TRUST ESTABLISHMENT IN MOBILE AD HOC NETWORKS

Richard Lawrence Gordon

Submitted in fulfilment of the academic requirements
for the degree of Master of Science in Engineering
in the School of Electrical, Electronic and Computer Engineering
at the University of KwaZulu-Natal, Durban, South Africa
05 February 2010

EXAMINER'S COPY

Supervisor: Prof. D. S Dawoud

As the candidate's supervisor I have approved this dissertation for submission.

Signed: ___________________________

Name: Prof. D. S. Dawoud

Date: 05 February 2010

DECLARATION

I .... declare that

(i) The research reported in this dissertation/thesis, except where otherwise indicated, is my original work.

(ii) This dissertation/thesis has not been submitted for any degree or examination at any other university.

(iii) This dissertation/thesis does not contain other persons' data, pictures, graphs or other Information, unless specifically acknowledged as being sourced from other persons.

(iv) This dissertation/thesis does not contain other persons' writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:
a) their words have been re-written but the general information attributed to them has been referenced;
b) where their exact words have been used, their writing has been placed inside quotation marks, and referenced.

(v) Where I have reproduced a publication of which I am an author, co-author or editor, I have indicated in detail which part of the publication was actually written by myself alone and have fully referenced such publications.

(vi) This dissertation/thesis does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the dissertation/thesis and in the References sections.

Signed: ___________________________

Date: 05 February 2010

Acknowledgements

I would like to thank my parents for always supporting me and giving me the opportunity to study toward my Masters Degree.

I would like to give a big thank to my supervisor, Prof. D. S. Dawoud for all his assistance in this research, for supporting me and believing in me. I am gratefully for his gracious assistance and help through my academic work. Many thanks also to Mr Tahmid Quazi who was a consistent help and encouragement through my research. Thank you to Libby Quinlan for all her gracious help. And thank you to my Lord and Saviour Jesus Christ.

I would also like to thank ARMSCOR for their generous support for this project. Thank you specifically to Mr Peter Handley and Mrs Franzette Vorster for believing in me and supporting me through my work. Without the funding received by ARMSCOR, I would have not been able to complete my research.

Abstract

The central focus of this dissertation is mobile ad hoc networks (MANETs) and their security. MANETs are autonomous networks of wireless nodes connected in an ad hoc manner, and have unique characteristics that make them difficult to secure. The principal aims of this investigation are to discuss the research and evaluation of existing mechanisms to secure MANETs and to design the implementation of a unique security mechanism. Key management is a major challenge in these networks due to the lack of fixed network infrastructure. In presenting a survey of the existing key management solutions for MANETs, the findings indicate that most security attacks target the network layer and more specifically the routing protocol. Consequently, the provision of secure routes is a vital element for trust establishment, and accordingly a survey is provided of the existing secure ad hoc routing protocols. The observation is made that most secure ad hoc routing protocols assume the existence of a key management system to certify, authenticate, and distribute keying information. Mobile ad hoc networks cannot assume the existence of a centralized authority member to perform key management tasks, and the problem of key management must be addressed.

A novel key management solution called Direct Indirect Trust Distribution (DITD) is proposed for an on-demand ad hoc routing protocol. The solution includes a trust evaluation mechanism and a key distribution scheme to distribute keying information in the form of certificates. The key distribution scheme performs localized certificate exchanges following the routing procedure. A security evaluation metric is proposed that aggregates trust along a path based on a security metric and the path distance. The proposed solution is implemented on a modified AODV routing protocol, and simulated on the ns2 Network Simulator. Simulations are conducted in order to compare the performance of the AODV and DITD protocols. The simulation results show that the DITD model provides key distribution and trust path selection with minimal effect on the routing agent. The findings of the investigation confirm that DITD can be used as a basis for the operation of existing security protocols requiring a secure key distribution mechanism.

Preface

The research work presented in this dissertation was performed by Richard Lawrence Gordon, under the supervision of Prof. D. S. Dawoud, in the School of Electrical, Electronic and Computer Engineering., University of KwaZulu-Natal. This work was supported by ARMSCOR, the Armament Corporation of South Africa.

Publications from this work include:

R. L. Gordon, D. Dawoud, Trust Establishment in Ad Hoc Networks, in proc World Multi-Conference on Systems, Cybernetics and Informatics (WMSCI 2009), Orlando, Florida, USA, July 10-13, 2009.

R. L Gordon, D. Dawoud, Direct and Indirect Trust Establishment in Ad Hoc Networks by Certificate Distribution and Verification, in proc. Wireless Communication Society, Vehicular Technology, Information Theory and Aerospace & Electronics Systems Technology (Wireless VITAE'09), Aalborg, Denmark, May 17-20, 2009.

R. L. Gordon, D. Dawoud, A Hybrid Trust Model for Mobile Ad Hoc Networks, in proc. Southern African Telecommunication Networks and Applications Conference (SATNAC'09), Swaziland, August 30 - September 2, 2009.

R. L. Gordon, D. Dawoud, Establishing Hybrid Trust for Military Type Mobile Ad Hoc Networks, in proc. Military Information and Communications Symposium of South Africa (MICSSA'09), Pretoria, South Africa, July 20-23, 2009.

R. L. Gordon, S. McDonald and D. Dawoud, Securing Wireless Ad Hoc Networks, in proc. Military Information and Communications Symposium of South Africa (MICSSA'07), Pretoria, South Africa, July 23-26, 2007

Table of Contents

Chapter 1

Introduction

1.1 Introduction

Mobile ad hoc networks are complex, wireless networks that have little or no existing network infrastructure. These networks can be established in a spontaneous manner, allowing organizations and network members to work together and communicate without a fixed communication structure. The mobility, spontaneity and ad hoc nature of these networks makes them optimal solutions for disaster area communication and tactical military networks. Due to recent wireless technology advances, mobile devices are equipped with sufficient resources to realize the implementation of these dynamic communication networks. However, before ad hoc networks can experience large scale development within both the military and commercial world, they must be secured against malicious attackers.

Mobile ad hoc networks have distinct characteristics that make them very difficult to secure. Such characteristics include: the lack of network infrastructure; no pre-existing relationships; unreliable multi-hop communication channels; resource limitation; and node mobility. Users cannot rely on an outside central authority, such as a trusted third party (TTP) or certificate authority (CA), to perform security and network tasks. The responsibility of networking and security is distributed among the network participants. Users have no prior relationship with each other and do not share a common encryption key. Therefore, only after the network has been formed, do the users establish trust and networking links. The establishment of networking links is identified as being vulnerable to security attacks. Trust establishment should allow protection for the network layer and ensure that honest links are created.

The sporadic connectivity of the wireless links, inherent to mobile ad hoc networks, results in frequent link breakages. These characteristics introduce unique challenges to trust establishment. Both the routing and trust establishment protocols must be designed to handle the unreliable wireless communication channels, the dynamic topology changes, and the distributive nature, all of which are common to mobile ad hoc networks. The security solutions used for conventional wired networks cannot simply be applied to mobile ad hoc networks. More complex network management must be implemented to achieve trust establishment in mobile ad hoc networks.

1.1 Outline of dissertation

This dissertation focuses largely upon establishing trust in mobile ad hoc networks, and concentrates more specifically on secure key management on the network layer. Our research focuses upon providing a solution for the security issues found in mobile ad hoc networks. The dissertation is organised as follows.

Chapter 2 provides a theoretical background to mobile ad hoc networks. The terms ‘mobile ad hoc network’ and ‘wireless ad hoc networks’ will be used interchangeably. These networks and their characteristics are defined in terms of trust establishment. As the focus of this research is on the network layer, attacks specific to this layer are identified and explained. Chapter 2 offers an introduction to the security issues present in mobile ad hoc networks.

Chapter 2 identifies that key management is a difficult task to achieve in mobile ad hoc networks. Chapter 3 presents a survey of the existing key management solutions for mobile ad hoc networks. Discussions are based on functionality; availability; security services; scalability; efficiency; and computational cost. A comparative summary is presented that identifies the difference in the requirements and the application of each solution.

Chapter 2 recognizes that most security attacks target the network layer and, more specifically, the routing protocol. Chapter 4 offers a survey of the existing secure routing protocols for mobile ad hoc networks. This chapter makes a pertinent observation that most secure routing protocols assume the existence of some kind of key management authority. Mobile ad hoc networks have little fixed network architecture, and it is unlikely that there is a centralised authority member.

Chapter 3 and 4 identifies the principal problem addressed by this dissertation. Secure routing mechanisms exist to address the unique characteristics of mobile ad hoc networks. However, these solutions assume that key management is addressed prior to network establishment. A novel, on-demand solution to the key management problem for mobile ad hoc networks is then described. Chapter 5 details the functionality and operation of the proposed model: “Direct Indirect Trust Distribution” (DITD). The DITD model focuses on the task of distributing keying information. The DITD model also includes a verification optimization protocol and trust evaluation metric that maximises the security of distribution.

In Chapter 6, the implementation and simulation of the DITD model is examined. There are various packages used to compare existing and proposed routing protocols. One such package is the ns2 Network Simulator, which is commonly used in the relating literature. A comparative ns2 simulation study between the DITD and the AODV protocols is presented. The DITD model is based on the Ad Hoc On-demand Distance Vector (AODV) routing protocol. Simulations show the performance overhead of including key management functionality and the performance of DITD in the presence of malicious attacking nodes.

Chapter 7 summarises the contribution of the dissertation to the field of trust establishment in mobile ad hoc networks, and provides recommendations alluding to future directions in related research.

Chapter 2

Mobile Ad Hoc Networks

2.1 Ad Hoc Networks

2.1.1 Definition

An ad hoc network is a network with no fixed infrastructure. It allows for users to enter and exit as they please, while seamlessly maintaining communication between other nodes. Mobile Ad Hoc Networks (MANETs) are advanced wireless communication networks that operate in an ad hoc manner.

The term ad hoc is defined as:

"For this purpose. Meaning "to this" in Latin, it refers to dealing with special situations as they occur rather than functions that are repeated on a regular basis." [1]. This definition suggests that it is a network that is formed in a spontaneous manner in order to solve an immediate communication need between mobile nodes. Mobile ad hoc networks differ from existing wired networks because [2, 3] they do not rely on a fixed network infrastructure, such as base stations or mobile switching centres. Instead, network functionality is adopted by the nodes themselves. When using a multi-hopping routing protocol, mobile nodes within each other's radio range communicate directly via wireless links. However the nodes that are far apart depend on the other nodes to relay the message in a multi-hop fashion. Figure 2-1 [4] demonstrates these autonomous, multi-hop characteristics. Connection between nodes is made by means of other nodes within the network. The circle represents node A's wireless range. Thus when node D exits A's range the topology changes to maintain the connection. Note that all network functions are performed by the nodes, and no host or outside authority exists.

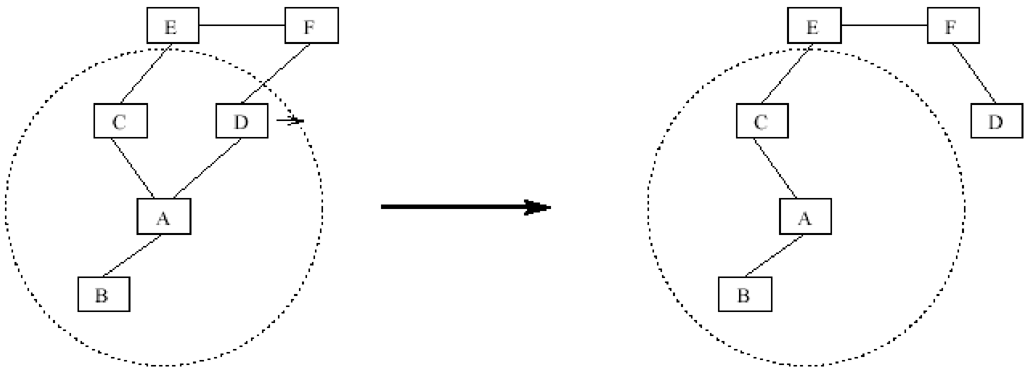


Figure 2-1: Ad Hoc Network Topology

2.2 Application

Mobile ad hoc networks have become widely desired in military and commercial applications, due to the ever-increasing development of mobile technology. The network's lack of infrastructure and independent nature allows for a robust network to be created within an unlikely networking environment.

2.2.1 Military Application

The first ad hoc networks were primarily deployed in the military domain in the early 1970's by the US Department of Defence, under the projects of DARPA and Packet Radio Network (PRnet) [3]. Ad hoc networks remain an important part of current and future military communication. They feature prominently in the following areas of military application: sensor networks, tactical networks and positional systems.

Their application within the military field is based on the network's high mobility, survivability, and self-organized nature. This allows mobile military units to communicate effortlessly, irrespective of the distance between each detachment. In a hostile environment, such as the battle field, an ad hoc network's distributive architecture eliminates the problem of a vulnerable network host or the loss of the network host. The modern battle field is characterized by highly mobile forces and the effect of a network that fails to maintain communication and high mobility is disastrous. An example of this can be seen in the experience of the Iraqi forces during the 1991 Gulf War. For this reason, soldiers would prefer mobile ad hoc networks, as opposed to existing local networks. Both invading and defending soldiers would avoid using the local operator, therefore ensuring communication stealth required for battle. Another illustration of the downfall of using an existing local network can be seen in Chechnya, where a general was killed by a missile which tracked the uplink signal of his portable phone. It is clear from these examples that mobile ad hoc networks provide stealth, mobility, and security in the battle field.

The military context represents the most obvious field of application and consequently research for mobile ad hoc networks. More recently in July 2008, DARPA invested $8.5 million in the Intrinsically Assurable Mobile Ad Hoc Network program (IAMANET) [5]. This project aims to improve the integrity, availability, reliability, confidentiality, safety, non-repudiation of MANET communication and data in the future.

2.2.2 Commercial Application

Early application and developments had a military focus. However, non-military applications have grown rapidly due to the availability and advances in mobile ad hoc research. The introduction of new standards such as IEEE 802.16e, IEEE 802.11g and IEEE 802.15.4 have significantly helped the deployment of wireless ad hoc network technology in the commercial domain [3]. In this sector, the aforementioned networks are desirable due to their dynamic and self organized nature that allows rapid network deployment. This is particularly useful in situations where infrastructure is damaged or does not exist, and where existing conventional networks are unaffordable or lack sufficient network coverage and need to be side-stepped. Some examples of these applications include personal area networks, sensor networks, and emergency networks.

Personal area networks are created when a small number of nodes meet spontaneously to form a network for the purpose of teleconferencing, file sharing, or peer-to-peer communication. An example of this can be seen when attendees in a conference room share data-using laptops or handheld devices.

Sensor networks are used to monitor data across an area. An example of these networks is small sensor devices that are located in animals and other strategic locations that collectively monitor and analyze the environmental conditions. Sensor networks have also been developed, by the PermaSense Project, to monitor the permafrost found in the Swiss Alps [6].

The application of this network to an emergency context often occurs in a hostile environment, similar to the military context. For example, natural or man-made disasters may result in the existing network infrastructure being unavailable or unreliable. Ad hoc emergency services could allow communication and sharing of video updates of specific locations, among relief workers and the command centre. A pertinent illustration of this scenario can be seen in the events surrounding the New York World Trade Centre disaster of September 11, 2001. The majority of the phone base stations were knocked out less than twenty minutes after the attack.

The remaining base stations were unable to operate because they could not work in ad hoc mode. The Wireless Emergency Rescue Team recommended afterwards that telecom operators provide ad hoc mode for their infrastructure in the event of emergency situations to enable co-operation between police, firemen and hospital networks [7]. Mobile ad hoc networks can allow for rapid network deployment in an emergency situation. Emergency networks can be set up in remote or hostile areas where there is no existing communication infrastructure, thereby assisting relief work and rescue missions.

A vehicular ad hoc network provides communication between vehicles, roadside equipment and vehicles travelling in close proximity. Data is exchanged between nearby vehicles to provide traffic information and early warnings for accidents and road works. The purpose of vehicular ad hoc networks is to provide a communication network of safety and information for users [8].

The benefits of ad hoc networks have realized new non-military communication opportunities for the public. Companies are starting to recognize the potential for commercial ad hoc network applications; and as a result, laptops and handheld devices are being equipped with wireless functionalities. Businesses are offering products using ad hoc networking technology in areas of law enforcement, intelligent transport systems, and community networking. These dynamic networks have still not reached their full potential, and it is clear that ad hoc technology has an imminent role to play in the ongoing development of commercial technology.

2.3 Ad Hoc Network Challenges

An ad hoc network is a dynamic type of network that is both similar and very different to its parent fixed communication network. The properties of an ad hoc network will define its shortcomings and highlight security challenges.

2.3.1 Dynamic Network Architecture

Ad hoc networks have no fixed or existing network infrastructure. The network architecture is continuously changing as the network evolves. There is no pre-existing or fixed architecture that handles all network tasks such as routing security and network management. Instead, the network infrastructure is spontaneously set up in a distributive manner. Each participating node shares the network's responsibilities. Distribution of network functionality avoids single point attacks and allows for the network to survive under harsh network circumstances.

A fixed entity structure, such as a base station or central administration, is crucial for security mechanisms. A trusted third party member [9] that is expected in traditional networks is similar to a fixed entity as both define security services, as well as manage and distribute secret keying information thereby allowing the secure communication of data through encryption and decryption techniques. Conversely, the absence of such a control entity introduces new opportunities for security attacks on the network.

2.3.2 Self Organized Nature

Wireless ad hoc nodes cannot rely on an off-line trusted third party member. The security functions of the trusted third party member are distributed among the participating nodes. Each node takes responsibility for establishing and maintaining its own security and consequently is the centre of its own world and authority [2, 10]. A wireless ad hoc network is therefore referred to as a 'self organized network' [2].

2.3.3 No Prior relationships

In ad hoc networks, nodes can have no prior relationships with other nodes within the network. Prior acquaintance between nodes can be considered as pre-trust relationships between nodes. However, the ad hoc nature of these networks does not allow for these assumptions, as it cannot be assumed that secrets exist between the respective pair of nodes [11]. If nodes can join and leave the network at random, and without prior trust relationships with nodes, access control becomes a difficult task for the security mechanism.

2.3.4 Multi-hop communication channel

Wired networks include fixed nodes and fixed wired communication lines. Wireless ad hoc networks have mobile wireless nodes, often in the form of hand-held devices; and their communication medium is wireless. This allows for greater network availability and easy network deployment. Each node's transmission range is limited and network communication is realized through multi-hop paths. Co-operation and trust along these paths is a crucial aspect of the security mechanism and ensures successful communication. The shared wireless communication medium means that any user can participate in the network. This creates access control problems for security mechanisms as adversaries are able eavesdrop on communication or launch active attacks to alter message data.

2.3.5 Mobility

Nodes are expected to be mobile within an ad hoc network, creating a dynamic and unpredictable network environment. In certain situations, the nodes' mobility is not totally unsystematic and assumptions can be made in the form of mobility patterns [10]. An example of these patterns is evident in a Vehicular ad hoc network where vehicles move along fixed paths or roads at speeds that have a high probability of being within the local speed limit. However, nodes demonstrate random mobility within these predictions [8].

Connectivity between nodes is sporadic. This is due to the shared, error-prone wireless medium and frequent route failures caused by the unpredictable mobility of nodes [12]. Increased mobility can result in the multi-hop communication paths being broken and network services becoming unavailable. Security mechanisms must account for the weak connectivity and unavailability. Furthermore, due to mobility and sporadic connectivity, these mechanisms must also aim to be scalable with the changing network density.

2.3.6 Resource Limitations

Wireless nodes allow for the freedom of mobility and easy network establishment and deployment. Wireless nodes are often smaller hand-held devices that do not experience the same resource privileges of traditional wired nodes [3]. Mobile nodes are ideally low cost and small in size thereby maximizing node availability and mobility. In attempting to achieve these objectives, wireless nodes have limited resource, specifically in the following areas:

- Battery life
- Communication range
- Bandwidth
- Computational capacity
- Memory resources.

If mobility is to be attained, nodes must be battery-powered. Battery-powered nodes suffer from the consequences of power failures that break connectivity. They also run a high possibility of failing to be on-line for the entire duration of the network. This could hinder network service availability. Cost and power restrictions limit the design features of wireless nodes. Power and transmission range are directly related, resulting in wireless devices having limited transmission ranges and bandwidths. Low powered, low cost CPU's are preferred, as this reduces the computational capacity and memory resources available for routing and security operations. As discussed above, network and security tasks are not performed by a central authority, but rather distributed among all the nodes. This creates a heavy burden upon the nodes to perform their own tasks as well as the network services. If the security mechanisms do not distribute the load fairly, adversaries can act in a selfish manner, forcing other nodes to perform extra tasks. In some instances, malicious nodes will flood a single node with service requests in the aim of depleting its limited resources. A well designed system of security algorithms optimizes computational processing and operation to meet the limited resource requirements of these dynamic networks.

2.3.7 Physical Vulnerability

Another challenge in ad hoc networks is the physical vulnerability of nodes. In a mobile ad hoc network, nodes are mobile and often small devices. This contributes to a higher probability of being captured or compromised when compared to traditional wired networks with stationary entities [13]. This means that wireless ad hoc networks are more prone to insider attacks and security mechanisms, and must be designed with this in mind. An inside attacker could analyze the node to gain secret keying information or use the node to compromise other nodes. The same threats exist in wired formal networks. Although they may rely on a secure host to detect and recover compromised nodes, sensitive security information may also be stored on that host, minimizing the consequences upon the network if a single node is captured. In an attempt to enhance security within hybrid ad hoc networks [14], a fixed architecture is combined with a volatile distributive architecture.

2.4 Security Objectives and Services

Securing mobile ad hoc networks requires certain services to be met. A security service is a made available by a protocol that ensures sufficient security for the system or the data transferred. The security objectives for mobile ad hoc networks are similar to those of fixed wired networks. The security objects are described in six categories, adapted from discussions in Stallings [15].

2.4.1 Authentication

An authentication service validates a user, node, or some digital entity to ensure that it is what it claims to be. Two specific authentications are defined, namely peer entity authentication and data origin authentication. Peer entity authentication provides information that offers confidence in the identity of the connected node. Data origin authentication provides assurance that the identity of the source of the received data is as stated and verifies all claims that are made by the node.

2.4.2 Access Control

Access control service restricts who can join the network. Pure mobile ad hoc networks are open systems that allow for users to enter or exit unhindered, as they have no access control. Mobile ad hoc networks with access control services are known as closed networks where a closed set of nodes communicate in an ad hoc manner, as is common in a hostile military environment.

2.4.3 Data Confidentiality

Services providing data confidentiality protect data from unauthorized disclosure. This prevents the transmitted data from being captured by malicious nodes that may be monitoring a communication channel to obtain sensitive information transmitted. Attackers may also analyze the traffic flow to identify communication patterns, and determine the location and identities of the source or its destination. Data confidentiality serves to protect traffic and message data from eavesdropping attackers.

2.4.4 Data Integrity

The security services must assure that the data received is exactly as it was sent by the authorized source. This prevents the actions of malicious nodes which attempt to modify, delete, or replay messages.

2.4.5 Non-repudiation

This service provides protection against nodes, as it denies their involvement in communication, whether or not the nodes participate in all or part of the communication. Non-repudiation allows the sender to prove that the receiver has received the message, and it allows the receiver to prove that the sender has sent the message.

2.4.6 Availability Services

A number of attacks can result in a loss of availability. In mobile ad hoc networks where protocol tasks are distributed, selfish nodes and denial of service attacks are more harmful. The availability service ensures availability and fair distribution between nodes. It is dependent on protocol management, control of resources, among other security services discussed in this chapter.

2.5 Attacks

Threats or attacks upon the network come from entities, known as adversaries. Mobile ad hoc networks inherit all the threats of wired and wireless networks. With these networks' unique characteristics, new security threats are also introduced [4]. Before the development of security protocols, it is essential to study the attacks associated with these unique networks.

2.5.1 Attack Characteristics

Attacks will be launched against either the vulnerable characteristics of a mobile ad hoc network or against its security mechanisms. Attacks against the security mechanism in all types of networks, including mobile ad hoc networks, include authentication and secret key sabotage.. Mobile ad hoc networks have distinctive characteristics, as identified in Section 2.3. Attackers are expected to target these points of vulnerability, for example the multi-hop nature of communication routes. The attacks are classified by their different characteristics, and the expected types of attacks are identified below.

2.5.1.1 Passive and Active Attacks

Security attacks can be classified by the terms ‘active’ and ‘passive’ [16]. Passive attacks attempt to steal information from the network without altering the system resources. Examples of passive attacks include, eavesdropping attacks and traffic analysis attacks. It is difficult to detect passive attacks as they leave no traceable affect upon the system resources or network functionality. Although the results or the need for securing against these attacks may not be monitored or visibly present, it is still a priority to protect networks from these seemingly harmless attacks, particularly in a military context. “If security is too successful, or perfect then the security expenditures are seen as wasteful because success is too invisible.” [17] However, Schneier assures one that, despite the lack of visible results, the need to secure information still exists.

Active attacks attempt to modify system resources or network functionality. Examples of these attacks are message modification, message replay, impersonation and denial of service attacks.

2.5.1.2 Insider and Outsider Attacks

Malicious nodes are unauthorized participants in the network that launch outsider attacks. Impersonation, packet insertion, and denial of service are some examples of outsider attacks. In contrast to outsider attackers, inside attackers are more difficult to defend against. Inside attacks are launched from nodes that are authorized participants in the network. Insider attacks are common in pure mobile ad hoc network where any user can freely join or exit . Security mechanisms become vulnerable when participants are malicious and the confidentiality of keying information can be compromised. Hence, an advantage of the non-repudiation and authentication techniques is that malicious insider nodes can be identified and excluded.

2.5.1.3 Layer Attacks

There are threats at each layer of the mobile ad hoc network communication protocol. The physical layer is vulnerable to passive and active attacks. The attacks found at the physical layer are as follows eavesdropping, denial of service, and physical hardware alterations. Encrypting the communication links and using tamper-resistant hardware helps to protect the physical layer. However, at the data link layer, adversaries can flood the communication links with unnecessary data to deplete network resources. Security mechanisms that provide authentication and non-repudiation can prevent this, as they allow invalid packets transfers to be identified. At the application layer, messages are exchanged in an end-to-end manner using wireless multi-hop routes established by the network layer. The wireless multi-hop routes are invisible to the application layer. Conventional security techniques used for wired networks can be used to prevent expected attacks upon the application layer. The application layer is dependent upon the network layer to provide secure routes between the two communicating parties.

The network layer provides a critical service to the mobile ad hoc network and the routing protocol. In the context of trust and security, the provision of secure routes is one of the most vital elements for trust establishment.

2.5.2 Attack Types

The different types of attacks are identified and described below. While there is a focus on the networking layer, attacks such as impersonation and denial of services can occur on any layer.

2.5.2.1 Wormhole attack

In a wormhole attack a compromised node receives packets at one place in the network. The attacker tunnels the packets to another destination (i.e. an external attacker) in the network where the packets are re-sent back into the network [18]. The tunnel created by the adversary is known as a wormhole. A wormhole allows adversaries to disturb the routing protocol, by intercepting routing messages and creating denial of service attacks. If the routing mechanism is not protected against such an attack, mobile ad hoc routing protocols may fail to find valid routes.

2.5.2.2 Black hole attack

During route discovery, a malicious node may falsely advertise itself as possessing the optimal route to the requested destination. Consequently, the adversary attracts all routing messages. The attacker then creates a black hole attack by dropping all routing packets, and disrupting the routing protocol and discovery phase.

2.5.2.3 Byzantine attack

During this type of attack, a malicious node or a group of malicious nodes will launch attacks on the routing protocol. The aim is to direct routing packets to follow non-optimal routes, routing loops, and selective dropping of packets [19]. Byzantine behaviour is difficult to detect. A network could be operating with Byzantine failures and be unaware of the attack on its routing mechanism.

2.5.2.4 Eavesdropping

An eavesdropping attack involves message or routing packet monitoring. It is a passive attack on the mobile ad hoc network. Eavesdropping attacks are performed by adversaries and can reveal confidential information about the network regarding its topology, geographical locations, or optimal routes in the network. Attackers can use this information to launch other attacks at identified points of vulnerability. All networks are prone to passive eavesdropping attacks. It is the nature of wireless, mobile ad hoc networks that make them more vulnerable. In wireless networks adversaries do not need a physical wired communication link to monitor the routing packets. The wireless communication medium allows for any users within range to analyze the traffic. Attackers can also exploit the multi-hop nature of routes in mobile ad hoc networks. An adversary can position itself along a route path and forward the routing messages along the multi-hop path. This allows adversaries to also analyze every packet that is forwarded along the path. Eavesdropping is a common problem in networks and encryption techniques can protect routing protocols from these attacks.

2.5.2.5 Packet Replay

Like eavesdropping, packet replay is a passive attack where data is captured by monitoring adversaries. Old routing messages are then re-transmitted to other nodes disturbing the routing process. Adversaries can therefore cause other node's routing tables to be updated with outdated information. Malicious attackers can also record authorized routing messages and replay them to gain unauthorized access to protected nodes.

2.5.2.6 Resource consumption attack

Mobile ad hoc nodes are restricted by their limited resources. Attackers exploit this by launching attacks that consume a node's resources thereby hindering them from network participation. Resources targeted by attackers are bandwidth, computational power and battery life.

Sleep deprivation attacks are resource attacks that are specifically aimed against mobile ad hoc node's battery power. Nodes attempt to save power by going into a sleep mode where a periodic scanning occurs and less battery power is used. Sleep deprivation attacks prevent nodes from going into sleep mode thereby draining the battery life and disabling the node itself. Attackers will flood a target node with redundant routing requests or routing packets to be processed, thereby keeping the node and its resources unnecessarily busy.

Packet replication is another type of resource attack where adversaries duplicate out-of-date packets and re-transmit them. This not only consumes battery life, bandwidth and computational power, but also disrupts the routing protocol.

Sleep deprivation attacks, flooding attacks and packet replication result in the depletion of precious resources. If this is not protected against, it will result in nodes and services becoming unavailable in the network.

2.5.2.7 Routing Table Poisoning

Malicious nodes will target the routing table in an attempt to sabotage the establishment of routes. One such attack is the routing table poisoning attack where malicious nodes send counterfeit routing updates or modify existing routing updates. This results in conflicting link information, unnecessary traffic congestion or denial of service.

2.5.2.8 Rushing attack

Mobile ad hoc networks that use on-demand routing protocols are vulnerable to rushing attacks [20]. On-demand routing protocols, such as AODV [21] and DSDV [22], use route request messages to discover the optimal route to a destination node. The network is flooded with route request messages. These messages are forwarded until the optimal route is found between the source and destination nodes. An adversary that receives a route request performs a rush attack by hurriedly flooding the network with that route request before other nodes, receiving the same route request, can respond. When other nodes receive the legitimate routing request, it is assumed to be a duplicate of the request that is distributed by the adversary, and the legitimate routing request is dropped. Consequently, the adversary will become part of the route that is discovered and this will result in a generally insecure route.

2.5.2.9 Selfish attack

Misbehaving nodes will act in a greedy or selfish manner, resisting cooperating or participating in the network operations. This is a denial of service and the attack causes the nodes to refuse to make their resources available. Selfish nodes do not cooperate in network operations that do not benefit them. They rather conserve their limited resources, such as battery life. Nodes may refuse to forward route request packets or turn off their devices when they are not transmitting data. The distributive architecture and multi-hop nature of mobile ad hoc networks means the network relies upon node cooperation [23]. A security protocol should ensure fair distribution of network operation in order to provide reliable network services, and prevent node's resources becoming depleted because of selfish node attacks.

2.5.2.10 Impersonation

Impersonation attacks are also known as masquerading or spoofing attacks. The attacks occur when adversaries take the identity of an authorized node and breach the security of the network. Masquerading nodes are able to receive routing packets destined for other nodes. Mobile ad hoc networks can help protect against impersonation attacks by authenticating their routing messages.

Pure mobile ad hoc networks are more vulnerable as they have no access control. If there is no strong binding between the physical entity and the network identity, malicious nodes can adopt different identities. A severe attack that is prone to mobile ad hoc networks is the Sybil attack [24, 25]. A single adversary node launches a Sybil attack by adopting multiple identities and participating in the network with all identities at once. The result of such an attack gives the attacker a majority vote or considerable control in the network.

2.6 Security Model

A security model for mobile ad hoc networks is illustrated in general terms in Figure 2-2. A message *M* is to be transmitted from the source *A*, across a network of nodes, to a destination node *B*. The two entities who are primary participants must collaborate for the transaction to occur. A routing protocol establishes a multi-hop route between the primary participants. The multi-hop route will involve secondary participating nodes. Security is provided by two accompanying techniques, namely a security related transformation applied to the message (resulting in an encrypted message *C)* and secret keying information shared by the principal participants.

The general mobile ad hoc security model shows four basic tasks for a security mechanism:

1. The design of a security algorithm.
2. Generation of secret keying information used in conjunction with this security algorithm.
3. Distribution of secret keying information.
4. Protocol for the participants to follow which will achieve the required security services.

Tasks 1 and 2 deal with the cryptographic algorithm used to provide security services. It is widely recognized that existing cryptographic technology can provide sufficiently strong security mechanisms to ensure routing message confidentiality, authentication and integrity. The establishment of these security mechanisms is a dynamic problem in wireless ad hoc networks, as this network cannot adopt the same approaches of its wired predecessors. The focus of this dissertation is upon tasks 3 and 4, which involve the establishment of the security protocols in the mobile ad hoc environment.

