until the destination "buys" it.

Both the models have their advantages and disadvantages. While the Packet Purse Model deters nodes from sending useless data and avoids the network overloading, the Packet Trade Model can lead to an overload of the network and the destination receives packets it does not want. On the other hand, in the Packet Purse Model it is difficult to estimate the number of nuglets that are required to reach a given destination. But the Packet Purse Model does not need to consider this problem.

To take advantages of the two models and avoid the disadvantages, a ***hybrid model*** is suggested. In this model, the sender loads the packet with some nuglets before sending it. The packet is handled according to the Packet Purse Model until it runs out of nuglets. Then it is handled according to the Packet Trade Model until the destination buys it.

3.3.2 Counter

To address the problems encountered by the nuglets approach such as difficulty in estimating pre-load nuglets and possible network overload, another payment approach based on credit counter is suggested [18]. In this approach, the current state of each node is described by two variables *b* and *c*, where *b* is the remaining battery power and *c* stands for the value of its nuglet counter. More precisely, *b* is the number of packets that

the node can send using its remaining energy and c is the number of packets a node can originate. A node can originate a number of packets N only when the condition $c \geq N$ holds. When a node forwards a packet, nuglet counter c is increased by one and b is reduced by one. Thus in order to originate packets, each node must earn credits by forwarding packets. The counter solution requires tamper resistant hardware security module.

3.3.3 Sprite [19]

S. Zhong et al. proposed Sprite, a credit-based system for MANET. As opposed to Nuglets or Counter they do not require tamper-proof hardware to prevent the fabrication of payment units. Instead, they introduce a central Credit Clearance Service (CCS) [19]. The basic scheme of the system is as follows. When a node receives a message the node keeps a receipt of the message and reports to the CCS when the node has a fast connection to CCS. The CCS then determines the charge and credits to each node involved in the transmission of a message, depending on the reported receipts of a message.

In this scheme, the sender charges money. A node that has forwarded a message is compensated, but the credit that a node receives depends on whether or not its forwarding action is successful. Forwarding is considered successful if and only if the next node on the path reports a valid receipt to the CCS.

3.3.4 Discussion on the Payment Systems

The payment systems we discussed in above sections either assume a tamper resistant hardware module is available to ensure that the behavior of the node is not modified or requires a central authority server to determine the charge and credit to each node involved in the transmission of a message. Tamper resistant hardware may not be appropriate for most mobile devices because it demands advanced hardware solution and increases the cost of the devices. Lacking of central authority server is right the inherent property of MANET that causes security challenges so it is also not appropriate.

Furthermore, all the approaches described above suffer from locality problems [20] that nodes in different locations of the network will have different chances for earning virtual currency, which may not be fair for all nodes. Usually nodes at the periphery of the network will have less chance to be rewarded.

3.4 Reputation System

Reputation systems have emerged as a way to reduce the risk entailed in interactions among total strangers in electronic marketplace [21]. Centralized reputation systems have been adopted by many on-line electronic auctions to collect and store reputation ratings from feedback providers in a centralized reputation database. Decentralized reputation systems used by MANET, on the other hand, do not use centralized reputation database. Instead, in these reputation systems, each node keeps the ratings about other node and updating the ratings by direct observation of the behaviors of neighboring nodes or second hand information from other trusted nodes.

Most of the reputation systems in MANET are based on trust management system. Trust is such a subjective and dynamic concept that different entities can hold different opinions on it even while facing the same situation. Trust management system can work without reputation system. For example, a mobile node can form opinion about other nodes by direct experience with the nodes.

We can unify reputation system and trust management system to dynamic feedback mechanisms. The former one is a global reputation system and mobile nodes share their own experiences of interaction with other nodes. The later one is a local reputation system in which mobile nodes rating the trustability of other nodes based on its own observation.

3.4.1 Cooperation Of Nodes: Fairness In Dynamic Ad-hoc Network (CONFIDANT)

CONFIDANT is a reputation system aiming at coping with misbehavior in MANET [22] [23] [24]. The idea is to detect the misbehaved nodes and isolate them from communication by not using them for routing and forwarding and by not allowing the

misbehaved nodes to use it to forward packets. CONFIDANT stands for Cooperation Of Nodes: Fairness In Dynamic Ad-hoc Network. It usually works as an extension to on demand routing protocols.

With CONFIDANT each node has four components: Monitor, Reputation System, Trust Manager and Path Manager. These components interact with each other to provide and process protocol information. Figure 3-1 illustrates the architecture of the CONFIDANT components based on DSR protocol [22].

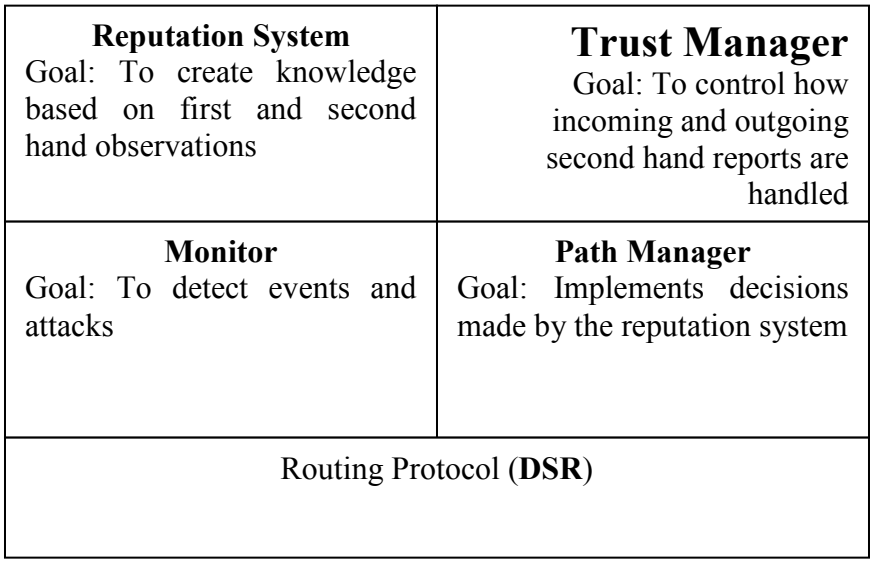

Figure 3-1 CONFIDANT components [24]

1. MONITOR is responsible for gathering firsthand information about the behavior of other nodes in the network. This is achieved by observing and detecting various attacks. A typical misbehavior is packet dropping. The monitor detects it by an enhanced Passive Acknowledge mechanism. The monitor can also detect other attacks such as message modification and fabrication through overhearing the packets forwarded by next hop.
2. REPUTATION SYSTEM is the core component of CONFIDANT. It is responsible for maintaining reputation rating about other nodes in the network. The reputation rating about other nodes is updated based on the firsthand

information observed by the node or the second hand information published by other nodes. Reputation System decides whether to accept secondhand information and how much the information is incorporated to update reputation ratings. Based on reputation ratings, Reputation System identifies misbehaved nodes.

3. TRUST MANAGER maintains the trust rating about other nodes in the network. Trust rating represents a node's opinion about how honest another node is as an actor in the reputation system. It is used as an alternative way to decide whether to accept second hand information. The benefit of using trust is to speed up the detection of misbehaved nodes.
4. PATH MANAGER performs actions once a misbehaved node is identified, e.g. deletion of paths containing misbehaved nodes, action on receiving request for a rout containing a misbehaved node in the source route, etc. With CONFIDANT each node collects two major types of information about other nodes that it has communicated or heard about in the network: first-hand information and second-hand information. Based on the information the reputation rating is updated. To have an accurate estimation of misbehavior, Bayesian estimation is employed to form reputation ratings and making other decisions.

CONFIDANT distinguishes trust from reputation. For each node, reputation-rating represents how well a node behaves while trust rating represents how honest a node is. Reputation rating is used to decide whether the node is regular or misbehaved, while trust rating is used to decide whether the node is trustworthy or not as one who can recommend.

The following description illustrates how CONFIDANT works to mitigate the misbehavior in the network:

When a node sends a packet to its neighboring node that is supposed to forward the packet, the node detects whether the neighboring node forwards the packets by listening the packet in a promiscuous mode, which is called Passive Acknowledgement. If it hears the packet is forwarded, it updates the first-hand information and increases the reputation

rating about the neighboring node. If it doesn't hear the packet within a certain time, it thinks the neighboring node misbehaves and decreases its reputation rating.

Whenever the reputation rating about another node is updated, the node will identify whether it is misbehaved node or not by comparing the reputation rating with a misbehaved threshold. The identified misbehaved nodes will be reported to Path Manager that will take further actions.

Every node periodically spreads the firsthand information it has collected to its neighboring nodes. If a node receives the published firsthand information and thinks the source of the information is trustable, it will incorporate the information to update the reputation ratings it keeps. It is very important to know that with CONFIDANT a node only forwards or responds to nodes with good behavior. In this way, it isolates misbehaved node by bearing grudges to it.

3.4.2 COllaborative REputation mechanism (CORE) [17]

Similar to CONFIDANT, CORE also provides a mechanism to enforce node cooperation based on a collaborative monitoring technique. However, CORE is different from CONFIDANT in reputation model and the way to spread rumor. Three types of reputations are used in the CORE.

- Subjective reputation of a target node is the reputation calculated directly from a subject's observation of the target node's behavior.
- Indirect reputation is evaluated only considering the direct interaction between a subject and its neighbors.
- Function reputation is the subjective and indirect reputation calculated with respect to different functions such as forwarding a data packet, reply route request.

The final reputation information is combined from the three reputations with different weight associated to the functional reputation value. CORE consists of two basic components:

1. Reputation Table (RT) is a data structure stored in each network entity, keeping the reputation data pertaining to the nodes in the network, and
2. The Watchdog mechanism (WD) is used to detect misbehaved nodes.

With CORE only positive rating factors are distributed among the entities to avoid a misbehaving entity to distribute false information about other entities in order to initiate a denial of service (DoS) attack.

3.4.3 Locally Aware Reputation System (LARS) [25]

Jiangyi Hu proposed a simple reputation based scheme, called LARS to mitigate misbehavior and enforce cooperation. Different from global reputation based schemes, with LARS each node *X* only keeps the reputation values of all its one-hop neighbors *N (X)*. The reputation values are updated based on the direct observation of the neighbors. If the reputation value of a neighbor node *M* is above threshold, then *M* is considered by *X* as misbehaved node. *X* will notify its neighbors about *M*'s misbehavior by initiating a warning message. To avoid false accusation, conviction of the uncooperative node is co-signed by *m* different nodes, where *m*-1 is an upper bound on the number of malicious nodes in the one-hop neighborhood. If the warning message is verified, it is then broadcasted to the *k*-hop neighborhood and *M*'s k-hop neighbors become aware of its misbehavior and refuse to serve for it.

3.4.4 Discussion on the Reputation Systems

Although different in reputation model and detailed implementation, all the reputation systems we have introduced have three common parts:

1. All the systems detect misbehavior using mechanisms similar to Passive Acknowledgement.
2. All the systems maintain reputation ratings about all or part of other nodes in the network and identify misbehaved nodes.
3. All the systems react to the misbehaved nodes.

Besides the systems introduced in this section, there are many other reputation systems for MANET. Basically they can be classified into following categories:

1. Global reputation system in which each node knows reputation value of every other node in the network. This is achieved by exchange indirect reputation message among the network. CONFIDANT and CORE are examples of global reputation system.
2. Local reputation system in which each node only keeps the reputation value of its neighboring nodes. Instead of distributing reputation value or information periodically, the local reputation systems usually update reputation value based on its own observation.

3.5 Watchdog and Pathrater [26]

Two techniques have proposed that improve throughput in an ad hoc network in the present of misbehaved nodes.

The ***watchdog*** method is used for each node to detect misbehaving nodes in the network. When a node sends a packet to next hop, it tries to overhear the packet forwarded by next hop. If it hears that the packet is forwarded by next hop and the packet matches the previous packet that it has sent itself, it considers the next hop behaves well. Otherwise it considers the next hop misbehaves.

The ***pathrater*** uses the knowledge about misbehaving nodes acquired from watchdog to pick the route that is most likely to be reliable. Each node maintains a trust rating for every other node. When watchdog detects a node is misbehaving, the trust rating of the node is updated in negative way. When a node wants to choose a safe route to send packets, pathrater calculates a path metric by averaging the node ratings in the path.

3.6 Key Management System [17]

We employ cryptographic schemes, such as digital signatures, to protect both routing information and data traffic. Use of such schemes usually requires a key management service.

We adopt a public key infrastructure because of its superiority in distributing keys and in achieving integrity and non-repudiation. Efficient secret key schemes are used to secure further communication after nodes authenticate each other and establish a shared secret session key.

In a public key infrastructure, each node has a public/private key pair. Public keys can be distributed to other nodes, while private keys should be kept confidential to individual nodes. There is a trusted entity called Certification Authority (CA) for key management. The CA has a public/private key pair, with its public key known to every node, and signs certificates binding public keys to nodes.