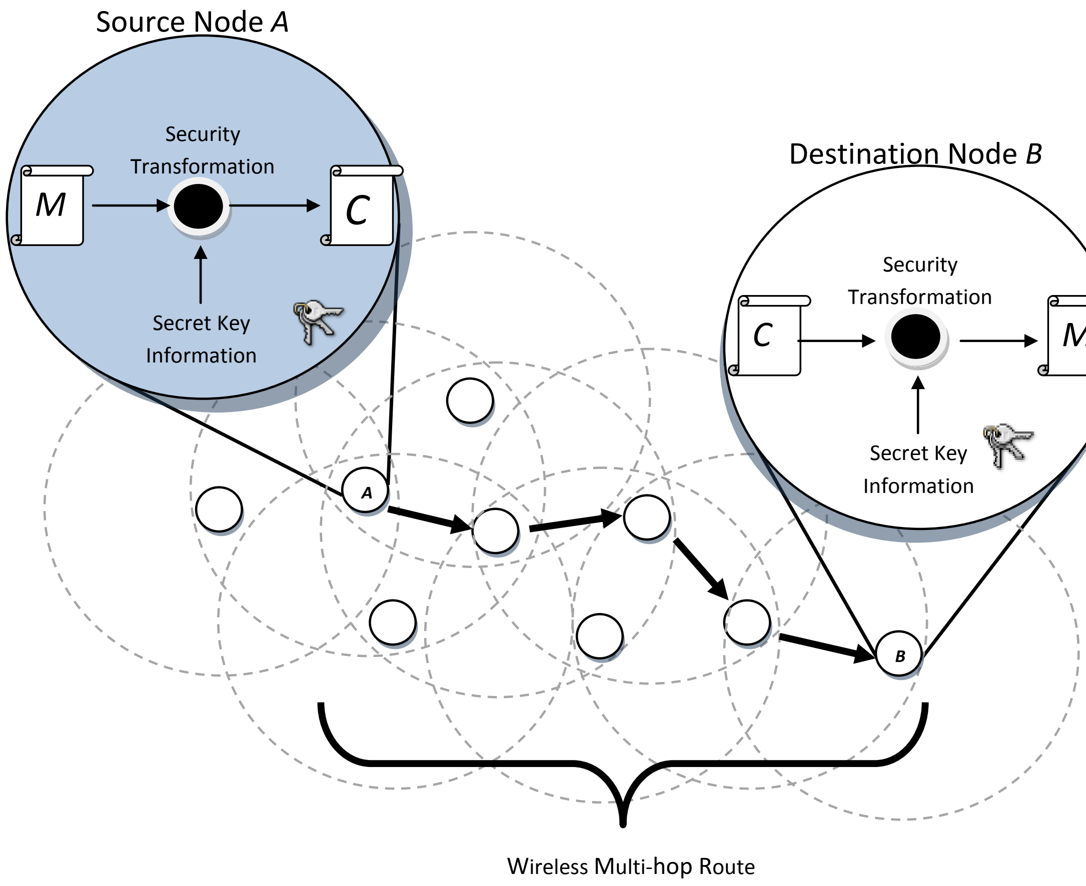


Figure 2-2: General Security Model

2.7 Conclusion

In conclusion chapter 2 provides essential background knowledge to mobile ad hoc networks. Mobile ad hoc network's distinct characteristics are investigated, providing insights into the environment and challenges that these networks will experience when deployed. A review of possible attacks on the network layer is presented showing that mobile ad hoc networks are vulnerable on the network layer specifically.

A security mechanism is reliant on some sort of central authority or shared key. It is noted that mobile ad hoc networks lack a fixed network infrastructure and prior relationships between nodes are non-existent. This will create dynamic security challenges for security solutions that are discussed in the chapters to follow.

Chapter 3

Key Management in Mobile Ad Hoc Networks

3.1 Introduction

Chapter 2 provided a discussion of the different types of attacks upon wireless ad hoc networks. In Chapter 3 the techniques used to prevent these malicious attacks and specifically key management techniques, will be reviewed. Security solutions that use cryptographic techniques rely on proper key management to establish trust. This chapter focuses upon key management that aids these cryptographic solutions.

3.2 Cryptography background

Cryptography is a common technique used to establish security. Cryptography techniques are not specific to mobile ad hoc networks, but can be applied to any communication network. Cryptography is defined as the study of the principals and design of algorithms for encryption and decryption. It is intended to ensure the secrecy and authenticity of messages and provide secure communications [15]. The procedure of encryption and decryption is governed by keying material. Keys are vital portions of information that are used to complete cryptographic algorithms. Keys may need to be kept confidential to ensure the security of a communication network. Key management deals with the secure administration of cryptographic keys to ensure the secure communication objectives, as defined in Chapter 2. There are two types of key cryptography: symmetric and asymmetric key cryptography.

3.2.1 Symmetric Cryptography

In symmetric key cryptography, both parties share the same key. This key is agreed upon before transmission and is used for both encryption and decryption techniques. There are two types of symmetric key algorithms: block ciphers and stream ciphers. Block ciphers divide the message into fixed-length parts that are called blocks. Each block is then individually encrypted. Examples of methods of symmetric key cryptography include substitution, transposition, and steganography [15]. Commonly used symmetric ciphers, such as the Data Encryption Standard

(DES) and the Advanced Encryption Standard (AES), are block ciphers. A stream cipher is essentially a block cipher with blocks of length one. Symmetric key algorithms are recognized as faster than asymmetric algorithms. However, they require a secret key to be shared first between the two communicating parties. In the event of a group of nodes communicating, each sender-receiver pair is required to share a separate secret key. This approach is un-scalable to large networks. If a group of nodes share a single key, the system is rendered vulnerable as the entire group would be endangered if the key was to be compromised.

3.2.2 Asymmetric Cryptography

Asymmetric key cryptography, also known as public key encryption, uses different keys for encryption and decryption. Each user has two keys: a private key (PR_i) which is secret, and a public key (PU_i) which is open to the network. Asymmetric encryption provides confidentiality and authentication. For example, to ensure message confidentiality between users A and B, A firstly encrypts (E), the message m intended for B, with *B's* public certificate PU_B. Thus the cipher text, *c, is formed*. The intended receiver, B, holds the secret matching private key PR_B, and performs decryption (D) with its private key obtaining the message m created by A. The cipher text is protected against malicious nodes, as long as B's private key remains a secret. The most widely used asymmetric key system is Rivest-Shamir-Adleman (RSA). This is a scheme based on the integer factorization problem and provides robust cryptographic security.

$$\textit{Encryption:} \quad c = E(PU_B,m)$$
$$\textit{Decryption:} \quad m=D(PR_B,c)$$

Authentication is realized in a similar manner by digital signatures. If A wants to sign a document m, it encrypts it using its private key PR_A (which is only known to A). Any user node who has A's public key can decrypt it and obtain the original document. If PR_A is maintained as secret then only A could have signed the document. This, therefore, provides authentication for A.

$$\textit{Authentication: } m = E(PR_A,m) = D(E(PR_A,m)$$

An outside trusted entity is primarily responsible for the issuing of these digital signatures (PR_i, PU_i) and for resolving disputes regarding signatures. The symmetric and asymmetric procedure is elaborated more in [9, 26].

3.2.3 Discussion

Symmetric key cryptography is less complex and computationally taxing than asymmetric key cryptography. Complex security algorithms result in communication overheads that pose major problems to mobile ad hoc networks, specifically those with high resource restrictions. Chan states that due to the resource constraints, asymmetric key cryptography is unsuitable for these networks.

"Due to their inefficiency, asymmetric/public key cryptosystems are unsuitable for ad hoc networks where there are constraints on computation and energy. In fact, symmetric key systems, like DES, AES and keyed hash functions, are still the major tools for communication privacy and data authenticity in most networks." [27]

Resource limitations must be addressed when designing a key management system and processing power cannot be assumed, even with current IC technology advances.

Despite the computationally taxing procedure of asymmetric or public key cryptography, this method is still identified as the most effective mechanism for providing security services to a dynamic network like the mobile ad hoc network. It is accredited by its superiority in distributing ,keys, providing authentication, achieving integrity; and non-repudiation [13, 26]. Symmetric key cryptography requires a connection that provides data integrity and confidentiality to distribute the shared secret keys. The possibility of such a connection cannot be guaranteed. With no prior trust relationships between nodes, or the possibility for key pre-distribution, the asymmetric key system proves superior to the more simplistic symmetric schemes. However, with a pre-distributive mobile ad hoc network, symmetric key cryptography must be considered if nodes have low computational strength or battery power. An example of this can be seen in a distributive sensor network.

3.3 Description of Key management

In any communication network, the cryptographic network security is dependent on proper key management. Mobile ad hoc networks vary significantly from standard wired networks. A specific efficient key management system is required to realize security in these networks.

Key management is defined as a set of procedures employed to administrate the establishment and maintenance of secure key base relationship. The purposes of key management, as stated by Menezes et al [28], is to:

1. Initialize system users within a network.
2. Generate, distribute and install keying material.
3. Control the use of keying material within the network.
4. Update, revoke, destroy and maintain keying material.
5. Store, backup and recover keying material.

Key management systems are responsible for the secure distribution of keys to their intended destinations. Keys that are required to remain secret must be distributed in a way that ensures confidentiality, authenticity and integrity. For example, in symmetric key cryptography both or all the participants must receive the key securely. For asymmetric key cryptography, the key management system must ensure that private keys are kept secret and only delivered to the required, authorized participants. Although, authentication and integrity is vital for public keys, they do not require confidentiality. The key management system is concerned with protecting the confidentiality and authenticity of keys, and must also prevent the unauthorized use of keys, including for example the use of keys that are outdated and invalid.

Cryptographic algorithms can provide confidentiality, authentication and integrity. However, the primary goal of key management, and the achievement of security specifications, is realized in the guarantee that the secret keying material is shared among the specific communication participants. There are several methodologies of sharing of the keying material. The main approaches are: key transport; key arbitration; key pre-distribution; and key agreement [28].

3.3.1 Key Transport

In a key transport system, one entity generates keys, or obtains keying material, and securely transports them to other entities in the network. The simplest key transport method is the key encrypting key method (KEK). This method assumes that a prior shared key exists among the participating nodes. The prior shared key is used to encrypt new keys and transport them to all participating nodes. Prior shared keying relationships cannot be assumed in networks, especially in mobile ad hoc networks. If a public key infrastructure exists, then the new keys can be encrypted by their respective receiver’s public key and transported without the existence of prior keying relationships. This approach assumes the existence of a trust third party (TTP) member which transports all the keying material. In pure mobile ad hoc networks a TTP member would not be available. Shamir’s three-pass protocol [29] is a key transport method without prior shared keys.

3.3.2 Key Arbitration

A key arbitration system is a division of key transportation. In key arbitration, a central arbitrator is assigned to create and distribute keys to all participants. The arbiter is often a wired node with no resource constraints. In mobile ad hoc networks, nodes are wireless with resource constraints. The arbiter would be required to be online throughout the network communication and be accessible to every member in the network. This is difficult in mobile ad hoc networks because of the resource constraints such as bandwidth; transmission range; and energy. A solution to these potential problems is a distributive system, where the arbiter is replicated at different nodes. Simple replication of the arbiter incurs severe resource expenses on certain nodes and creates multiple points of vulnerability in the network. If a single replicated arbiter is compromised, the entire network can be at risk.

3.3.3 Key Pre-distribution

Keys are distributed to all participating members before the start of communication. Key pre-distribution requires prior knowledge of all participating nodes. Its implementation is simple and involves much less computation than other schemes. This method is suitable for mobile ad hoc sensor networks, as they have highly restrictive resource capabilities. The set of sensor nodes is also established before the network is deployed and data is tracked. Once the network is deployed, there is no service that allows for new members to join or for keys to be changed. This method is extended by allowing sub-groups of communication to form in the network. Similarly, the decision is made prior to deployment, and not during communication.

3.3.4 Key Agreement

Key agreement is used to enable two participants to agree upon a secret key. In this way, keys are shared and establish a secure communication line over which a session can be run. Key agreement schemes are often based on asymmetric key cryptography and have high computational complexity, but little pre-configuration is required. The most widely used key agreement scheme is the Diffie-Hellman key exchange [30]. This is an asymmetric keying approach based on discrete logarithms.

3.4 Key management in Mobile Ad Hoc Networks

Ad hoc wireless networks have unique characteristics and challenges that do not allow the simple replication of conventional key management methods that are used for wired networks. Mobile ad hoc networks’ lack of infrastructure poses the greatest threat to the establishment of a secure key management scheme. Fixed infrastructure such as a trusted third party member, an administrative support or certificate authority, dedicated routers, or fixed reliable communication links cannot be assumed in wireless ad hoc networks. Unique solutions are required for such unique networks. The focus of this chapter is on the investigation of the existing key management schemes for mobile ad hoc networks.

Key management schemes are investigated with regard to functionality, scalability, availability, security services, efficiency, and computational cost. A key management solution that is scalable will effectively provide security services in a network that dynamically changes in size, as nodes join and leave the network. Availability is essential for a network whose topology is rapidly changing. Nodes should have easy access to authority members and keying services. A high priority is the success and efficiency that a key management solution can provide in crucial security services for the keying material. Such services include key confidentiality, key authenticity, key integrity, and fresh key updates. These services are congruent with the security services described in Chapter 2.

Of the existing key management solutions, asymmetric cryptography is predominantly used when managing trust via a public key infrastructure (PKI) of some sort. Existing PKI schemes utilize either the hierarchical or web-of-trust model.

3.4.1 Hierarchical Trust models

The hierarchical trust models are more structured, as they use a PKI and a certificate authority as a source of trust. The certificate authority (CA) is a trusted entity used to verify, issue, and revoke certificates, thereby enabling successful public key cryptography. A key management service for public key cryptography would include the certificate authority service that has a public key, *K*, and private key, *k*. The CA's public key is distributed to all the nodes in the network. The nodes know that any certificate signed by the CA's private key may be trusted. Each node also has its own public/private key pair, which allows for nodal communication. The CA stores the public keys of all the network nodes and distributes the respective keys to the nodes that request to set up a secure communication with another node [9]. A fixed CA is not considered in this investigation, due to the limitations caused by having no TTP.

The CA distributes trust in a hierarchical manner, as seen in Figure 3-1. A root CA issues certificates to delegated CA's or end users. The CA can issue certificates to user nodes or other CA nodes. The PKI X.509 framework is an example of such an infrastructure [15].

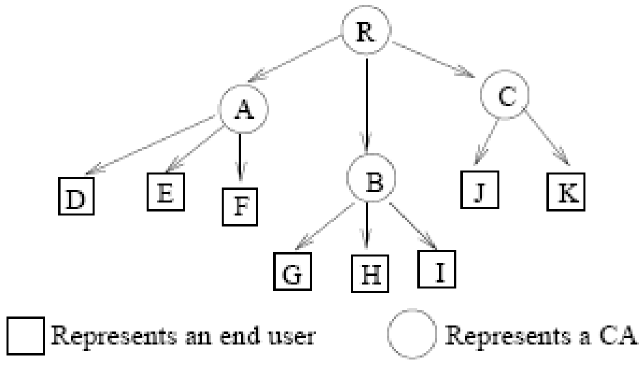


Figure 3-1: Hierarchical trust

The following types of hierarchal trust models have been investigated in the context of mobile ad hoc networks:

1) *Off-line trusted third party models* use a trusted outside entity to achieve a large portion of the key management tasks.
2) *Partially distributed certificate authority models,* distribute the functionality of the CA to a small set of nodes.
3) *Fully distributed certificate authority models,* are self-organized models that are similar to the previous category. However, this model distributes across the entire network in a self-organized manner.
4) *Cluster-based model,* is a special kind of hierarchical trust in the form of group authentication, where *clustered groups* of nodes are treated as single trust entities and authenticated as a group.

3.4.2 Web-of-Trust Models

The Pretty Good Privacy model (PGP) [31], also known as a "web-of-trust-model", enables nodes to act as independent certification authorities. There is no distinction between a CA and an end user node. Nodes provide individual trust opinions of other nodes, thereby creating a "web of trust", as illustrated in Figure 3-2. Each user node is the "centre of its own world" and is responsible for certificate management. The advantage of a PGP model is its dynamic, autonomous nature that is seemingly ideal for application in decentralized environments such as ad hoc networks [32].

Certificates are issued by the nodes themselves. However, a public certificate directory is required for their distribution. This directory is often located at an online, centralized, trusted third party entity. This makes the PGP model unsuitable for ad hoc network application. Steps are needed to be taken to localize such directories and realize certificate distribution. The autonomous nature of the "web-of-trust" model means that it is more susceptible to malicious attackers, than to more structured networks. For example, if one entity is compromised, a corrupt set of certificates is filtered throughout the network. The self -issued certificate model is investigated as a foundation for PGP-based solutions in mobile ad hoc networks. Figure 3-3 illustrates the key management solutions investigated in this chapter.

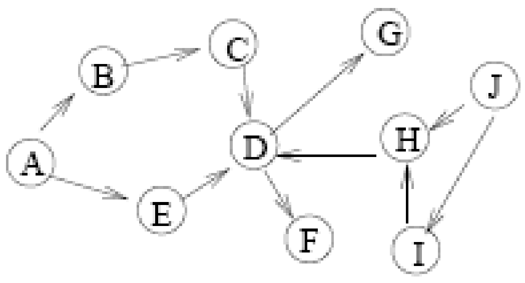


Figure 3-2: PGP web-of-trust

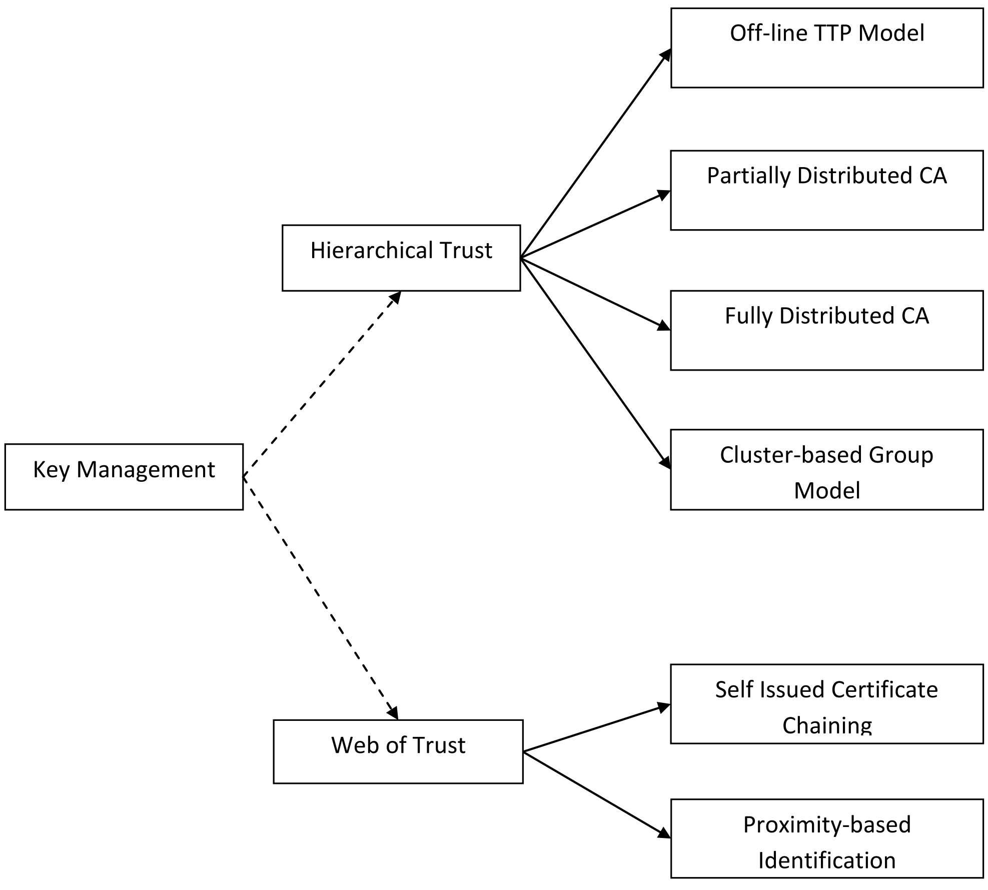


Figure 3-3: Key Management Solutions

3.5 Off-line Trusted Third Party Models

A progress trust negotiation scheme was introduced by Verma [33]. It is a hierarchical trust model where authentication is preformed locally, but an off-line trusted third party performs trust management tasks like the issuing of certificates. The off-line trusted third party also manages the certificate revocation process. This scheme is extended through a localized trust management scheme proposed by Davis [32]. Davis attempts to localize Verma's solution. The only trust management task that is not implemented locally is the issuing of the certificates.

3.5.1 System Overview

Each node possesses its own private key and the trusted third party's public key. The maintenance of these keys is the responsibility of each node. Trust is established when the trustor provides the trustee with a certificate that has not expired or has not been revoked, and the trustee can verify it with the trusted third party's public key (possessed by the trustee). Furthermore, to realize certificate revocation, each node must possess two certificate tables: a status and profile table. The profile table, illustrated in Figure 3-4, describes the conduct or behaviour of each node. The status table describes the status of the certificate, i.e. revoked or valid. These two tables are maintained locally by the nodes themselves, with the purpose of maintaining consistent profiles.

Davis's scheme is a fully distributed scheme. It requires that a node broadcasts its certificates and its profile table to all the nodes in the network. It also requires that each node's profile table be kept updated, and distributed with synchronization of data content. The profile table contains information from which the user node may calculate, if a certificate can be trusted or of it must be revoked. Node *i*'s profile table stores three pieces of data:

1) *Accusation info:* the identity of nodes that have accused node *i* of misbehaving.
2) *Peer n ID:* the identity of nodes that node *i* has accused, acting almost as a CRL (certificate revocation list).
3) *Certificate status:* a 1-bit flag indicating the revocation status of the certificate.

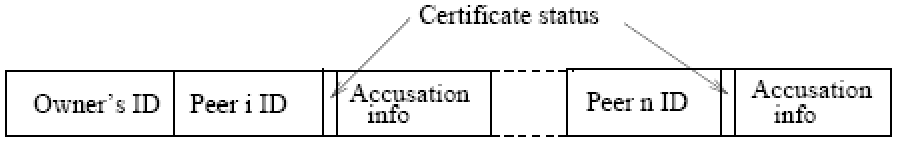


Figure 3-4: Profile Table

The fully distributed information in the profile tables should be consistent. If any inconsistency is detected, an accusation is expected to be launched against the node in question. Inconsistent data can be defined as data that differs from the majority of data.

The status table is then used to calculate the certificate's status, i.e. revoked or not revoked. The node *i*'s status table stores and analyses the following factors: A_i (total number of accusations against node *i*) ; a_i (total number of accusations made by node *i*) ; *N* (expected maximum number of nodes in the network). These factors are used to calculate the weight of node *i*'s accusation and the weight of other nodes' accusations against node *i*. A revocation quotient is then calculated, R_j , as a function of the sum of the weighted accusations. It is then compared to a network defined revocation threshold R_T. If $R_j > R_T$ then the node *i*'s certificate is revoked.

3.5.2 Analysis

This scheme uses a hierarchical trust model that relies upon an off-line trusted third party for aspects of key management. The off-line trusted third party is to be resident as a trusted source if required. This scheme assumes the existence of a trusted off-line entity which initializes certificates, and securely distributes them amongst the network participants. This scheme is a pre-distributive key exchange model. It provides robust security, however its implementation is more realistic within a hybrid infrastructure. A key management scheme with a hybrid infrastructure is a scheme that makes use of both wired and wireless architecture. A wired trusted off-line node performs all or a portion of the key management services to maximise security and efficiency. Hybrid infrastructures allow for greater security and offer a relatively simple solution to the central problem of key distribution in mobile ad hoc networks.

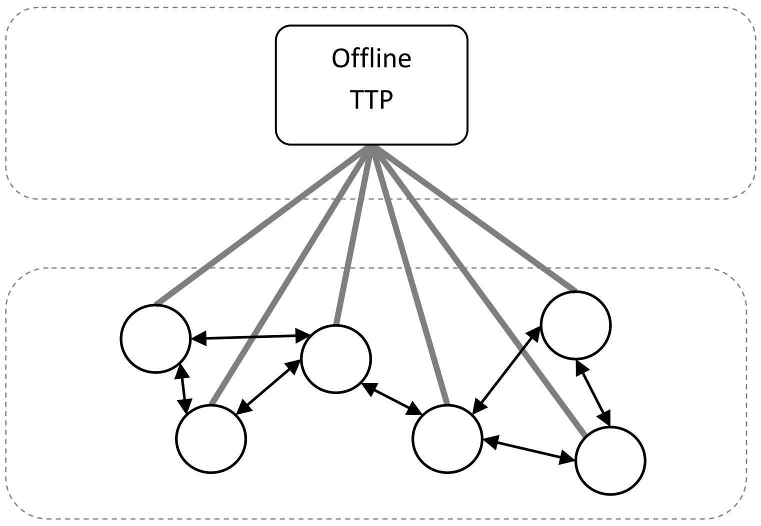


Figure 3-5: Hybrid progressive trust negotiation scheme

Verma and Davis's solution does not specify that a wired node be the off-line authority for key pre-distribution. Nevertheless, a separate trusted entity capable of intense computation, high security and network distribution must exist for the success of Verma and Davis's model. Such assumptions cannot be made in pure mobile ad hoc networks. The hybrid nature of Davis's solution is displayed in Figure 3-5.

Verma localizes the task of authentication. Davis goes one step further by localizing the revocation module of the scheme by proactively maintaining accusation information in profile tables and locally, calculating revocation decisions. This scheme mitigates against malicious accusation exploits. This could result in a node being revoked based on a single malicious offender's broadcast information. To solve this problem, one must not treat all accusations equally but rather use a sum of weighted accusations that are calculated before the node is revoked. Davis' scheme succeeds in taking steps toward self-organization in ad hoc network trust establishment as it provides a protocol that enables the revocation of certificates without continual trusted third party involvement.

3.6 Partially Distributed Certificate Authority

The solution proposed by Zhou and Haas [13] allows for the functionality of the certificate authority to be shared amongst a set of nodes in the network. This solution aims to create the illusion of an existing trusted third party. Zhou and Haas's proposal in 1999 was instrumental in the initial research of key management solutions for ad hoc networks. This approach has been extended to incorporate the heterogeneous nature of nodes in [34].

3.6.1 System overview

The CA's public key, *K*, is known by all nodes (*m*) and the CA's private key, *k*, is divided and shared by *n* nodes where $n < m$. The distributed CA signs certificates by recreating the private key via a *t* threshold group signature method. Each CA node has a partial signature. The CA's signature is successfully created when *t* correct partial signatures are combined, at a combiner node. To prevent the distributed CA nodes from becoming compromised and the authentication becoming compromised, a preventive proactive scheme is implemented in order to refresh the CA nodes. A simple partially distributed CA system is illustrated in Figure 3-6.

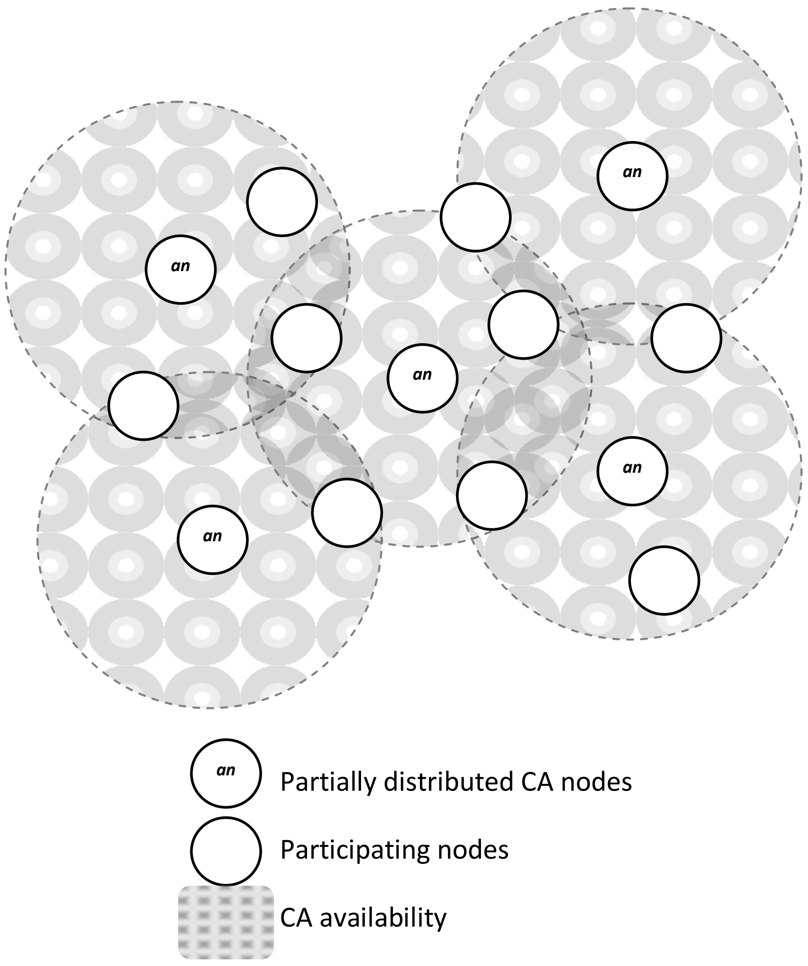


Figure 3-6: Partially Distributed Certificate Authority

3.6.2 Threshold Scheme

Threshold cryptography is used to share the CA service between nodes. A threshold cryptography scheme allows the sharing of cryptographic functionality. A *(t-out-of-n)* threshold scheme allows *n* nodes to share the cryptographic capability. However, it requires *t* nodes from the *n* node set to successfully perform the CA's functionality jointly. Potential attackers need to corrupt *t* authority nodes before being able to exploit the CA's functionality and analyze secret keying information. Hence, a *(t-out-of-n)* threshold scheme tolerates *t-1* compromised nodes, from the *n* node set[35].

When applying threshold cryptography to the shared CA problem, the CA service is shared by *n* nodes across the network called authority nodes. The private key *k*, crucial for digital signatures, is split into *n* parts ($k_1,k_2,k_3,...,k_n$) assigning each part to an authority node (*an*). Each authority node has its own public key, $K_{n,}$ and private key, $k_{n,}$ (as seen in Figure 3-7).It stores the public keys of all the network nodes (including other authority nodes). Nodes wanting to set-up secure communication with node *i* need only request the public key of node *i* (K_i) from the closest authority node, thereby increasing the CA's availability. For the CA service to sign and verify a certificate, each authority node produces a partial digital signature using its respective private key, k_p, and then submits the partial digital signature to a combining node. Any node may act as a combiner in the ad hoc network. The partial digital signatures are combined at a combiner (*c*) to create the signature for the certificate, *t* correct partial digital signatures are required to create a successful signature. This effectively protects the network against corrupt authority nodes, and up to *t-1* corrupt authority nodes may be tolerated [13].

For example, in figure 3-8 a (*2-out-of-3*) threshold scheme where the message *m* is signed by the CA, two partial signatures (*PS*) are accepted, while the third (an_2) was corrupted. The partial signatures meet the threshold requirements, and the partial signatures are combined at *c* and applied to the message.

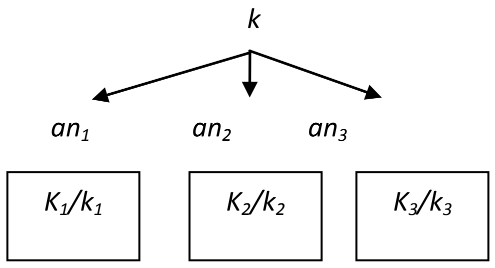


Figure 3-7: (2-out-of-3) Threshold Key Management

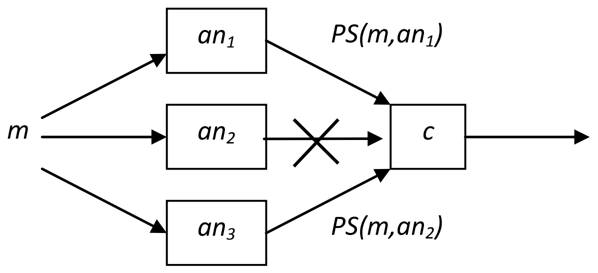


Figure 3-8: (2-out-of-3) Threshold Signature

3.6.3 Proactive security

Threshold cryptography increases the availability and security of the network by de-centralizing the CA. Security is maintained with the assumption that all CA authority nodes cannot be simultaneously corrupt.

It is possible for a malicious attacker to compromise all the CA's authority nodes over time. An adversary of this type is then able to gain the CA's sensitive keying information. Proactive schemes [36-39] are implemented to avoid such adversaries.

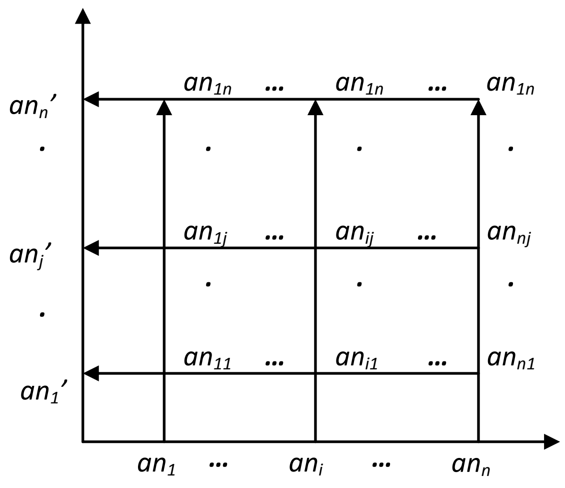
A proactive threshold cryptography scheme uses share refreshing. This enables CA authority nodes to compute new key shares from old ones without disclosing the CA's public/private key. The new key shares make a new (*t-out-of-n)* sharing of the CA's public/private key pair. These are independent of the old pair [40].

Share refreshing relies on the following mathematical property:

If (s_{11} , s_{21} , ... ,s_{n1}) is a (*t-out-of-n)* sharing of k_1 and (s_{12} , s_{22} , ... ,s_{n2}) is a (*t-out-of-n)* sharing of k_2, then ($s_{11} + s_{12}$, $s_{21} + s_{22}$, ... ,$s_{n1} + s_{n2}$) is a (*t-out-of-n)* sharing of $k_1 + k_2$. Therefore if k_2 is 0, then we get a new (*t-out-of-n)* sharing of k_1 [13].

The share refreshing scheme is applied to a threshold CA. A threshold CA is a (*t-out-of-n)* system that shares the CA's private key *k* among *n* authority nodes (an_1 , ... , an_n) each with a share of the CA's private key. To generate a new (*t-out-of-n)* sharing (an_1', ... , an_n') of *k,* each authority node an_i generates sub-shares (an_{i1} , an_{i2} , ... , an_{in}) a (*t-out-of-n)* sharing of 0that represents the *i*'th column, as seen in figure 3-9. Each sub-share an_{ij} is sent to the authority node an_j. When authority node an_j has received all sub-shares (an_{1j} , an_{2j} , ... , an_{nj}) that represents the *j*th row seen in figure 3-9, it then generates its new share an_1' by using the mathematical property described above.

The communication of the sub-shares requires a secret redistribution protocol [41, 42] to ensure secure transmission. Note that share refreshing does not change the CA's private key pair. Share refreshing may occur periodically and be extended to occur upon events. These events can include the detection of compromised nodes or a change in network topology. Therefore, the key management service is able to transparently adapt itself to changes in the network and maintain secure communication.



an_n'
an_j'
an_1'
an_{1n} ... an_{1n} ... an_{1n}
an_{1j} ... an_{ij} ... an_{nj}
an_{11} ... an_{i1} ... an_{n1}
an_1 ... an_i ... an_n

Figure 3-9: (t-out-of-n) Share Refreshing

3.6.4 Heterogeneous Extension

An extension to Zhou and Haas's scheme can be seen in the Mobile Certificate Authority (MOCA) scheme by Yi and Kravets [43]. Similarly, the MOCA scheme uses threshold cryptography to implement a public key that is a partially distributed certificate authority solution. The functionality of the certificate authority is distributed to *n* nodes called MOCAs. The assumption is made that all nodes have heterogeneous visible qualities. These visible qualities act as initial trust evidence and are used when selecting the MOCA nodes to distribute authority. Such visible evidence can include computational power, physical security, or position. This evidence is based on a trust decision and authority is distributed accordingly. Similar to Zhou and Haas's scheme, nodes require $t+1$ partial signatures from a set of *n* MOCAs to allow for certificate verification and trust relationship establishment with a threshold of *t*. The MOCA scheme further builds on Zhou and Haas's solution by adding a revocation of certificates. Certificate revocation lists are stored at each MOCA. For certificates to be revoked, $t+1$ MOCAs must sign a revocation certificate request with $t+1$ partial signatures from the MOCAs. Once the partial signatures are gathered, the certificate revocation list is updated. Malicious nodes wanting to unnecessarily revoke another node's certificate can only do so with the approval of $t+1$ trusted MOCAs, thereby ensuring the reputation of each node's certificate.

3.6.5 Analysis

This solution demonstrates some of the problems of an ad hoc network. Despite its obvious weaknesses, it is noted as one of the earliest key management solutions to ad hoc networks.

The partial distributive scheme proposed by Zhou and Haas requires that an off-line TTP member exists at the initialization phase in order to establish the distributive CA. The off-line TTP: generates the threshold private key, shares it among the appointed CA authority nodes, and distributes the CA's public key to all participating nodes in the network. All certificate related tasks including signatures, generation, distribution, refreshing, and revocation are performed by the participating nodes, without the involvement of a TTP. The off-line TTP is not as involved, as in Verma [33] and Davis' [32] proposals. However, in spontaneous ad hoc networks such a trusted entity cannot be assumed at initialization.

The advantage of distributing the CA allows for the functionality of the CA to be distributed among the nodes. This avoids single point attacks and allows the computational overhead of the CA's services to be distributed. Although the CA is distributed, it still remains centralised between a few nodes.

The centralization of authority creates availability issues. In addition to this, availability issues are heightened as communicating nodes are required to communication with *t* authority nodes before acquiring a signature. The CA's availability is dependent on the threshold parameters *t* and *n*. These parameters must be selected to provide a suitable trade-off between availability, security, and cost of computation. The larger the threshold (*t*), the higher the security is, at the cost of availability. The centralization of authority also results in a select group of nodes carrying the burden of security computations. This breaks the value of fair distribution in a network.

This solution requires that the CA authority nodes store all the certificates issued, and necessitates a costly synchronization mechanism. Furthermore, a share-refreshing or proactive method is required. This is achieved by using a secret redistribution protocol [41]. With this in place, it is ascertained that all the CA authority nodes are not compromised. The procedure of synchronization, updating and proactive refreshing is costly to resource constrained nodes.

Another potential problem is related to network participants addressing the CA authority nodes. A node requesting a service from the CA entity is required to contact *t* out of *n* nodes. The CA can then be given a multicast address and participating nodes can multicast their requests to the CA. The CA authority nodes can then unicast replies to the requesting participant. In ad hoc networks that do not support multicasting, a participating node can broadcast its request. This approach is more common in mobile ad hoc networks, despite its potential of a large amount of network traffic.

Zhou and Haas's partially distributed certificate authority approach provides much of the groundwork for future solutions through the implementation of threshold cryptography in ad hoc networks.

3.7 Fully Distributed Certificate Authority

The threshold scheme investigated in [44, 45] uses ideas proposed by the partial distributive threshold scheme found in [13]. *Luo* and *Lu* propose a scheme that embraces the distribution of the CA. In a network of *m* nodes, the network and security services are shared across *m* nodes. Therefore, a fully distributed system is realized, as seen in figure 3-10. This scheme further differs from [13] in that there is no need to select specialized nodal authorities, as all nodes perform this role. Like the partial distributive scheme, the fully distributive scheme includes the use of share refreshing. This allows proactive security against significant nodes that are compromised. This scheme is designed for and aimed at planned, long-term ad hoc networks that have the capacity to handle public key cryptography.

3.7.1 System overview

The Fully Distributive Certificate Authority scheme is a public key cryptography scheme. It takes the functionality of the certificate authority and distributes it across *m* nodes, where *m* is the total number of nodes in the network. This threshold scheme requires *k* or more nodes to act in collaboration to perform any operations of the CA. The CA's private key is divided and shared among all the participating nodes. This effectively enhances availability and allows nodes that are requesting the CA to contact any *k* one-hop neighbour nodes. It is assumed that each node will have more than *k* one-hop neighbours [44]. Therefore, only one-hop certificate communication can occur. This allows for more reliable communication, in comparison with multi-hop communication. It is also easier to detect compromised nodes. Figure 3-10 illustrates the fully distributive network where all nodes have a portion of authority in the form of a partial CA signature. Figure 3-10 shows a network with threshold $k=3$, where nodes *B*, *C* and *D* can find a coalition of partial CA nodes to form a group authentication CA signature. Node *A* is unable to find a sufficient coalition of nodes.

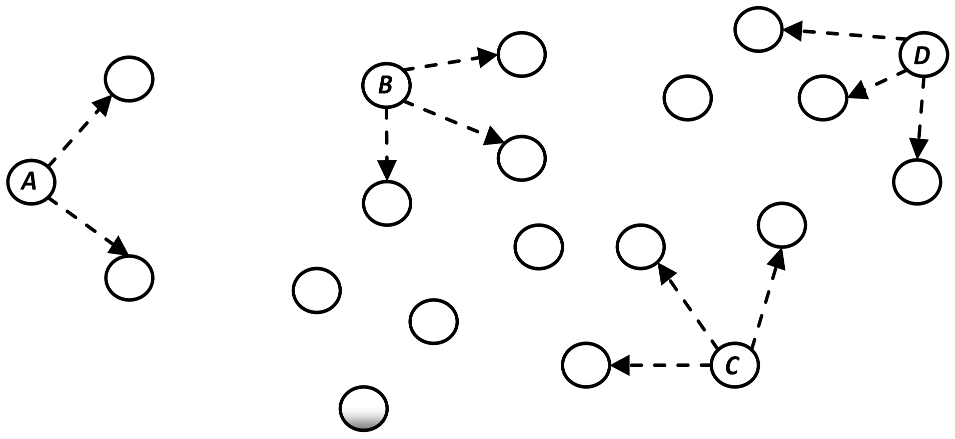


Figure 3-10: Fully distributive CA system

3.7.2 Off-line Initialization

The initial phase of [44, 45] requires an off-line trusted third party (TTP) to establish the initial set of nodes. The off-line TTP will provide each node *i* with its own certificate, the CA's public key, and a share of the CA's private key. A certificate is a binding between a node's ID and its public key. The certificate is signed by CA's private key k_{CA} and can be verified by the CA's public key K_{CA}. that is made available to all the participating nodes. The off-line TTP initialises the threshold private key to the first *k* nodes by the following steps:

1. Generate the sharing polynomial $f(x) = a_0 + a_1x + ... + a_{k-1}x^{k-1}$ where $a_0 = k_{CA}$
2. Securely distribute node *i* identified by ID_i where $i \in k$ with its secret share $S_i = f(ID_i)$

3. Broadcast k public witnesses of the sharing polynomial's coefficients $\{h^{a_0}, \ldots, h^{a_{k-1}}\}$ and then the off-line TTP involvement is over.
4. Each node with ID_i that has received a secret share S_i verifies it by checking the sharing polynomial's coefficients such that $h^{S_i} = h^{a_0} \cdot (h^{a_0})^{ID_i} \cdot (h^{a_1})^{ID_i^2} \cdot \ldots \cdot (h^{a_{k-1}})^{ID_i^{k-1}}$

After the initial establishment of the shared secret key amongst the first k nodes, the TTP is no longer responsible for the full distribution of the CA's private key. The off-line TTP maintains the responsibility of issuing new nodes with their initial certificates binding, and as a result impersonation attacks are prevented.

3.7.3 On-line Shared Initialization

New nodes entering the network need to be provided with their own share of the CA private key k_{CA} so that they can be part of the signing process. The participating nodes in the network perform this initialization process, without the interference of an off-line TTP. Shared initialization is modelled on Shamir's threshold secret sharing scheme [29]. This scheme allows for a culmination of t nodes to initialize a joining node with a share of the CA private key k_{CA}.

A node i, already initialized by the off-line authority can generate a partial secret share $S_{p,i}$ for a joining node p. The combination of k partial secret shares results in node p's secret share $S_{p.}$ This is a partial share of the CA's private key.

$$S_p = \sum_{i=1}^{k} S_{p,i}$$

Node i's secret share S_i can be derived from each partial secret share S_p that is sent to node p. The joining node p must not be allowed to know the secret shares of other nodes, as this would breach confidentiality. The aim is to hide the actual partial secret shares $S_{p,I}$, while still transporting the combined secret share S_p to node p. A shuffling scheme is used to solve this problem. The shuffling scheme is illustrated in figure 3-11. From Figure-1, nodes i and j wish to initialize node p with a secret share Sp. Nodes i and j agree upon a shuffling factor d_{ij}. The shuffling factor is combined with the partial secret shares $S_{p,i}$ and $S_{p,j}$. The sum of the shuffling factors is null. This allows for the secret share S_p to be calculated while hiding the secret shares of i and j. Figure 3-11 illustrates a system with a threshold of two nodes to scale this to k nodes. Each pair of contributing nodes must decide on a shuffling factor resulting in $k(k\text{-}1)/2$ shuffling factors that need to be distributed.

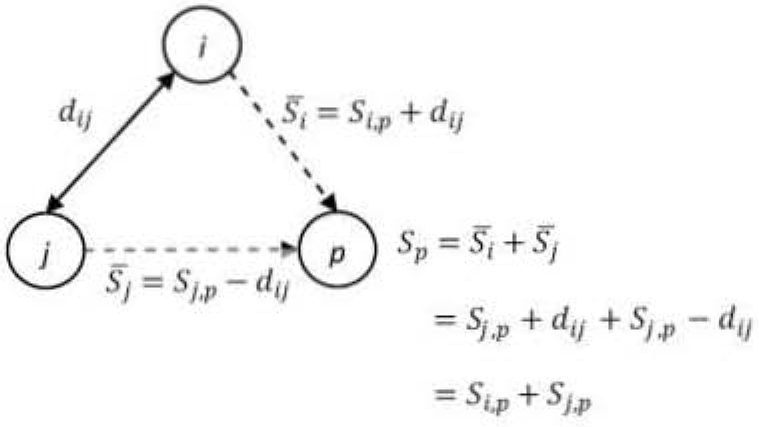

d_{ij}
$\overline{S_i} = S_{i,p} + d_{ij}$
$\overline{S_j} = S_{j,p} - d_{ij}$
$S_p = \overline{S_i} + \overline{S_j}$
$= S_{j,p} + d_{ij} + S_{j,p} - d_{ij}$
$= S_{i,p} + S_{j,p}$

Figure 3-11: Shuffling scheme of partial secret sharing

This key transport mechanism is described in the following steps:

1. Node *p* broadcast an initial request to a coalition of *k* neighbouring nodes.
2. The coalition of nodes divides into *i* and *j* pairs, and agrees upon appropriate shuffling factors. An associated public witness $h^{d_{ij}}$ is generated and signed to identify any misbehaviour. The shuffling factor and the witnesses are sent to node *p*.
3. Node *p* routes all the shuffling factors and witnesses to the *k* coalition nodes.
4. Each coalition node *j* generates the partial secret share $S_{j,p}$ and shuffles it with the shuffling factors received by *p* such that $\overline{S_{j,p}} = S_{j,p} + \sum_{i=1}^{k} d_{ij}$ and sends $\overline{S_{j,p}}$ to *p*.
5. Node *p* verifies the shuffled share values $\overline{S_{j,p}}$ by checking the public that witnesses $h^{\overline{S_{j,p}}} = h^{S_p} \prod_{i=1}^{k}\left(h^{d_{ij}}\right)$. If the verification is successful, the shuffled share values are combined such that $S_p = \sum_{i=1}^{k} \overline{S_{p,i}}$.

After the joining node *p* has been issued with a part of the CA private key, it can perform the services of the CA in the network including certificate renewal and certificate revocation. System maintenance includes the initializing of joining nodes. System maintenance also encompasses the renewal of certificates, certificate revocation and proactive updating of the CA private key shares, thereby protecting against the CA's private key becoming compromised.

3.7.4 Share Updating

In a k threshold system, attacks can compromise *k* nodes over a period of time and allow them to impersonate the CA and perform malicious communication attacks. A solution to this is secret share updates by the use of a proactive security method, similar to that used in partial distributed certificate authority methods [13].

The network will have an operation phase and an update phase where periodic updates occur of the secret shares of the CA's private key. During the update phase all nodes participate in the updating procedure. Each node will have an equal probability of initiating the update phase, enabling a fair distribution of the load. The secret share update phase involves the following steps:

1. The node that is to initiate the update phase requests a coalition of *k* nodes and generates an update polynomial $f_{update}(x) = b_1 x + b_1 x^1 + \cdots + b_1 x^{k-1}$.
2. Each co-efficient of the polynomial is signed by the coalition CA and flooded through the network such that each node possesses the $f_{update}(x)$ polynomial.
3. Each node *i* generates its secret update share $S_i = f_{update}(ID_i)$ and verifies it by a coalition of *k* nodes. Each node in the coalition returns a partial update to node *i* that combines them to form its update share. This update share is added to the current share, and a new updated share of the CA's private key is formed.

The share update procedure provides robust security against multi-point attacks but security comes at a high computational cost.

3.7.5 Certificate Renewal

Certificate issuing is assumed to be handled by the off-line TTP that registers, initialises, and certifies new nodes joining the network. The issue of certificate renewal is performed by the distributed CA in the network. Each node's certificate is only valid for a specified time period, after which it must be renewed before it expires. For successful certificate renewal in a *k* threshold fully distributive system, node *i* must request the renewal of certificate $Cert_i$ from a coalition of *k* nodes. One-hop neighbours are identified as more trust worthy coalition members. Each coalition node then generates a new partial signature and will send it to node *i*. Node *i* then acts as a combiner (all nodes may act as combiners in the fully distributive certificate authority scheme) and combines the *k* partial signatures to produce the new certificate $\overline{Cert_i}$ [44]. In a similar manner, messages are signed by the coalition nodes and form a group signature as described in providing authenticity and security.

3.7.6 Certificate Revocation

Certificates can be revoked if nodess are found to be corrupt or compromised. This revocation service assumes that all nodes monitor their one-hop neighbour nodes and are capable of retaining their own certificate revocation list (CRL) [44]. When a user node identifies that a neighbouring node is corrupt, it adds the node in question to its CRL and announces this to all neighbouring nodes. The neighbouring nodes in turn check to ascertain if this announcement is from a reliable source, i.e. the source is not on the receivers CRL. If the source is reliable, the announced node is marked as suspect. If a threshold of *k's* reliable accusation is made against a single node, then the node's certificate is revoked. This procedure allows for compromised nodes to be identified and explicitly quarantined from CA involvement until such a time as they have become secure again. Implicit revocation is implemented by setting lifetimes for certificates t_{cert}. When the time has expired and the certificate has not been renewed, it is implicitly revoked.

3.7.7 Analysis

This scheme is a hierarchical model. It is similar to the partially distributed certificate authority scheme. One can see that fully distributive networks possess similar weaknesses to partial distributive networks. Both schemes require prior knowledge and an off-line TTP for the initialization of certificates. The main advantages of the fully distributive scheme are its availability and implement revocation mechanism.

The fully distributive nature of the CA allows for high availability. It does require that each requesting node have *k* one-hop neighbours that form a CA coalition. The localization of the coalition to the one-hop neighbours avoids transitive trust and reduces network traffic.

One can choose for the threshold parameter *k* to be larger thereby providing a higher level of security. This change requires an attacker to compromise a larger number of nodes in order to obtain the CA's private key. Increased security comes at the cost of availability. This scheme is non-scalable, as it lacks a mechanism that increases the threshold parameter *k* dynamically as the network density increases.

As the CA is distributed through the network its availability is greatly increased. However, an increase in availability of the CA requires a greater security and more focus upon the proactive share-refreshing scheme. This scheme is a complex and computationally taxing maintenance protocol. It includes the share initialization and share update protocols. The trade-off between security and resources is an important issue in wireless ad hoc networks.

The revocation mechanism allows for explicit and implicit revocation, while the assumption is made that all nodes are computationally capable of monitoring the behaviour of their one-hop neighbours. However, this assumption may not be true for certain ad hoc networks.

3.8 Cluster-based Model

This solution investigates the Secure Pebblenets [46] which is a cluster or group-based scheme. This solution is a group-based solution that uses symmetric key cryptography. It is a hierarchical distributive key management system. The focus of this scheme provides group authentication for user nodes, as well as message integrity and confidentiality. Group authentication is achieved by grouping nodes into clusters and treating them with blanket authentication. This solution is suited for planned, long-term distributed ad hoc networks. It is specifically aimed toward networks with low capacity nodes that lack the resources to perform public key encryption.

3.8.1 System overview

This solution requires an initial infrastructure for setup. A secret group identity key k_{GI} is set. This identity provides every node with authentication and integrity. Its key is kept constant for the duration of the network unless an off-line authority re-initializes the network. k_{GI} is used to generate further keys to provide message confidentiality [46].

The life of the network is illustrated in figure 3-12. The lifetime is divided into time slices, with three phases: the cluster generation phase; the operation phase; and the key update phase. Each time slice will consist of these three phases. Assuming a network has low processing capacity nodes, authentication is complex and costly. Therefore authentication, confidentiality and integrity are provided for nodal groups or clusters. This maximizes efficiency and minimizes computational cost.

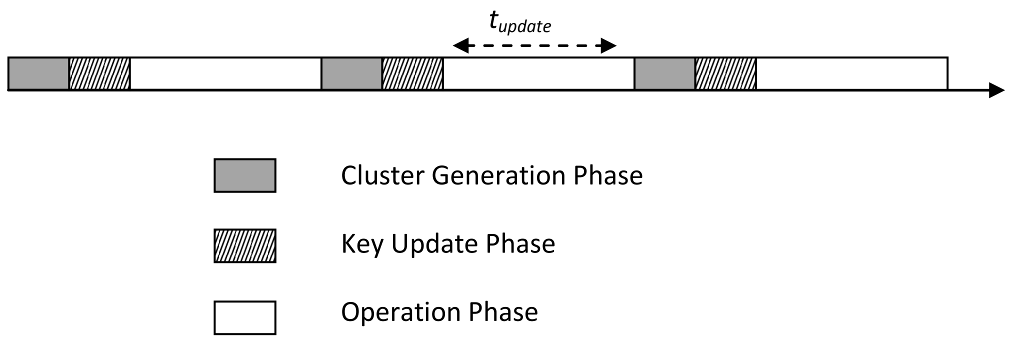


Figure 3-12: Phases of the network lifetime

3.8.2 Cryptographic keying material

The network uses the following cryptographic keying material to provide message and group confidentiality and authentication:

1. Group identity key k_{GI} is shared prior to network establishment between all network nodes and is used to derive additional keys for security services.
2. Traffic encryption key k_{TEK} is used for symmetric data encryption and is updated during the network lifetime.
3. Cluster key k_C is used for cluster specific communication.
4. Backbone key k_B is used to encrypt communication between cluster heads.
5. Hello key k_H is used between neighbours in cluster generation phase.

The cluster key is generated by the cluster head. The k_{TEK} is randomly generated by the key manager who is selected in the key update phase. The group identity key is used to derive the backbone and hello keys in the following manner, where k^i represents the key in the i time slice and h^i represents a hash function to the order i:

$$k_B^0 = k_{GI}$$

$$k_H^i = h\left(k_B^{i-1}\right) = h^i(k_{GI})$$

$$k_B^i = h\left(k_H^{i-1}\right) = h^{i+1}(k_{GI})$$

The three phases of operation use the described, cryptographic keying material to provide cluster-based security in a hierarchical manner.

3.8.3 Cluster Generation Phase

During the cluster generation phase, nodes decide to be either cluster heads or cluster members. This decision is based on a variable called weight [46]. Node i's weight w_i is a representation of the node's current capacity status that is made up of factors such as: battery power, and distance from other nodes etc. The cluster head will manage the group keying services for that cluster. The cluster heads then discover each other and establish a cluster head backbone that is used to distribute updated traffic encryption key k_{TEK}.

The cluster generation phase follows the following three steps:

1. Nodes share their weights. Each node i calculates its weight w_i . It then broadcasts its *id* and w_i to its one-hop neighbours, and encrypts it with the hello key k_H . This provides confidentiality and, along with the group identity key, also provides authentication. The message is as follows.

$$E_{k_H}(w_i|id_i|E_{K_{GI}}(w_i|id_i))$$

2. After receiving the weighted messages from all its neighbours, node *i* will decide if it is a cluster head or cluster member. Once a role has been selected by node *i*, it broadcasts its role to its neighbours in the following message:

   $$E_{k_H}(w_i|id_i|role|E_{K_{GI}}(w_i|id_i|role))$$

   The *role* of node *i* is decided by its weight. The highest weighted node will broadcast a role of *ch*, cluster head, while other nodes will broadcast a role of id_j, where *j* is the identity of the cluster head that node *i* will belong to.
3. The cluster heads are then interconnected. All cluster members inform their cluster head of any other cluster heads within a three hop radius. The network is effectively segmented and clusters are interconnected by a cluster head backbone, as illustrated in figure 3-13.

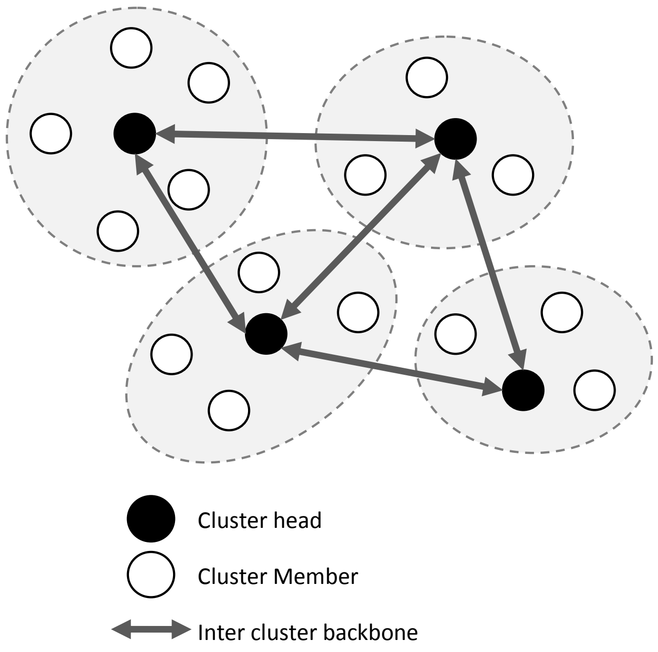


Figure 3-13: Segmented network with cluster backbone

3.8.4 Operation Phase

During the operational phase, the nodes use the group identity key k_{GI} to authenticate nodes and provide message integrity. The traffic encryption key k_{TEK} is used to encrypt the application data and provide message confidentiality. These services are provided using the cryptographic functions of symmetric encryption algorithms and the one-way hash function [46].

3.8.5 Key Update Phase

The traffic encryption key is updated periodically. This period is measured by an externally set parameter t_{update} (key update period). Updating occurs during the key update phase. Firstly, a key manager is selected from the pool of all the cluster heads. Selection is done by each cluster head that checks if it is a potential key manager by comparing its weight with the neighbouring cluster heads. Secondly, an exponential delay period, statistically averaged to Δ, is set aside, in order to minimize the risk of multiple nodes becoming key managers [46]. Thirdly, the cluster head with the highest weight value will arise as the selected key manager. The key manager's purpose is to generate a new traffic encryption key k_{TEK} and then distribute this to all the cluster heads, effectively updating the traffic key that provides message confidentiality. The new k_{TEK} is generated, k_{TEK} using a secure key generation algorithm. This new traffic key is distributed to the cluster heads securely using the backbone key k_B. The message sent to the cluster heads is:

$$E_{k_B}(w_c|id_c|k_{TEK}|E_{K_{GI}}(w_c|id_c|k_{TEK}))$$

Once the cluster heads have received the new traffic key, this is distributed to the cluster members using the cluster key k_c that is generated by the cluster head. The message sent to the cluster members is:

$$E_{k_C}(w_c|id_c|k_{TEK}|E_{K_{GI}}(w_c|id_c|k_{TEK}))$$

These three phases are repeated every network time-slice. The shorter this time-slice, the greater the security obtained. Similarly, this applies to the t_{update} period for the key update phase. However, in this case, it stands that the shorter the update period or time-slice, the more resources are required.

3.8.6 Analysis

This scheme is designed for large ad hoc networks that are made up of nodes with limited processing, power and storage capacity. Public key cryptography is unsuited for such a design, as this solution is realized through symmetric key cryptography. This solution requires a TTP to initialise the network nodes with the group identity key k_{GI} , as well asto set the parameters, such as the t_{update} time period.

The group identity key, which is distributed to all participating nodes, is required to remain secret throughout the lifetime of the network. In [46], the authors of the Secure Pebblenets solution propose that nodes have tamper-resistant storage that securely holds the group identity key. Standard network devices do not have such features and this limits its application for mobile ad hoc networks. If an attacker were to compromise the group identity key, all the nodes in the network would need to be re-initialized with a new group identity key, given by a TTP.

The clustering approach does benefit large ad hoc networks, as routing algorithms for long distances or large networks can become complex and expensive. Cluster-based communication allows for packets travelling long distances to travel via the cluster backbone, until they reach their desired neighbourhood or cluster. From there, the cluster head can transmit the packets more specifically. This approach reduces security computation and routing complexity in large networks.

A cluster head centralizes the authority in a network. In doing so, it provides a central point of attack for adversaries. Nodes within mobile ad hoc networks have unreliable characteristics because of their mobility and wireless sporadic connectivity. Selecting a reliable cluster head may become a problem in these dynamic networks. Nodes may also refuse to adopt the computational burden of being the cluster head. This is due to resource constraints inherent to mobile ad hoc networks.

Authentication is limited to groups to reduce computational requirements of nodes. It was found that if authentication was to be extended to the individual nodes, it would require the management of $n \times \frac{(n-1)}{2}$ symmetric keys [9]. Therefore, this solution is not feasible for peer-to-peer communication.

3.9 Proximity-based Identification

Smetters et al [47] proposes a solution called demonstrative identification . This solution allows nodes to establish initial trust relationships without prior knowledge or relationship and without the existence of an off-line TTP that most key management systems assume. This solution uses close proximity channels to establish initial bootstrapping and provides a basis for more complex key establishment. Demonstrative identification approach is designed for spontaneous, small, localized short term ad hoc networks. An example of such a network can be seen in the gathering of people in a coffee shop, where each person wishes to establish temporary communication network via their PDA's.

3.9.1 System Overview

Two nodes desiring to establish a secure communication link, initially engage across a location-limited channel. This channel is separate to the main communication channel, as displayed in figure 3-14. Location-limited channels include infrared, physical contact, and audio etc. Across the location-limited channel, pre-authentication information is exchanged. For example, a user with a PDA who wants to communicate with a second user's PDA can use an infrared channel. They can direct the PDA's infrared device towards the second device and an exchange is made. The user can be assured that the pre-authentication information is from the chosen PDA, due to the nature and characteristics of infrared communication.

After the user has exchanged the pre-authentication information, a two-party (for example Diffie Hellman) or group key exchange scheme can be implemented over the main communication channel. This is done in order to establish the keying material required for secure communication. A limited localized communication channel allows for communication without the existence of an off-line TTP or prior knowledge.

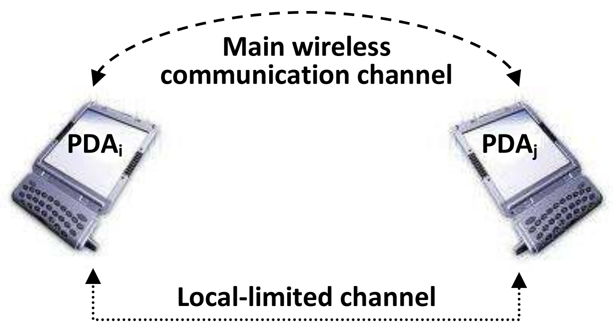


Figure 3-14: Proximity-based identification with location-limited channel

3.9.2 Two-Party Key Exchange

The key exchange between communication pair *i* and *j* is explained in the following steps:

1. Nodes *i* and *j* make close proximity contact with each other using a common location-limited channel.
2. Pre-authentication information is exchange across the common location-limited channel. Node *i* sends $h(K_i)$ to node *j* and *j* sends $h(K_j)$ to node *i*, where $h(K_j)$ is the irreversible one-way hash function of a node *j*'s public key.
3. Nodes *i* and *j* now exchange their public keys over the main channel such that *j* receives $\overline{K_i}$ and *i* receives $\overline{K_j}$. To avoid the impersonation attack that is common to mobile ad hoc networks, the public keys are then authenticated in step 4 using the pre-authentication information from step 2.
4. Authentication is checked using the one-way hash function *h* and verifies that $h(\overline{K_i}) = k(K_i)$ and $h(\overline{K_j}) = k(K_i)$.

5. Upon successful verification, any asymmetric key-exchange protocol can be implemented to allow for nodes *i* and *j* to share a secret key.

The two-party key-exchange described above is the basic formulae for demonstrative identification. This protocol can also be applied to heterogeneous nodes, where public key encryption is available to only one of the two communication members. This allows for nodes with limited complexity and computational capacity to participate in pair-wise secret key exchange. The procedure for a two-party key exchange, where only one of the members (node *i*) is the public key competent, is described as follows:

1. Nodes *i* and *j* make contact on a location-limited channel, allowing *i* to send *j*, $h(K_i)$ and *j* to send *i*, $h(S_j)$, where S_j is a secret from *j*.
2. Node *i* sends *j*, $\overline{K_i}$ over the main communication channel to realize authentication.
3. Node *j* authenticates node *i*'s public key, K_i, by verifying that $h(\overline{K_i}) = h(K_i)$.
4. Upon successful authentication, node *j* sends $E_{K_i}(S_j)$ to *i*.
5. $E_{K_i}(\overline{S_j})$ is decrypted at node *i* using K_i. $\overline{S_j}$ then verified by checking that $h(S_j) = h(\overline{S_j})$.

   Upon successful verification the two heterogeneous parties share a secret S_j that can be used to establish secure communication keying material.

3.9.3 Analysis

This solution allows for a fully self-configured ad hoc network, as the initial trust establishment phase does not require the assistance of an off-line TTP. Users realize the initial trust relationship by localized communication. For example, a user with a PDA would point its PDA to another PDA to automatically exchange authentication information and establish a secure communication line.

This solution requires that nodes are equipped with location-limited communication devices. Examples of these devices are infrared, audio or a wired link. This requirement limits the network participants to those possessing specific peripherals. The assumption is made that most portable wireless devices are equipped with some type of localized communication medium, such as infrared.

The location-limited pre-authentication exchange realizes demonstrative identification [47]. It only allows key-exchange to occur in a localized manner, where nodes are in close proximity to each other. As a result, this solution is not suited to large networks, and is best suited to small spontaneous networks. A solution presented by Capkun [10] extends the self-issued certificate chaining approach as it implements a demonstrative identification approach in a PGP-based network. Capkun's proposal uses location-limited communication to establish initial trust and relies upon mobility to distribute this trust in large networks. Such a proposal allows for demonstrative identification to be implemented in large-to-moderate networks.

More recently, the Amigo proximity-based authentication system proposed by Scannell et al [48], uses shared radio environment evidences as proof of physical proximity to authenticate localized mobile communication nodes.

3.10 Self Issued Certificate Chaining

A PGP-based security solution for ad hoc networks is proposed by Capkun and Hubaux [2, 49]. This solution uses a certificate chaining approach. It outlines a fully self-organized public key management system that allows users to generate their public-private key pairs, issue certificates, and perform authentication without the presence of an off-line trusted third party. Capkun and Hubaux focus on the key management and key distribution system. Without the need of prior relationships or an organizational TTP member, this solution is best suited to spontaneous ad hoc networks. However, due to its complex initialization phase, it is not suitable for small short-term networks.