The trusted CA has to stay on-line to reflect the current bindings, because the bindings could change over time: a public key should be revoked if the owner node is no longer trusted or is out of the network; a node may refresh its key pair periodically to reduce the chance of a successful brute-force attack on its private key.

It is problematic to establish a key management service using a single CA in ad hoc networks. The CA, responsible for the security of the entire network, is a vulnerable point of the network: if the CA is unavailable, nodes cannot get the current public keys of other nodes or to establish secure communication with others. If the CA is compromised and leaks its private key to an adversary, the adversary can then sign any erroneous certificate using this private key to impersonate any node or to revoke any certificate.

A standard approach to improve availability of a service is replication. But a naive replication of the CA makes the service more vulnerable: compromise of any single replica, which possesses the service private key, could lead to collapse of the entire

system. To solve this problem, we distribute the trust to a set of nodes by letting these nodes share the key management responsibility.

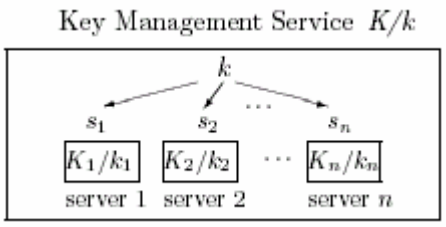

Figure 3-2 Configuration of a key management service: the key management service consists of n servers. The service, as a whole, has a public/private key pair K/k. The public key K is known to all nodes in the network, whereas the private key k is divided into n shares s_1, s_2, . . ., s_n, one share for each server. Each server i also has a public/ private key pair K_i/k_i and knows the public keys of all nodes [27].

3.6.1 Self Organized Public Key Infrastructure

A fully self-organized public key management system has been proposed that can be used to support security of ad hoc network routing protocols [28]. The suggested approach is similar to PGP [29] in the sense that users issue certificates for each other based on their personal acquaintances. However, in the proposed system, certificates are stored and distributed by the users themselves, unlike in PGP, where this task is performed by on-line servers (called certificate directories). In the proposed self-organizing public-key management system, each user maintains a *local certificate repository*. When two users want to verify the public keys of each other, they merge their local certificate repositories and try to find appropriate certificate chains within the merged repository that make the verification possible.

The success of this approach very much depends on the construction of the local certificate repositories and on the characteristics of the certificate graphs. By a certificate graph is meant to be a graph whose vertices represent public-keys of the users and the edges represent public-key certificates issued by the users. The authors investigate on several repository construction algorithms and study their performance. The proposed

algorithms take into account the characteristics of the certificate graphs in a sense that the choice of the certificates that are stored by each mobile node depends on the connectivity of the node and its certificate graph neighbors.

More precisely, each node stores in its local repository several directed and mutually disjoint paths of certificates. Each path begins at the node itself, and the certificates are added to the path such that a new certificate is chosen among the certificates connected to the last node on the path (initially the node that stores the certificates), such that the new certificate leads to the node that has the highest number of certificates connected to it (i.e., the highest vertex degree). The authors call this algorithm the *Maximum Degree Algorithm*, as the local repository construction criterion is the degree of the vertices in a certificate graph.

In a second, more sophisticated algorithm that is called the *Shortcut Hunter Algorithm*, certificates are stored into the local repositories based on the number of the shortcut certificates connected to the users. The shortcut certificate is a certificate that, when removed from the graph makes the shortest path between two users previously connected by this certificate strictly larger than two.

When verifying a certificate chain, the node must trust the issuer of the certificates in the chain for correctly checking that the public key in the certificate indeed belongs to the node identification (ID) named in the certificate. When certificates are issued by the mobile nodes of an ad hoc network instead of trusted authorities, this assumption becomes unrealistic. In addition, there may be malicious nodes that issue false certificates. In order to alleviate these problems, the authors propose the use of authentication metrics [30]: it is not enough to verify a node ID key binding via a single chain of certificates. The authentication metric is a function that accepts two keys (the verifier and the verified node) and a certificate graph and returns a numeric value corresponding to the degree of authenticity of the key that has to be verified: one example of authentication metric is the number of disjoint chains of certificates between two nodes in a certificate graph.

The authors emphasize that before being able to perform key authentication, each node must first build its local certificate repository, which is a relatively expensive operation (in terms of bandwidth and time). However this initialization phase must be performed rarely and once the certificate repositories have been built, then any node can perform key authentication using only local information and the information provided by the targeted node. It should also be noted that local repositories become obsolete if a large number of certificate are revoked, as then the certificate chains are no longer valid; the same comment applies in the case when the certificate graph changes significantly. Furthermore, PGP-like schemes are more suitable for small communities because that the authenticity of a key can be assured with a higher degree of trustiness. The authors propose the use of authentication metrics to alleviate this problem: this approach however provides only probabilistic guarantees and is dependent on the characteristics of the certificate graph on which it operates. The authors also carried out a simulation study showing that for the certificate graphs that are likely to emerge in self-organized systems, the proposed approach yields good performances both in terms of the size of the local repository stored in each node and scalability.

3.6.2 Threshold Cryptography - System model [31] [32]

Our key management service is applicable to an asynchronous ad hoc network; that is, a network with no bound on message-delivery and message-processing times. We also assume that the underlying network layer provides reliable links. The service, as a whole, has a public/private key pair. All nodes in the system know the public key of the service and trust any certificates signed using the corresponding private key. Nodes, as clients, can submit query requests to get other clients' public keys or submit update requests to change their own public keys.

Internally, our key management service, with an (n, t+1) configuration (n >= 3t+1), consists of n special nodes, which we call servers, present within an ad hoc network. Each server also has its own key pair and stores the public keys of all the nodes in the network. In particular, each server knows the public keys of other servers. Thus, servers

can establish secure links among them. We assume that the adversary can compromise up to t servers in any period of time with certain duration.

If a server is compromised, then the adversary has access to all the secret information stored on the server. A compromised server might be unavailable or exhibit Byzantine behavior (i.e., it can deviate arbitrarily from its protocols). We also assume that the adversary lacks the computational power to break the cryptographic schemes we employ.

The service is correct if the following two conditions hold:

- (**Robustness**) The service is always able to process query and update requests from clients. Every query always returns the last updated public key associated with the requested client, assuming no concurrent updates on this entry.
- (**Confidentiality**) The private key of the service is never disclosed to an adversary. Thus, an adversary is never able to issue certificates, signed by the service private key, for erroneous bindings.

3.6.2.1 Algorithm

Distribution of trust in our key management service is accomplished using threshold cryptography. An (n, t + 1) threshold cryptography scheme allows n parties to share the ability to perform a cryptographic operation (e.g., creating a digital signature), so that any t + 1 parties can perform this operation jointly, whereas it is infeasible for at most t parties to do so, even by collusion. In our case, the n servers of the key management service share the ability to sign certificates. For the service to tolerate t-compromised servers, we employ an (n, t+1) threshold cryptography scheme and divide the private key k of the service into n shares (s_1, s_2, . . . , s_n), assigning one share to each server. We call (s_1, s_2, . . . , s_n) an (n, t + 1) sharing of k. Figure 3-2 illustrates how the service is configured. For the service to sign a certificate, each server generates a partial signature for the certificate using its private key share and submits the partial signature to a combiner. With t + 1 correct partial signatures, the combiner is able to compute the signature for the certificate. However, compromised servers (there are at most t of them) cannot generate correctly signed certificates by themselves, because they can generate at

most t partial signatures. Figure 3-3 shows how servers generate a signature using a threshold signature scheme.

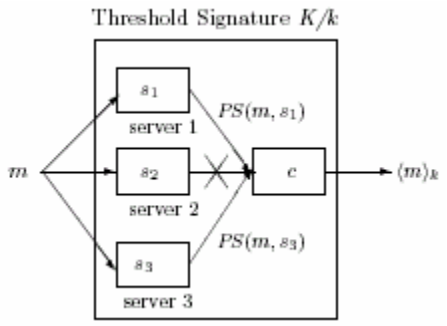

Figure 3-3 Threshold signature: given a service consisting of 3 servers. Let K/k be the public/private key pair of the service. Using a threshold cryptography scheme, each server i gets a share s_i of the private key k. For a message m, server i can generate a partial signature PS (m, s_i) using its share s_i. Correct servers 1 and 3 both generate partial signatures and forward the signatures to a combiner c. Even though server 2 fails to submit a partial signature, c is able to generate the signature <m>k of m signed by service private key k [27].

When applying threshold cryptography, we must defend against compromised servers. For example, a compromised server could generate an incorrect partial signature. Use of this partial signature would yield an invalid signature. Fortunately, a combiner can verify the validity of a computed signature using the service public key. In case verification fails, the combiner tries another set of t + 1 partial signatures. This process continues until the combiner constructs the correct signature from t + 1 correct partial signatures. More efficient robust combining schemes are proposed. These schemes exploit the inherent redundancies in the partial signatures (note that any t+1 correct partial signatures contain all the information of the final signature) and use error correction codes to mask incorrect partial signatures. A robust threshold DSS (Digital Signature Standard) scheme can also be used. The process of computing a signature from partial signatures is essentially an interpolation.

3.7 Authentication [33]

With only a valid key without knowing whom it belongs to, or integrity check on the data packets there are no real security achieved. Without knowing who you talk to, or at least, knowing that the one you are talking to are the one he or she claims to be, there is no real security in the system. This section is about the problem of authentication. Authentication is usually based on knowledge or trust and in the Internet this is commonly achieved using public key cryptography and digital signatures. Here two different approaches are presented. The first is based on a distributed model of trust and references, much like we act person to person in different situations. The second proposal is used to make the verification more efficient by not using public key cryptosystems as the main verification system. This second proposal fits neatly in broadcast environments and ad hoc networks where some packet might not reach the destination due to sporadic network connectivity changes.

3.8 Routing in Ad hoc Networks – DSR (Dynamic Source Routing)

3.8.1 Overview

Dynamic source routing is a Source routed On-Demand routing protocol in Ad Hoc networks. It uses Source Routing, which is a technique in which the sender of a packet determines the complete sequence of nodes through which the node has travel. The sender of the packet explicitly mentions the list of all nodes in the packet's header, identifying each forwarding 'hop' by the address of the next node to which to transmit the packet on its way to destination host. In this protocol the nodes don't need to exchange the Routing table information periodically and thus reduces the bandwidth overhead in the network. Each Mobile node participating in the protocol maintains a 'routing cache' that contains the list of routes that the node has learnt. Whenever the node finds a new route it adds the new route in its 'routing cache'. Each mobile node also maintains a sequence counter 'request id' to uniquely identify the requests generated by a mobile host. The pair < source address, request id > uniquely identifies any request in the Ad Hoc network. The protocol does not need transmissions between hosts to work in bi-direction.

3.8.2 Protocol Description

Route Discovery and Route Maintenance, which are the main mechanisms of the DSR protocol, allows the discovery and maintenance of source routes in the ad hoc network. DSR works entirely on an on-demand basis. DSR does not rely on functions like periodic routing advertisement, link status sensing or neighbor detection packets and because of the entirely on demand behavior, the number of overhead packets caused by DSR scales down to zero. As DSR works entirely on demand and as nodes begin to move continuously, the Routing packet overhead automatically scales to only that needed to react to changes in the route currently in use. In response to a single Route Discovery if a node learns and caches multiple routes to a destination, it can try another route if the one it uses fails. The overhead incurred by performing a new Route Discovery can be avoided when the caching of multiple routes to a destination occurs. In wireless networks, differing antenna, propagation patterns or sources of interference can cause the link between two nodes to not work efficiently in either direction. DSR improves the overall performance and network connectivity in the system by allowing unidirectional links to be used when necessary. Routing in DSR is integrated into standard Internet routing and Mobile IP routing and supports internetworking between different types of wireless networks [1].

3.8.3 DSR Route Discovery

The header of the packet, which originates from a source node S to a destination node D, contains the source route, which gives the sequence of hops that the packet should traverse. A suitable source route is found normally when searching the Route Cache of routes obtained previously but if no route is found then the Route Discovery protocol is initiated to find a new route to D. Here S is the initiator and D the target [1]. Node A transmits a ROUTE REQUEST message, which is received by all the nodes in the transmission range of A.