3.10.1 System Overview

Public keys (K) and certificates are modelled as direct graphs $G(V,E)$ where vertices, V, represent the public keys and the edges, E, represent a certificate between two vertices. The self-organized system proposed by Capkun and Hubaux [2, 49] differs from PGP in that it relies on the users to store and distribute the certificates in a self-issued manner. Each user node carries a certificate memory consisting of certificates limited to local neighbourhood. For a user to authenticate and certify another user's public key, a certificate chain is first found between the two users by combining the users' certificate memory. Figure 3-15 illustrates a situation where node u and v request secure communication [2]. Node u is required to verify the authenticity of the public key K_v for corresponding to node v. To do so, nodes u and v combine their certificate memories to find a certificate chain or path between K_u and K_v that is made up of valid public key certificates shared between the two communicating nodes.

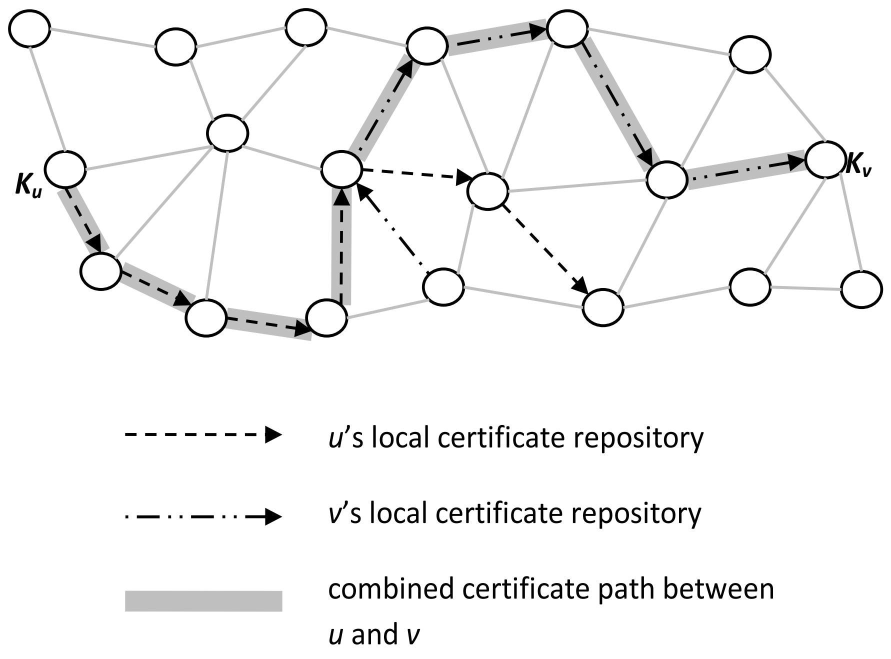

K_u
K_v
u's local certificate repository
v's local certificate repository
combined certificate path between u and v

Figure 3-15: A certificate chain or path between public keys K_u and K_v

The fully self-organized public key management system can be broken into four procedures of analysis, as follows:

- Public/private key creation
- Certificate exchange
- Authentication
- Certificate revocation
- Load sharing

During the initialization phase, the public-private keys are created and distributed with a certificate exchange procedure. Secure communication is realized and impersonation attacks are thwarted by the authentication of the available certificates. The certificate revocation protocol is outlined in order to maintain security and exclude malicious users. Optimization is implemented by a load sharing protocol that ensures fair distribution of the work load and prevents selfish nodes in a network. The initialization phase is executed in a four step procedure that establishes trust in the network, as follows:

1. The user creates a personal public/private key pair.
2. The user issues public key certificates (vertices) based on the knowledge of the other public keys.
3. The user performs certificate exchange and collecting certificates, and creates a non-updated certificate repository.

4. The user constructs an updated certificate repository, modelled as a graph G_u. This is done by communicating with certificate graph neighbours or by a second method of applying the repository construction algorithm to the non-updated certificate repository.

After initialization is complete, authentication between two users can take place through certificate chaining. Each step is explained in more detail below.

3.10.2 Public/private Key Creation

Public and private keys for users are created locally. Public key certificates are issued by the user. If the user *u* believes that a public key K_v belongs to *v*, then the user *u* can issue a public key binding K_v to user *v*, by the signature of *u*. This certificate has an expiry time T_v. A periodic update may be issued that simply extends expiry time T_v. The reason for trust is not identified but assumed, for example through a physical side channel.

3.10.3 Certificate exchange

The certificate repositories are created automatically by exchanging certificates. A user *u* has two certificate repositories, namely: an update certificate repository G_u and a non-updated certificate repository G_u^N. All certificates are stored twice, as when a certificate is issued, it is stored in both the certificate issuer *u* and certificate owner *v'*s repository. Therefore, initially each certificate repository has only the certificates it has issued and those that have been issued to it. Certificates are exchange periodically. Each node periodically polls its physical neighbour for certificates.

A certificate exchange is performed by the following procedure:

1. Node *u* broadcasts G_u and G_u^N to its physical neighbours. The broadcast contains only identities (hash values).
2. Neighbours reply with identities of their update repository G and non-update repository G^N.
3. Node *u* crosschecks the received sub-graphs and its sub-graphs for any additions.
4. Node *u* requests those certificates it does not hold.

After the initial convergence phase, all the certificates of the nodes are stored by all users. As a result, users' non-update repositories are created. After this phase the nodes exchange only new certificates at a rate of T_{CE} , which represents the time for a certificate to be exchanged throughout the network. Note that certificate expiration times are not considered thus far.

3.10.4 Construction of updated certificate repositories

The exchange of certificates provides an incomplete view of the graph and allows each node to create its own non-updated certificate repository. The updated repository G_u will consist of certificates that user *u* keeps updated. There are two approaches in this creation, namely:

1. Apply algorithm *A* to G_u^N that results in G_u , and validity of each certificate is checked.
2. Communicate with certificate graph neighbours only.

The maximum degree algorithm is an algorithm *A* proposed by [2] that is applied to the non-update repository G_u^N to create the update repository G_u in [2, 49]. The algorithm selects a sub-graph that consists of two logically distinct paths: the out-bound path and the in-bound path, that are made up of outgoing edges and incoming edges, respectively. The selection of G_u's out-bound path is done in multiply rounds in the following manner [2, 49]:

1. Each round runs from vertex K_{vert}, starting with vertex K_u.
2. User *u* requests the outgoing edge list of vertex K_{vert}. This is possible as every vertex stores this list locally.
3. An outgoing edge (with its terminating vertex *z*) is selected from the list in 2. Selection is based on the highest number of shortcuts of the terminating vertex *z*. Where a shortcut is defined as an edge and removed, the shortest indirect path between the nodes previously connected by that edge becomes larger than two. User *u* can determine its number of shortcuts by gathering information about the outgoing and incoming edges of its adjacent users.
4. The selected vertex *z* is added to a set N_{out} of vertices selected thus far. This is done to ensure that the selected out-bound paths are disjointed.
5. The round is finished and now the terminating vertex *z* becomes K_{vert} and a new round begins, starting from step 1.

The in-bound path selection is done in a similar way, as follows:

1. Each round runs from vertex K_{vert} , starting with vertex K_u.
2. User *u* requests the incoming edge list of vertex K_{vert}. Every vertex stores this list locally. Therefore, this step requires that each user be notified whenever another user issues a certificate to that user.
3. An incoming edge (with its originating vertex *y*) is selected from the list in 2. Selection is based on the highest number of shortcuts of the originating vertex *y*.
4. The selected vertex *y* is added to a set N_{in} of vertices selected so far, to ensure that the selected in-bound paths are disjointed.

5. The round is finish and now the originating vertex *y* becomes K_{vert} and a new round begins, starting from step 1.

The update repository is the union of the inbound sub-graph and outbound sub-graph. The pure method will operate on a single round. However, it is extended so the update repository consists of several vertex disjoint out-bound and vertex disjoint in-bound paths. The final sub-graph is star-like in formation.

3.10.5 Authentication

When initialization is complete, the user is prepared to perform authentication. Authentication is preformed between users *u* and *v* with public keys K_u and K_v respectively, as follows:

Firstly, user *u* and user *v* merge their update certificate repository (G_u and G_v) to find a certificate chain between *u* and *v*. User *u* then looks for a path in G_u and G_v. Validity and correctness checks are done to all certificates in the discovered path. Validity checks that the certificates are not revoked. Correctness checks that the certificates contain the correct user-key bindings.

If no certificate chain is found, the user *u* combines its two repositories of the updated and non-updated certificates to find a chain. User *u* searches for a path in G_u and G_u^N. If a chain is found, then *u* requests the updates of the expired certificates. Subsequently, the validity and correctness checks are made.

If there is still no certificate chain found between K_u and K_v then authentication is aborted. During authentication, nodes that are one-hop physical neighbours (also known as helper nodes) are given precedence as to maximize performance. When a path is found, the certificates (edges) along this path are then used by user *u* to authenticate K_v.

3.10.6 Certificate revocation

Certificates are revoked when it is believed that the user-key binding is no longer valid. If users believe their own private keys are compromised, then they can revoke their public key certificate binding. This is done in two ways, explicitly and implicitly:

1. Explicitly, a user *u* would revoke a certificate issued by *u,* by broadcasting a revoke statement to its G_u nodes. The certificate exchange scheme allows for this revoke to reach all other nodes at a time delay of T_{CE}.
2. Implicit revocation is based on the expiration of certificates. Certificates are valid for a given time T_v after which they must be updated.

This allows for comprised certificates and private keys to be dealt with explicitly, and provides a higher level of confidence by implicitly maintaining validity.

The fully distributive nature of this scheme means that every certificate is stored at each node, allowing for nodes to cross-check conflict and detect inconsistent certificates.

To combat false certificate bindings, the following two procedures are taken:

1. If a certificate is received that doesn't exist in G_u or G_u^N , then it and the issuer are labelled *unspecified* until a period T_p where $T_p > T_{CE}$, after which if no conflicting certificates are received, it is marked *non-conflicting*. This procedure is not a preventative against Sybil Attacks.
2. If a certificate conflict is found where a user u has two certificate bindings (v,K_v) and (v,K'_v), both certificates and the certificates that certified them are labelled as *conflicting*. To resolve such a conflict, the validity of certificates is first checked with their issuers. If validity status remains true, then u will try to find chains of non-conflicting valid certificates to public keys K_v and K'_v. Confidence values are calculated based on the number and length of chains, and values compared to compute the correctness of the bindings. If no decision is made these bindings are labelled as *conflicting* and the node waits for more information to resolve the conflict.

In this case, a confidence algorithm is not identified but assumed. This conflict resolution mechanism can be further used: to evaluate trust in users, as well as to issue correct certificates and to detect malicious users.

3.10.7 Load Sharing

For an update to occur, nodes contact the issuer of the certificates that they store. This approach is not efficient because one certificate issuer could be overloaded and consequently be unable to handle the computational workload. Simple load sharing can be implemented to allow for relief. Each node u provides updates to up to s other nodes, where s is equal to size of u's updated repository. After node u has provided s updates, it replies to update requests with a list of nodes that get updates directly from u. The requesting node then randomly selects a node from u's list and requests its update from that node.

3.10.8 Analysis

The self organized, self certificate issuing trust model is a web of trust type models inheriting PGP characteristics and applying them to an ad hoc network environment. In a similar way that PGP [3] realizes trust, the certificate chaining approach is used to create chains of hierarchical trust between users.. The main difference between PGP and the certificate chaining solution is that PGP stores certificates in a centralized manner, and this scheme decentralizes this procedure through local certificate repositories.

The main advantage of this scheme is that it is fully self-organized and does not require the presence of a TTP. Trust is established in a self-organised manner with self-certificates being issued by the nodes themselves. The initial phase requires nodes to interact and establish trust. Trust relationship can take time to establish. Therefore, in the early stages of the network, an initial time delay can be expected, limiting the effectiveness of communication. For this reason, this network is not suited for short term mobile ad hoc networks. An example of this shortcoming is illustrated in figure 3-16, where node *A* wants to communicate with node *B*. At the early stage of the network, only *D* and *C* have issued certificates and as a result no certificate chain exists between *A* and *B*. Only once the intermediate nodes have issued certificates, will a certificate chain between *A* and *B* be possible.

The use of certificate chains is identified as vulnerable because a chain of trust is 'only as strong as its weakest link'. A PGP hierarchical trust model is adopted that assumes transitive trust. This web-of-trust based approach allows for more flexibility than the other certificate approaches. However, a no central administration is present to enforce policy and trust assessment. Therefore, because of this lack of structure, it is more prone to attacks by malicious nodes. This solution is best suited to open mobile ad hoc networks, but may not be suited to applications where high degrees of security are required [32], such as closed military mobile ad hoc networks.

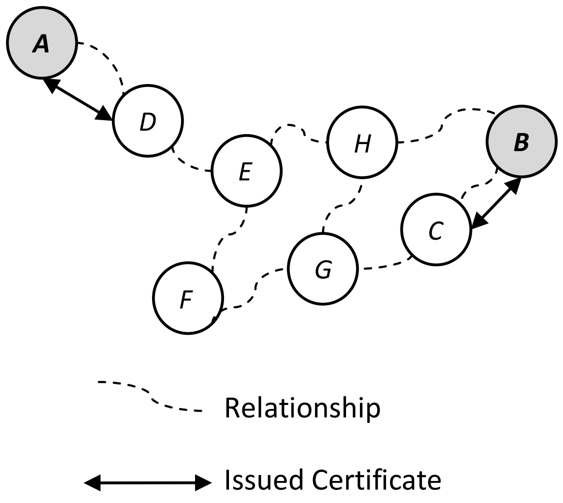


Figure 3-16 Initial phase delay problem

This self-organized scheme is fully distributive and would result in a certificate update being computationally taxing. Certificate update repositories and load sharing relieve this expense.

However, better load balancing data management schemes can introduced to further relieve the load [49].

The maximum degree algorithm *A* (or Shortcut Hunter Algorithm) is implemented to maximise effectiveness and optimise the update procedure. This proposal has been tested on PGP trust graphs. Nevertheless, an ad hoc network does not have the privilege of every node having public knowledge of all the certificates available. Step 3 of the maximum degree algorithm requires that an edge is selected from *vert* to *z*, where *z* is the vertex with the highest number of shortcuts. To determine *z,* knowledge of the surrounding trust graph is required, and this may not be available to all ad hoc network members.

One of the main disadvantage of a fully self-organized model is that nodes can adopt as many identities as they have resources, in order to support [2] further steps that need to be taken to protect this solution from Sybil and impersonation type attacks.

3.11 Discussion and Summary

The solutions presented in this chapter provide a summary of the work related to key management in mobile ad hoc networks. Surveys of key management solutions exist such as [50, 51], but the majority of literature fails to identify a criteria on which key management solutions can be compared. The solutions differ considerably in requirements, complexity and functionality. Each solution is suited to different types of ad hoc network environments. Criteria by which these key management solutions can be grouped or differentiated include:

- Pre-configuration: *Planned vs Spontaneous*

This criterion describes the pre-requisites and assumptions that are made for the nodes participating or joining the network. If an ad hoc network is planned, then nodes can be assumed to have some pre-configured information, such as: initial shared secret, certificate or authenticated identification. If the network is spontaneous, then nodes have no prior security relationships or initial data assumptions. Pure ad hoc networks are more spontaneous, allowing for nodes to join and leave the network without complex pre-configurations and assumptions being made.

- Network Area: *Local vs Distributive*

This criterion describes the area or space in which the key management scheme is operating. The physical topology of the network would result in more close proximity interaction or more multi-hop distributive interaction. A localized area is a network in which nodes come within a close proximity range of each other, such as in a classroom. A distributive area is a network where nodes are located some distance apart with little possibility of physical interaction. Certain key management schemes do not function in a distributive network area.

- Network Duration: *Short Term vs Long Term*

The duration of the network can dictate the initialization period of the key management scheme. For short term ad hoc networks, a group of nodes establish communication for a short time period and may never come into contact again. Short term ad hoc networks require speedy initialization and require communication to be available at the start of the network, without an initial period of weakened or delayed secure communication. Long term ad hoc networks consist of nodes that plan to be part of a network and in relationship with other nodes for a longer time period. Furthermore, nodes retain information and relationships with other nodes even when they leave the network. Long term ad hoc networks require more complex trust establishment.

- Off-line TTP Involvement

Ad hoc networks are characterized by their lack of infrastructure. Key management schemes often rely on an off-line trusted third party (TTP) for initialization and operational security. The extent of the off-line TTP involvement describes the self-organized nature of the network. Ideally, an ad hoc network has no off-line TTP involvement at the initialization or operational stages.

A summary of the presented key management solutions is provided in Table 3-1 in reference to the criteria discussed above. The off-line TTP model relies on an external TTP to establish and maintain security. This model is suited for networks that have available fixed infrastructure and will therefore have limited mobility. The partially and fully distributive CA solutions are similar, using threshold cryptography as they distribute the hierarchical trust of a certificate authority. They are suited to large planned ad hoc networks like military battlefield networks or disaster area networks. The Secure Pebblenet scheme is a cluster-based model that is ideal for hierarchical group-oriented ad hoc networks where all nodes are distributed in a large network area and nodes have limited resources. An application of this cluster-based approach is sensor networks.

The Self-Issued Certificate model or certificate-chaining model uses a localized PGP web-of-trust approach. Its self-organized nature makes this solution most suited to spontaneous networks, such as peer-to-peer communication in a classroom or coffee shop. The proximity-based identification solution is suited to localized networks. Its greatest advantage is that it requires no prior knowledge to establish trust. The proximity-based identification method, as used in Capkun's mobility-based approach, uses mobility of nodes to establish initial trust relationships across a large network.

This chapter shows that many of the solutions presented have issues that need to be resolved. Key management is an integral part of providing security and (as identified in Chapter 1) the routing layer is the focus of attack for adversaries. If these MANETS are to be recognized as secure, then mobile ad hoc network's security mechanism must strive to provide security on the routing and application layer.

Table 3-1: Summary of Key Management Solutions

	Pre-Configuration	***Network Duration***	***Network Area***	***Off-line TTP Involvement***
Off-line TTP Model	Planned	Long-term	Distributive	Full
Partially Distributed CA	Planned	Long-term	Distributive	Initialization
Fully Distributed CA	Planned	Long-term	Distributive	Initialization
Self Issued Certificates	Spontaneous	Long-term	Distributive	None
Cluster-based Model	Planned	Long-term	Distributive	Initialization
Proximity-base Identification	Spontaneous	Short-term	Localized	None

Chapter 4

Secure Routing in Mobile Ad Hoc Networks

4.1 Introduction

A mobile ad hoc network's routing protocol has unique challenges due to the dynamic nature of ad hoc networks. Mobile ad hoc networks do not have the same privileges as fixed, wired networks. The routing mechanisms are uniquely designed to deal with the lack of infrastructure and unreliable wireless multi-hop communication channels.

This chapter investigates the procedure of securing these routing protocols. The routing solutions are briefly visited, and an extensive survey is presented for the existing security mechanisms that are used to secure these routing protocols.

Routing in mobile ad hoc networks is divided into two categories: table driven methods and on-demand methods. Table driven methods are also known as proactive routing. They maintain routing tables that contain routes to all the nodes in the network. Theses tables are periodically updated thereby allowing routing information to be available at all times. Examples of table methods include Destination Sequence Distance Vector Routing (DSDV) [22] and Optimized Link State Routing (OLSR). Source initiated on-demand routing methods establish routes in a reactive manner. Routes are established through a route discovery phase. During a route discovery phase, node *S* will broadcast a request message *RREQ* into the network. This request message is forwarded until it reaches its target destination node *D.* Node *D* then replies with a reply message *RREP* that is unicast along the reverse route, until it reaches the source and the route is established. Routes are maintained as long as they are required. Examples of on-demand methods include Ad Hoc On-Demand Distance Vector (AODV) [21] and Dynamic Source Routing protocol (DSR) [52]. The reactive on-demand approach is less computationally expensive, in comparison with the proactive table-driven approach.

In chapter 2 it was identified that most security attacks target the network layer, and more specifically the routing protocol. These attacks include black-hole attacks, wormhole attacks, eavesdropping attacks, byzantine attacks, resource consumption attacks, and routing table poisoning. The routing protocol is found on the network layer and is a significant service for mobile ad hoc networking. Adversaries, specifically, target the routing protocol. Thus, a secure routing solution is needed for ad hoc networks to be securely implemented.

This chapter focuses on a survey and an analysis of the existing secure routing protocols. Secure routing surveys exist such as [53, 54], but fail to focus upon the facet of key management. Each protocol is presented and investigated based on functionality, operational assumptions, and security. A summary and discussion of these aspects is provided at the close of this chapter focusing upon the aspects that influence key management.

4.2 Secure Efficient Ad Hoc Distance Vector Routing Protocol (SEAD)

Secure efficient ad hoc distance vector (SEAD) [55] is a secure routing protocol that is used in conjunction with the table driven destination-sequenced distance vector (DSDV) routing protocol [22]. The DSDV routing protocol uses a distributed version of the Bellman-Ford algorithm to discover the shortest path between two nodes. The SEAD protocol uses symmetric key cryptography and one-way hash functions to protect against security attacks such as denial of service and resource attacks.

4.2.1 System Overview

The DSDV routing protocol discovers the shortest path based on a route's hop count. Routing packets are assigned sequence numbers to ensure the most recent route is processed. The hop count and sequence number variables are stored in the routing packets. Attackers can create an incorrect routing state in nodes resulting in a denial of service attack (DoS) where the attacker attempts to make other nodes consume excess bandwidth and processing time. SEAD makes the routing process robust against multiple uncoordinated attackers by authenticating the hop count and sequence number of routing packets with a one-way hash function h. Hash chaining is used so that only nodes that are in possession of the previous routing update (identified by a sequence number) can broadcast a new routing update. Authenticated routing updates are computed to safeguard against malicious routing updates broadcast by attackers.

4.2.2 One-Way Hash Function

SEAD uses a one-way hash function to authenticate routing updates and minimize resource consumption attacks. A formal definition of the hash function H is provided in [15]. The most commonly used hash functions are MD-5 [56] and SHA-1 [57].

A one-way hash function *H* is used to generate a one-way hash chain *h*. The one-way hash function *H* has an input of any bit length * and outputs a variable of fixed bit length *p*. The one-way hash function *H* must be computationally impossible to invert.

$$H: (0,1)^* \rightarrow (0,1)^p$$

A hash chain h_i is created when a node selects a random number $x \in (0,1)^p$ and uses it to generate a list of variables that make up a hash chain $h_0, h_1, h_2, h_3, \ldots, h_n$. Here $h_0 = x$ and h_i is calculated using the irreversible one-way hash function *H* such that:

$$h_i = H(h_{i-1}) \text{ where } 0 \leq i \leq n$$

Assuming there is an existing authenticated element, a node can verify elements later in the chain's sequence. For example, if an authenticated element h_i exists, a node can authenticate h_{i-4} by checking that $h_i = H(H(H(H(h_{i-4}))))$. SEAD assumes the existence of an authentication and key distribution mechanism to distribute an authenticated element such as h_n allowing for authentication by hash chaining [55].

4.2.3 Authenticating routing updates

SEAD uses the elements of the hash chain to provide authentication and secure the routing updates in DSDV. SEAD assumes an upper bound on the variable to be authenticated. For example if it were the hop count, then SEAD would assume a maximum route distance *n* in the network (the maximum hop count between two nodes allowed). This also eliminates any routes with a length greater than *m* to exist, eliminating possible routing loops or the routing infinite problem.

The sequence values that make up the hash chain are calculated from the *H* function such that $h_1, h_2, \ldots, h_n$ where *n* is divisible by *m*. For a routing table entry with sequence number *i* , let $k = {}^{n}/_{m} - i$. An element from $h_{km}, h_{km+1}, \ldots, h_{km+m-1}$ is used to authenticate the routing entry with sequence number *i*. If the hop count is *j* where $0 \leq j < m$, then h_{km+j} is used to authenticate the routing entry found with sequence number *i* and hop count *j* [55].

Routing updates are sent with the appropriate routing information, and a hash chain value is used to authenticate the update. If the authentication value appended is h_{km+j} , then only attackers with h_{km+j-1} can modify the authentication value. Nodes receiving a routing update, check the authentication value h_{km+j} by calculating the new hash chain value. Receiving nodes can calculate the new hash chain value by using the earlier hash chain value h_{km+j-1} and the received sequence number i and hop count j. If the new calculated hash value is equal to h_{km+j} , then the routing update is verified.

SEAD proposes two methods for routing update authentication. One method uses clock synchronization and a broadcast authentication mechanism such as TESLA [58]. The second method requires a shared secret between each communicating node pair. The secret can be used to implement a message authentication code (MAC) between nodes authenticating routing update messages.

4.2.4 Analysis

The SEAD protocol protects the ad hoc network from routing attacks that target resource consumption. The SEAD protocol does protect against multiple uncooperative attacks. While it has the capacity for preventing routing loops, its routing loop prevention cannot be guaranteed in the presence of co-operating attackers. The SEAD protocol is vulnerable to intelligent attackers that use the same sequence number and same hop count of the most recent update to corrupt routing information. The SEAD protocol provides protection against denial of service attacks, replay attacks and routing table poisoning by authenticating routing updates so malicious nodes cannot corrupt the routing procedure.

4.3 A secure on-demand routing protocol for ad hoc networks (Ariadne)

Ariadne [59] is a secure on-demand routing protocol that uses symmetric cryptography. Ariadne is based on the on-demand DSR [52] routing protocol and is developed by the same authors as the SEAD protocol [55]. Ariadne provides end-to-end authentication on the routing layer.

4.3.1 System Overview

Ariadne assumes a shared secret key between communicating node pairs and uses message authentication code (MAC) to authenticate end-to-end packets between the communication pair. Broadcast authentication is employed with loose time synchronization to authenticate route request and other broadcast packets. The TESLA [58] broadcast authentication scheme is used. In TESLA the source generates a one-way key chain, and a schedule is made that defines at which time keys of the chain are revealed. This mechanism limits Ariadne's operation to ad hoc networks that have time synchronization. Ariadne provides end-to-end authentication in an on-demand manner over the DSR routing protocol [59].

4.3.2 End-to-end Authentication

For communication from a source node *S* to a destination node *D*, the source *S* will broadcast a route request into the network and expect a reply from *D*. Ariadne assumes a shared secret between *S* and *D*, K_{SD} and K_{DS}, that enables message authentication for each respective direction.

Nodes *S* wanting to start a route discovery for node *D* will first generate an initial hash chain h_0 consisting of: a packet identifier identifying the type of packet (a request packet *RREQ* in this case), the source's address (ID_S), the destinations address (ID_D), a broadcast identity (*bi*) identifying the current route discovery, and a TESLA time interval (*tes*) identifying the expected time of arrival at the destination.

$$h_0 = MAC_{K_{SD}}(RREQ|ID_S|ID_D|bi|tes)$$

Node *S* will broadcast a route request packet that includes a packet identifier, the hash chain h_0, the source's address (ID_S), the destinations address (ID_D), the broadcast identity (*bi*), the TESLA time interval (*tes*), a node list *N()* and a MAC list *M()*. The packet broadcast is as follows:

$$S \rightarrow broadcast\ :\ RREQ|h_0|ID_S|ID_D|bi|tes|N()|M()$$

A neighbouring node that receives the route request checks the validity of the TESLA time interval, *tes*. The TESLA time interval is valid if the corresponding key that it points to has not been revealed yet and the time interval does not point too far in the future. The neighbouring node *A* will then compute a new hash chain h_1 using the previous hash chain h_0. A message authentication code of the packet to be broadcast is created (MAC_A). MAC_A is calculated using the TESLA key (K_{Ates}). Before forwarding the packet, the neighbour node *A* includes the hash chain h_1, itself in the node list *N*, and the MAC_A calculated in the MAC list *M*. The hash function and broadcast packet are as follows:

$$h_1 = H(A|h_0)$$

$$A \rightarrow broadcast\ :\ RREQ|h_1|ID_S|ID_D|bi|tes|N(A)|M(MAC_A)$$

Intermediate node *P* receiving a forwarded route request, first calculates a new message authentication code MAC_P and a new hash chain $h_i = H(P-1|h_{i-1})$ where *P*-1 is the previous node and h_{i-1} is the previous hash chain value. Secondly, it includes this information and forwards the route request as follows:

$$P \rightarrow broadcast : \; RREQ|h_i|ID_S|ID_D|bi|tes|N(A, \ldots, P)|M(MAC_A, \ldots, MAC_P)$$

The route request is propagated to the destination node *D*. When *D* receives the route request it validates the authenticity of the route request by checking that the TESLA time intervals indicate that no keys have been released as of yet and that the hash chain is valid. *D* then generates a message authentication code MAC_D. MAC_D and an empty key list *K()* are included in the packet and sent back along the reverse path indicated by the node list and DSR protocol. The MAC_D and reply message are as follows:

$$MAC_D = MAC_{K_{DS}}(RREP|ID_D|ID_S|bi|tes|N(\ldots)|, M(\ldots)$$

$$D \rightarrow P : \; RREP|ID_D|ID_S|bi|tes|N(\ldots)|M(\ldots)|MAC_D|K()$$

Intermediate nodes that receive a reply message will wait for the *tes* time interval to lapse so the corresponding key can be revealed and included in the key list *K()*. The reply message is forwarded until it contains all the TESLA keys of the intermediate nodes, and it finally reaches the source node *S*. The source then verifies the validity of all the keys, and the message authentication code, MAC_D..

4.3.3 Maintenance

Ariadne achieves secure route maintenance by authenticating the DSR error messages. Ariadne authenticates error messages preventing malicious nodes from broadcasting false broken links and causing denial of service type attacks. When an error message is generated, TESLA authentication information is included. If authentication is delayed as a result of the TESLA time intervals, the intermediate nodes buffer the error message until the appropriate keys are revealed and the message can be authenticated and action taken [59].

4.3.4 Analysis

The Ariadne proposal is proposed by the same authors as the SEAD [55] protocol. Ariadne employs an end-to-end technique to authentication, while SEAD uses a hop-by-hop technique because of the DSDV routing procedure. The Ariadne proposal is based on the on-demand DSR routing protocol. Ariadne implements TESLA broadcast authentication and message authentication code to provide authentication for routing packets in an ad hoc network environment. The Ariadne proposal assumes the existence of some shared secret between a communication pair, therefore assuming the existence of an authentication and key distribution mechanism. Ariadne relies on TESLA authentication which requires time synchronization in the ad hoc network. Synchronization is difficult to achieve without the presence of an outside authorized member or TTP.

Ariadne implements end-to-end authentication to prevent unauthorized nodes from sending error messages and incorrect routing packets in the form of repays attacks. However this proposal does not consider the case where attackers do not cooperate with the routing protocol and drop routing packets that are supposed to be forwarded. An extension is proposed in [60] that uses packet leashing to solve this problem.

4.4 Authenticated Routing for Ad Hoc Networks

The authenticated routing for ad hoc networks (ARAN) protocol [61] is a securing routing solution that uses cryptographic certificates. ARAN is designed for an on-demand ad hoc routing protocol and achieves authentication, integrity and non-repudiation on the network layer but assumes prior shared secrets at initialization.

4.4.1 System Overview

The ARAN secure routing protocol establishes trust in three stages:

1. Issuing of certificates
2. Route Discover process
3. Shortest path Optimization

Initially, ARAN assumes the presence of a trusted third party (TTP) that issues valid certificates, and a shared public key for all participating nodes. The route discovery process of ARAN provides end-to-end authentication for communicating nodes. The source node broadcasts a route request which carries the source's certificate. The route request is propagated to the destination node by an end-to-end authentication process. The destination node responds by unicasting a reply message back along the found route using the end–to-end authentication protocol.

4.4.2 Issuing of Certificates

This section describes how the certificates are issued and distributed to the participating nodes. The assumption is made that an authenticated trusted third party (TTP) member exists and plays the role of an initial certificate authority (CA). This TTP CA is known to all the nodes in the network. The ARAN protocol assumes that certificates are generated by the TTP CA and distributed to nodes before they officially join the wireless ad hoc network. No specific key distribution mechanism is described for the ARAN protocol. Node *i* entering the network will receive a certificate $cert_i$ from the TTP CA that has the following contents:

$$TTP - CA \rightarrow i \quad : \quad cert_i = E_{k_{TTP-CA}}(ID_i|K_i|t|et)$$

The certificate *cert*$_i$ is signed by the private key of the TTP-CA ($k_{CA\text{-}TTP}$), and has the following contents: ID_i representing the identification of node *i* , for example a specific IP address; K_i the public key of node *i*; *t* the time stamp for the *cert*$_i$; and *et:* the expiry time of the certificate.

4.4.3 Route Discovery Process

The route discovery process provides end-to-end authentication that ensures that the packets sent from a source node *S* reach their intended destination node *D*. Each node maintains a routing table that contains the active communication routes between the different source and destination pairs. The route discovery process begins with a source *S* broadcasting a route request. The route request is signed by the source node's private key k_S and contains the certificate of the source node (*cert*$_S$), the identification of the destination node (ID_D), a nonce (N_S), a timestamp (*t*), and a packet identifier identifying that the packet is a route request packet (*RREQ*). The authenticated route request broadcast by node *S* is:

$$S \rightarrow broadcast \quad : \quad E_{k_S}(cert_S|ID_D|N_S|t|RREQ)$$

The nonce value is incremented every time the source sends a route request. The nonce value acts like a sequence number ensuring that the most recent route request is dealt with. Each node that receives the route request will process it if it has a higher value of the source's nonce than previously received route requests from the same source node. Each intermediate node *P* receiving the route request will validate the signature with the certificate, update the routing table with the neighbour from which it received the route request, sign the route request, and broadcast it to its neighbours. Node *P* will remove the signature and certificate of the previous node if the previous node was not the source itself. Hence each forwarded route request is authenticated by the source, and the intermediated node and will contain two certificates *cert*$_S$ and *cert*$_P$:

$$P \rightarrow broadcast \quad : \quad cert_P|E_{k_P}(E_{k_S}(cert_S|ID_D|N_S|t|RREQ)$$

The route request is propagated to the destination node *D* that will reply with a reply message *RREP*. The reply packet is signed by the destination node's private key k_D and the packet contains the identity of the source node (ID_S), the destination's certificate (*cert*$_D$), a nonce of validity (N_D), a timestamp (*t*), and a packet identifier (*RREP*). The reply packet is unicast along the reverse path toward the source node with a similar authentication procedure to the forwarding of the route request.

$$D \rightarrow reverse\ path \ : \ E_{k_D}(cert_D|ID_S|N_D|t|RREP)$$

$$P \rightarrow reverse\ path \ : \ cert_P|E_{k_P}(E_{k_D}(cert_D|ID_S|N_D|t|RREP)$$

The source node will receive the reply packet *RREP* and check the signature and nonce (N_D) to verify that the packet was sent by the destination node and not a malicious attacker. If the nonce or certificate fails, an error message is broadcast and the route request process restarted.