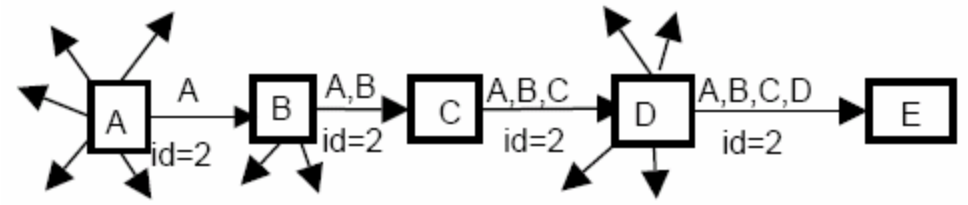

Figure 3-4 Node A is the initiator and Node E is the target [1]

Each ROUTE REQUEST message identifies the initiator and target of the Route Discovery and also contains a unique request ID, determined by the initiator of the REQUEST. Each ROUTE REQUEST also contains a record listing the address of each intermediate node through which this particular copy of the ROUTE REQUEST message has been forwarded. The initiator of the Route Discovery initializes the route record to an empty list [1]. When the target node receives the ROUTE REQUEST message, it returns a ROUTE REPLY message to the ROUTE Discovery initiator with a copy of the accumulated route record from the ROUTE REQUEST. This route is cached in the Route Cache when the initiator receives the ROUTE REPLY and is used in sending subsequent packets to this destination. When the target node finds a ROUTE REQUEST message from the same initiator bearing the same request ID or if it finds its own address is already listed in the route record of the ROUTE REQUEST message, it discards the REQUEST.

If the target node does not find the ROUTE REQUEST message from the initiator, then it appends its address to the route record in the ROUTE REQUEST message and propagates it by transmitting it as a local broadcast packet. When Route Discovery is initiated the copy of the original packet is saved in a local buffer called Send Buffer. The Send Buffer contains copies of each packet that cannot be transmitted by the sending node. The packets are kept until a source route is available or a timeout or Send Buffer overflow occurs. As long as a packet is in the Send Buffer, the node should initiate new Route Discovery until time out occurs or overflow of Buffer occurs. An exponential Back off algorithm is designed to limit the rate at which new ROUTE Discoveries may be initiated by any node for the same target [1].

3.8.4 DSR Route Maintenance

When a packet with a source route is forwarded, each node in the source route makes sure that the packet has been received by the next hop in the source route. The confirmation of receipt will be received only by re-transmitting the packet for a number of times [1].

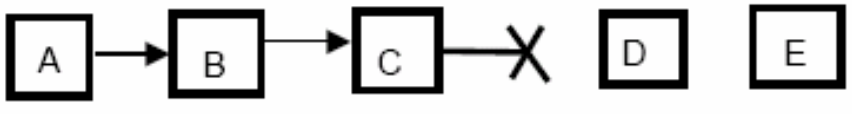

Figure 3-5 Node C is unable to forward a packet from A to E over the next node D [1]

Node A is the originator of a packet to the desired destination E. The packet has a source route through intermediate nodes B, C and D. Node A is responsible for receipt of the packet at B, node B at C, node C at D and node D at E. Node B confirms receipt of packet at C by overhearing C transmit the packet to forward it to D. The confirmation of acknowledgement is done by passive acknowledgements or as link-layer mechanisms such as option in MAC protocol. The node receiving the packet can return a DSR specific software acknowledgement if neither of the acknowledgements is available. This is done by setting up a bit in the packet's header and then requesting a DSR specific software acknowledgement by the node transmitting the packet. When a node is unable to deliver a packet to the next node then the node sends a ROUTE ERROR message to the original sender of the packet. The broken link is then removed from the cache by the originator of the packet and retransmissions to the same destination are done by upper layer protocols like TCP [1].

3.8.5 Additional Route Discovery Features

- **Caching Overhead Routing Information:** When a node is forwarding a packet, the routing information of the packet is added to the Route Cache of the node. A node may cache the source route, the accumulated route record and the route being returned in a ROUTE REPLY. The presence of unidirectional links in the ad hoc network is a limitation on caching of overhead information [1].
- **Replying to Route Requests using Cached Routes:** Before forwarding packets, nodes receiving a ROUTE REQUEST examine their Route Caches even if they are not targets. A ROUTE REPLY is send to the initiator when the ROUTE is

found. The ROUTE REPLY contains the route record to list the sequence of hops and the route from the node to the target from the node's Route Cache. There should be no duplicate nodes listed in the route record of the resulting route being returned in the ROUTE REPLY [1].

- **Preventing Route Reply Storms:** When a packet is sent to the local broadcast address, it might cause a large number of nodes to send Route Replies back to one source. If a node puts its network interface into promiscuous receive mode, it delays sending its own ROUTE REPLY for a short period. During the delay, the node receives and can verify all data packets from the initiator and can infer whether the initiator of the Route Discovery has already received a ROUTE REPLY showing a better route [1].
- **Route Request Hop Limits:** This mechanism is used to determine whether the target is a neighbor of an initiator or if the neighbor node has a route to the target cached. A hop limit is used to limit the number of intermediate nodes allowed to forward the copy of the ROUTE REQUEST. Before finding the target and when the REQUEST is forwarded, the limit is decreased and the REQUEST packet is discarded if the limit reaches zero [1].

3.8.6 Additional Route Maintenance Features [1]

- **Packet Salvaging:** A node may want to salvage the data packet that caused the ROUTE ERROR. The Route Cache of the node is searched for a route from itself to the destination of the packet causing the ERROR. The original source route on the packet with the route from its Route Cache is replaced for the route being found. The packet is forwarded to the next node in the source route. Backtracking from a current node to an earlier node is allowed in this method of packet salvaging but the packet can be salvaged only once.
- **Automatic Route Shortening:** Automatic shortening is the method when one or more of the intermediate hops become unnecessary, the source route gets shortened. Automatic shortening of routes is similar to passive acknowledgements. A node by operating its network interface in promiscuous receive mode is able to overhear a packet carrying a source route. If so, then the

node examines the route's unused portion. The intermediate nodes in the source route are not needed if the node is not the intended next hop for the packet but is named in the later unused portion of the source route of the packet.

- **Increased Spreading of Route Error Messages:** When a source node receives a ROUTE ERROR for a data packet that it originated, it propagates it to its neighbors by piggybacking it on its next ROUTE REQUEST. Stale information in the caches of nodes around this source node will not generate ROUTE REPLYs that contain the same invalid link for which this source node received the ROUTE ERROR.

Chapter 4

Problem Statement

4.1 Problem Statement and Motivation

Most current ad hoc routing protocols assume that the wireless network is benign and every node in the network strictly follows the routing behavior and is willing to forward packets for other nodes. Most of these protocols cope well with the dynamically changing topology. However, they do not address the problems when misbehavior nodes present in the network.

A commonly observed misbehavior is packet dropping. Practically, in a MANET, most devices have very limited computing and battery power while packet forwarding consumes a lot of such resources. Thus some of the mobile devices would not like to forward the packets for the benefit of others and they drop packets not destined to them. On the other hand, they still make use of other nodes to forward packets that they originate. These misbehaved nodes are very difficult to identify because we cannot tell that whether the packets are dropped intentionally by the misbehaved nodes or dropped due to the node having moved out of transmission range or other link error. Packet drop significantly decreases the network performance.

Traditional security mechanisms are generally not suitable for MANET because:

- The network lacks central infrastructure to apply traditional security mechanism such as access control, authentication and trusted third party.

- Limited bandwidth, battery lifetime, and computation power prohibits the deployment of complex routing protocols or encryption algorithms. New security models or mechanisms suitable for MANET must be found.
- Network topologies and memberships are constantly changing. Thus new intrusion detection system and entity recognition mechanisms that are suitable for mobile ad hoc networks must be designed to avoid or mitigate the behavior to the networks.

TRUST MANAGEMENT SYSTEMS have been recently introduced as a security mechanism in MANET. In a trust management system, a communicating entity collects evidence regarding competence, honesty or security of other network participants with the purpose of making assessment or decisions regarding their trust relationships [34].

Here trust means the confidence of an entity on another entity based on the expectation that the other entity will perform a particular action important to the one who trusts, irrespective of the ability to monitor or control that other entity [35]. For example, a trust-based routing protocol can collect the evidence of nodes misbehaving, form trust values of the nodes and select safest routes based on the trust metrics.

Reputation systems are often seen as a derivation of trust management system. In the reputation system, an entity forms its trust on another entity based not only on the self-observed evidence but also on the second hand information from third parties. One of the influential reputation systems is the CONFIDANT protocol.

In the trust management system, reputation system and other trust-based systems, route selection is based on the sending node's prior experience with other nodes in the network. Its opinions about how other entities are honest are constantly changing. Thus, the trust management systems and their derivations are known as dynamic feedback mechanisms [2]. The dynamic feedback mechanisms are usually applied on the current ad hoc routing protocols to rate the trust about other nodes in the network and make routing decisions based on the trust matrix, which is formed according to the evidence collected from previous interactions. By incorporating the dynamic feedback mechanism in the routing

protocol, misbehaved nodes are identified and avoided to forward packets. In this way, misbehavior can be mitigated.

4.2 Objective and Sub-tasks

The primary objective of this thesis is to —

To analyze, implement and evaluate CONFIDANT protocols to demonstrate how the dynamic feedback mechanisms improve the network performance and what are the side effects of introducing the mechanism to the mobile ad hoc network. Design a protocol for the distribution of reputation data based on the above analysis.

The above problem can be broken into following sub-tasks:

1. Investigate security issues of mobile ad hoc network and current dynamic feedback mechanisms or protocols that are used to solve or mitigate the issues.
2. Investigate and learn how to use the network simulation tool. There are several popular network simulation tools available and we need to choose the one that best suits our needs. The selected network simulator should be studied so that we can use it as platform to implement protocol and conduct simulations.
3. Analyze and implement the CONFIDANT protocol based on Dynamic Source Routing protocol (DSR); evaluate the network performance.
4. Based on the above analysis, propose a protocol for the distribution of reputation data using reputation and direct trust.

Chapter 5

Simulation, Performance Evaluation & Design

This chapter introduces the most important features, models and existing problems in the DSR protocol, CONFIDANT protocol and ns2. These features, models and problems will impact the software design and performance evaluation.

5.1 Network Simulation [36] [37] [38]

5.1.1 Simulation

Simulation can be defined as "Imitating or estimating how events might occur in a real situation." It can involve complex mathematical modeling, role-playing without the aid of technology, or combinations. The value lies in the placing you under realistic conditions that change as a result of behavior of others involved so you cannot anticipate the sequence of events or the final outcome

5.1.2 NS2-Overview

NS is an event driven network simulator developed at UC Berkeley that simulates variety of IP networks. It implements network protocols such as Transmission Control Protocol and User Datagram Protocol, traffic source behavior such as File Transfer Protocol, Telnet, Web, Constant Bit Rate and Variable Bit Rate, router queue management mechanism such as Drop Tail, and Random Early Discard, routing algorithms such as Dijkstra, and more. NS also implements multicasting and some of the MAC layer

protocols for LAN simulations. The NS project is now a part of the VINT project [39] that develops tools for simulation results display, analysis and converters that convert network topologies generated by well-known generators to NS formats. Currently, NS (version 2) written in C++ and OTcl (Tcl script language with Object-oriented extensions developed at MIT) is available. This document talks briefly about the basic structure of NS, and explains in detail how to use NS mostly by giving examples. Most of the figures that are used in describing the NS basic structure and network components are from the 5th VINT/NS Simulator Tutorial/Workshop slides and the NS Manual (formerly called "NS Notes and Documentation") modified little bit as needed.

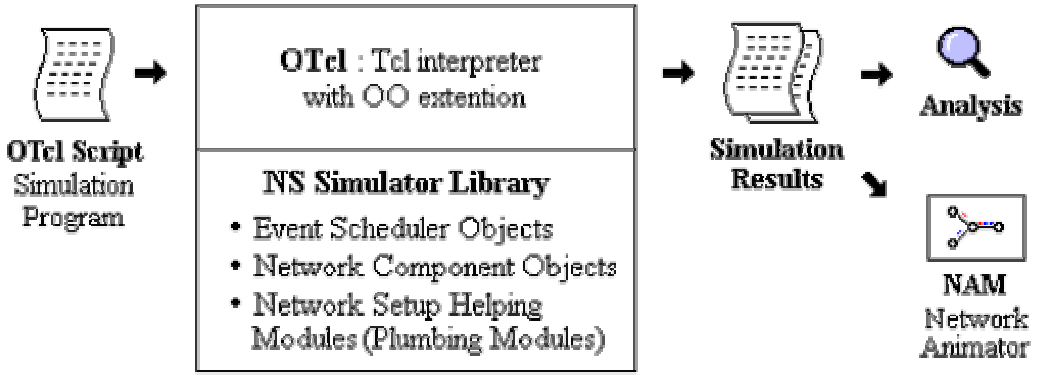

Figure 5-1 Simplified User's View of NS [37]

As shown in Figure 5-1, in a simplified user's view, NS is Object-oriented Tcl (OTcl) script interpreter that has a simulation event scheduler and network component object libraries, and network setup (plumbing) module libraries (actually, plumbing modules are implemented as member functions of the base simulator object). In other words, to use NS, you program in OTcl script language. To setup and run a simulation network, a user should write an OTcl script that initiates an event scheduler, sets up the network topology using the network objects and the plumbing functions in the library, and tells traffic sources when to start and stop transmitting packets through the event scheduler. The term "plumbing" is used for a network setup, because setting up a network is plumbing possible data paths among network objects by setting the "neighbor" pointer of an object to the address of an appropriate object. When a user wants to make a new network object, he or she can easily make an object either by writing a new object or by making a

compound object from the object library, and plumb the data path through the object. This may sound like complicated job, but the plumbing OTcl modules actually make the job very easy. The power of NS comes from this plumbing.

Another major component of NS beside network objects is the event scheduler. An event in NS is a packet ID that is unique for a packet with scheduled time and the pointer to an object that handles the event. In NS, an event scheduler keeps track of simulation time and fires all the events in the event queue scheduled for the current time by invoking appropriate network components, which usually are the ones who issued the events, and let them do the appropriate action associated with packet pointed by the event. Network components communicate with one another passing packets, however this does not consume actual simulation time. All the network components that need to spend some simulation time handling a packet (i.e. need a delay) use the event scheduler by issuing an event for the packet and waiting for the event to be fired to itself before doing further action handling the packet. Another use of an event scheduler is timer. Timers use event schedulers in a similar manner that delay does. The only difference is that timer measures a time value associated with a packet and does an appropriate action related to that packet after a certain time goes by, and does not simulate a delay.

NS is written not only in OTcl but in C++ also. For efficiency reason, NS separates the data path implementation from control path implementations. In order to reduce packet and event processing time (not simulation time), the event scheduler and the basic network component objects in the data path are written and compiled using C++. These compiled objects are made available to the OTcl interpreter through an OTcl linkage that creates a matching OTcl object for each of the C++ objects and makes the control functions and the configurable variables specified by the C++ object act as member functions and member variables of the corresponding OTcl object. In this way, the controls of the C++ objects are given to OTcl. It is also possible to add member functions and variables to a C++ linked OTcl object. The objects in C++ that do not need to be controlled in a simulation or internally used by another object do not need to be linked to OTcl. Likewise, an object (not in the data path) can be entirely implemented in OTcl.

5.2 Analysis of DSR Protocol

Apart from the basic functions of DSR protocol, some additional features of DSR that will impact the network performance especially after the integration with CONFIDANT. We analyze these features and decide whether to include them in the simulation. In the following section, the DSR protocol is the standard DSR and the CONFIDANT fortified DSR protocol is called CONFIDANT.

5.2.1 Importance of Redundant Routes

All the dynamical feedback mechanisms investigated in chapter 3 rely on inherent redundancies – multiple routes available to a single destination. As long as there are enough good nodes and alternative routes, packets can go around those misbehaved nodes and arrive at a destination. Thus increasing the number of available routes to the same destination is very important.

The route cache is used to store routes in the standard DSR. When a node gets a new route, either by initiating a new route discovery or by overhearing a packet that contains route information, it adds the route into the route cache. When the node wants to send a packet to a destination node, it searches a shortest route to the destination in the route cache. If no route is found, the node will send out a Route Request to find new routes.

With standard DSR, a node selects a shortest route to the destination from the route cache.

In CONFIDANT, however, it works in different way. Rather than calculating the shortest hops, a node selects the route that contains no misbehaved nodes. The more alternative routes available in route cache, the more possibility that a node can find a qualified route. The additional feature should be enabled if it can increase the number of routes discovered or cached. Otherwise it should be disabled.

5.2.2 Only Forwarding the First Route Request Message

When a node initiates a Route Discovery, it broadcasts the Route Request message to all its neighboring nodes and these neighboring nodes will append its address to the address list of the message and propagate the copy of the message to their own neighbors. There will be message flooding if the Route Request is forwarded unlimitedly. To mitigate the problem, DSR protocol specifies that a node should only forward the first copy of the same Route Request message it receives.

Figure 5-2 illustrates why the propagation of Route Request should be controlled. The intermediate node A receives a Route Request message for the first time and forwards it to B (see the dashed arrow). B forwards the Route Request to its neighboring nodes (see the dashed-dotted arrow). A receives the Route Request a second time and drops it. If the propagation of Route Request message is not controlled, A will forward the message again to B (see the dotted arrow) and the message will be forwarded endlessly.

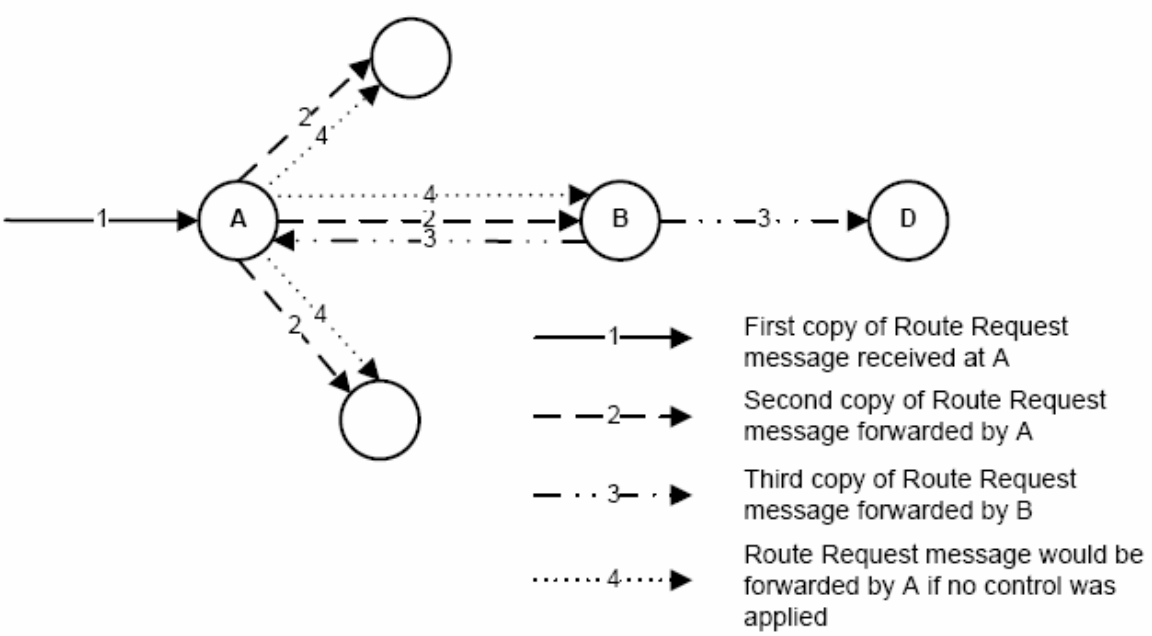

Figure 5-2 Node A only forwards the first Route Request message

Figure 5-3 (given on next page) illustrates another scenario of Route Request propagation control. To discover a route to the destination E, the source node S broadcasts a Route Request message to its neighboring nodes A, B and C. The node D receives the three

copies of Route Request message from node A, B and C. (We assume D receives the first copy from B.) If D forwards all the three copies of the Route Request to E, then the number of packets is tripled compared with the case that D only forwards the Route Request it receives from B. Thus in this case, DSR requires that D only forwards the first Route Request and discards the other two.

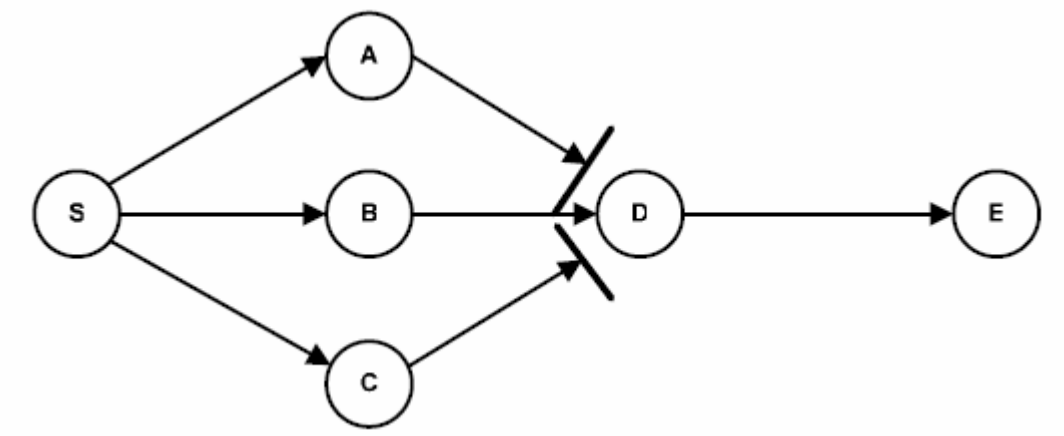

Figure 5-3 Node D only forwards the first Route Request message

Thus, the feature has two contradictory consequences in the implementation: enabling the feature will increase the number of Route Discovery; disabling the feature will cause message flooding. Since message flooding can cause much more serious problem we think it is reasonable for the feature to be applied.

5.2.3 Replying to Route Requests Using Cached Routes

DSR allows a node receiving a Route Request for which it is not the target to reply with the route found in its own route cache. In the Route Reply, this node appends the source route to the target node obtained from its own route cache after a sequence of hops over which the route request has been forwarded to it. After the node sends out the Route Reply, it will not propagate the Route Request any further.

Figure 5-4 illustrates the scenario of replying to Route Request using cached routes. Source node S initiates a Route Request to find routes to destination D. When B receives the Route Request, it searches its route cache and finds a route B →C →D to the

destination D. Then B concatenates the route with the accumulated source route S→A in the Route Request and sends the Route Reply to source node S. Node B will not broadcast the route request further in this case.

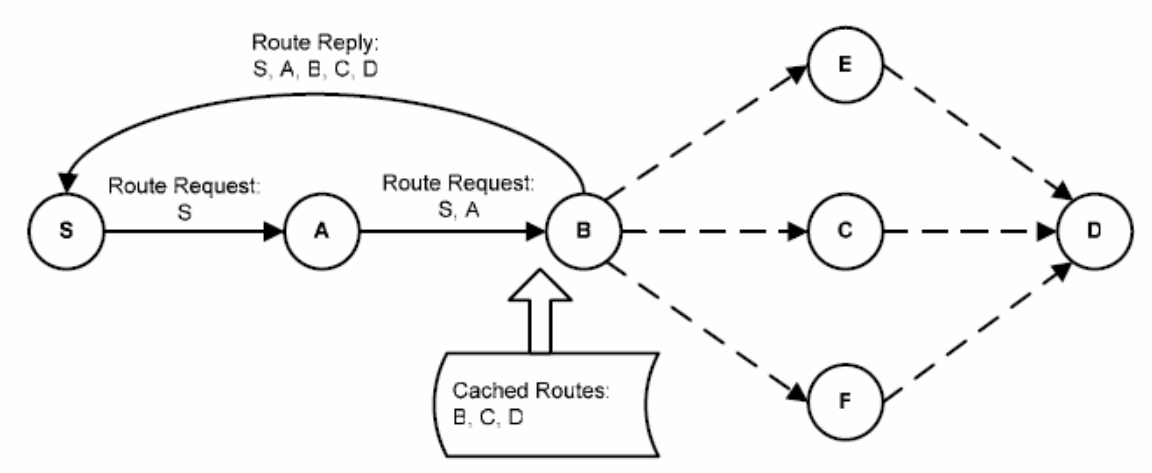

Figure 5-4 Replying to Route Requests using cached routes

This feature, however, will decrease the number of routes discovered. For example, if B does not reply the route request with the cached routes and instead forwards the route request to node C, E, and F. Then we can get two additional routes: S→A→B→E→D, S→A→B→F→D. (As seen in Figure 5-3, the dashed lines indicates the routes could be found). But with this feature we can only get one route. Thus, we will disable this feature.

5.2.4 Route Request Hop Limit

In DSR, each Route Request message contains a "hop limit" that may be used to limit the number of intermediate nodes allowed to forward the Route Request. DSR allows a ring search feature to discover routes. In the ring search mode, the source node sends out a Route Request with hop limit zero, which means only the neighboring nodes, can receive the Route Request. If no route was found in the first round of route discovery, the hop limit will be increased to allow the Route Request to be forwarded to greater range of nodes.

This feature can save the routing overhead in case the destination node is one of the neighbors of the source node. However, it will also limit the number of the routes being discovered and thus will be disable in the implementation.

5.3 Analysis of CONFIDANT Protocol

In this section we will analyze the CONFIDANT to make our way towards design. We will discuss the misbehavior detection and data representation that affects the implementation. We will also discuss the network overhead introduced by CONFIDANT that is one of the tasks of performance analysis.

5.3.1 States and Events

As introduced earlier, there are five modules in CONFIDANT: DSR, Monitor, Reputation system, Trust manager and Path manager. These modules are interrelated together to provide and process all kinds of information. The modules also interact with DSR to send and receive packets.

The dashed lines describe how the first hand information is collected. When a node i receives a packet in the promiscuous mode from another node *j* within the DSR module, it passes the tapped packet to the Monitor to detect whether it is the PACK packet. If it is, the ratings about *j* will be updated. If the reputation rating is greater than misbehaved threshold, it will inform Path manager to delete all the paths that contains the node *j* from the route cache of node *i*.