4.4.4 Shortest Path Confirmation

This is an optional procedure employed by ARAN to ensure that the shortest path is found between source and destination. Path confirmation has a high computational cost. After a route has been found between *S* and *D*, the shortest path confirmation process begins. The source will broadcast a packet signed by the public key of *D* (K_D) containing: the certificate of the source, the identity of the destination node, a nonce (N_S), timestamp (*t*), and packet identifier identifying that this is a shortest path confirmation packet (*SPC*).

$$S \rightarrow braodcast \ : \ E_{K_D}(cert_S|ID_D|N_S|t|SPC)$$

Each intermediate node that receives the *SPC* packet updates its routing table, signs the packet, includes its certificate, and signs it with the public key of the destination node.

$$P_1 \rightarrow braodcast \ : \ E_{K_D}(cert_{P_1}|E_{k_{P_1}}(E_{K_D}(cert_S|ID_D|N_S|t|SPC)$$

The destination node will verify all the signatures and reply to the first and subsequent *SPC* packets with a recorded shortest path packet *RSP*. The *RSP* is propagated to the source which confirms the shortest path by verifying the nonce N_S sent with the *SPC* packet.

4.4.5 Maintenance

The ARAN solution uses error messages and implicit revocation of certificates to maintain routes. Error message packets (*ERR*) are broadcast by any node *P* that discovers a broken route. An *ERR* packet is signed by its originator and includes the certificate of the originator, the source and destination pair describing the broken route, a nonce, a timestamp, and a packet identifier. Each node receiving an ERR packet will check its routing table if it contains the accused route. If it does, then the *ERR* packet is rebroadcast unchanged.

$$P \rightarrow braodcast \ : \ E_{k_P}(cert_P|ID_S|ID_D|N_P|t|ERR)$$

The expiration (*et*) attribute included in each certificate allows for implicit revocation of certificates. Certificates are implicitly checked during the route discovery process. Explicit revocation is achieved by the TTP CA broadcasting a certificate revocation message to nodes that can then forward it. Routes are re-calculated as a result of certificate revocation.

4.4.6 Analysis

The ARAN solution uses asymmetric key cryptography to provide authentication, integrity and non-reputation. Asymmetric cryptography will result in high complexity and computational cost. A trusted certificate authority (TTP CA) is required so that authentication can be made available. In the route discovery process, unlike AODV, ARAN disallows intermediate nodes that have a path to the destination to reply with a *RREP* message. This creates additional routing overheads but ensures authentication [61].

4.5 Secure Ad hoc On-demand Distance Vector (SAODV)

Zapata et al [62] proposes the Secure Ad hoc On-demand Distance Vector (SAODV) protocol as an security extension to the AODV protocol. SAODV secures the AODV protocol by using a hybrid cryptographic approach involving asymmetric cryptography in the form of digital signatures and symmetric cryptography in the form of hash chains.

4.5.1 System Overview

SAODV defines the fields in a routing packet according to two categories: mutable and non-mutable. The non-mutable fields are authenticated using asymmetric cryptographic signatures. The only mutable field in an AODV routing packet is the hop count. A new hash chain is created for each route discovery phase that is used to secure the hop count. SAODV requires that the AODV routing packet is extended to include security information such as digital certificates. The implementation of digital signatures and hash chains provides end-to-end authentication for the AODV routing protocol.

SAODV uses asymmetric cryptography and assumes the presence of a key management scheme to distribute keys in a secure manner [62]. It also assumes that it is possible to verify the relationship between a public key and an IP address or identity.

4.5.2 Packet Extension

SAODV proposes an extension to the standard AODV message format so that security mechanism can be implemented. The SAODV extension contains the following fields as described in Table 4-1 and Figure 4-1 [62].

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     | Hash Function | Max Hop Count |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Top Hash                           |
 ...                                                         ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Signature                          |
 ...                                                         ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              Hash                             |
 ...                                                         ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4-1: RREQ Single Signature Extension

Table 4-1: RREQ and RREP Signature Extension Fields

Field	Description
Type	This is a packet identifier where the value 64 identifies a request packet *RREQ* and the value 65 identifies a reply packet *RREP*.
Length	The length of the packet data.
Hash Function	This describes the hash function used for example MD5 [56] or SHA-1 [57].
Max Hop Count	The maximum hop count (*mhc*) is used for hop count authentication. It defines the maximum number of nodes a packet is allowed to pass through.
Top Hash	The top hash is the result of the hash function *H* applied *mhc* times to a random generated number *x* such that: $top\ hash = H^{mhc}(x)$. Top Hash is vital in the hop count authentication process.
Signature	This field is 32-bit aligned and contains the signature used to authenticate all the non-mutable fields in the AODV packet.
Hash	This field is 32-bit aligned and contains the hash value h_i used to authenticate the mutable hop count variable.

The standard AODV protocol uses packet sizes of 512 bytes but the SAODV extension requires the AODV packet size to be extended to use packets of size 1024 bytes.

4.5.3 Route Discovery Process

A source node *S* initiates a route request to a destination node *D* by performing the following steps:

1. *S* sets the max hop count (*mhc*) variable equal to the *TTL* (time to live) variable found in the IP header.
2. *S* generates a random number *x* and sets it as the value in the hash field such that $h_0 = x$.
3. The top hash is then generated by applying the hash function *H*, max hop count (*mdc*) times to h_0 such that: *top hash*= $h_{mhc} = H^{mhc}(h_0)$. The hash function *H* is defined in the hash function field in the packet header.
4. *S* digitally signs all the fields in the packet except hop count and hash field, and stores the digital signature in the 32-bit signature field.
5. *S* then broadcasts the route request packet to its neighbours.

When an intermediate node receives a route request, it verifies the authenticity of the hop count and the integrity of the digital signature. The digital signature is verified using asymmetric cryptography. The hop count is verified by checking $h_{mhc} = H^{mhc-i}(h_i)$, where h_{mhc} is the top hash; h_i is the hash field of the route request; and H^{mhc-i} is the application of the hash function max hop count minus the hop count (*i*) times. The one-way hash chain technique used to authenticate the hop-count is similar to the authentication technique used in the SEAD protocol [55]. After the intermediate node verifies the digital signature and hop count, it replaces the packet's hash field with a new hash value computed by applying the hash function to the existing hash value. The intermediate node then rebroadcasts the route request and propagates it until it reaches the destination *D*.

When the route request *RREQ* reaches the destination *D*, *D* checks the validity of the packet and will reply with a reply packet *RREP* if the route request is valid. The *RREP* packet is forwarded along the reverse route to the source following the same authentication and integrity procedure that the *RREQ* message experienced.

AODV allows for intermediate nodes to also reply to route requests if they have a valid route to the destination node. SAODV proposes two solutions to this security problem. The first is the simplest, disallowing intermediate nodes to reply thereby ensuring that the destination node sends the reply message *RREP* and guaranteeing authentication. The second solution uses a double signature extension that allows an intermediate node *P* to reply to a route request from *S* for *D* $RREQ_{SD}$. Intermediate node *P* will reply with a double signature *RREP* message. One signature will sign the intermediate node *P*'s standard reply, and the second signature will sign the original *RREP* packet received by *P* for its route to *D*. Both reply message headers are included and sent to the source *S* to establish a secure route to *D*.

4.5.4 Maintenance

AODV uses error messages *RERR* to report broken links. SAODV secures these messages using digital signatures. The originator of the error message signs the entire message except the destination sequence number. Each forwarding node also signs the message to prevent unauthorized error messages being broadcast. Nodes using SAODV do not change their destination sequence number after receiving an error message because the error message's sequence number is not authenticated.

4.5.5 Analysis

SAODV authenticates the AODV routing packets preventing certain impersonation attacks. The assumption is made that a node's identity and address can be securely bound to a public key. Such an assumption leaves SAODV vulnerable to Sybil attacks.

SAODV employs asymmetric cryptography that is computationally taxing. The packet size has to be extended to incorporate the security mechanism resulting in a severe communication overhead. For every route discovery, a new one-way hash chain is computed, resulting in further computational overheads.

The SAODV protocol assumes that a key management entity is available in the ad hoc network that can successfully distribute public keys among participants. Such infrastructure is difficult to execute in mobile ad hoc networks. Before SAODV is to be implemented, either an off-line TTP or distributive key management scheme must be employed.

The SAODV solution uses hash chaining and digital signatures providing security against impersonation routing attacks. It also helps towards preventing denial of service attacks and eavesdropping attacks where malicious users may re-direct a route through a malicious node where eavesdropping may occur.

4.6 Secure Link-state routing (SLSP)

A hybrid scheme is proposed in [63] called Secure Link State Routing Protocol (SLSP). It is a proactive security solution that uses digital signatures and one-way hash chains to provide security to link-state updates and neighbour discovery. SLSP secures link state information in a localized manner, and can operate alone or be combined with a reactive ad hoc protocol routing protocol such as ZRP where SLSP would be the intra-zone routing protocol for the Zone Routing Protocol (ZRP) [64].

4.6.1 System Overview

SLSP provides secure neighbour discovery for nodes in a limited hop radius called a zone. Link state updates are authenticated using a hybrid cryptographic method and flooding attacks prevented by a priority ranking mechanism. The main assumption of SLSP is that nodes have existing asymmetric key pairs. SLSP assumes that a key management scheme is present to certify the public keys in the network.

SLSP uses four components: key distribution, secure neighbour discovery, link state update management, and a priority ranking scheme. SLSP's priority ranking scheme prevents denial of service type attacks.

4.6.2 Key distribution

SLSP assumes that each node has an existing signed public key before it joins the network, and that the certification of keys is performed by an assumed key management method. Public keys are bound to the IP addresses of the network nodes. Nodes then broadcast their public keys into their neighbourhood zone, for example a two hop radius. The received public keys are used to authenticate future packets from the source node.

4.6.3 Secure Neighbour Discovery

SLSP uses a Neighbour Location Protocol (NLP) to proactively check that neighbouring nodes do not perform impersonation attacks. Link state information is periodically broadcast by nodes in the form of a NLP hello message. These messages are signed by the source and contain the source's MAC address (a unique hardware address) and IP address (a distinctive network address). A NLP hello message broadcast by source *S* is described here:

$$S \rightarrow broadcast \; : \; E_{k_S}(IP_S|MAC_S)$$

Notification messages are generated when conflicting link state information is broadcast. An inconsistent mapping of the IP and MAC addresses is considered as conflicting link state information, as for example, when two nodes with different IP addresses have the same physical MAC address.

4.6.4 Link State Update Management

Link state update packets are periodically broadcast to a limited hop radius of nodes. A link state update packet (LSU) contains the IP address of the source, a 32-bit sequence number used for updating and a hop count variable. The LSU hop count variable is authenticated using hash chains, as discussed in SEAD [55] and SAODV [62], and the rest of the packet content is authenticated using digital signatures.

When a LSU is received, the digital signature is verified using the previously distributed public key. The hash chain is verified and the time to live (TTL) variable is decremented. The hop count authentication protects the LSU packet from travelling too many hops or from link state updates not being received by nodes.

4.6.5 Priority Ranking Scheme

SLSP uses a lightweight flooding prevention scheme that gives priority ranking to nodes. A priority list is maintained for neighbouring nodes that ranks node's priority based on the number of link state updates a node broadcast. Malicious nodes will flood the network with link state update packets to cause resource and denial of service attacks. SLSP gives high priority to nodes that send less link state updates limiting the affects of flooding attacks.

4.6.6 Analysis

The Secure Link State Routing Protocol is a hybrid cryptographic scheme using digital signatures to provide authentication for its NLP hello messages and link state update (LSU) packets. Hash chains are used to authenticate the limited hop broadcast of LSU. Impersonation type attacks are prevented by monitoring the IP and MAC address bindings of neighbours through a neighbour location protocol (NLP). Link state updates are authenticated using digital signatures and hash chains. Flooding type attacks are prevented using priority ranking.

The SLSP protocol provides security to topology discovery but cannot act as a standalone security mechanism as it lacks a data transmission protection agent. Nodes that securely join the network can misbehave during data transmission without being detected or prevented. SLSP lacks a disciplining agent like a revocation mechanism. For example, a malicious node *B* can impersonate another node *A* and flood *A*'s neighbours with LSU packets. SLSP's priority mechanism will limit the effectiveness of the flooding attack. The NLP protocol will detect the impersonation attack, but node *A* has no mechanism to correct to the attack and *A* will remain with a low priority.

4.7 On-Demand Secure Routing Byzantine Resilient Routing Protocol (ODSBR)

The on-demand secure routing byzantine resilient routing protocol (ODSBR) is proposed in [65]. Byzantine behaviour is defined by the authors as any action taken by an authenticated node to disrupt the routing procedure. ODSBR is a secure reputation-based routing protocol that prevents the effects of byzantine failures on successful routing.

4.7.1 System Overview

ODSBR uses weighted paths to select routes in the route discovery process. Paths are assigned weights based on a fault path detection method. A high weight is assigned to an unreliable path. ODSBR is divided into three components: route discovery process, fault detection, and weight path management. ODSBR assumes that a public key infrastructure is present to manage public key authentication.

4.7.2 Route Discovery Process

Routes are discovered in an on-demand manner like in DSR. ODSBR extends the standard route request *RREQ* by adding a weight list instead of a node list as in DSR. A weight list includes the list of chained nodes with their associated weights. These weights are defined by link failures detection mechanism. The *RREQ* is signed by the source and broadcast into the network, updating its weighted list after each hop until it reaches the destination. The destination then verifies the signature and broadcasts the reply message *RREP*. Each node that receives a *RREP* will then calculate the total weight of the path by summing the weights of the specific path to the current node. The *RREP* is forwarded if the total weight is less than the total weight of any previously forwarded *RREP*. Before an intermediate node *P* forwards a suitable *RREP*, all the signatures are verified and node *P* appends its signature. The source node receives the *RREP*, calculates the total weight and verifies all the signatures.

4.7.3 Fault Detection Method

Each node *i* has a list of probe nodes. Each probe node sends node *i* an acknowledgement message for each data packet *i* sends. If a threshold *t* of acknowledgements is not received, a fault accusation is logged against a specific path. Using a binary search technique, ODSBR identifies a path as faulty after *log(n)* fault accusations, where *n* is the length of the accused path.

4.7.4 Weight Path Management

A low weight is associated with a secure path. The weights associated with paths are influenced by two factors: time and fault detection. When ODSBR identifies a path as faulty based on the fault detection method, then the weight for that path is doubled. Path weights are halved after a counter reaches zero, and each path has an associated counter.

4.7.5 Analysis

The on-demand secure routing protocol (ODSBR) provides ad hoc on-demand routing with byzantine failure prevention. Weights are assigned to paths by a fault detection method and paths are selected, based on the weights. The Secure Routing Protocol (SRP) proposed in [66] introduces the metric specific path selection method but this proposal is not a standalone secure routing protocol like ODSBR [65].

ODSBR assumes the existence of a public key management system. ODSBR further assumes that a shared key exists between source and destination nodes to ensure authenticity and integrity of acknowledgement of messages sent by probe nodes. This helps avoid expensive asymmetric per packet computations for acknowledgement messages.

The route discovery process of ODSBR broadcasts reply messages, instead of unicasting them which would result in a computationally expensive operation. This method will result in $2^{\frac{n}{2}+1} - 1$ reply packet transmissions where *n+2* is the number of nodes in a path from node *A* to *B* including nodes *A* and *B* [65]. Furthermore, the cost monitoring of data packet transmission is computationally high because the fault detection method requires a threshold of *t* probe nodes to reply with an acknowledgement for every data packet sent.

ODSBR is identified by authors [65] to be vulnerable to wormhole attacks. The wormhole attack may be avoided in the case where the wormhole link node exercises byzantine behaviour.

4.8 Reputation-based CONFIDANT

The CONFIDANT protocol representing the ‘Cooperation Of Nodes: Fairness In Dynamic Ad Hoc Networks’ [67] is a reputation-based solution that operates over the DSR routing protocol.

4.8.1 System Overview

The CONFIDANT solution does not use a cryptographic approach to achieve secure routing. The system is solely reputation-based and operates in an on-demand ad hoc network environment as an extension of the DSR routing protocol. Each node in the ad hoc network is required to be involved in the four components of CONFIDANT: monitoring, trust management, reputation system and path management.

4.8.2 Monitoring

Each node is a monitor and is responsible for the packets that it sends or forwards. For every packet that a node forwards, it watches that the next hop node forwards the packet properly. The monitor looks for inconsistent behaviour and triggers an alarm to the reputation system if misbehaviour is discovered.

4.8.3 Trust Management

The trust management system manages the alarm messages. The alarm messages are generated by each node’s monitoring system and exchanged between other nodes in order to build and maintain a local rating list. The trust management manages the input of alarm messages and assigns more influence to alarm messages that come from trusted nodes and less influence from other nodes. CONFIDANT assumes pre-existing relationships between a selection of nodes called friends [67]. Friend nodes are highly trusted nodes. Local rating lists are exchanged as well, and their influence is managed by the trust management system.

4.8.4 Reputation System

The reputation system manages and maintains the local rating list. This list contains node identities and corresponding ratings. A rating will correspond with the amount of misbehaviour a node has displayed. The ratings will be updated from alarm messages and direct observations.

4.8.5 Path Management

Paths are selected based on a rating threshold, local rating lists and a blacklist containing all untrusted nodes. A node is blacklisted when its rating is below the rating threshold *t*. The path manager removes the paths in the network that contain the blacklisted node. The path manager manages the route discovery process by reacting to route requests from blacklisted nodes or route requests that have passed through a blacklisted node.

4.8.6 Analysis

CONFIDANT is an exclusively reputation-based routing protocol. Local rating lists of nodes' behaviour are recorded and used during the route discovery process. The authors note that only negative evidence is gathered against nodes. Consequently nodes can only be identified as less trusted as the network continues. Like most reputation-based schemes, a counter system is employed where each rating list entry has an associated counter. When the counter reaches zero, the rating is reset to the default state of null misbehaving accusations. The CONFIDANT protocol assumes the existence of prior trust relationships between a select number of nodes called friends [67].

4.9 Discussion

Several different secure routing protocols are presented in Chapter 4. They differ in the areas of security and operational requirements. The diversity of their design makes it difficult to assess their comparative success, but this section outlines the diverse characteristics of the presented protocols.

4.9.1 Security Analysis

The proposals can be categorized by the security approaches that include the asymmetric, symmetric and hybrid cryptographic security. The last category is the reputation-based solutions. The security mechanism of each protocol is presented, and the attacks that the protocol protects against are highlighted. A summary of the evaluation is presented in Table 4-2.

4.9.1.1 Symmetric Cryptography

The symmetric cryptographic approaches include the Secure Efficient Ad Hoc Distance Vector Routing protocol (SEAD) and the Ariadne protocol. Hash functions and hash chains like SHA-1 [57] and MD5 [56] are used for authentication purposes usually for the hop count variable. The hash function is lightweight compared to asymmetric security techniques.

The SEAD protocol uses a hop-by-hop authentication technique. SEAD authenticates the sequence number and hop count of routing packets protecting the routing procedure from resource consumption attacks, such as: denial of service attacks, route table poisoning, and replay attacks.

The Ariadne protocol is proposed by the same authors of SEAD. Ariadne uses message authentication code (MAC) to provide end-to-end authentication between communication nodes. Ariadne protects against similar attacks to SEAD but uses end-to-end authentication.

4.9.1.2 Asymmetric Cryptography

The only solely asymmetric cryptographic protocol investigated is the Authenticated Routing for Ad hoc Networks protocol (ARAN). Asymmetric cryptographic is computationally costly compared to symmetric cryptography, and it requires the existence of a trusted third party or self organized key management system.

ARAN provides end-to-end authentication for an on-demand mobile ad hoc network. ARAN provides authentication and protection from replay attacks and unauthorized routing table attacks. ARAN is vulnerable to flooding attacks. A malicious node can flood nodes with fake routing packets signed with illegitimate keys. This will result in many unsuccessful verifications and ultimately denial of service and resource consumption.

4.9.1.3 Hybrid Cryptography

The SAODV and SLSP protocols are hybrid solutions that employ both asymmetric cryptography and symmetric cryptography. The common approach is for all the mutual fields to be digitally signed and the immutable fields, like the hop count, to be protected using hash chains. The Secure Ad Hoc On-demand Distance Vector protocol (SAODV) employs this tactic to provide end-to-end authentication but at the cost of extending the routing packet header.

SAODV protects against impersonation attacks on the routing protocol. It also helps prevent replay and denial of service attacks.

Secure Link State Routing Protocol (SLSP) provides secure neighbour discovery and authenticated link state updates but lacks a secure data transmission protocol. The Neighbour Location Protocol of SLSP protects against impersonation type attacks where malicious nodes adopting conflicting IP and MAC addresses would want to corrupt the routing table. Furthermore, flooding attacks that result in resource consumption and a denial of services are prevented by a priority ranking scheme.

4.9.1.4 Reputation-based

Reputation-based or conduct-based systems allow for a node's behaviour in the network to affect its assigned security or trustworthiness. Reputation-based protocol can be computationally costly because they usually require packet monitoring systems and the proactive exchange or behavioural evidence between nodes. The On-demand Secure Routing Protocol Resilient to Byzantine Failures (ODSBR) and the CONFIDANT protocol are reputation-based systems.

ODSBR uses a reputation-based system to select the most secure routes. A fault detection method monitors the success of each packet transmission and faults are logged against specific paths. Reputation is path specific in ODSBR. ODSBR couples with an asymmetric cryptographic approach to provide end-to-end authentication along the selected secure path. The CONFIDANT uses a reputation-based approach to provide security. Similar to ODSBR, only negative evidence is considered. Nodes monitor every packet that they forward and maintain a local rating list. The CONFIDANT reputation or ratings are node specific, unlike those of ODSBR.

Both CONFIDANT and ODSBR monitor packet forwarding that protects the system from black hole attacks. Replay attacks that use a flooding method are prevented by the use of path reputation and node reputation in ODSBR and CONFIDANT respectively. A disadvantage of the negative reputation approach for ODSBR and CONFIDANT is that black list nodes or faulty path entries have an expiration time, after which their confidence is reinstated. This allows malicious nodes to continue disrupting the network until they are caught again.

Table 4-2: Summary of security analysis for secure routing in ad hoc networks

Protocol	**Security Approach**	**Techniques**	**Attack Prevention**
SEAD [55]	Symmetric Cryptography	• Hop-by-hop authentication • Hash chains	• Resource consumption • Denial of Service • Route table attack • Replay
Ariadne [59]	Symmetric Cryptography	• End-to-end authentication • Hash chains	• Resource consumption • Denial of Service • Route table attack • Replay
ARAN [61]	Asymmetric Cryptography	• End-to-end authentication • Certificate Authority	• Route table attack • Replay attacks
SAODV [62]	Hybrid Cryptography	• End-to-end authentication • Hash chains • Digital Signatures	• Denial of Service • Route table attack • Replay
SLSP [63]	Hybrid Cryptography	• Secure neighbour discovery • Authenticated link state updates	• Denial of Service • Route table attack • Replay
ODSRP [65]	Reputation-based	• Path specific reputation lists • Digital Signatures	• Denial of Service • Route table attack • Replay • Byzantine Failures
CONFIDANT [67]	Reputation-based	• Node specific reputation lists	• Black Hole • Replay

4.9.2 Operational Requirements

The presented secure ad hoc routing protocols have certain assumptions that each makes to realize its design. Furthermore, protocols are designed specifically for operation in specific routing environments. This section summarizes the operational requirements of the presented secure ad hoc routing protocols. Table 4-3 summarizes this discussion.

The symmetric cryptographic approaches do not rely on a public key infrastructure, and still require some kind of key management in the ad hoc network. The SEAD protocol is designed for a table-driven routing protocol and is based on the DSVD routing protocol. SEAD requires a key management mechanism to distribute an authenticated initial hash element. Ariadne is a DSR based on-demand protocol that assumes that there are shared secrets between each communication pair. The shared keys are used in TESLA authentication, and a key management system is assumed to be present to distribute the keys. TESLA authentication also requires time synchronization between each node. This is difficult without the presence of an online TTP.

ARAN, SAODV, SLSP and ODSBR use asymmetric cryptography, and key management is simply assumed for each of these protocols. ARAN assumes that an online TTP is present that acts as a certificate authority (CA). Prior shared secrets are assumed between all participating nodes and the CA. ARAN is an on-demand protocol. SAODV protocol is based on the AODV on-demand routing protocol and assumes the presence of a key management system to distribute keys. SLSP assumes that nodes enter the network with asymmetric key pairs, and a key management scheme is present to certify the keys in the network. SLSP is a table-based routing solution. ODSBR is based on DSR routing and authenticates its routing packets with digital signatures, and a public key infrastructure is assumed to manage the keys. Shared keys are also assumed to allow for authentic acknowledgement message communication between source and probe nodes.

CONFIDANT does not use a cryptographic approach and does not require the existence of a key management scheme. CONFIDANT does assume pre-existing relationships between a small number of nodes called friends. The CONFIDANT solution is designed for on-demand routing.

From this discussion, the conclusion is made that most secure ad hoc routing protocols assume the existence of a key management system to certify, authenticate, and distribute keying information. Mobile ad hoc networks cannot assume the existence of a TTP and must address the problem of key management.

Table 4-3: Operational requirements of the present secure routing protocols

Protocol	**Routing**	**Assumption**
SEAD [55]	Table Driven DSDV based	• Key management system to distribute an authenticated element for hash chaining
Ariadne [59]	On-Demand DSR based	• Time synchronized network • Shared secret key between each node pairs for MAC • Key management system to manage TESLA keys
ARAN [61]	On-Demand Not protocol specific	• TTP acting as a certificate authority (CA) • Prior shared CA public key
SAODV [62]	On-Demand AODV based	• Key management scheme • Secure IP public key binding
SLSP [63]	Table Driven Not protocol specific	• Nodes have existing asymmetric key pair • Key management system
OSRP [65]	On-Demand DSR based	• Key management system • Shared keys between source and probe nodes
CONFIDANT [67]	On-Demand Not protocol specific	• Pre-existing relationships between a selection of nodes called friends

Chapter 5

Proposed Security Scheme: Direct Indirect Trust Distribution (DITD)

5.1 Introduction

A security establishment scheme is proposed for a mobile ad hoc network. Key management is central to the establishment of trust in these networks. The proposal focuses on key management. The proposal is for a mobile ad hoc network that operates in a self-organized and fully distributive network environment. These networks allow for nodes to join and exit the network without restriction. These networks find application in the military and commercial filed. For example, application can be found in tactical positional networks for military-based communication or personal area networks for secure peer-to-peer data and file-sharing. A current example is sited within The Armaments Corporation of South Africa, (ARMSCOR) sensor network development department. The application of these networks was discussed in Chapter 2. The proposed protocol is planned for these self-organized distributive networks. It is noted that these networks do not allow for rigorous access control, but extensions can be made to the proposal, that allow for access control services.

In a self-organized mobile ad hoc network, there is no presence of a separate authority member, such as a trusted third part or certificate authority. Instead, each node that enters the network is considered the security authority of its own domain. Security is established by nodes creating and issuing certificates that bind nodal identities to their respective public keys. These certificates are issued and distributed in order to realize secure communication. A bi-directional security association is made between nodes *A* and *B*, when node *A* holds a certificate binding the public key of *B* and *B*'s identity; and *B* holds the certificate binding the public key of *A* and *A*'s identity. Malicious adversary nodes that wish to disrupt communication will target the network layer, and more specifically the routing mechanism, as identified in Chapter 2. The network layer is identified as the sphere of design.

The problem is then to provide secure communication that is implemented on the network layer. Secure communication is achieved when node *A* is able to set up a secure communication channel, where no other entity can interrupt or eavesdrop on its communication with node *B*. The question as to whether node *B* is worthy of trust is not the concern. That question must be decided by the nodes themselves, based on available trust evidence. The proposal made on the network layer aims to provide the most secure route between *A* and *B*, preventing malicious adversaries from sabotaging communication.

The term trust is defined as the "belief by a trustor with respect to the competence, honesty, security and dependability of a trustee within a specific context."[68]. There are two trust variables: direct trust and indirect trust. Direct trust is a result of independent or local trust evaluation, between two immediate nodes. Indirect trust is evaluated using the advice from other nodes. In the context of certificate base trust, direct trust is defined as trust between local neighbours. Indirect trust is created by certificate chaining.

A hybrid trust model is proposed, uniting certificate and conduct-based trust to provide a more secure communication. A key management model is proposed. This model supports an existing routing protocol. The proposed scheme is called Direct, Indirect Trust Distribution (DITD), and it follows the following procedure: direct trust is established by requesting that all nodes involved in the route discovery stage share their self certificates with each others' one-hop neighbours involved in the route discovery phase. Indirect trust is further established by requesting that the sender's self certificate propagate with the route request toward the destination. The routing messages trigger certificate distribution, allowing direct trust relationships between one-hop neighbours. These trust relationships are then chained together, providing a trusted route to the destination node. Keying material is allowed to be propagated along these chains of trust. A disadvantage of the self-organized nature of these networks is that the established security of trust chains will rely on transitive trust [2]. The DITD model proposes coupling the security provided by the certificates with a conduct-based trust analysis model. Conduct trust is affixed, allowing for more secure communication. This is achieved by calculating the trust of routes, based on the conduct of the nodes involved, and selecting the most trusted route for communication.

This chapter is structured as follows. In Section 5-2, the related work is briefly reviewed. In Section 5-3, the Direct Indirect Trust Distribution (DITD) scheme is proposed, and the requirements, environment and a high level system model of the scheme's design is presented.

In Section 5-3-2, the key distribution mechanism is proposed. Section 5-3-3 introduces the key verification mechanism. In Section 5-3-4 the conduct-based trust analysis model is proposed.

In this chapter, the focus is placed on the introduction and proposal of the new security scheme DITD. Chapter-6 will discuss the performance and characteristics of the proposed mechanism and provide concluding comments for the proposal.

5.2 Related Work

A detailed survey was presented on key management schemes for mobile ad hoc networks in Chapter 3. Chapters 4 focused on the network layer and presented a survey of existing secure routing protocols. This section provides work directly related to the DITD model.

The authors of [2] propose a completely self organized public key system for mobile ad hoc wireless networks. This is a PGP based solution that provides key management in ad hoc networks without the presence of an off-line or on-line authority, such as a CA, TTP or server. Each node distributes its self certificates and maintains its own certificate repositories. Nodes participating in the network share their certificate repositories, and repository updates are preformed in a proactive manner. Certificates are reciprocally authenticated, and trust chains are formed linking remote nodes to each other. Security is realized on the application layer.

Zapata [69] addresses the issue of verification delays in secure mobile ad hoc networks. Zapata proposes a protocol to optimize the number of verifications made in a single secure route discovery phase. Only when a route is established, are the shared certificates verified. This helps reduce the computational overhead of verifications on multi-hop paths. By reducing the total number of verifications made in a network's life time, there is a resultant end-to-end delay upon the delivery of routes.

Theodorakopoulos *et al.* [70] proposes a fully distributive conduct-based trust model that has PGP characteristics. This model operates on the application layer and allows for trust to be established without the presence of a central authority member. PGP models share certificates to establish trust, while the work proposed in [70] allows for other trust evidence such as conduct and location to influence the trust establishment. Trust is fully distributed in a proactive manner, allowing all nodes to give trust opinions about other nodes.

Semiring mathematics presented in [71] has been used to model trust calculations more recently in [70]. Trust opinions are mathematically aggregated along a path, and trust decisions are mathematically represented. The work in [70] uses Dijsktra's extended algorithm proposed by Mohri [72] to include trust. This finds the most trusted path between two remote nodes in a proactive manner.

The majority of literature mentions functions in a proactive manner for application layer solutions. The DITD model is designed on the network layer for a reactive, fully distributive, self organized, mobile ad hoc network environment. The ideas of some of these protocols have inspired the creation of the DITD model, and the impact of these protocols is discussed in Chapter-6.

5.3 Proposed Security Scheme

The aim is to design and investigate a security mechanism to specifically provide: *public key certificate distribution*, *optimal verification*, and a *conduct trust model* to optimize trust decisions. The security mechanism is to provide secure communication in a mobile ad hoc network environment while satisfying the following requirements based on environment and functionality.

5.3.1 Design Requirements

5.3.1.1 Environment

- **Network layer design:** The security mechanism is to be implemented on the network layer protecting these dynamic networks from attacks and avoiding multi-layer design.
- **Self organised:** Nodes are responsible for their own security services, including the distribution of keying information.
- **Fully distributive:** The certificate distribution scheme is to be designed in a fully distributive manner where all nodes participate in the operation and implementation.
- **On-demand:** The DITD model is to be designed in an on-demand environment optimizing the limited resources of ad hoc networks. On-demand models provide security to nodes upon request. The proactive approach provides security to an entire network at once and requires computationally taxing periodic updates.