Figure 5-5 shows the interactions between the modules of CONFIDANT and DSR in each node. The ovals within each module represent the most important states of the module and the arrow lines indicate events or message between the states.

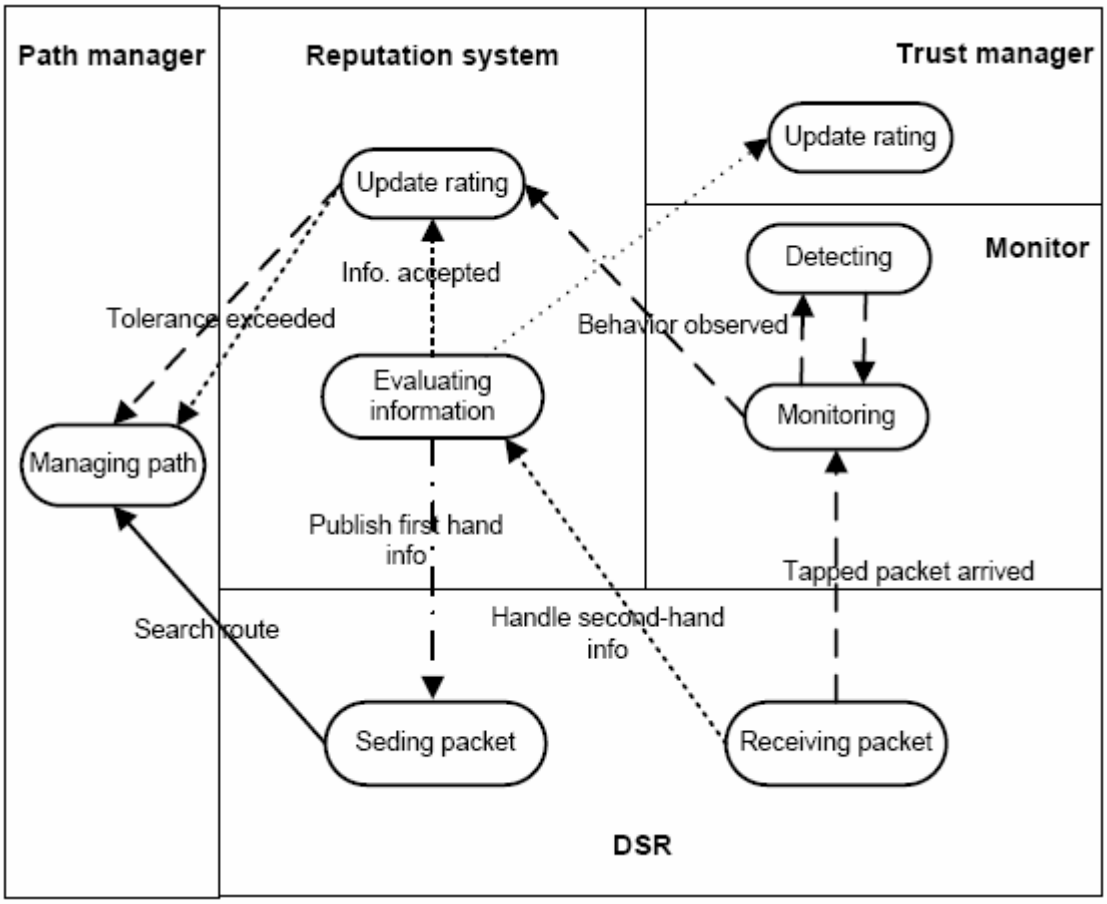

Figure 5-5 Finite state machine implemented in each node [24]

The dotted lines describe how second hand information published by the other nodes is handled. As seen in the figure, when node *i* receives published information it passes the information to the Reputation system to decide whether it should be accepted. If the information is accepted, the ratings about node *j* are updated. If the reputation rating after updating exceeds tolerance threshold, all the paths that containing the node *j* will be deleted from Path manager.

The dashed-dotted line describes that a node periodically publishes the reputation ratings it has about other nodes in the network. As seen, reputation system periodically calls the DSR to send out the firsthand information.

5.3.2 Detection of Misbehavior

Misbehavior detection is an important part of CONFIDANT. Here, the various misbehaviors detected by CONFIDANT will be introduced.

5.3.2.1 Improved Passive Acknowledgement

CONFIDANT protocol uses the Passive Acknowledgement not only for an indication of the correct reception at the next hop but also to detect whether a node forwards packets that it is supposed to forward or not. In CONFIDANT the passive acknowledgement is improved so that it can have capability to detect more misbehavior types other than dropping packets. When a node overhears a packet, it checks following fields to see whether the packet matches the one it has sent previously.

- IP header: The TTL value must be decremented by one and only one.
- Route reply option: All fields
- Route error option: All fields

If any one of above fields does not match, the next hop node is considered misbehaved and its reputation rating is updated in favor of misbehaving. Otherwise, reputation rating is updated in favor of honest.

5.3.2.2 Detectable Misbehavior

Generally the misbehaved nodes can be classified into two categories. One is the selfish node that drops packets only for the purposes of saving battery since transmission consumes energy. The other is the evil node that may intentionally drop, modify or fabricate packets. The later one could cause much more serious problems. For examples, by modifying a data packet, incorrect information will be sent to destination; by sending forged routing packets.

Whatever the purposes of the attackers, CONFIDANT can effectively detect drop, modification and fabrication attacks through the PACK mechanism. If a node does not hear the next hop forwards its packet within PACK timeout, it knows the next hop drops

the packet. If the packet a node hears is different from the original one, it knows the next hop modifies the packet. If a node hears a packet which indicating the node participated to forward or originate but actually it does not, it know the next hop fabricates the packet.

Although CONFIDANT is effective in detecting and mitigating the security problems such as dropping packets and big liar, there are also some types of attacks that CONFIDANT cannot efficiently cope with. Here we present some of these attacks.

5.3.3 Bearing Grudges

One of the major features of CONFIDANT is to mitigate misbehavior by bearing grudges to the nodes identified as misbehaved nodes. On the other hand bearing grudge can also serve as incentive for nodes to behave well and thus improve network performance. Following are the possible ways to bear grudge to the misbehaved nodes.

- Do not select a route containing misbehaved nodes to forward a packet.
- Do not forward data packets originated from misbehaved nodes.
- Alert the source node when finding a misbehaved node in the source route of the packet that is forwarded.
- Do not forward or reply Route Requests originated from misbehaved nodes.
- Do not forward or reply Route Request when misbehaved node(s) are present in the source route.

All the nodes only publish firsthand information but not their opinions! On the other hand, if the node does not forward the packet, it will be considered misbehaved by its previous hop.

5.3.4 Naming Conventions

In CONFIDANT protocol, a node classifies the other nodes in the network either as misbehaving or honest based on its reputation rating about those nodes. The identified misbehaved node may actually be a normal one. To avoid confusion when we describe

the nature of the nodes, we define following name conventions about the nodes in the aspects of reality and opinion.

We define nodes in reality as follows:

- Evil node: a node that intentionally drops packet.
- Normal node: a node behaves as the routing protocol specifies.

We define nodes in the opinion of other nodes as follows:

- Misbehaved node: a node is considered misbehaving by another node. A misbehaved node may be actually a normal node but is misidentified.
- Good node: a node is considered normal by another node. A good node may be actually an evil node but is misidentified.

5.4 Assumptions

5.4.1 Assumptions about Mobile Ad Hoc Network

Since the CONFIDANT protocol relies on Passive Acknowledgement mechanism to detect nodes that fail to forward packets, we assume following items about the mobile ad hoc network.

- The wireless communication between any two nodes is bi-directional.
- The network interface of any node is in the promiscuous mode.

In this way, a node is able to find out whether the next node forwards the packet if both of them are still in the transmission range of one each.

5.4.2 Assumptions about Misbehaved Nodes

CONFIDANT can detect dropping attacks, modification attacks and fabrication attacks. Due to the time limitation, we only simulate dropping attacks in this project.

In the dropping attack, an evil node could drop all the packets it is supposed forward or destined to it. It could also be partial dropping, which is restricted to specific types, e.g. data packets or routing packets containing Route Error. When we simulate the evil nodes, we want to maximize the bad effect of evil nodes in the network and see whether CONFIDANT can mitigate the problems. Based on this guideline, we make following assumptions about evil nodes:

- The primary interest for evil nodes is to drop data packet since dropping data packet can have more serious consequence than dropping routing packets in general. The purpose of the routing protocol is to enable the data packets exchange between any communication ends in the network. Dropping data packets will also decrease the network throughput significantly.
- Evil nodes forward Route Requests. Dropping Route Requests is meaningless for evil nodes if its main target is to drop the data packets. In the mobile ad hoc network where dynamic routing protocol is used, the source node can often find alternative routes to the destination. Thus blocking a route by dropping Route Request may lead to opposite consequence: the data packets will be forwarded by safe routes.
- Evil nodes reply Route Request destined to them because their aim is to drop messages sent to other nodes but they don't want to lose any message destined to them.

5.5 Evaluation Parameters

5.5.1 Throughputs and Evil Drop Rate

Throughput is the most important metrics in our performance evaluation. Since the purpose of CONFIDANT is to improve the throughput for good nodes while bearing grudges to evil nodes, we evaluate the throughputs of good nodes and evil nodes separately. For simplicity, we call them good throughput and evil throughput. The goal of the CONFIDANT protocol is to increase the good throughput while decrease the evil throughput as much as possible. The formula used to calculate the good throughput is expressed in Equation 5-1.

$$GT = \frac{\sum_{i=1}^{n} \text{Received good packets}}{\sum_{i=1}^{n} \text{All Good Packets}}$$

Equation 5-1

where good packets are the packets originated by good nodes. The formula for evil throughput is expressed in Equation 5-2.

$$ET = \frac{\sum_{i=1}^{n} \text{Received evil packets}}{\sum_{i=1}^{n} \text{All evil Packets}}$$

Equation 5-2

Evil drop rate is used to evaluate how effective CONFIDANT is to mitigate packet drop attack. Evil drop rate means how much packets are dropped by evil nodes compared to the total number of packet dropped. Equation 5-3 is used to calculate the evil drop rate.

$$EDR = \frac{\sum_{i=1}^{n} \text{Packets Dropped by Evil Nodes}}{\sum_{i=1}^{n} \text{Total Packets Dropped}}$$

Equation 5-3

5.5.2 Overhead

CONFIDANT introduces network overhead by publishing firsthand information periodically. The quantity of the overhead depends on the publish timeout. CONFIDANT may also increase Route Request message since it uses stricter route selection strategy.

There are two factors in calculating overhead, the number of packets and the size of an individual packet. These two values cannot be simply multiplied since sending off a packet and transmitting a packet have different cost. Due to the time limitation we only consider the number of packets in our evaluation.

5.5.3 Misbehavior identification rate

In this performance analysis, we evaluate how much percentage of misbehaved nodes IS identified in the network. The rate reflects how effective CONFIDANT is in identifying misbehaved nodes. To evaluate the identification rate, the average number of identified misbehaved nodes should be calculated periodically.

5.6 Simulation of CONFIDANT Fortified DSR

To form the comparison, simulations are conducted for Standard DSR, CONFIDANT and other modified versions of CONFIDANT. The most important factors that impact the simulation results are the topology and traffic connections of the network. To dynamically simulate the network, following files are generated randomly and automatically to get different topology and traffic pattern.

- Node-movement scenario – specify how mobile nodes move in the network. E.g. the number of nodes participating in the network, the position of each node, the maximum speed of the movement.
- Traffic pattern – specify how a node sends packets to another. e.g., the packet sent rate, the application protocol and the size of packet.

NS2 provides automation tools to generate the different node-movement scenario files and the traffic pattern files.

5.6.1 NS2 Related Parameters

Table 5-1 lists the most important NS2 related parameters and the values that will be used in the simulations. However the value of transmission rate, packet size and maximum speed are different.

Parameter	Value
Application traffic	CBR
Radio Range	250 m
Packet Size	64 bytes
Transmission rate	2 packets/s
Pause time for node	100 s
Maximum Speed	1m/s
Simulation time	900 s
Number of nodes	50
Data connections	30
Area	1000 m x 1000 m
Available Bandwidth	2 Mbps

Table 5-1 NS-2 related parameters

5.7 Performance Evaluation

In this section, we will evaluate the performance of CONFIDANT protocol when different percentages of evil nodes are present in the network.

Apart from evaluating the performance of the classic CONFIDANT protocol proposed in [22], we will also analyze a variation of CONFIDANT, Path Re-ranking [23], to see how they will impact the performance of the network and whether they have advantages over the classic CONFIDANT. During the evaluation, we use following name conventions for different variations of CONFIDANT protocol.

Standard DSR – the DSR protocol specified in [40].

CONFIDANT – the classic CONFIDANT protocol specified in [22]

Path Re-ranking – the CONFIDANT protocol with path re-ranking as route selection strategy.