5.3.1.2 Functionality

- **Distribution of keying material:** DITD is to provide direct and indirect trust relationships between local and remote nodes by efficiently distributing self certificates between nodes.
- **Minimize the overhead:** DITD aims to minimize the overheads upon the network routing performance while still providing trust establishment. DITD aims to avoid alterations to the routing control packets and strives for independence between routing and trust establishment.

- **Provide secure communication from the start:** Secure communication is requested from the start to the end of the network lifetime unlike [10, 73] which are flawed by an initial setup phase with weak security.
- **Trust evaluation mechanism:** Security should be supported by a trust evaluation mechanism allowing for more secure routes to be established and ensuring the secure distribution of certificates.
- **Robust in the presence of topology change:** The DITD model should be robust to poor connectivity and routing failure due to changing mobility that is an inherent characteristic of a mobile ad hoc network.

5.3.2 System Model

To fulfil the constraints given in Section 5-3-1, we assume the following system model. There is no pre-existing infrastructure and no online trusted third party present during communication. The model is a fully distributive network of wireless nodes using an ad hoc on-demand routing mechanism. It is assumed that nodes have their own keying material before joining the network generated by a fully self organized mobile ad hoc network [2], or by an off-line authority issuing keying material before a node enters the network, as for example in [10]. Each node is assumed to have a public and private key pair, a self certificate binding the public key and user identification of the node, and a set of network security parameters common to all nodes in the network. Secure communication is requested from the start to the finish of the network's lifetime. Users can join and leave the network as they please. Any user with the correct keying material may participate in the network. It is assumed that conduct information is available to each node from node monitors [74].

The DITD model uses certificate-based trust coupled with conduct-based trust to develop a hybrid trust protocol maximizing trust in the network. The DITD model addresses the issues of *key exchange*, *verification protocol* and *conduct trust evaluation*. It is designed on the network layer accompanying an on-demand routing protocol. Figure 5-1 describes the high-level system model.

The DITD scheme performs the task of *key exchange*, exchanging self certificates on the network layer following an on-demand routing mechanism. Direct trust is established among one-hop neighbours by self certificate exchanges triggered by the route discovery process. This allows for a bi-directional security association to be made between immediate nodes. This relationship is referred to as direct trust. Direct trust associations are chained together creating trusted paths and allowing for two nodes out of each other's communication range to exchange self certificates. This describes the *key exchange* element of the DITD model. It is divided into two parts: the exchange of direct and indirect trust relations. Figure 5-2 illustrates the direct, indirect trust relations established by DITD.

An on-demand route discovery phase will flood the network with route requests in search of a destination. This will couple with the *key exchange* mechanism of DITD. The result of this is a flood of self-certificate exchanges. Verification of these certificates is optimized by the DITD model. Trust chains will have an accumulative certificate verification delay because possibly each direct trust association will need to be verified. The DITD model proposes a *verification protocol* that optimizes and manages the verification process.

DITD uses the existence of conduct trust evidence to maximize the security provided by public key certificates. Trust is aggregated in a reactive manner using semi-ring mathematics and a reactive shortest path algorithm. The most trusted routes are selected by a *conduct evaluation protocol* that includes an implicit revocation mechanism and trust evaluation metric. Direct and indirect trust establishment is strengthened by DITD's conduct trust evaluation.

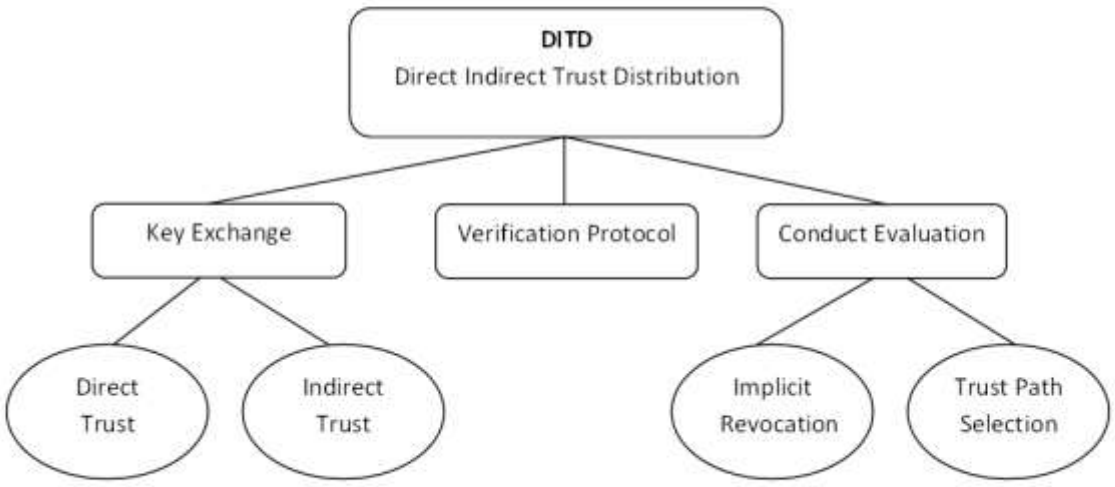


Figure 5-1: High level system model

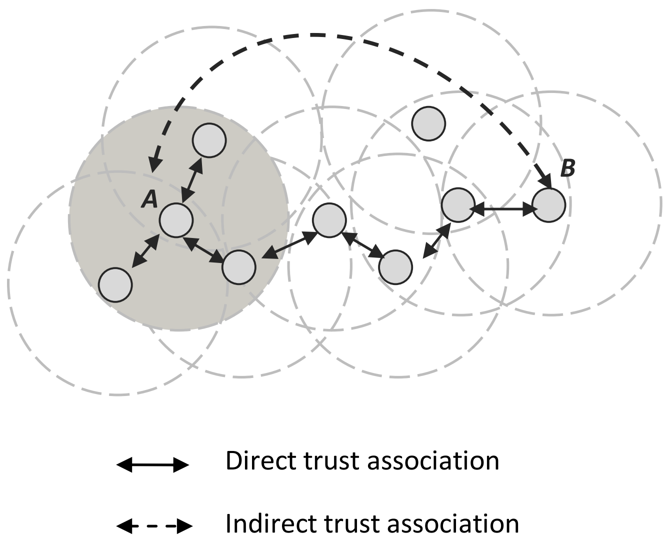


Figure 5-2: Direct and Indirect trust establishment in DITD model

5.3.3 Key exchange

A certificate trust model is used to create trust between nodes. A secure bidirectional communication path can be set up between node *A* and *B* if both nodes have exchanged each other's self certificates, as seen in Figure 5-3. The self certificate binds the user *ID* and the public key, and the exchange of self certificates allows keys to be exchanged authentically. Node *A* can trust node *B* if node *A* has a certificate verifying the identity of node *B*, and likewise concerning node *B's* communication with node *A*.

The DITD certificate trust model appends an existing on-demand mobile ad hoc routing protocol. Its principles can be applied to any on-demand routing scheme. Knowledge of the operation of an ad hoc routing mechanism will help visualise the explanation of DITD.

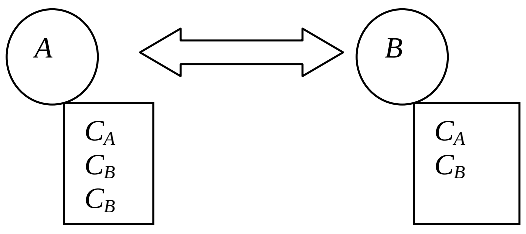


Figure 5- 3: Secure bidirectional communication between node A and B

The application of an ad hoc on demand routing procedure is briefly revisited to aid the DITD explanation. The AODV [21] routing procedure follows three stages during route discovery:

1. Sending of request message (*RREQ*)
2. Receiving of request message (*RREQ*)
3. Sending reply message (*RREP*)

In stage one, the source node *A* requests communication with destination node *B* by broadcasting a routing request $RREQ_{AB}$ into the network. This request is forwarded and propagated through the network to *B*. The *RREQ* message may have been sent by *A* or forwarded by an intermediate node P_i. When the *RREQ* is received by an intermediate node P_i, stage two begins. At stage two, a reverse route to *A* is then set up and P_i checks if it is the destination *B* or has a fresh route to the destination node *B*. If not, then the *RREQ* is further broadcast by P_i and propagates until the destination is found. When the destination or a fresh route to the destination is found, stage three commences. At stage three, a reply message *RREP* is propagated along the reverse route until it reaches the source node *A*, establishing the communication route.

When a node receives a routing control packet, certificate requests are triggered and sent using separate unicast messages. The certificate distribution process is added at stage two and stage three, i.e. the receiving of a route request and the sending of a reply message, respectively. The following symbols are defined for explanation purposes:

Table 5-1: Definition of symbols

P_i	intermediate node *i* receiving the *RREQ*
P_{i-1}	previous node P_{i-1} who forwards *RREQ* to its neighbour P_i
A	originator node of RREQ message
B	destination node of RREQ message
$Cert_i$	certificate of node P_i

At stage two, involves the reception of a route request packet. Before this packet is processed and the routing table updated, direct trust and indirect trust establishment is set up.

5.3.3.1 Direct Trust

At stage two, direct trust relationships are made by sharing neighbouring nodes self certificates. When intermediate node P_i receives a route request *RREQ*, it first checks its certificate repository for the certificate of the neighbour who forwarded the request, P_{i-1}. If it does not possess such a certificate, $Cert_{i-1}$, a local self certificate exchange is done between node P_i and its previous hop neighbour P_{i-1} as follows: a *unicast* message is sent from P_i to P_{i-1} with P_i's self certificate $Cert_i$ appended; P_{i-1} receives the message, updates its certificate repository and replies with a unicast message to P_i containing P_{i-1}'s certificate. If the certificate $Cert_{i-1}$ is found in P_i*'s* certificate repository, there is no need for a self certificate exchange. This procedure follows the *RREQ* as it floods the network in search of a route to the destination node. Direct trust establishment is illustrated in Figure 5-4. What can be expected is an increased in control packet overhead, as the DITD model transmits additional certificate packets into the network.

A second direct trust establishment approach is proposed that exploits the *HELLO* packets of the AODV routing protocol. The AODV protocol uses periodic one-hop broadcast packets to maintain and establish communication between neighbouring nodes. The DITD model proposes that direct trust can be established independent of the route discovery process by including a certificate query in the *HELLO* packets. When a $HELLO_A$ packet is broadcast by node *A* and received by node *B*, the receiver *B* checks its certificate repository for the certificate $cert_A$. In a similar way to the first approach, if no certificate is found a localized certificate exchange is performed. The certificate exchange messages are independent from the *HELLO* packets. The second approach allows for direct trust establishment with the least amount of dependence on the routing procedure.

5.3.3.2 Indirect Trust

Still at the stage two (receiving the routing request) level, indirect trust is established between remote nodes *A* and *B* by the exchange of remote self certificates. Similar to the direct trust set up, before node P_i processes the received routing request *RREQ*, a certificate search and exchange is made. Node P_i searches for the source *A*'s certificate, $Cert_A$. If the certificate is not found, P_i sends a separate unicast certificate request for $Cert_A$ to the previous node P_{i-1}, whose address can be found at the next hop on the reverse route in the routing table. This addition allows for the source's certificate $Cert_A$ to be propagated towards the destination *B*. It is noted that by not appending the certificate to the route requests, this reduces dependency between the route establishment and certificate trust establishment.

For indirect trust to be complete between nodes *A* and *B*, the source *A* is required to possess the destination's certificate, $Cert_B$. Further additions to stage three, sending the reply message, are required to complete the indirect trust establishment. A reply is sent is sent under two conditions. Firstly, when the destination node is found; and secondly, when a fresh route to the destination node is found.

For the first condition, the reverse route to the source *A* is already set up with localized direct trust existing between nodes on the route. Hence, a trusted chain of nodes is available from *B* toward the originator node *A*. All that is required is for the certificate of the destination node, $Cert_B$, to be piggy- backed on the routing reply message *RREP* toward *B*. Each intermediate node stores $Cert_B$ and updates its certificate repository and the forward route to *B*.

For the second condition: if a fresh route to *B* is found at P_i, there exists a route from the intermediate node P_i to the destination *B* and a route from P_i to the source *A*. Both routes have localized direct trust existing already. Two *RREP* messages are then propagated, one toward *B* with $Cert_A$ appended and one toward A with the $Cert_B$ appended. Indirect trust is therefore set up by certificate chaining as illustrated in Figure 5-4.

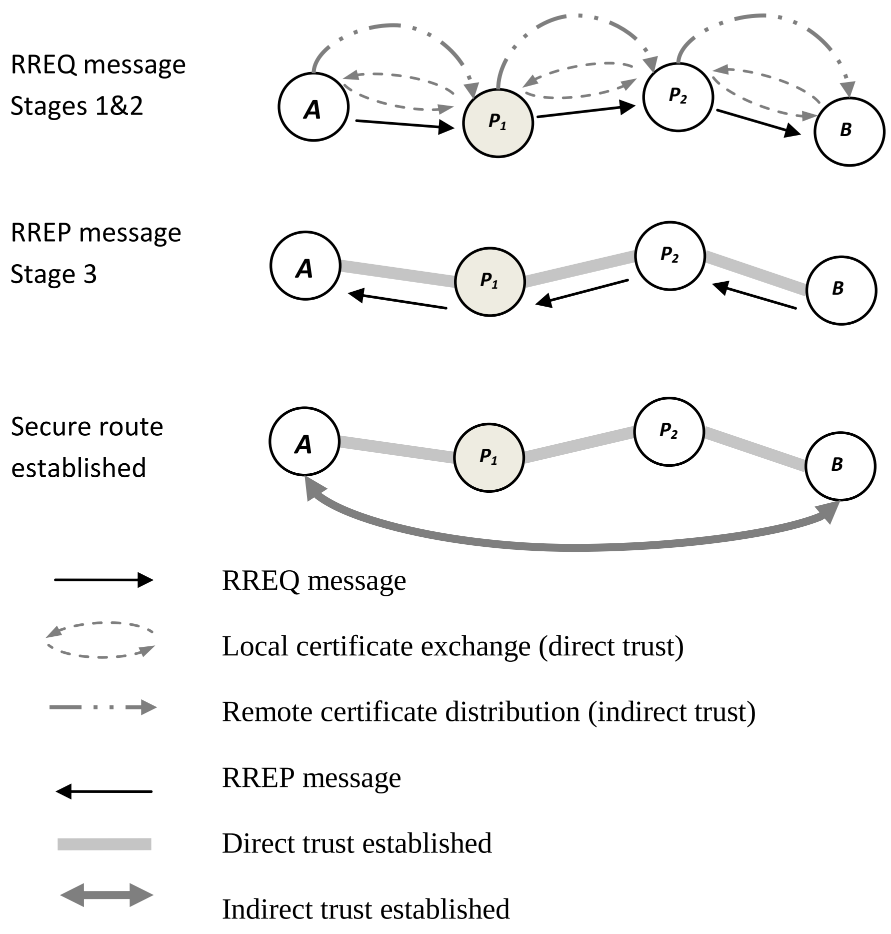


Figure 5-4: Illustrating the certificate exchange protocol

5.3.4 Verification Protocol

For trust to be established between two entities they must not only share the certificates but the certificates must be verified for the users to be authenticated. Ideally, verification will take place immediately after a certificate is received but a single verification can take up to 1ms delay [75] on 1024-bit RSA key, and even more for a ECC key. These verifications can accumulate across multi hop routes. For application, specific networks that are time-dependent such as audio applications and military automation networks, a delay of milliseconds is critical. A requirement of DITD is for the security additions not to cripple or delay the existing routing mechanism.

Verification for direct trust establishment can be done immediately without incurring a delay upon the routing mechanism. This is because the localised certificate messages are separate and independent from the request messages. Furthermore during route discovery, request messages (*RREQ*) can be forwarded without waiting for verification to be processed [69], as verification can be confirmed on the reply route. Delayed confirmation of verification is not possible for the reply message (*RREP*) because the exchange of the destination node's certificate, $Cert_B$, follows the *RREP* message. The $Cert_B$ certificate must be verified before the *RREP* message can be securely forwarded and trusted routes established. This means that a certificate trust chain will have an accumulative processing delay due to verifications. Therefore, the problem is that the verification of the destination certificate $Cert_B$ may cause a delay in route establishment because $Cert_B$ is distributed with the *RREP* message.

A solution to this is that if any intermediate node has $Cert_B$, it can distribute $Cert_B$ to the reverse route, during *RREQ* message propagation. When a *RREQ* message is forwarded, a *flag* is appended identifying whether the forwarder has the destination certificate $Cert_B$. Intermediate node *P* receives the *RREQ* message and updates the reverse route entry with $flag_{cert}$ indicating if the previous hop has $Cert_B$. *P* checks if it has $Cert_B$ in its certificate repository and assigns an appropriate value to $flag_{cert}$ before forwarding the *RREQ* message. If *P* has $Cert_B$ and the reverse route variable $flag_{cert}$ indicates that the previous hop does not have $Cert_B$, then *P* sends a unicast certificate message containing $Cert_B$ to the previous hop, whose identity can be found from the reverse route in the routing table. The $Cert_B$ is propagated along the reverse route by checking the routing table entry $flag_{cert}$ and responding in a similar fashion. This allows the destination certificate $Cert_B$ to be distributed during the route discovery phase independent from route establishment. The verification protocol is illustrated in Figure 5-5 where source *A* sends a *RREQ* for destination *B* and intermediate node P_2 has $Cert_B$ but no route to *B* itself. In this case while route discovery continues, $Cert_B$ is transmitted to the nodes in the reverse route which do not have $Cert_B$. These nodes are indicated by the $flag_{Cert}$ variable. Certificate verification is done concurrently with route discovery, thereby minimising the amount of verifications that delay the route discovery. Outstanding verifications are done following the *RREP*. Verification checks are preformed with the *RREP* message. Figure 5-5's example allows for three less verification delays at P_2, P_1 and *A* because of back-tracked verification.

The condition under which the above scenario will be most effective is when a node has received $Cert_B$ during a previous route establishment but no longer has an existing route in its routing table for *B*. Such an occurrence is a result of a node previously involved in a route to *B* but due to route expiry, loss of connectivity or node mobility the node is no longer part of such a route. Therefore the benefits of the verification protocol are most evident in ad hoc networks with moderate to high speeds. The DITD model implements verification optimization to reduce the delay incurred on the routing mechanism by verification.

The authentication protocols ARAN, SAODV, SEAD or Ariadne discourage intermediate nodes with a route to the destination to reply to route requests. If the DITD model were used in conjunction with such a protocol, then although intermediate nodes with a validated destination certificate are unauthorized to reply, DITD maximizes the availability of the destination certificate. The verification protocol would be used to distribute the destination's certificate along the reverse path and perform verification checks so lesser time delay is incurred from the route reply message.

Direct and indirect trust establishment is realised through the route establishment phase of the ad hoc routing scheme. During the initial stage of route establishment the network is flooded with routing requests and in turn certificate exchange messages. It can be expected that there will be a large packet overhead as a result of additional certificate packets.

Mobility produces erratic connectivity problems and unexpected routing failure. Multi-hop routes are vulnerable to failure under increased mobility, while localised one-hop route connections are less vulnerable. If the proposed solution was dependent upon such multi-hop routes, as [10] is, it would suffer severely from inherited link breakages common to highly mobile networks. The proposed solution prevents the certificate exchange procedure from using multi-hop routes by exchanging certificates in a strictly localized manner. This allows the DITD certificate distribution scheme to operate in ad hoc networks with varied mobilities and changing connectivity without the worry of routing failure interfering with security.

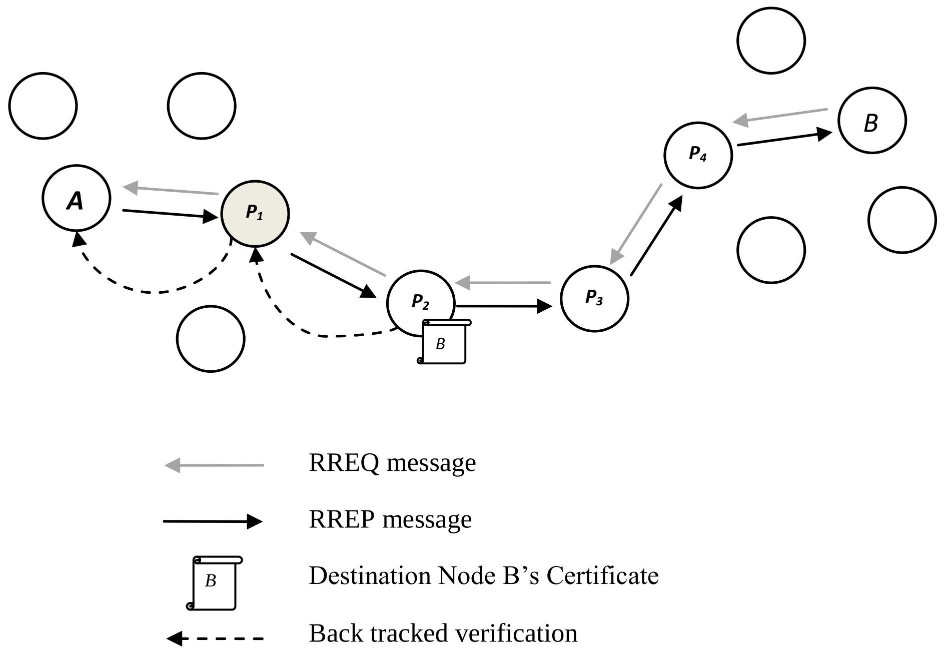


Figure 5-5: Illustration of verification protocol

5.3.5 Conduct Trust Evaluation

The provision of conduct-based trust enhances the trust decision made by nodes and thereby affects keying decisions. "Conduct trust influences decisions like access control, choice of public keys, etc. It could be useful as a complement to a public key infrastructure (PKI), where an entity would accept or reject a public key according to the trustworthiness of the entities that vouch for it; this is the idea behind PGP web of trust" [31]. It also provides trust influence at the network layer, allowing for routes to be selected based on trust. Trust Establishment incorporates the following functions: specification of evidence, generation, distribution, discovery and evaluation of trust evidence. The scope of the work focuses upon trust evaluation rather than the collecting of trust evidence from the network and the semantics of such trust evidence. These issues are still important, and need to be addressed in a complete system.

5.3.5.1 Trust Representation

The DITD model represents trust on a weighted trust graph *G(V,E)* by a trust opinion. The trust opinion is a numeric trust variable that is a function of the available confidence and trustworthiness evidence.

$$trust_i(t_{evidence}, c_{evidence}) = t_i \in [0,5] \tag{5.1}$$

A high trust opinion means that the node is a good node, or that the node provides highly accurate location information, or that the certificate issued by the node is highly trusted. Trust is further influenced by network operation confidence that includes the duration of a node's participation in the network, or the lack of negative evidence against the node.

The trust function, *trust*, computes the available evidence ($t_{evidence}$ and $c_{evidence}$) into a semantic numeric representation of trust. Trust is represented at each node or vertex of the trust graph. The work focuses upon the evaluation of routes, and the assignment of trust to individual nodes is assumed to be taken care of by a network monitoring system.

A trust variable, t_i, will be assigned and stored at each node or vertex of the trust graph. Each node entering the network with a valid self-certificate is provided with a default trust value t_d. The DITD model can be extended to include access control and allow for a trusted outside member to assign trust values to nodes entering the network. This would allow for a more secure system with limited or specified users, and still maintain the self organized nature of the network.

Trust is assigned to the established routes including both one-hop neighbouring routes and multi-hop routes. In an on-demand routing environment, nodes maintain a routing table storing the routes to each known node. A trust variable t_{AB} will be assigned to each of these routes representing the aggregated trust from the source node *A* to node *B*. The duration of time for which these routes are maintained securely will influence the weight of trust assigned to the edges of the trust graph. The trust of nodes (t_i) and trust of routes (t_{AB}) will change as the network progresses and new trust evidence is made available. The representation of trust is illustrated on weighted trust graph *G(V,E)* in Figure 5-6.

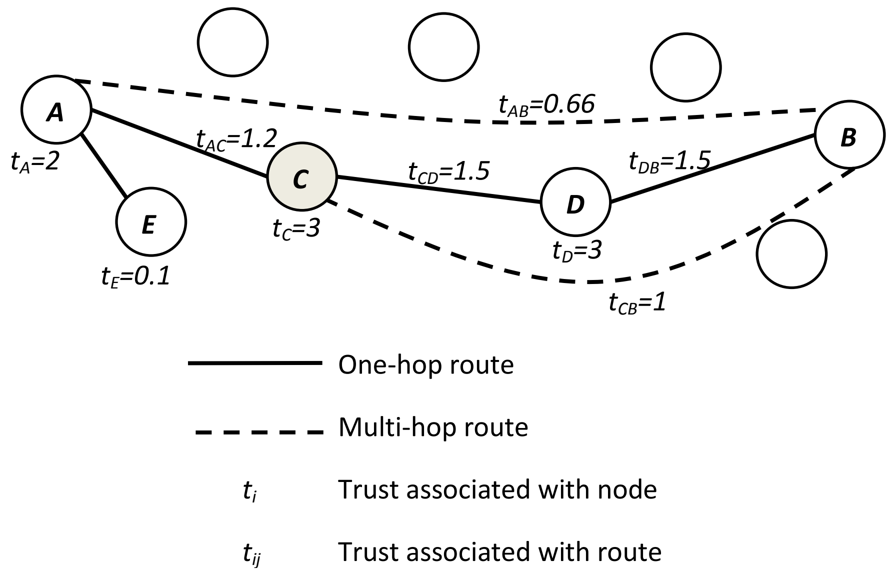


Figure 5-6: Weighted Direct Trust Graph

Certificate-based trust provides the user with binary trust, i.e. when two nodes share certificates, they trust each other or alternatively do not trust each other. DITD represents trust with a range from 0 to 5. Where 0 represents a malicious node or a node unworthy of trust, and 5 represents full confidence in the certificate and trustworthiness of the node. This gives the trust graph system some flexibility.

When inconsistent data is shared, then a trust accusation may be made against offenders reducing the trust of the node and the routes in which it participates. A proposal is made to use the route maintenance mechanism implemented by the on-demand routing protocol to help establish the confidence and trust variables. The purpose of route maintenance is to maintain the routes and to share neighbourhood information. This allows for provided trust evidence to be shared in a localized manner. This maintenance protocol allows nodes to "monitor" their neighbours and when inconsistent data is shared, a trust accusation may be made against the offender.

The DITD model inherits aspects of the semiring mathematical trust representation following semiring properties that are used for aggregating trust opinions along and across paths. The distance semiring operators $\oplus$ and $\otimes$ are applied to optimize trust accumulation. The $\otimes$ operator is used to add trust values along a trusted certificate chain. The $\otimes$ operator allows for a final trust value to be calculated representing a chain of nodes with different trust values. Trust values will be aggregated along a path like parallel resistors would be summed, i.e. $\frac{1}{R_T} = \frac{1}{R_1} + \frac{1}{R_2} + ... + \frac{1}{R_n}$. The trust value decreases along the path, and the final trust of the path can be no larger than the lowest trust value. This aligns with the description of a trust chain that states that a chain is as strong as its weakest link. The distant semiring approach is based on Eiser's proposal [50]. Figure 5-6 illustrates how trust is aggregated along a path and stored representing the trust of a specific route between node *A* and *B*.

In summary, methods are proposed to allow for a trust semantic, but this is not the focus of the work. The assumption is made that trust evidence is available and that each weighted vertex has been assigned a trust value.

5.3.5.2 Trust Evaluation

The conduct-based proposal complements the certificate exchange and verification mechanism forming a hybrid security model that embraces certificate trust establishment as well as conduct-based trust. Ideas from the modified proactive generic-single-source-shortest-distance algorithm [70, 72] are inherited, and this semiring mathematical formulae can be applied to the reactive on-demand trust path discovery phase of the routing protocol. The generic-single-source-shortest-distance algorithm calculates the shortest path from a source node to all nodes in the network, working in a proactive manner. The DTID proposal is a reactive path specific model. DITD will have a hand in the selection of the multi-hop routes. Hence, its operation will lie in the network layer. The DITD model modifies and optimizes the shortest path algorithm on the network layer. The following modifications are made to the trust path discovery phase in order to compliment the certificate-based model with a conduct-based model evaluating trust along a path.

1. **Trust is aggregated along the *RREQ* path**

The distance semiring mathematic operator $\otimes$ [72] is used to allow for trust to be calculated for a route from source node *S* through intermediate nodes *1* to *n* toward destination node *D*. Trust for this path is a function of the participating nodes.

$$trust_{SD}(t_S, t_1, t_2, t_3 ..., t_n, t_D) = t_S \otimes t_1 \otimes t_2 \otimes t_3 \otimes ... \otimes t_n \otimes t_D = t_{SD}$$

$$= \left(\frac{1}{\frac{1}{t_S} + \frac{1}{t_1} + \frac{1}{t_2} + \frac{1}{t_3} + ... + \frac{1}{t_n} + \frac{1}{t_D}} \right) \tag{5.2}$$

Trust is aggregated along the path that the *RREQ* propagates. The trust of the route is updated at every hop and the trust value is stored in the routing table of the intermediate nodes with respect to the level of trust of the reverse path to the source.

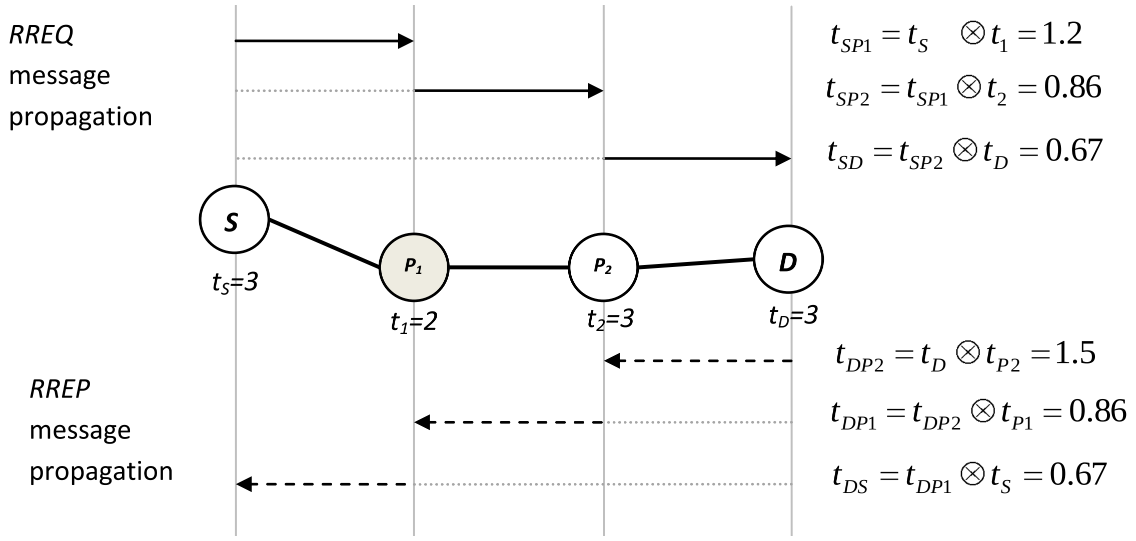
2. **Trust is aggregated along the *RREP* path**

As was the modus operandi with the RREQ path, the trust from the destination to the source is aggregated using the distance semiring formulae, as follows:

$$trust_{DS}(t_D, t_{n-1}, t_{n-2}, t_{n-3} ..., t_1, t_S) = t_D \otimes t_{n-1} \otimes t_{n-2} \otimes t_{n-3} \otimes ... \otimes t_1 \otimes t_S = t_{DS} \tag{5.3}$$

Although the total trust between the source and destination is already calculated after *RREQ*'s propagation, this step is necessary to provide appropriate trust values for the forward path recorded in the intermediate node's routing table.

The aggregation of trust is illustrated in Figure 5-7 where source node, *S*, sends a *RREQ* message to destination node *D*; and at each hop of the *RREQ* message, the trust is calculated and stored as a trust value associated with the reverse route to *S*. The trust associations are t_{SP1} , t_{SP2} and finally t_{SD} which is the trust for the route between *S* and *D*. Figure 5-7 also follows the *RREP* message calculating the trust associated with the forward routes stored in the routing table. Figure 5-7 shows that trust is route-specific, and trust must be aggregated along both the *RREP* and *RREQ* paths.



RREQ message propagation
$t_{SP1} = t_S \otimes t_1 = 1.2$
$t_{SP2} = t_{SP1} \otimes t_2 = 0.86$
$t_{SD} = t_{SP2} \otimes t_D = 0.67$
S
P₁
P₂
D
t_S=3
t_1=2
t_2=3
t_D=3
$t_{DP2} = t_D \otimes t_{P2} = 1.5$
RREP message propagation
$t_{DP1} = t_{DP2} \otimes t_{P1} = 0.86$
$t_{DS} = t_{DP1} \otimes t_S = 0.67$

Figure 5-7: Illustration of trust aggregated along RREQ and RREP path

3. Implicit revocation: Filter trust path discovery participation

The nodes that participate in the trust path discovery process must all have an acceptable value of trust. This eliminates untrusted nodes from participating in the multi-hop routes, and also eliminates low level trust paths from being discovered. Before a *RREQ* message is processed and forwarded, the aggregated trust of the propagating route request is compared to a trust threshold t_{thresh}. If the trust is lower than the t_{thresh} then the *RREQ* message is discarded.

$$t_{RREQ} < t_{thresh} \tag{5.4}$$

This procedure will act as an implicit revocation mechanism for DITD. A trust chain is as weak as its weakest link. Therefore if the weakest links are not considered, then their corresponding weak trust chains are not considered either. This modification helps find the most trusted path and reduces unnecessary network computation and message propagation.