5.7.1 Good Throughput

Figure 5-6 presents the good throughputs of CONFIDANT and standard DSR. It is surprised to see that the good throughput of CONFIDANT does not show any improvement over that of standard DSR, whereas according to CONFIDANT [22], we can improve good throughput by two times. Our result does not fulfill the goal of CONFIDANT that the throughput of the network should be increased by discouraging misbehavior.

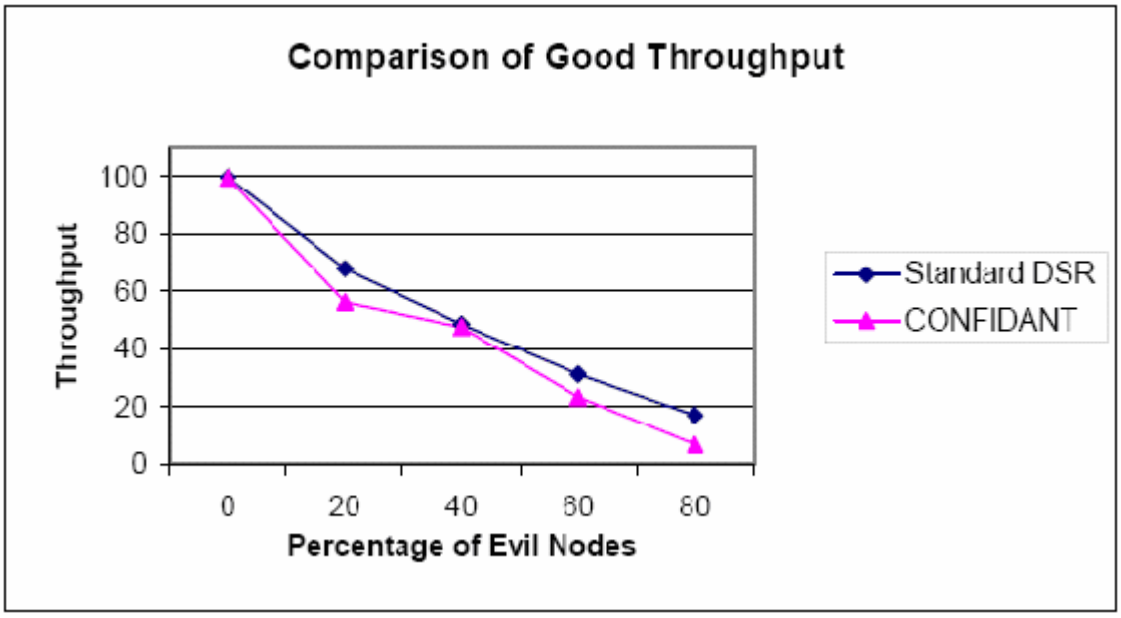

Figure 5-6 Comparison of good throughput

5.7.2 Evil Throughput

Figure 5-7 presents the evil throughputs of CONFIDANT and standard DSR. As seen, the evil throughput of CONFIDANT significantly decreases up to 50% compared to that of standard DSR. This result fulfills the goal of CONFIDANT that the evil throughput should be suppressed. In the figure, the throughput at zero percentage of evil nodes is empty because we cannot calculate the evil throughput when there are no evil nodes in the network.

5.7.3 Evil Drop Rate

The evil drop rate is the percentage of packets dropped by evil nodes in all the dropped packets. It reflects how much the evil drop contributes in overall packet drop compared to other drop reasons. Equation 5-3 is used to calculate the evil drop rate. Figure 5-8 shows the evil drop rates of CONFIDANT and standard DSR. As seen in the figure, the evil drop rate of CONFIDANT is significantly lower than that of standard DSR for all percentages of evil nodes. It is kept under 6%. This means that fewer packets are routed by evil nodes. This result fulfills the purpose of CONFIDANT that the misbehaved nodes should be avoided in forwarding packets.

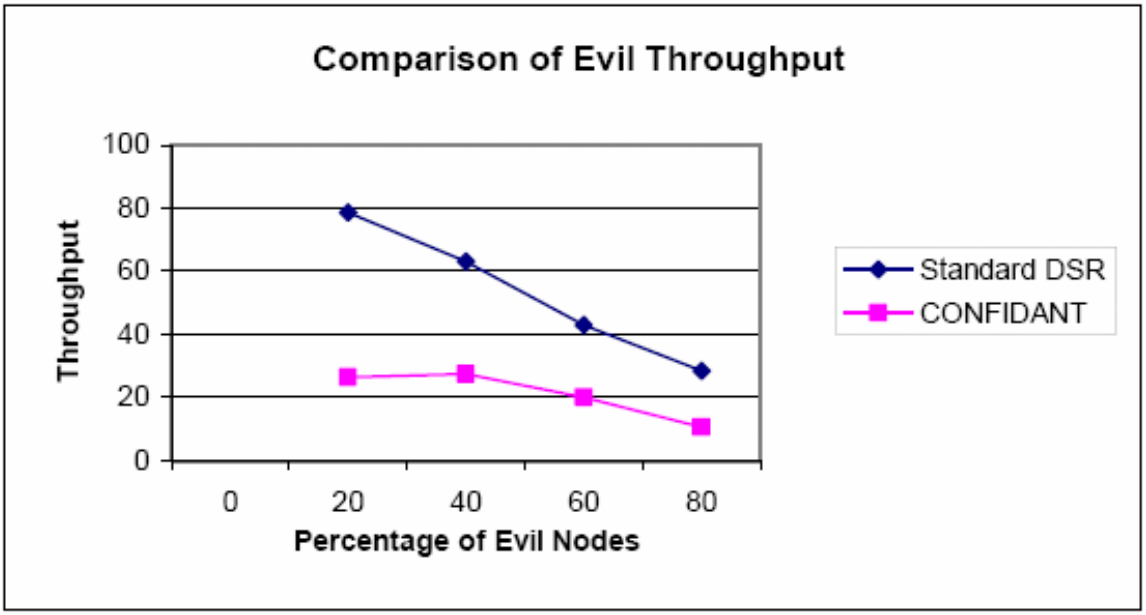

Figure 5-7 Comparison of evil throughput

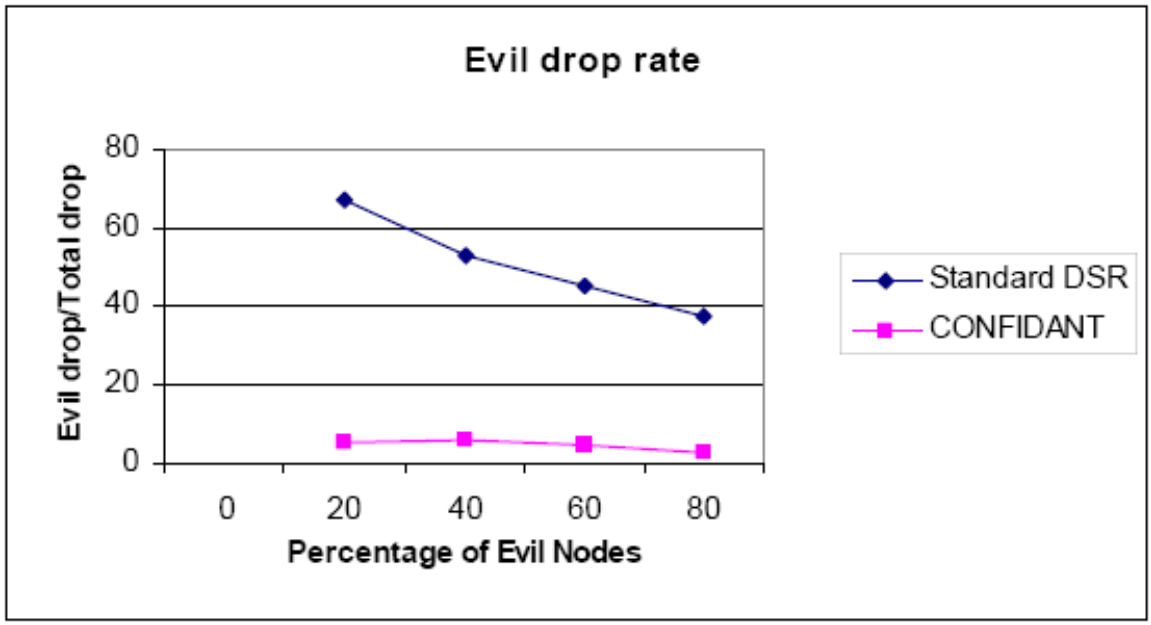

Figure 5-8 Comparison of Evil drop rate

5.7.4 Overhead

As stated earlier, CONFIDANT increases the network overhead by publishing firsthand information. It may also increase Route Request and Route Reply since it uses stricter route selection strategy and initiate Route Discovery to find safe routes.

Figure 5-9 shows the network overhead of CONFIDANT and standard DSR. The overhead is divided into two categories: published information and routing overhead. Only Route Request and Route Reply are calculated for routing overhead because we think CONFIDANT increases these two kinds of messages most. As seen, CONFIDANT significantly increases the routing overhead compared to standard DSR. The increase of routing overhead is also partly due to no enough good routes in the network. DSR keeps sending Route Request to discover new routes.

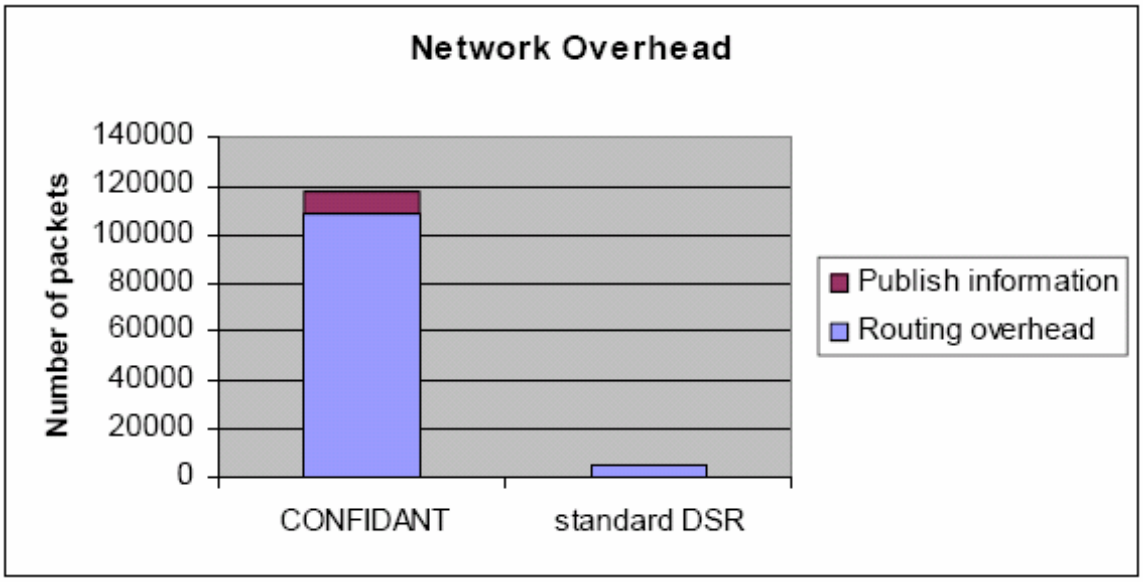

Figure 5-9 Network overhead evaluations

5.7.5 CONFIDANT with Path Re-ranking

An alternative route selection strategy is path re-ranking. With classic CONFIDANT, the Path manager only selects a route containing no misbehaved nodes, whereas with Path re-ranking the Path manager selects a route based on the reputation metrics of the route. In our implementation, we use a simple reputation metrics that is the average of the mean reputation values of all the nodes along the route. The advantage of Path re-ranking is that the packets will be sent out as long as there exists a route to the destination. Thus the send buffer drop rate would be decreased. However, Path re-ranking may have higher evil drop rate compared to the classic CONFIDANT since the selected route may contains misbehaved nodes.

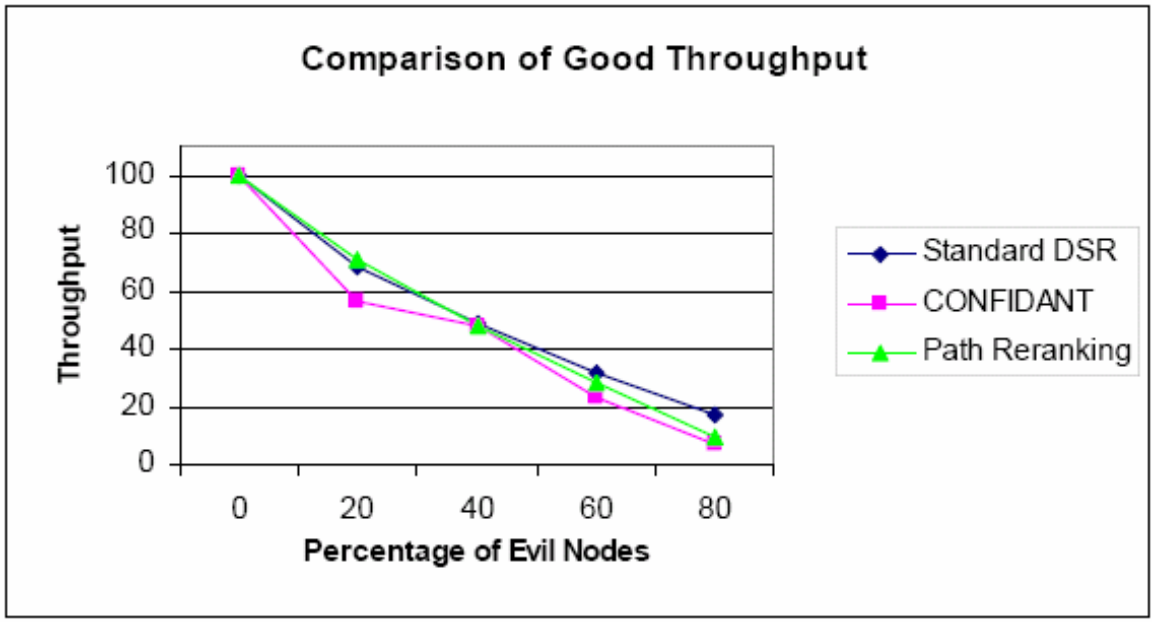

Figure 5-10 Good throughput of Path re-ranking

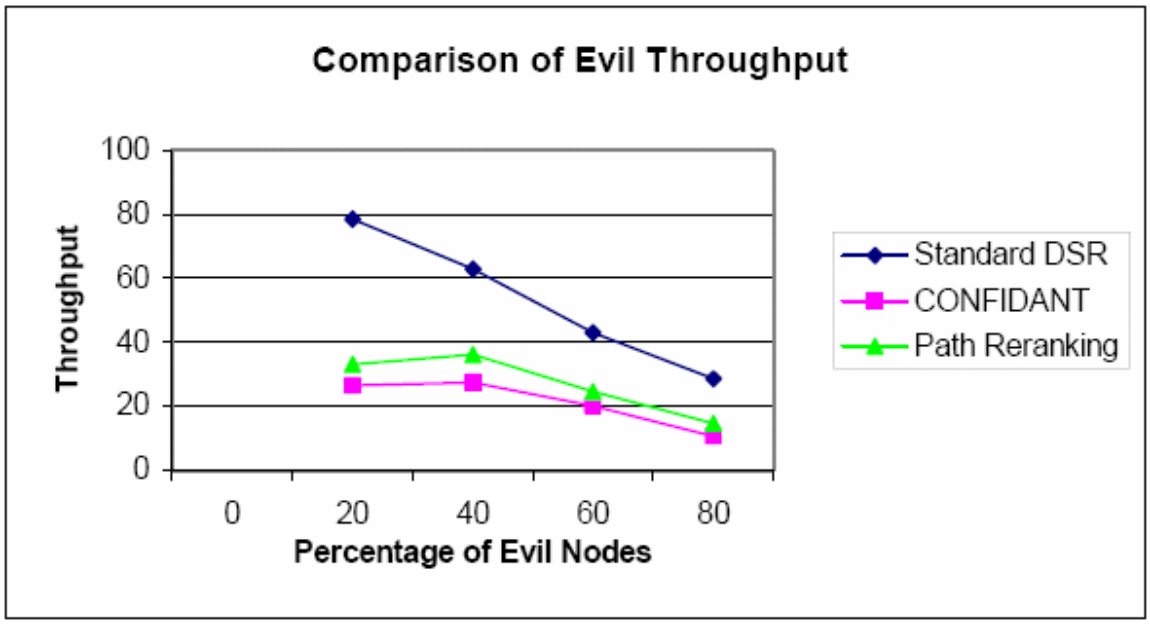

Figure 5-11 Evil throughput of Path re-ranking

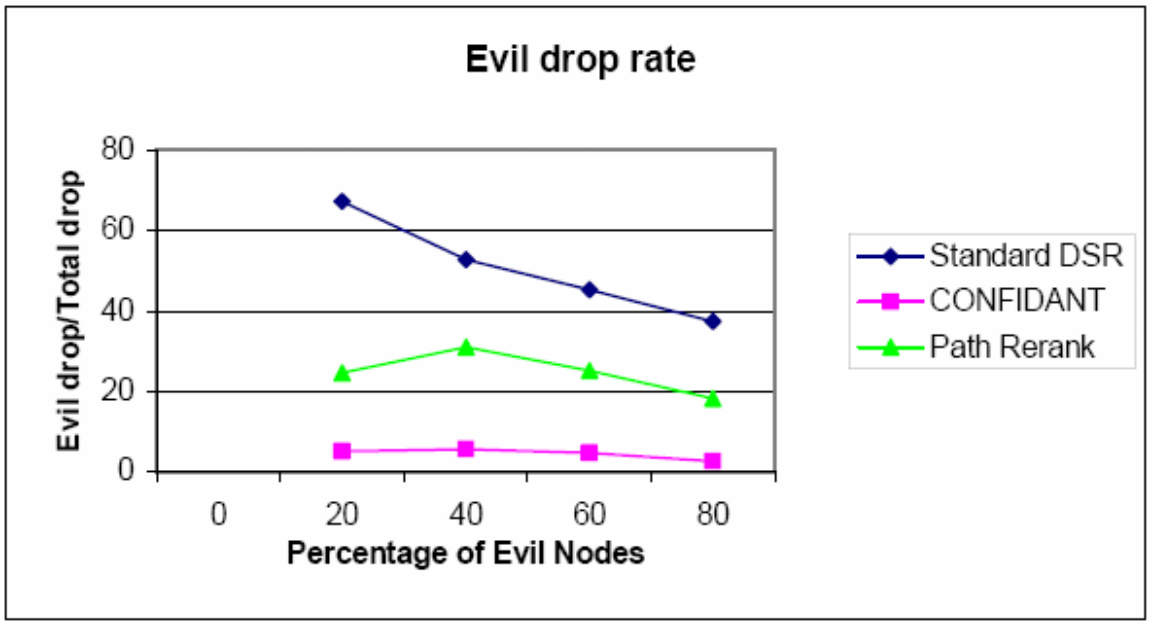

Figure 5-12 Evil drop rate of Path re-ranking

As seen in the figures, the good throughput, evil throughput and evil drop rate of Path re-ranking lie between those of standard DSR and classic CONFIDANT. Although Path re-ranking has slightly higher good throughput than classic CONFIDANT, it is not very effective at deterring evil nodes from misbehaving because the evil throughput and the evil drop rate remain high. Thus we get the conclusion that the classic CONFIDANT can better cope with misbehavior than Path re-ranking does.

5.8 Trust Based Routing – A Proposal

Here we present the proposal of a new trust based routing protocol. The goal is to prohibit the incorrect forwarding attack. For the moment we want to focus on the incorrect forwarding of data packets. We start with the assumptions we made, and then we state how trust should be determined. Depending on this, we state which nodes should be looked at as misbehaving and how this will affect routing.

5.8.1 Assumptions

To proof the concept of our design we will first implement our protocol in a network simulator. We assume certain things:

- Each node has a unique id

- Links are bi-directional
- Source IPs are not spoofed
- Nodes do not have a priori "trust" relationships
- All nodes give correct reputation
- Misbehaving nodes do not forward data packets, but act correctly for everything else (which is selfishness)

5.8.2 How to Determine Trust

5.8.2.1 Direct Trust

When we want to know, whether we can trust some node B, we can route some packets via B and see (by sniffing in promiscuous mode) if B forwards them correctly. The fraction of correctly forwarded packets in relation to the total amount of packets then gives us some idea on how trustworthy B is.

To be a bit more concrete: For every packet A sends to B, A puts a copy of it in a cache. If A sees B forwarding the packet correctly A promotes B for that. If A sees that B changed the packet or if A does not see the packet for some time, A punishes B. Then the packet is deleted from the cache.

A node can change its behavior over time, so we are only interested in its current behavior. Because of this we create an array of some size x (for B), where we can store the behavior of B for the last x packets we routed via B. A promotion gets a '1', a punishment gets a '0'. If we now sum up all the values and divide this sum by x (if we have only seen y < x packets we divide by y), we get the fraction of correctly forwarded packets by B. A can then use this value as his *direct trust* for B:

$$dt(A,B) = \frac{\#\text{ forwarded} = \text{number of packets (coming from A) correctly forwarded by B}}{\#\text{ sent} = \text{number of packets sent to B (by A)}}$$

Equation 5.4

All these arrays (one for each neighbor node) are managed in a sorted list. In this way all nodes establish some direct trust values for their (direct) neighbors.

What if we want to take other measures (than forwarding) into account? We can create an array per measure and node - i.e. a matrix - (at each node) and then get one trust value per measure in the way we did it before. These different values can then be combined to one by giving each measure a weight and adding up the weighted values (the result should be between 0.0 and 1.0). This value can then be used as our direct trust in the same way than before

5.8.2.2 Reputation - General Idea

What happens if there is a new node? We can either watch it and establish some direct trust values or we can ask our neighbors for references. Asking for references is done in everyday life and works quite well. So why should we not do it in MANETs? The question is how do we ask for references, what answers will we get, and how do we combine them?

So, lets create two new types of packets *reputation request* and *reputation reply* analogous to the existing route request and route reply packets of DSR. If A now wants to get references for B, he creates a reputation request, sets himself as source, sets B as target and broadcasts it to his neighbors (ttl = 1). Every node N receiving this request then looks if he has a direct trust value for B and if yes creates a reputation reply (from him to A) that is carrying this value. After some time, A can then combine the received values to a reputation value for B:

$$\text{Reputation}(A, B) = \frac{\sum_{i=1}^{n} dt(A, N_i) * dt(N_i, B)}{\sum_{i=1}^{n} n}$$

Equation 5.5

This reputation depends on when it is calculated and how many answers (route replies) have been received (and from whom). It can be completely different at some other node and/or time.

5.8.2.3 Whom to Ask?

The question is whom do we ask for reputation? Only our neighbors (as described above) or everybody or trusted nodes? If we ask only our neighbors we can determine trust only for nodes near us (worst case: maximal 2 hops away). If we ask everybody we can get trust values for nearly every node. But in this case we have two problems: First, reputation replies are routed over other nodes (we do not get them directly) and thus can be manipulated on their way. Using cryptography could prevent manipulation of packet, except dropping, but this is not easy to achieve (establishing a public key infrastructure in MANETs is a problem of it's own). Second, we get a big overhead on the protocol by flooding the net with reputation requests. Because of this it is better to send reputation requests only to neighbors. In this way we will get a local but "correct" view (we cannot exclude spoofing) with not so much overhead. And when moving around we can expand this local view in a more complete one.

5.8.2.4 How to combine reputation replies?

The question is how do we combine all the direct trust values from the reputation replies together to one reputation value. One possibility is to weight them with the direct trust values we have (as described above). Another possibility is to look at the answers and compare them. If we have a lot of nearly identical values we can treat them as correct and the few "differing" values we can treat as incorrect. We then can either just average the correct values to get our reputation. Or we can remember which nodes gave a correct answer and which ones did not and use this information to weight future answers.

The question is if a value of 0.1 really is incorrect if all the other nodes answer with 0.9? Maybe 0.9 just tells us that the forwarding is perfect and 0.1 tells us that delay is very

bad. Or maybe there is a node dropping all the packets from one node but correctly forwarding all the packets of all the other nodes. This will also result in very different direct trust values.

If we assume that a node is either good or bad, i.e. either behaves correctly for everything or nothing, answers of nodes can be weighted with the direct trust values we have for them. Therefore the first possibility (as described above) seems fairer.

5.8.2.5 Reputation Request and Reply Packets

Reputation request and reply are very similar to route request and reply. The following information is contained in these two packet types:

- Reputation request: sender, target, node reputation is requested for, TTL
- Reputation reply: sender, target, node reputation is given for, reputation value

5.8.2.6 Total Trust

Now we have some direct trust values and some reputation values. They can be combined in the following way:

Total Trust $tt(A, B) = a * dt(A, B) + (1-a) * r(A, B)$ $0 < a < 1$	Equation 5.6

5.8.3 Which Nodes are Misbehaving?

First we need to observe that it is not possible for us to differentiate the different types of misbehavior. We cannot say if a node is misbehaving because he is malicious, just selfish, has no battery left and so on.

The idea is to exclude misbehaving nodes from the net. Nobody wants to send his packets via a misbehaving node where one cannot be sure if it reaches its destination (unchanged), but when nobody sends packets via misbehaving nodes they are relieved from the burden of forwarding packets, and therefore rewarded for their misbehavior.