4. Filter the most trusted path

Figure 5-8 illustrates this step. When the *RREP* is propagated back to the source node, it is very possible that multiple routes are found. Hence, an intermediate node may receive more than one *RREP* message. In this case, the first *RREP* is forwarded and successive *RREP*'s are only forwarded based on their sequence number or total trust value, effectively filtering the most recent and most trusted routes to the destination. The generic-single-source-shortest-distance algorithm would unicast *RREQ* messages in order of trust to their neighbours. By doing this, cyclic paths are avoided and the procedure of discovering the most trust path is maximised. The possibility of unicasting *RREQ* messages instead of broadcasting them is unfeasible for mobile ad hoc networks due to resource limitations. Instead, DITD's proposal of filtering the routes by sequence number and trust will effectively realizes the relaxation process of the Dijkstra's shortest path algorithm in a reactive rather than a proactive manner.

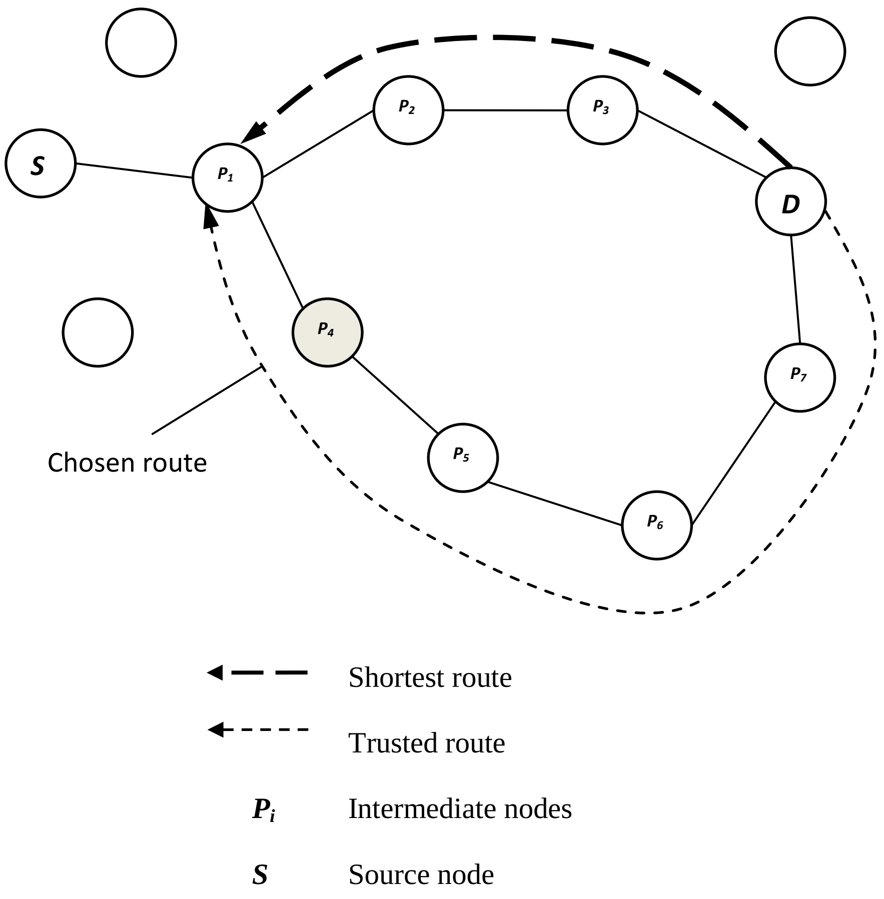


Figure 5-8: Illustration of the filtering the most trusted route

The four additives to the trust path discovery phase allow for conduct trust evaluation to be added to the on-demand routing protocol of the ad hoc network, increasing the security of trust chains created during indirect trust establishment. The conduct model is explained with an example.

We have discussed how the proposed hybrid trust scheme is incorporated into an ad hoc on-demand routing scheme with a low level complexity. Direct and indirect trust is established by localized one-hop certificate exchanges in a reactive manner, and conduct trust is appended by aggregating trust along paths. The following chapter discusses the performance of the proposed scheme with use of simulations.

Chapter 6

Performance and Simulation Study of the Proposed DITD Model

6.1 Introduction

Two basic approaches pertain to the evaluation of routing applications for mobile ad hoc networks, namely: simulations and real test beds [76, 77]. Real test beds can provide realistic results. However, they are impractical to set up. A real test bed for a large network of nodes would require 50 nodes in operation, which is considerably costly. It is also difficult to compare different protocol because of the difficulty in repeating test conditions, such as mobility and erratic wireless connectivity. Therefore, real test beds are logistically unfeasible. Currently, simulations are widely used to compare proposed routing protocols. Simulation packages such as ns2 [78] and GloMoSim [79] provide an environment to design and compare proposed and existing protocols. The majority of literature on this subject uses ns2 because its enhanced functionality is suitable for wireless scenarios. In view of these considerations, the ns2 network simulator was selected to perform a simulation study for the DITD model.

This chapter presents the effects of adding the security functionality proposed by the DITD model to the AODV routing protocol. This functionality includes a certificate distribution mechanism and a trust evaluation mechanism. The environment investigated is a large mobile ad hoc network that uses an on-demand routing algorithm.

The following section describes the simulation environment, discussing the simulation scenario, the traffic model, and the mobility model. Section 6.3 describes the performance metrics used to analyze the simulated routing protocols. The focus of this chapter is located in Section 6.4 where a comprehensive simulation study is presented. This is done by comparing the proposed DITD model with the AODV routing protocol. Results are presented in simple line graphs and discussed accordingly.

6.2 Simulation Setup

The goal of the simulation experiments was to measure the proposed routing protocol's performance to a changing network topology and network conditions. To measure this, protocols are simulated at varied mobility conditions. A comprehensive simulation study is presented of the proposed security scheme for mobile ad hoc networks implemented on the network layer. This section defines the simulation setup. A summary of the simulation set is given in Table 6-1.

6.2.1 Simulation Scenario

The network was set up with 50 wireless nodes allowing data communication to occur in a peer-to-peer manner. Nodes are mobile in a rectangular space of 1500m x 300m and the simulation is run for 900 seconds. A rectangular area is preferred to a square area as longer routes can be expected. Nodes were configured to use the 802.11b standard communicating over wireless channels with a two-ray ground radio propagation model with a bandwidth of 2Mbps and a nominal transmission range of 250m.

6.2.2 Traffic Model

Traffic was simulated using a constant bit rate (CBR) traffic generator that models UDP traffic. TCP traffic was not used because it uses its own flow control mechanism which schedules data packets based on the network's ability to carry them. CBR traffic is more useful for a routing protocol analysis as it allows the routing protocol to manage the flow of traffic. All traffic is started within the first 180 seconds of the simulation. Simulations were performed with data packets sizes of 64, 256, 512 and 1024 bytes. At higher data packet sizes, traffic congestion causes a few nodes to drop most of their received packets, as was observed from test simulation runs. A data packet size of 64 bytes was selected for the simulation analysis. The focus of the simulation study is to compare the performance of routing protocols against changing topology. As no load balancing is employed in any simulated protocol, congestion is factored out by selecting a lower data packet size. The traffic analysis model is consistent with routing protocol analysis in [80].

For topology analysis, the traffic load is fixed with a rate of 4 packets per second. The maximum number of connections is set to 30 connections with a traffic model with 20 sources.

6.2.3 Mobility Model

A modified "random waypoint" mobility model was used to prevent mobility concerns highlighted in [81]. The modified random waypoint model improves upon the standard model by selecting a speed that is between 10% and 90% of the given maximum speed. This addition provides a more balanced mobility and prevents extreme drops in speed during simulation.

Changing network topology is simulated based on network participant speed. The maximum speed was varied from 0 to 30m/s with 6 different mobility patterns (0.1, 1, 5, 10, 20 and 30m/s) for two different pause time scenarios, 0 and 250 seconds, representing a network with continuous motion and a partially stable network.

6.3 Performance Metric

The following quantitative metrics are used to analyze the performance of the routing protocols in mobile ad hoc networks.

6.3.1 Packet Delivery Ratio

The packet delivery ratio (PDR) represents the percentage of data packets that are successfully received by their intended destination. This metric is also known as 'throughput' and is considered a measurement of the effectiveness of a routing protocol. The equation for PDR is defined below where $\sum_1^n CBRrec$ and $\sum_1^n CBRsent$ are the number of CBR data packets received and sent respectively.

$$PDR\% = \frac{\sum_1^n CBRrec}{\sum_1^n CBRsent} \times 100$$

6.3.2 Routing Overhead

A routing protocol uses control packets to establish routes on which data packets are transmitted. Control packets are separate from data packets but share the same communication channel. Due to the lack of channel capacity in mobile ad hoc networks, a large number of control packets can result in poor network performance. Key management would require additional control packets to achieve key management functionality, and this will be reflected in the simulations. The routing overhead is also known as a routing protocol's internal efficiency, and will represent the number of control packets used for a given protocol.

6.3.3 Average End-to-End Delay

Average end-to-end delay is a qualitative measurement of the delay of data packets. The average end-to-end delay of a data packet is the duration of the time from which it is created at the source until it arrives at the intended destination. The delay includes propagation and queuing delay. Delay can be caused by a high number of control packets propagating in the network or a high computational overhead for the given protocol. The average end-to-end delay is calculated as follows, where *CBRsendtime* and *CBRrecvtime* represent the recorded times that a CBR data packet was sent and received.

$$End\ to\ End\ Delay = \frac{\sum_1^n (CBRsendtime - CBRrecvtime)}{\sum_1^n CBRrec}$$

Table 6-1: Simulation Setup for varied topology

Simulation Scenario	
Physical and MAC model	IEEE 802.11b standard
Nominal bit rate	2Mbps
Transmission Range	250m
Number of nodes	50 nodes
Simulation duration	900 seconds
Simulation area	1500m x 300m
Traffic Model	
Traffic type	CBR
Data packet size	64 byte
Traffic rate	4 packets per second
Traffic started	0 – 180 seconds
Number of connections and sources	30 and 20
Mobility Model	
Model	Random Waypoint
Max speed	0.1 , 1, 5, 10, 20, 30 m/s
Pause time	0 and 250 seconds

6.4 DITD Simulation study

6.4.1 Implementation

A linux based server was set up to run the Network Simulator ns-2.31 [78]. A routing protocol was designed in C++ based on the AODV routing protocol available in the ns-2.31 package. The routing protocol DITD is programmed as a routing agent class. The routing agent handles the establishment of routes, certificate distribution and trust evaluation. Modifications are made to the AODV routing agent at the *RecvRequest*, *SendRequest*, *RecvReply*, and *SendReply* functions. These modifications allow for the distribution of separate certificate packets, triggered by the routing packets. The routing agent's packet header was modified to include a certificate control packet *CertS*. The size of the certificate included is 450 bytes which correlates with experiments in [69]. The size of the certificate control packets is increased resulting in an effective delay in communication simulating the transfer of actual certificates. The authors of [65] use a similar approach to simulate the effect of security processing. A certificate table is included at each node *CertTable* which is updated by certificate control packets. The certificate table is linked to the routing table, and each node is responsible for managing its own certificate table.

The trust evaluation scheme assumes that monitoring trust evidence is available. Routing control packets are modified to include an associated trust variable. As each routing packet propagates through the network, the trust of the specific route is calculated and stored in the routing table of each node. Implicit trust revocation and trust path selection is performed at the *RecvRequest* and *RecvReply* functions respectively.

A simulation *tcl* file is written to set up the mobile ad hoc network's desired simulation scenario, traffic and mobility model. The trace support files in ns-2.31 were modified to support the DITD routing agent, allowing the inclusion of certificate control packets and trust information. As a result, the output trace and nam files reflect the operation of the DITD routing agent. Figure 6-1 shows a sample output of the nam simulation file, and Figure 6-2 shows a sample trace file output. AWK, an extremely versatile programming language for unix based systems, was used to write script files to analyze the trace data and provide the measured performance metrics. Finally, unix-based shell script files were written to allow for multiple iterations and simulation scenarios to be run simultaneously resulting in over 1000 simulation runs and 430 Gb of data analyzed and presented in simple line graphs.

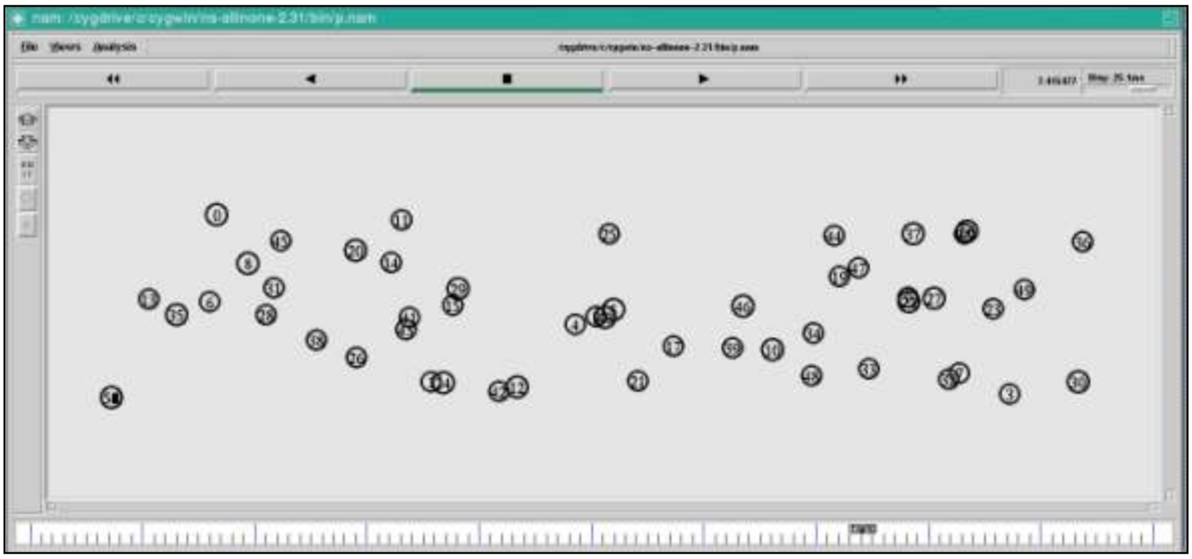


Figure 6-1: Sample nam simulation file illustrating typical network topology

```
r 19.867 _9_ RTR  --- 0 AODV 60 [0 ffffffff 16 800] --- [22:255 -1:255 26 0] [0x2 5 1 [19 0] [17 20]] 20 (REQUEST)
r 19.867 _43_ RTR  --- 0 AODV 60 [0 ffffffff 16 800] --- [22:255 -1:255 26 0] [0x2 5 1 [19 0] [17 20]] 20 (REQUEST)
r 19.868 _25_ RTR  --- 0 AODV 508 [13a 19 27 800] --- [39:255 25:255 1 25] [0x14 [25 32] [39] 10 [0 0] 4 0 1] (CERT_R)
r 19.871 _13_ RTR  --- 125 cbr 84 [13a d 11 800] --- [17:3 19:0 30 13] [0] 1 3
f 19.8711 _13_ RTR  --- 125 cbr 84 [13a d 11 800] --- [17:3 19:0 29 39] [0] 1 3
r 19.8724 _26_ RTR  --- 0 AODV 508 [13a 1a 28 800] --- [40:255 26:255 1 26] [0x14 [26 28] [40] 10 [0 0] 4 0 1] (CERT_R)
r 19.8733 _39_ RTR  --- 125 cbr 84 [13a 27 d 800] ---[17:3 19:0 29 39] [0] 2 3
f 19.8733 _39_ RTR  --- 125 cbr 84 [13a 27 d 800] --- [17:3 19:0 28 19] [0] 2 3
r 19.877 _48_ RTR  --- 0 AODV 508 [13a 30 12 800] --- [18:255 48:255 1 48] [0x12 [48 30] [18] 10.00 [18 17] 1] (CERT_S)
```

Figure 6-2: Sample trace file output for DITD simulation run

6.4.2 DITD Performance Results

The DITD model is compared with the AODV routing protocol. Further comparisons are presented against a conventional approach to key distribution. The simulation scenario used is described in Section 6.2 which is consistent throughout the simulation study. The traffic model simulates a moderate traffic load at a rate of 4 packets per second. The effects of changing topology is investigated by varying the node speed for a continuously moving network and a partially stable network. The simulation results were averaged at over 10 seeds per scenario, resulting in a total of 360 iterations for the speed analysis.

6.4.2.1 Packet delivery

The packet delivery results for the AODV and DITD routing protocols are presented in Figure 6-3 and Figure 6-4. Figure 6-3 represents a simulation environment with a pause time of 0 seconds. This represents a network of nodes that are continually moving, while Figure 6-4 represents a partially stable network. The observation is made that as the speed increases, both protocols' throughput decreases. At high speeds, the network topology changes rapidly causing breakages in routing links. The reduction in packet delivery at high speeds is because both protocols will drop data packets as a result of increased routing breakages. The curves for the AODV and DITD packet delivery ratio have similar shapes. This is expected because the DITD model is based on the AODV model. In Figure 6-3, the DITD model shows a 0–10% reduction gap in packet delivery when compared to the AODV model. This reduction gap is as a result of DITD's extra security packets producing more traffic congestion and lower packet delivery for the DITD model. The gap increases uniformly as the speed increases, leveling at 10% for speeds of 20 m/s and higher. Likewise for the more stable network presented in Figure 6-4, there is a reduction in packet delivery ratio of 0-5% when compared to the AODV model. The stable network in Figure 6-4 shows better performance at higher speeds because the number of route link breakages is reduced as a result of a larger pause time. A large pause time represents a network that will move at a given speed, then pause in a fixed location for a set amount of time.

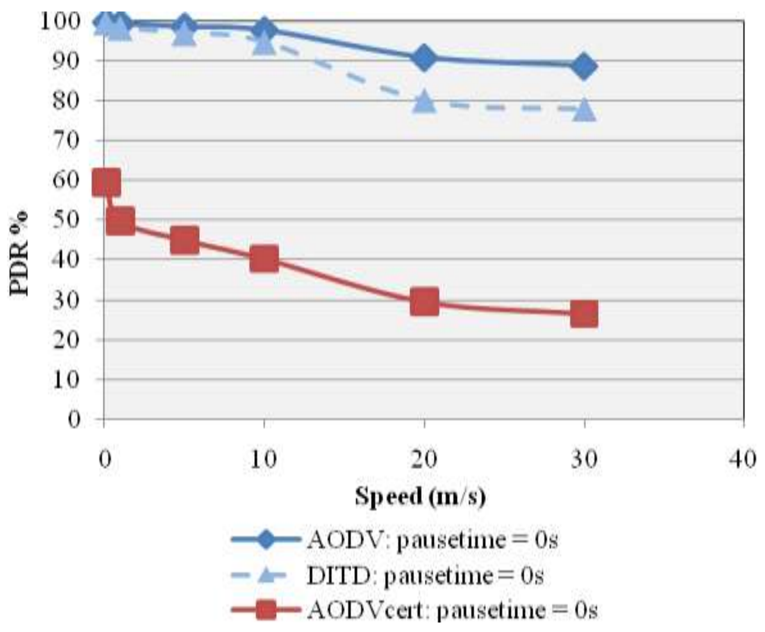


Figure 6-3: Packet Delivery Ratio for highly mobile network (0 second pause time)

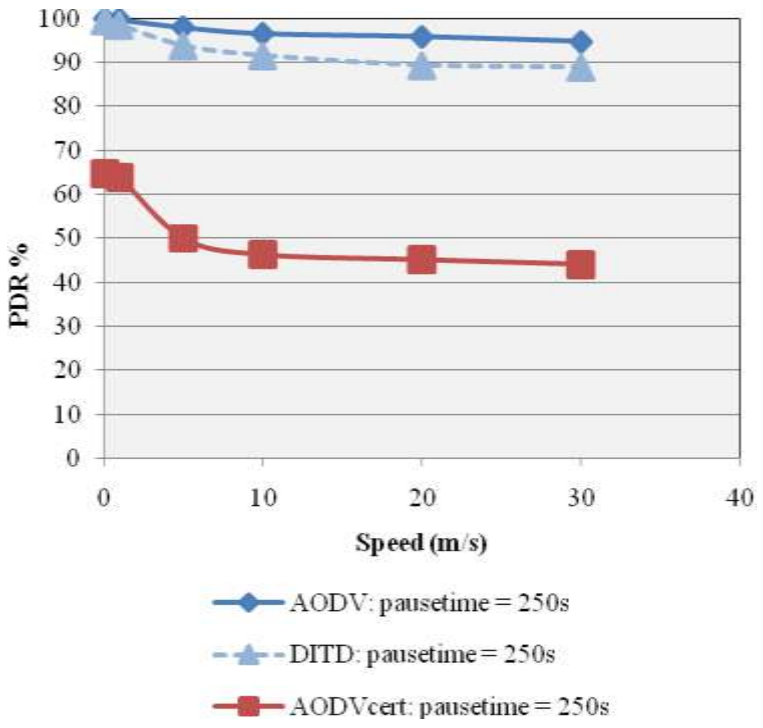


Figure 6-4: Packet Delivery Ratio for partially stable network (250 second pause time)

During this time routing link breakages are not expected until movement commences again. The reduction in packet delivery ratio of DITD, when compared to AODV, can be attributed to the additional certificate packets being distributed and handled by the routing agent. The packet queue for the routing protocol has a limited capacity and when it is overloaded, packets are dropped. This will cause a resultant drop in throughput. The DITD model optimizes its throughput by processing the routing and certificate control packets independently of each other.

A certificate distribution scheme would expect a severe reduction in performance due to an excessive number of packets being transmitted in the network or the additional size of the control packet. A conventional certificate distribution scheme, suggested as a possible solution in [62], simply includes the source's certificate in the request packets *RREQ* and includes the destination's certificate in the reply packets *RREP*. This method was implemented as a separate routing agent *AODVcert* in ns2. A similar method is suggested in [61]. Implementation includes increasing the packet size of the routing control packets to include a 450 byte certificate. This effectively increased the regular 56 byte AODV route control packets to 506 bytes. Such an approach would result in the simplest method of certificate distribution but the result of transmitting 450 bytes more data per control packet would severely reduce the network performance. The *AODVcert* routing agent was simulated under the same simulation conditions as AODV and DITD, and the packet delivery ratio is presented in Figure 6-3 and Figure 6-4. It can be observed that the packet delivery ratio is severely less than both the AODV and DITD model. For a pause time of 0 seconds, there is an average gap of 55% between *AODVcert* and AODV and an average gap of 49% between *AODVcert* and DITD. Similar results are observed for the stable network in Figure 6-4. This simulation shows that DITD optimizes the distribution of certificates by sending them as separate certificate control packets independent of the route control packets. The certificate control packets are processed independently of the routing packets, allowing concurrent processing in a fully distributive system. The operation of DITD allows for certificate distribution with minimal effect upon the routing procedure.

Figure 6-3 shows that the DITD model has a 10% reduction in throughput for high speed mobile ad hoc networks. A high speed network is described by a maximum node speed of 20 and 30 m/s. This simulates mobile units travelling at a maximum speed of 70–100km/h which is typical of mobile military vehicles. Mobility aids the distribution of certificates as nodes come in close contact with each other and are able to establish direct trust relations thereby reducing end-to-end certificate distribution. These benefits are similar to Capkun's solution which relies upon mobility to establish trust in a localized manner [10]. Capkun's solution is aided by mobility but is also dependent upon mobility for trust relations to be established. Because of this dependency, a period of weakened security is expected as nodes exchange certificates. DITD does not only distribute certificates in a localized manner, but Figure 6-3 shows that the DITD model has a 0 - 3% reduction in throughput for low speed mobile ad hoc networks where nodes move at a maximum speed of 0–10 m/s. This type of network is typical of infantry units or a man-on- the-ground scenario. DITD allows for mobility to aid the distribution of certificates but does not rely on mobility for throughput success. This allows DITD to operate successfully in slow moving and stationary-type networks. The packet delivery ratio results show that DITD provides certificate distribution at a low performance cost for high speed networks and for low speed networks.

6.4.2.2 Control Packet Overhead

The control packet overhead presents a comparison between the AODV and DITD models. The overhead is presented in terms of the number control packets. The AODV model will have only routing control packets, while the DITD model will have both routing and certificate packets. The results are presented in Figure 6-5 and Figure 6-6 for a highly mobile network with a pause time of 0 seconds and a partially stable network with a pause time of 250 seconds. The DITD model aims to distribute certificates while routes are discovered and a resultant packet overhead is expected. AODV and DITD are similar in shape and it is observed that the number of control packets increases as the speed increases. As the speed increases, the topology of the network changes more rapidly, causing routing link breakages and forcing nodes requesting communication to re-establish routes by send new route request messages. For a partially stable network presented in Figure 6-6, the effects of speed are reduced. This confirms that a larger pause time provides a more stable network. Figure 6-5 and Figure 6-6 show a consistent control packet overhead for the DITD model. It is observed that the gradient of DITD's packet overhead decreases as speed increases. This is because mobility aids certificate distribution and as the speed increases, less certificate control packets are required. For example in Figure 6-5 at the low speed of 1 m/s, there is a 132% increase in the number packets when compared to the AODV protocol. This overhead decreases for higher speeds, showing a comparative 38% and 33% packet overhead for speeds of 20 m/s and 30 m/s respectively. This confirms that mobility aids certificate distribution.

A standard AODV request message is 48 bytes and a reply message is 44 bytes. The DITD model uses request messages of 60 bytes and reply messages of 56 bytes. Therefore, DITD increases the routing control packet size by 12 bytes. DITD's routing control packets contain trust-associated variables and flags to trigger back-tracked certificate distribution. The DITD certificate control packets are 508 bytes in size as they include a 450 byte certificate. It is noted that making the routing and certificate control packets separate and independent from each other has a greater impact than reducing the per byte packet overhead. This independency allows for concurrent processing of packets which is optimal in a fully distributive ad hoc network.

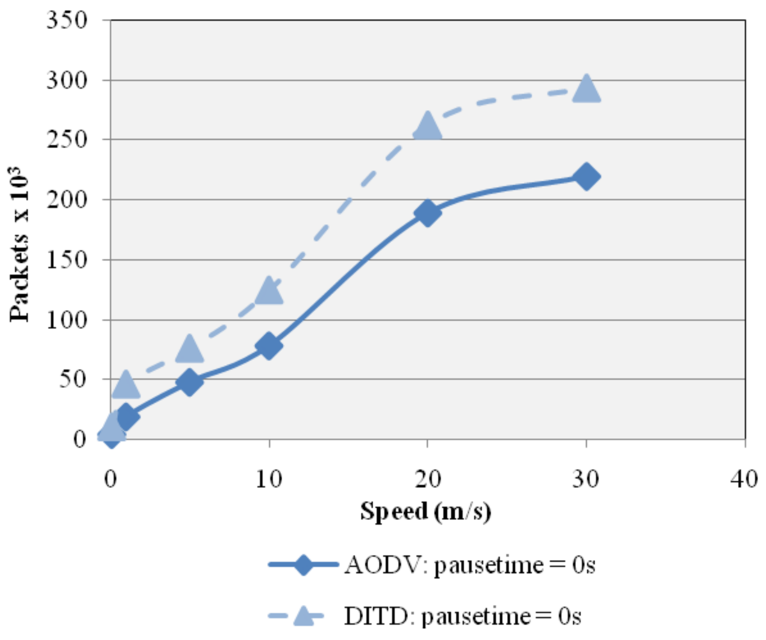


Figure 6-5: Control packet overhead for highly mobile network (0 second pause time)

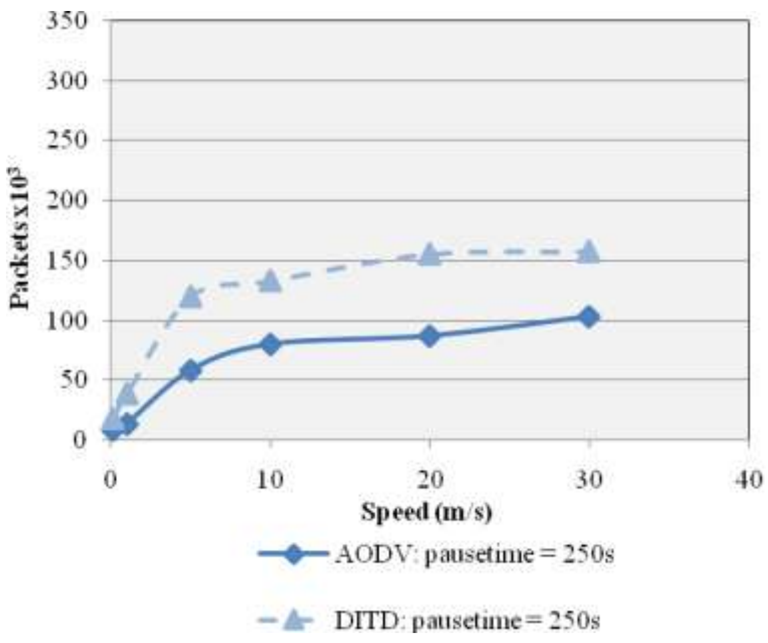


Figure 6-6: Control packet overhead for partially stable network (250 second pause time)

6.4.2.3 End-to-End Delay

The average end-to-end delay results are presented in Figure 6-7 and Figure 6-8. It is observed that the DITD model delivers packets with more delay than AODV. The additional delay is attributed to the transmission delay, the packet queuing delay, and the processing delay of additional certificate control packets. The processing delay includes verification. A conventional certificate distribution scheme that follows the route discovery process would require that certificates be verified before the routing packets are forwarded. DITD performs verifications independent of the routing procedure. The request route is established following the route request message *RREQ* to the destination, and DITD performs verifications independently without hindering the propagation of the *RREQ* message. DITD uses back-track verification to minimize the number of verifications performed on the reply route that follows the reply message *RREP* toward the source. The authors of [69] propose a solution that performs all verifications on the reply route. This method minimizes the number of verifications performed in a networks lifetime but results in delayed establishment of routes. If ECC (elliptic curve cryptography) type keys are used, the verification process could take up to 16ms per verification [69]. Such a delay is unrealistic for multi-hop routes requiring verification. DITD's approach attempts to minimize the delay incurred.

The average end-to-end delay of DITD follows a similar shape to the AODV model. It is observed that DITD has a lesser delay for low speeds. This is because the network topology is not rapidly changing and fewer packets are required to be transferred and processed. It is observed that the end-to-end delay and control packet overhead are closely related. This confirms the effect that additional control packets have on the network performance. For a high speed network with a pause time of 0 seconds and maximum speeds of greater than 20m/s, the average delay is a consistent 0.4 seconds more than AODV. For more stable networks with a pause time of 250 seconds, the average delay is reduced to an average of 0.2 seconds for speeds of 10m/s and higher. The additional delay of DITD is expected as the protocol performs certificate distribution and security evaluation which requires additional control packets to be transmitted and processed. Since packet queues are implemented in a first-in-first-out structure, the additional certificate control packets would result in data packets being queued for a longer time.

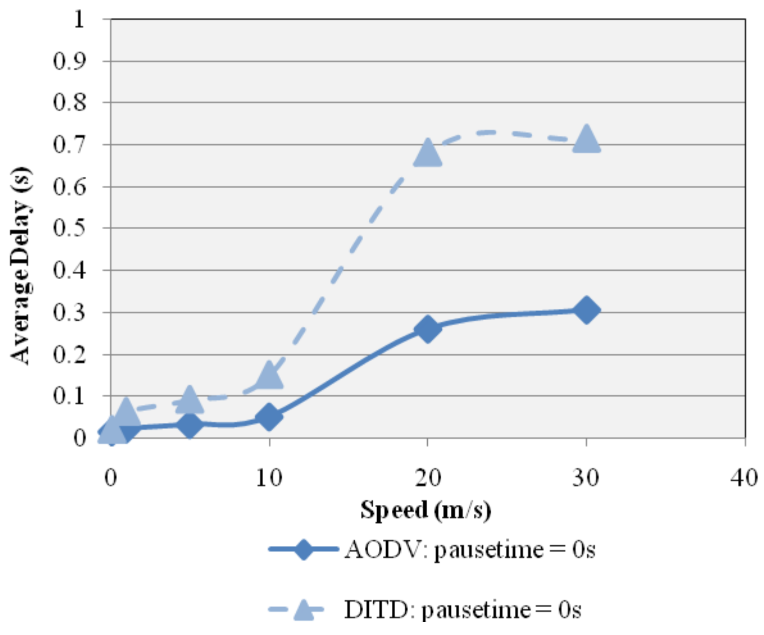


Figure 6-7: Average end-to-end delay for highly mobile network (0 second pause time)

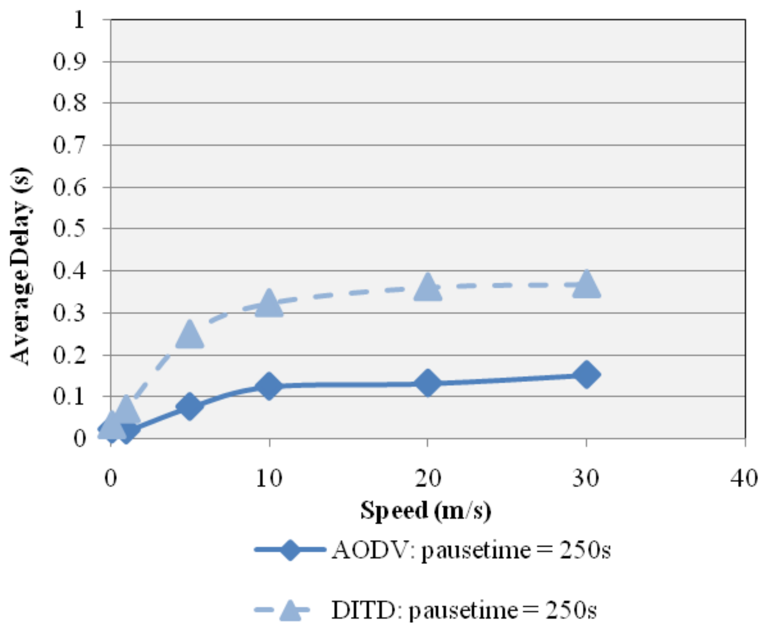
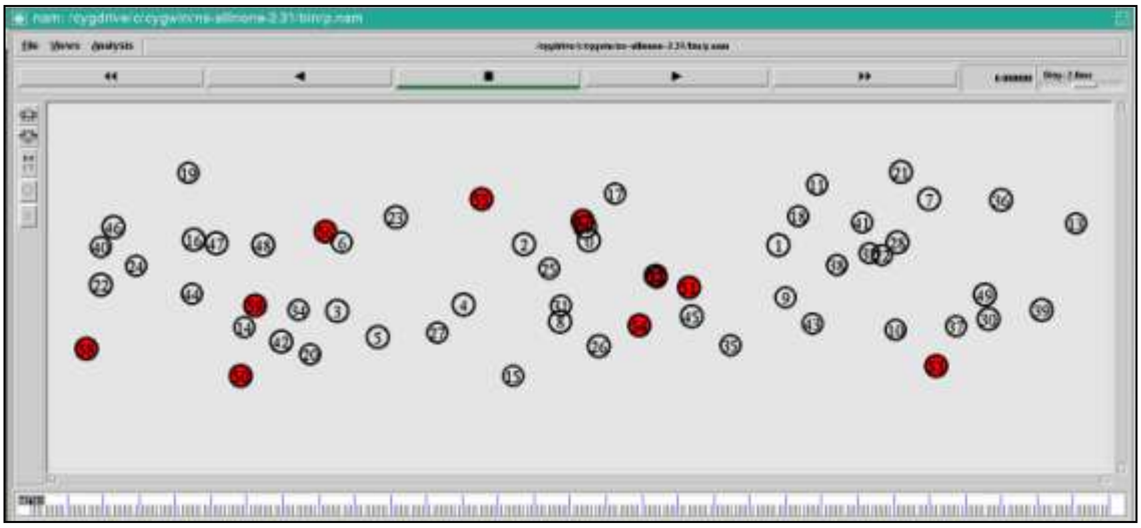


Figure 6-8: Average end-to-end delay for partially stable network (250 second pause time)



● Black hole adversary node

○ Trusted node

Figure 6-9 Sample nam simulation of black hole network simulation

6.4.3 Trust Evaluation Results

In order to test the performance of the security evaluation scheme, a black hole attack was simulated to show that DITD's security evaluation scheme excludes malicious nodes from trust and route establishment thereby protecting the network from black hole type attacks. A black hole adversary model was designed on the ns-2.31 link layer (LL) that lies below the routing layer. Modifications were made to the link layer agent *ll.cc* to simulate a black hole attack. Each packet sent by the routing layer is checked at the link layer. The adversary model silently drops all data packets while still allowing routing packets to be passed. This creates the affect of a black hole attack. A second black hole adversary model was implemented that included a rushing type attack. The rushing attack was implemented by allowing adversary nodes to forward routing packets immediately, removing the small jitter delay that AODV implements. AODV uses this small delay to reduce the number of collisions and ensure that the shortest path is selected. The rushing attack gives an adversary node a time advantage over normal nodes, resulting in the adversary node becoming part of considerably more routes.