Many proposed protocols work like this. But we do not want to encourage misbehavior; we want to enforce cooperation. This is achieved when packets of misbehaving nodes are dropped by the other nodes (instead of forwarded). In this way, misbehaving nodes are completely excluded from the network. Because we want to give misbehaving nodes a chance of changing their behavior we will route some of our packets through them (so that we can monitor their behavior), but we will not forward packets for them. How do we determine if a node is misbehaving? A trust value can be small if a node dropped packets, but also if they never reached him or if we have not seen the correct forwarding. For the forwarding of packets it does not matter why a node has a small trust value. We therefore choose nodes with high trust values to maximize the probability of reaching the destination. In the other case we want to drop packets of misbehaving nodes only.

Chapter 6

Conclusion and Future Scope

6.1 Conclusions

The presented work has carried out the sub-tasks and completed the objective as listed in chapter 4. We sincerely hope that our work will contribute in providing further research directions in the area of trust-based security. The contribution of the thesis can be summed up as follows.

- An investigation has been conducted on the state of the art technologies dealing with security issues in Mobile Ad Hoc Network. As a result, the general security issues/requirement of Mobile Ad Hoc Network has been summarized. The investigation also covers the security solutions of payment system, reputation system, trust-based system and a new intrusion diction system. We have compared the advantage and disadvantages of these systems.
- Detailed analyses have been presented about DSR protocol, CONFIDANT protocol and network simulator. DSR has many additional features that will impact CONFIDANT and influence the network performance. We designed criteria and discussed each feature to decide whether it should be enabled or not. We described the details of CONFIDANT since it is the basis of implementing the protocol. We also discovered a problem of ns2 that has significant impact on the project and worked out a solution.
- A large amount of simulations have been conducted to evaluate the performance of CONFIDANT fortified DSR. The simulation results show that CONFIDANT significantly decreases the evil throughput and evil drop rate by up to more than 50%. It proves that CONFIDANT can effectively mitigate misbehavior in the network. However CONFIDANT does not improve the good throughput due to the large increase of send buffer drop, which is caused because there are not enough good routes in the network.

- We also evaluated the performance of a variation of CONFIDANT: Path re-ranking. The simulation result shows that CONFIDANT is better at coping with misbehavior compared to Path re-ranking.
- We propose a design of routing protocol based on trust that prevents incorrect forwarding attack. Here we focus only on the incorrect forwarding of data packets. The basic principles on the basis of which we can provide the implement the routing protocol, was also dealt with.

6.2 Future Scope

We have implemented and evaluated the performance of the CONFIDANT protocol based on DSR. We also encountered some problems that we feel worth to investigate but were not able to complete in limited time. In this section we illustrate some of the topics that could be explored in future:-

- To investigate the performance of using trust. Using trust to accept secondhand information is said to be able to speed up the misbehavior detection time. However, our simulation results show that using trust actually slows down the detection. More investigation about why the detection time of using trust is slow could be done.
- To investigate ns2 simulator regarding wireless links. The reason why we get very low good throughput for CONFIDANT is that there are not enough good routes available in the network. The ns2 could be investigated further about why there are few wireless links. If the wireless links could be increased then the good throughput of the CONFIDANT could be re-evaluated.
- To evaluate the network overhead of CONFIDANT considering the size of packet. In our thesis we only consider the number of the increased routing packet for overhead. However, the published information packets are usually much larger than other normal routing packets. Furthermore, the number of packets introduces different cost than the size of packets. Thus a better method could be designed to evaluate the network overhead.

- The trust based routing protocol that focuses on preventing incorrect forwarding attack can be implemented and evaluated with current trust based routing protocols, for a comparative study.

[1] David B. Johnson, David A. Maltz, Josh Broch. DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks. In *Ad Hoc Networking*, edited by Charles E. Perkins, chapter 5, pages 139-172. Addison-Wesley, 2001.

[2] Dellarocas C. The digitization of word-of-mouth: Promise and challenges of online feedback mechanisms. In *Proceedings of Management Science 2003*, Volume 49, No. 10, pages 1407–1424. INFORMS, October 2003.

[3] Levente Buttyan and Jean-Pierre Hubaux. Stimulating cooperation in self-organizing mobile ad hoc networks. In *Mobile Networks and Applications*, Volume 8, Issue 5, pages 579 – 592. Kluwer Academic Publishers, October 2003.

[4] H. Armbruster. Broadcast Communications and Its Realization with Broadband ISDN. *IEEE Communications Magazine*, Volume 25, No. 11, pages 8-19. November 1997.

[5] Ad hoc Networks Image: http://www.acorn.net.au/telecoms/adhocnetworks/adhocnetworks.cfm

[6] Jean-Pierre Hubaux, Levente Buttyan and Srdan Capkun. The Quest for Security in Mobile Ad hoc Networks. In *Proceedings of International Symposium on Mobile Ad Hoc Networking & Computing*, pages146-155. ACM Press, 2001

[7] Y. Hu, A. Perrig, and D. Johnson. Packet Leashes: A Defense Against Wormhole Attacks in Wireless Ad Hoc Networks. Technical report TR01-384, Department of Computer Science, Rice University, December 2001.

[8] Sencun Zhu, Sanjeev Setia, Sushil Jajodia. LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks. In *Proceedings of 10th ACM Conference on Computer and Communications Security (CCS '03)*, Washington D.C., pages 62-72. ACM Press, October, 2003.

[9] Yih-Chun Hu, D. Johnson, A. Perrig. Rushing Attacks and Defense in Wireless Ad Hoc Network Routing Protocols. In *Proceedings of the ACM Workshop on Wireless Security (WiSe 2003)*, pages 30-40. Sep. 2003.

[10] J.R. Douceur. The Sybil attack. In Proceedings of 1st International Workshop on Peer-to-Peer Systems (IPTPS _02). Springer, March 2002.

[11] H. Deng, W. Li, and Dharma P. Aggarwal. "Routing Security in Ad Hoc Networks. In *IEEE Communications Magazine, Special Topics on Security in Telecommunication Networks*, Vol. 40, No. 10, pages 70-75. October 2002.

[12] L.M. Feeney and M. Nilsson. Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In *Proceedings of IEEE Conference on Computer Communications (IEEE InfoCom)*, Anchorage AK, USA, Vol. 3, pages 1548–1557. April 2001.

[13] Y. -C. Hu, D.B. Johnson, and A. Perrig. SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad hoc Networks. In *Proceedings of 4th IEEE Workshop on Mobile Computing Systems and Applications*, Callicoon, NY, pages. 3-13, June 2002.

[14] Archive for RFC 2560: http://www.ietf.org/rfc/rfc2560.txt

[15] Archive for Thwate: http://www.thawte.com/repository/

[16] Levente Buttyan and Jean-Pierre Hubaux. Nuglets: a Virtual Currency to Stimulate Cooperation in Self-Organized Mobile Ad Hoc Networks. In Technical Report DSC/2001/001, EPFLDI –ICA, January 2001.

[17] Pietro Michiardi and Refik Molva. CORE: A Collaborative Reputation Mechanism to enforce node cooperation in Mobile Ad hoc Networks. In *Proceedings of Communication and Multimedia Security 2002 Conference*, September 26-27, 2002.

[18] Levente Buttyan and Jean-Pierre Hubaux, Stimulating Cooperation in Self-Organizing Mobile Ad Hoc Networks. In Mobile Networks and Applications, Vol. 8, No. 5, pages 579-592. Kluwer Publishers, October 2003.

[19] Zhong, S., Chen, J., Yang, Y. R. Sprite: A simple, cheat-proof, credit based system for mobile ad-hoc networks. In *Proceedings of IEEE Infocom'03*, San Francisco, CA (2003)

[20] Yongwei Wang, Venkata C. Giruka, Mukesh Singhal. A Fair Distributed Solution for Selfish Nodes Problem in Wireless Ad Hoc Networks. In *Proceedings of Third International Conference, ADHOC-NOW 2004 (LNCS 3158)*, Vancouver, British Colombia, pages 211-224. July 22-24, 2004.

[21] Giorgos Zacharia, Alexandros Moukas and Pattie Maes. Collaborative Reputation Mechanisms in Electronic Marketplaces. In *Proceedings of the 32nd Annual Hawaii International Conference on System Sciences*, Volume 8, Page 8026. IEEE Computer Society, 5-8th Jan, 1999.

[22] Sonja Buchegger. Coping with Misbehavior in Mobile Ad-hoc Networks. Ph.D. Thesis number 2935, EPFL April 2004.

[23] Sonja Buchegger and Jean-Yves Le Boudec. Nodes Bearing Grudges: Towards Routing Security, Fairness and Robusteness in Mobile Ad hoc Networks. In *Proceedings of the Tenth Ecurfomicro Workshop on Parallel, Distributed and networks based Processing*, pages 403-410. January 2003.

[24] Sonja Buchegger, Jean-Yves Le Boudec. The Effect of Rumor Spreading in Reputation Systems for Mobile Ad-hoc Networks. In *Proceedings* of WiOpt 2003: *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, Sophia-Antipolis, France. March 2003

[25] Jiangyi Hu, "Cooperation in Mobile Ad Hoc Networks", Computer Science Department Florida State University, January 11, 2005

[26] Sergio Marti, T.J. Giuli, Kevin Lai, and Mary Baker, "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks", In *Proceedings of the Sixth annual ACM/IEEE International Conference on Mobile Computing and Networking*, Boston, Massachusetts, United States, pages 255-265. ACM Press, 2000.

[27] Lidong Zhou, Zygmunt J. Haas. Securing Ad Hoc Networks. Published in *IEEE network Journal, special issue on network security*, pages 24-30, November/December, 1999.

[28] S. Capkun, L. Buttyan and J-P Hubaux. Self-Organized Public-Key Management for Mobile Ad Hoc Networks. In *proceedings of IEEE Transactions on Mobile Computing*, Volume 2, Issue 1, pages 52-64. IEEE Educational Activities Department, January 2003.

[29] P. Zimmermann. The Official PGP User's Guide. MIT Press, 1995.

[30] M. Reiter, S. Stybblebine, Authentication metric analysis and design, In *ACM Transactions on Information and System Security (TISSEC)*, pages –138-158. ACM Press, May 1999.

[31] Y. Desmedt. Threshold cryptography. *European Transactions on Telecommunications*, 5(4): 449–457. July–August 1994.

[32] Y. Desmedt and Y. Frankel. Threshold cryptosystems. *Proceedings on Advances in Cryptography*, Santa Barbara, California, US, pages 307-315. Springer-Verlag New York, 1989.

[33] Perrig A., Canetti R., Tygar J. D. and Song D. The TESLA Broadcast Authentication Protocol. *Cryptobytes*, Volume 5, No. 2 (RSA Laboratories, Summer/Fall 2002), pages 2-13. 2002.

[34] Josang, S. Hird, E. Faccer. Simulating the Effect of Reputation System on E-markets. In *Proceedings of the 1st International Conference on Trust Management*, Crete, May 2003.

[35] Z. Yan and P. Cofta. Methodology to Bridge Different Domains of Trust. Trust management first international conference, In *Proceedings of First International Conference, 2003*, Heraklion, Crete, Greece, Volume 2692/ 2003, pages 211-224. May 28-30, 2003

[36] Network Simulator was installed from this website: www.isi.edu/nsnam/ns

[37] NS-2 by Example: http://nile.wpi.edu/NS/

[38] Tutorial on Ad hoc routing Protocols: http://wiki.uni.lu/secan-lab/Ad Hoc+Protocols.html

[39] Ad hoc Network Image: http://www.acorn.net.au/telecoms/adhocnetworks/adhocnetworks.cfm

[40] David B. Johnson, David A. Maltz and Yih-Chun Hu. The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks, IETF Draft version 10. 2005

Acronyms — Annex - II

CBR	Constant Bit Rate
CONFIDANT	Cooperation Of Nodes: Fairness In Dynamic Ad-hoc Network
CORE	COllaborative REputation mechanism
DSR	Dynamic Source Routing
DoS	Denial of Service
LARS	Locally Aware Reputation System
MAC	Medium Access Control
MANET	Mobile Ad Hoc Network
NS	Network Simulator
PACK	Passive Acknowledgement
PGP	Pretty Good Privacy
DSS	Digital Signature Standard
CA	Certification Authority
PPM	Packet Purse Model
PTM	Packet Trade Model

List of Publications ANNEX –III

ACCEPTED

1. Ghansham Sangar, Ravi Kumar Bansal, Ripan Kumar and A.K. Verma, "A Novel Technique for Securing Bluetooth Communication", 4th International Conference on Computer Science and its Applications (ICCSA-2006), San Diego, California, June 27-29, 2006.
2. Ghansham Sangar and A.K. Verma, "Secured Routing for Mobile Ad Hoc Networks: A Comparative Study", in Proceedings of National Conference on Computing Sciences and Information Technology, B.B.S.B.E.C, Fatehgarh Sahib, 18-19 March 2006.

COMMUNICATED

1. A. K. Verma and Ghansham Sangar, "Ad Hoc Networks- A solution for Value Creation in Modern Economy for NextGen Enterprise", SEARCC 2006 Conference, Colombo, Sri Lanka, Sept 13-15, 2006.