The same simulation scenario and traffic model was used to analyse the black hole attack. The mobility was fixed with a pause time of 0 seconds, and three speeds were investigated (0.1m/s, 5m/s and 20m/s). A 50 node network was simulated with 6 different attack scenarios. The attack scenarios were created by varying the number of black hole adversary nodes added by 0 to10. Figure 6-9 shows the nam simulation file for a simulation scenario with 10 adversary nodes. Each scenario was averaged over 10 speeds, resulting in 720 iterations for the security evaluation scheme analysis. The black hole attack aims to drop data packets and reduce the networks throughput. The effects of a black hole and rushing attack are analysed using the packet delivery ratio performance metric.

6.4.3.1 Packet delivery

A black hole type problem is implemented to simulate the success of DITD's security evaluation scheme. The scenario assumes that weighted nodes carry a security metric that identifies fault detection or data transmission errors carried out by a monitoring system at each node. An example of such a system is found in [67]. The weighted nodes are used to establish a weighted trust graph where each edge or route carries a trust calculated by DITD's security evaluation scheme. The effects of the black hole attack upon AODV and DITD are compared in Figure 6-10 and Figure 6-11. It is observed that as the number of adversary nodes increases, the packet delivery ratio for the AODV model decreases. The AODV model is vulnerable to black hole attacks and in the presence of 10 adversary nodes, the packet delivery ratio is below 65%. The reduction in throughput is expected as more data packets will be dropped by the presence many adversary nodes. DITD avoids the adversary nodes by implicitly excluding these nodes during route establishment. The success of the protocol at low speeds is presented in Figure 6-10; and it is observed that even in presence of 10 adversary nodes, the packet delivery ratio is not less than 90%. Figure 6-11 presents the success of the DITD model at a higher mobility of 20m/s. The DITD model prevents the severe effects of black hole attacks, showing better results when 4 and greater than 4 adversary nodes are present. There is approximately a 10% decrease in packet delivery ratio when compared to the low mobility scenario in Figure 6-10. This reduction in packet delivery ratio is attributed to the increase in link breakages apparent at higher speeds and the overhead incurred from the certificate exchange protocol. The results of DITD in Figure 6-11 correlate with the packet delivery ratio at 20m/s in Figure 6-3.

A rushing attack was included for the simulations presented in Figure 6-12 and Figure 6-13. An adversary node equipped with a rushing type attack will participate in more routes thereby maximising the effect of its attack. Figure 6-12 and Figure 6-13 show that when adversary nodes employ a rushing attack, the effects of the black hole attack are maximised. The packet delivery ratio of the AODV protocol is dropped to 40% when 10 adversary nodes are present. This is considerably less when compared to the 60-65% packet delivery ratio that AODV experiences under the same conditions with a standalone black hole attack. The results of DITD under rushing attacks are unnoticeable when compared to DITD with no rushing attacks. For low speeds, DITD provides a throughput rate of above 90%, even in the presence of 10 adversary nodes.

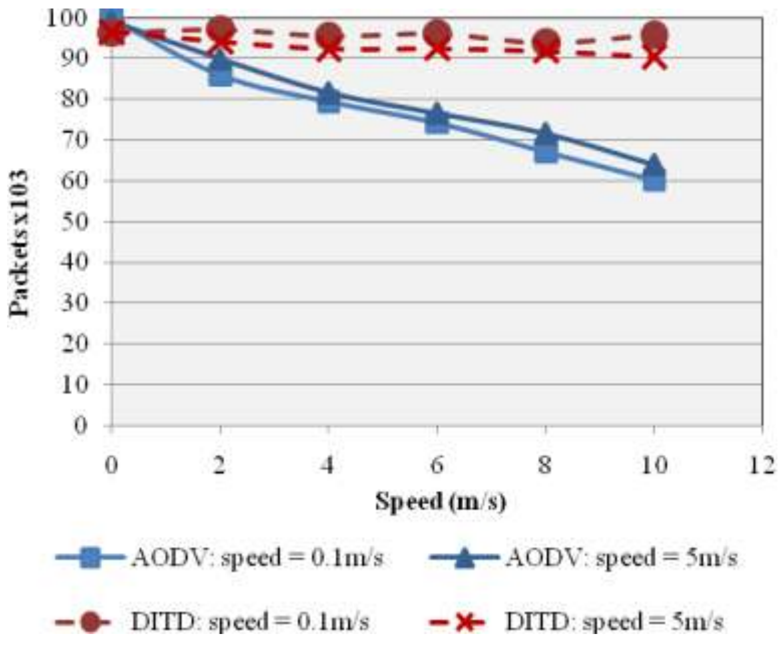


Figure 6-10: Packet Delivery Ratio for slow moving network under black hole attack

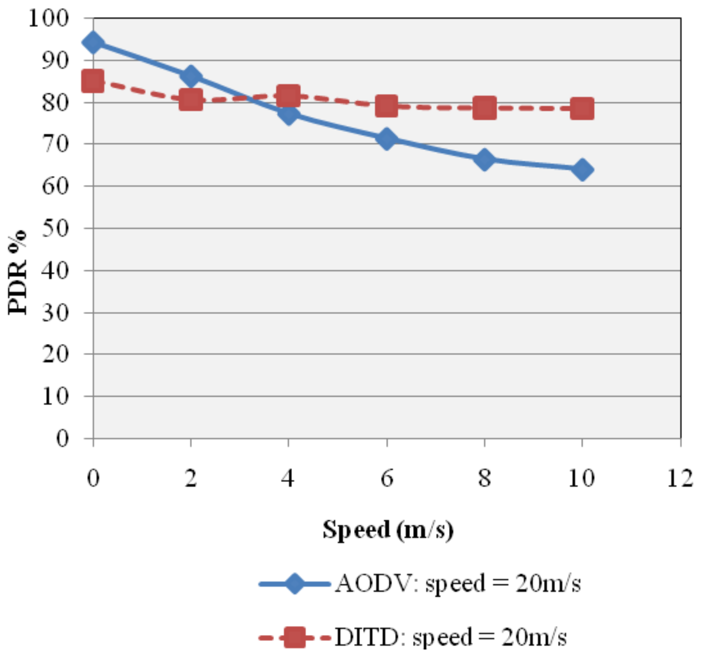


Figure 6-11: Packet Delivery Ratio for fast moving network under black hole attack

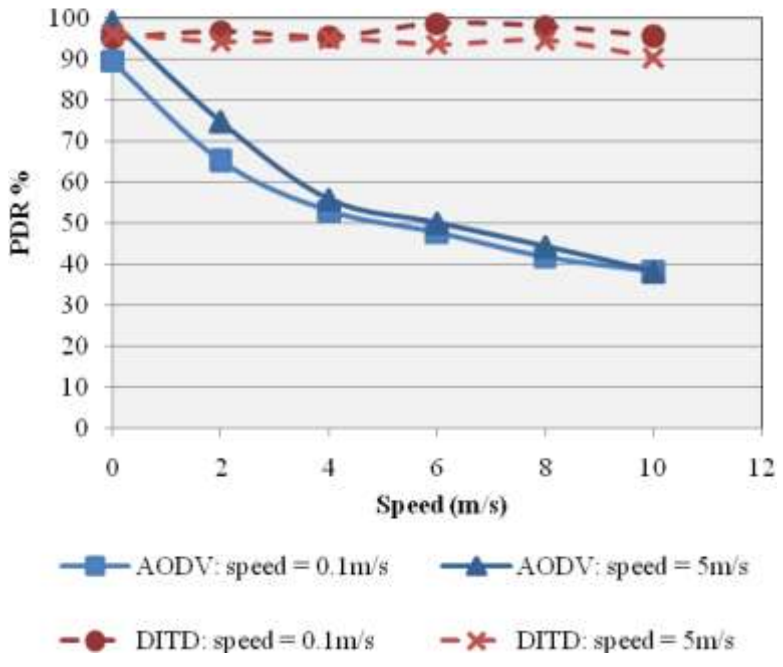


Figure 6-12: Packet Delivery Ratio for slow moving network under black hole rush attack

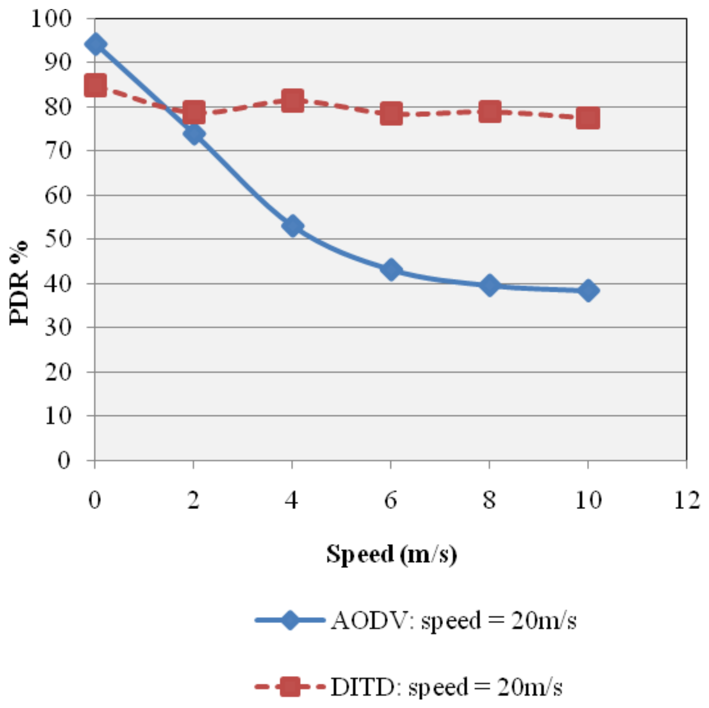


Figure 6-13: Packet Delivery Ratio for fast moving network under black hole rush attack

DITD provides a security scheme that excludes malicious nodes from participating in trusted routes, thereby preventing black hole attacks and a number of other attacks targeting the network layer. The inclusion of this trust evaluation scheme allows the distribution of certificates to operate in the most trusted routing environment.

6.5 Design verification

In this section, the DITD model will be reviewed in relation to the design requirements discussed in Chapter 5. These requirements are based on environmental factors and functionality. The design requirements are briefly revisited throughout the discussion that follows.

6.5.1.1 Environment

The DITD model is required to operate on the network layer in an on-demand, fully distributive, self-organized manner. Chapter 6.5.1 explains the implementation of DITD as an overlay of the on-demand routing protocol, AODV. Implementation was performed on the network layer, thereby avoiding multi-layer design problems. The simulation environment is set-up with no TTP member. This is similar to the way in which a certificate authority and network nodes are responsible for their own routing and trust establishment. The successful operation of DITD in the given environment is proven through simulation results, as presented in Chapter 6.

DITD is self-organized in nature. However, it is noted that DITD assumes that nodes are able to create their own keying material prior to joining the network. Self-certificates provide a strong binding between a user's key and a unique identity. The generation of keying material without the presence of a TTP is a complex problem. Solutions exist based on identity-based key generation [82, 83]. The author suggests that there is a need for further research to be carried out in this area.

6.5.1.2 Functionality

Certificate distribution is a requirement of the DITD model. DITD provides the distribution of keying material in the form of self-certificates. Local certificate exchanges are made between one-hop neighbors, thereby creating direct trust relations. These direct trust relations are chained together to share certificates across multi-hop channels. The DITD model assumes the existence of a weighted conduct value at each node. This allows the initial direct trust relations to have meaning. If this information is not available, direct trust relationships need to be established over a location-limited channel to ensure security, similar to infrared. Proximity-based solutions are used in [10, 48]. DITD's simulation model assumes the availability of conduct information. Certificates are observed in the trace table as they are transmitted successfully to their desired destinations.

A second design requirement is that DITD must minimize the network overhead. The DITD model distributes certificates that use separate unicast certificate control packets. The certificates are triggered by the routing control packets. In comparison to AODV, DITD has an approximate 38% increase in control packets for highly mobile, high speed networks. The routing control packet size is increased by 12 bytes to include trust information, and certificate control packets are 508 bytes in size. These packets result in a serve control packet overhead. The effects upon performance are reduced by: independency; concurrent processing; and back-track verification. Despite the significant control packet overhead, DITD merely reduces the packet delivery ratio by a 0-10% gap when compared to AODV. This reduction is notable if compared to a conventional certificate distribution method that increases the routing control packets by 450 bytes and results in over 50% reduction in packet delivery ratio. The performance of DITD is improved with more stable networks that have a higher pause time.

Simulations show that as the speed of nodes increases, the network performance decreases as a result of a rapidly changing topology and increased link breakages. Simulations also show that mobility aids certificate distribution. However, DITD is not reliant on mobility and can still successfully operate in low speed and stationary type networks. This allows DITD to meet the requirement to provide secure communication at the start of the network lifetime. Solutions in [10, 73] depend on mobility to establish trust and expect an initial time delay before trust is established. DITD provides secure communication in a reactive manner without a significant time delay. DITD is not limited by mobility, as it shows high throughput rates for low speed and stationary network environments.

DITD is required to be robust in spite of changing topologies. The simulations presented in chapter 6 were performed under varied pause times and speeds. This aided the investigation of the performance of DITD, under varying topology environments. The simulation results show that DITD is robust in the presence of changing mobility that will inherently have frequent routing failures. As mentioned above, DITD only reduces the throughput by a 0-10% gap across for changing topologies. It was observed that the DITD model has an approximate 0.7 second end-to-end delay (0.4 seconds greater than AODV) for high speed, highly mobile networks. This indicates that DITD is not feasible to use for audio application in highly mobile network environments. DITD's average end-to-end delay is reduced to 0.35 seconds (0.2 more than AODV) in a more stable network environment. This delay is within acceptable limits for audio application.

The last functional requirement was the inclusion of a trust evaluation scheme. The trust evaluation scheme allows for the most trusted route to be selected and for malicious nodes to be excluded from route participation. The success of the scheme is present in its prevention against black hole attacks. Simulations show that a black hole attack of 10 adversary nodes causes a 35-40% reduction in packet delivery for the AODV routing protocol. DITD avoids black hole and rushing attacks by excluding malicious nodes. In low speed networks, DITD achieves a 90-95% throughput rate in the presence of 10 adversary nodes.

6.6 Conclusion

This chapter has focused on discussing the implementation, simulation results, and design verifications of the proposed DITD security model. The purpose of the simulations is to compare the proposed DITD model with the AODV model on which DITD is based. The results obtained from the practical investigations presented in this chapter show that the proposed model can be implemented with low complexity, and that it provides the functionality of key distribution and security evaluation with trivial effects on network performance.

Chapter 7

Conclusion

7.1 Summary of Contribution

Mobile ad hoc networks allow for a new set of applications that benefit from the dynamic, autonomous, and spontaneous mobile nature, inherent to these networks. However, the very qualities that make these networks so attractive also provide designers with new security challenges.

The focus of this work is upon trust establishment in mobile ad hoc network. This dissertation contributes to the body of work in the following ways:

In the first part of the dissertation, background knowledge on mobile ad hoc networks is presented. Their application in the military and commercial arena is investigated. A review is presented of potential security attacks including black hole attacks, wormhole attacks, eavesdropping attacks, byzantine attacks, resource consumption attacks, and routing table poisoning. It was found that mobile ad hoc networks are most vulnerable to network layer attacks, and focus was placed on trust establishment on the network layer.

The second part of the dissertation provides a comprehensive survey of the existing key management solutions for mobile ad hoc networks. As respective solutions are intended for the needs of different types of ad hoc networks, their comparison is a difficult task. The solutions that are investigated are:

- Off-line Trusted Third Party Models
- Partially Distributed Certificate Authority
- Fully Distributed Certificate Authority
- Cluster-based Model
- Proximity-based Identification
- Self Issued Certificate Chaining

A discussion of the functionality and characteristics of each approach is presented. The self-issued certificate model is identified as providing the lowest level of pre-configuration and off-line trusted third party (TTP) involvement.

In the third part of the dissertation, the contribution is a secure ad hoc routing survey. This work is vital to understanding trust establishment on the network layer. The following solutions are presented:

- SEAD: Secure Efficient Ad Hoc Distance Vector Routing Protocol
- Ariadne: A secure on-demand routing protocol for ad hoc networks
- ARAN: Authenticated Routing for Ad Hoc Networks
- SAODV: Secure Ad hoc On-demand Distance Vector (SAODV)
- SLSP: Secure Link-state routing
- ODSBR: On-Demand Secure Routing Byzantine Resilient Routing Protocol
- CONFIDANT: Reputation-based solution

A comparative summary is presented focusing upon the security analysis and operational requirements of each solution. The Ariadne, ARAN, SAODV, OSRP and CONFIDANT are designed for on-demand ad hoc routing. All the protocols investigated, except the CONFIDANT protocol, are based on the assumption that pre-existing key relationships or the presence of a key management system to perform the tasks of key distribution and maintenance. The CONFIDANT protocol avoids key management by establishing trust based solely on conduct. In this part of the dissertation, an open research field is identified in the area of key management on the routing layer of mobile ad hoc networks.

The fourth part of the dissertation presents a novel security solution for mobile ad hoc networks. The solution is called Direct Indirect Trust Distribution (DITD) and is designed for an on-demand, fully distributive, self-organized, mobile ad hoc network. The scheme provides key distribution in the form of separate unicast certificate exchanges. The certificate exchange packets are independent of the routing control packets, and allow route establishment to operate concurrently but independently of trust establishment. A trust evaluation scheme is proposed that allows conduct-based trust to influence the selection of routes and implicitly exclude malicious attacking nodes. This scheme allows the keying information to be distributed in a more secure manner.

A comprehensive simulation study compares the performance of DITD and AODV, the protocol on which DITD is based. Simulation results show that under changing topologies DITD provides successful certificate distribution and trust evaluation with a minimal throughput reduction of 0-10%. Simulations show that DITD does not rely on mobility to distribute certificates and still performs in low speed communication networks. A black hole and rushing attack adversary model was designed on the link layer. Simulations show that DITD is successful in excluding malicious nodes from participating in route and trust establishment.

7.2 Future Research

Future development will be made to enhance the DITD protocol to further minimise the performance overhead. Such development would include the implementation of a load balancing agent to compliment and optimize the efficiency of DITD's key management.

The proposed model is not a standalone security solution. Future development would also include the integration of the DITD scheme with a secure ad hoc routing protocol to realize a complete security system.

The key management tasks are key distribution, key generation, key maintenance and key revocation [27]. The DITD model addresses key distribution, assuming that keys are generated by participating nodes. The generation of a secure certificate binding between a node and its public key is difficult without the presence of a trusted third party. Furthermore, the effects of adversary nodes with multiple identities performing Sybil attacks remain a problem that is difficult to solve.

Trust evaluation schemes depend upon the availability of trust evidence. Trust establishment is made up of the following services: gathering, generation, discovery and evaluation of trust evidence. This dissertation focuses upon the trust evaluation, and the indications are that future development will focus on the gathering and interpreting of trust evidence by using local network monitors.

Mobile ad hoc cluster-based networks have found increasing application in the military sector. Efficient and secure cluster-based key management is an open research area that calls for future investigation.

Bibliography

[1] "The American Heritage Dictionary of the English Language," Fourth ed: Houghton Mifflin Company, 2004.

[2] S. Capkun, L. Butty, and J.-P. Hubaux, "Self-Organized Public-Key Management for Mobile Ad Hoc Networks," *IEEE Transactions on Mobile Computing,* vol. 2, pp. 52-64, 2003.

[3] J. D. Z. Haas, B. Liang, P. Papadimitatos and S. Sajama, "Wireless ad hoc networks," in *Encyclopedia of Telecommunications* J. W. John Proakis, Ed., 2002.

[4] L. Zhou and Z. J. Haas, "Securing Ad Hoc Networks," *IEEE Network: special issue on network security,* vol. 13, pp. 24-30, 1999.

[5] H. Jameson, "Secure Military Networks: The war without weapons," 2008.

[6] I. Talzi, A. Hasler, S. Gruber, and C. Tschudin, "PermaSense: investigating permafrost with a WSN in the Swiss Alps," in *Proceedings of the 4th workshop on Embedded networked sensors* Cork, Ireland: ACM, 2007.

[7] W. C. Karl F. Rauscher, "Wireless Emergency Rescue Team (WRET) Final Report for the September 11, 2001 New York City World Trade Center Terrorist Attack," 2001.

[8] M. Raya and J. P. Hubaux, "The Security of Vehicular Ad Hoc Networks," in *proc. ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN'05)*, 2005.

[9] S. William, *Cryptography and network security (2nd ed.): principles and practice*: Prentice-Hall, Inc., 1999.

[10] S. Capkun, L. Buttyan, and J.-P. Hubaux, "Mobility Helps Peer-to-Peer Security," *IEEE Transactions on Mobile Computing,* vol. 5, pp. 43-51, 2006.

[11] L. Eschenauer and V. D. Gligor, "A Key-Management Scheme for Distributed Sensor Networks," *proc. 9th ACM Conf. on Computer and Communication Security (ACM CCS'02),* November, 17-21 2002.

[12] J. van der Merwe, D. Dawoud, and S. McDonald, "Fully Self-Organized Peer-to-Peer Key Management for Mobile Ad Hoc Networks," *proc. ACM Workshop on Wireless Security (WiSe'05),* September, 2 2005.

[13] Z. Lidong and H. Zygmunt, "Securing Ad Hoc Networks," Cornell University1999.

[14] N. B. Salem, L. Buttyan, J.-P. Hubaux, and M. Jakobsson, "Node Cooperation in Hybrid Ad Hoc Networks," *IEEE Transactions on Mobile Computing*, 2005.

[15] W. Stallings, *Cryptography and Network Security: Principles and Practices*: Prentice Hall, 2003.

[16] W. Stallings, *Cryptography and Network Security: Principles and Practice*: Pearson Education, 2002.

[17] S. Bruce, *Beyond Fear: Thinking Sensibly about Security in an Uncertain World*: Springer-Verlag New York, Inc., 2003.

[18] L. Qian, N. Song, and X. Li, "Detection of wormhole attacks in multi-path routed wireless ad hoc networks: a statistical analysis approach," *J. Netw. Comput. Appl.,* vol. 30, pp. 308-330, 2007.

[19] B. Awerbuch, D. Holmer, C. Nita-Rotaru, and H. Rubens, "An on-demand secure routing protocol resilient to byzantine failures," in *Proceedings of the 1st ACM workshop on Wireless security* Atlanta, GA, USA: ACM, 2002.

[20] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Rushing attacks and defense in wireless ad hoc network routing protocols," in *Proceedings of the 2nd ACM workshop on Wireless security* San Diego, CA, USA: ACM, 2003.

[21] C. Perkins, E. Belding-Royer, and S. Das, *Ad hoc On-Demand Distance Vector (AODV) Routing*: RFC Editor, 2003.

[22] C. E. Perkins and P. Bhagwat, "Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers," *SIGCOMM Comput. Commun. Rev.,* vol. 24, pp. 234-244, 1994.

[23] R. Molva and P. Michiardi, "A Game Theoretical Approach to Evaluate Cooperation Enforcement Mechanisms in Mobile Ad hoc Networks (extended abstract)," in *proc. Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt'03)*, 2003.

[24] S. Hashmi and J. Brooke, "Authentication Mechanisms for Mobile Ad-Hoc Networks and Resistance to Sybil Attack," in *Proceedings of the 2008 Second International Conference on Emerging Security Information, Systems and Technologies - Volume 00*: IEEE Computer Society, 2008.

[25] J. R. Douceur, "The Sybil Attack," in *Revised Papers from the First International Workshop on Peer-to-Peer Systems*: Springer-Verlag, 2002.

[26] A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook in Applied Cryptography*: CRC Press, 1996.

[27] A. C.-F. Chan, "Distributed Symmetric Key Management for Mobile Ad Hoc Networks," *proc. 23rd Conf. of the IEEE Communications Society,* March, 7-11 2004.

[28] A. J. Menezes, S. A. Vanstone, and P. C. V. Oorschot, *Handbook of Applied Cryptography*: CRC Press, Inc., 1996.

[29] A. Shamir, "How to share a secret," *Communications of the ACM,* vol. 22, pp. 612-613, 1979.

[30] M. Steiner, G. Tsudik, and M. Waidner, "Diffie-Hellman Key Distribution Extended to Groups," in *proc. Third ACM Conf. on Computer and Communication Security*, 1996.

[31] A. Abdul-Rahman, "The PGP trust model," *EDI-Forum: The Journal of Electronic Commerce,* vol. 10, pp. 27-31, 1997.

[32] C. R. Davis, "A localized trust management scheme for ad hoc networks. ," *In: 3rd International Conference on Networking (ICN'04),* pp. 671–675, 2004.

[33] R. Verma, D. O'Mahony, and H. Tewari, "NTM- Progressive Trust Negotiation in Ad Hoc Networks," 2001.

[34] S. Yi and R. Kravets, "Practical PKI for Ad Hoc Wireless Networks," Department of Computer Science, University of Illinois, Technical ReportAugust 2001.

[35] K. Aram, K. Jonathan, and A. A. William, "Toward Secure Key Distribution in Truly Ad-Hoc Networks," in *Proceedings of the 2003 Symposium on Applications and the Internet Workshops (SAINT'03 Workshops)*: IEEE Computer Society, 2003.

[36] S. Jarecki, "Proactive secret sharing and public key cryptosystems," Massachusetts Institute of Technology (MIT), 1995.

[37] J. van der Merwe, D. Dawoud, and S. McDonald, "A Proactively Secure Threshold-multisignature Scheme based on Publicly Verifiable Distributed Key Generation and Publicly Verifiable Secret Redistribution," *IEEE Transactions on Parallel and Distributed Systems,* 2004.

[38] A. Herzberg, M. Jakobsson, S. Jarecki, H. Krawczyk, and M. Yung, "Proactive Public Key and Signature Systems," *proc. 4th ACM Conf. on Computer and communications security,* April, 1-4 1997.

[39] Y. Frankel, P. Gemmell, D. MacKenzie, and M. Yung, "Optimal resilience proactive public key cryptosystems," *proc. 38th Annual Symposium on Foundations of Computer Science (FOCS '97),* October, 19-22 1997.

[40] A. Herzberg, S. Jaracki, H. Krawczyk, and M. Yung, "Proactive Secret Sharing Or: How to Cope With Perpetual Leakage," *proc. Advances in Cryptology - CRYPTO '95,* 1995.

[41] Y. Desmedt and S. Jajodia, "Redistributing Secret Shares to New Access Structures and Its Applications," Department of Information and Software Engineering, School of Information Technology and Engineering, George Mason University, Technical ReportJuly 1997.

[42] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch, "Verifiable secret sharing and achieving simultaneity in the presence of faults (extended abstract)," *proc. 26th IEEE Annual Symposium on Foundations of Computer Science*, October, 21-23 1985.

[43] S. Yi and R. Kravets, "MOCA: Mobile certificate authority for wireless ad hoc networks," in *proc. of the 2nd Annual PKI Research Workshop (PKI 2003)*, 2003.

[44] H. Luo and S. Lu, "Ubiquitous and robust authentication services for ad hoc wireless networks," Computer Science Department, University of California, Technical ReportOctober 2000.

[45] H. Luo, P. Zerfos, J. Kong, S. Lu, and L. Zhang, "Self-securing Ad Hoc Wireless Networks," *proc. Seventh International Symposium on Computers and Communications (ISCC'02)*, July 1-4 2002.

[46] S. Basagni, K. Herrin, D. Bruschi, and E. Rosti, "Secure pebblenets," in *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking \&amp; computing* Long Beach, CA, USA: ACM, 2001.

[47] D. B. Smetters, D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong, "Talking To Strangers: Authentication in Ad-Hoc Wireless Networks," 2002.

[48] A. Scannell, A. Varshavsky, A. LaMarca, and E. D. Lara, "Proximity-based authentication of mobile devices," *Int. J. Secur. Netw.*, vol. 4, pp. 4-16, 2009.

[49] J.-P. Hubaux, L. Butty, and S. Capkun, "The quest for security in mobile ad hoc networks," in *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking \&amp; computing* Long Beach, CA, USA: ACM, 2001.

[50] J. V. D. Merwe, D. Dawoud, and S. McDonald, "A survey on peer-to-peer key management for mobile ad hoc networks," *ACM Comput. Surv.*, vol. 39, p. 1, 2007.

[51] A. M. Hegland, E. Winjum, S. F. Mjølsnes, C. Rong, O. Kure, and P. Spilling, "A survey of key management in ad hoc networks," *IEEE Communications Surveys and Tutorials*, vol. 8, pp. 48-66, 2006.

[52] D. B. Johnson, D. A. Maltz, and J. Broch, "DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks," in *In Ad Hoc Networking, edited by Charles E. Perkins, Chapter 5*, 2001, pp. 139-172.

[53] D. Wang, M. Hu, and H. Zhi, "A Survey of Secure Routing in Ad Hoc Networks," in *Proceedings of the 2008 The Ninth International Conference on Web-Age Information Management*: IEEE Computer Society, 2008.

[54] P. Papadimitratos and Z. Haas, "Secure Routing for Mobile Ad hoc Networks," in *SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002)* San Antonio, TX, USA: , 2002.

[55] Y.-C. Hu, D. B. Johnson, and A. Perrig, "SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad Hoc Networks," in *Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications*: IEEE Computer Society, 2002.

[56] R. Rivest, *The MD5 Message-Digest Algorithm*: RFC Editor, 1992.

[57] F. I. P. S. Publications, "Secure Hash Standard (SHS)," National Institute of Standards and TechnologyOctober 2008.

[58] A. Perrig, R. Canetti, D. Song, and D. Tygar, "Efficient and Secure Source Authentication for Multicast," Network and Distributed System Security Symposium (NDSS'01), 2001.

[59] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Ariadne: a secure on-demand routing protocol for ad hoc networks," *Wirel. Netw.*, vol. 11, pp. 21-38, 2005.

[60] Y. C. Hu, A. Perrig, and D. B. Johnson, "Packet leashes: a defense against wormhole attacks in wireless networks," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*. vol. 3, 2003, pp. 1976-1986 vol.3.

[61] K. Sanzgiri, B. Dahill, B. N. Levine, C. Shields, and E. M. Belding-Royer, "A Secure Routing Protocol for Ad Hoc Networks," in *Proceedings of the 10th IEEE International Conference on Network Protocols*: IEEE Computer Society, 2002.

[62] M. G. Zapata, "Secure ad hoc on-demand distance vector routing," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 6, pp. 106-107, 2002.

[63] P. Papadimitratos and Z. J. Haas, "Secure Link State Routing for Mobile Ad Hoc Networks," in *Proceedings of the 2003 Symposium on Applications and the Internet Workshops (SAINT'03 Workshops)*: IEEE Computer Society, 2003.

[64] Z. J. Haas and M. R. Pearlman, "The performance of query control schemes for the zone routing protocol," *IEEE/ACM Trans. Netw.*, vol. 9, pp. 427-438, 2001.

[65] B. Awerbuch, R. Curtmola, D. Holmer, C. Nita-Rotaru, and H. Rubens, "ODSBR: An on-demand secure Byzantine resilient routing protocol for wireless ad hoc networks," *ACM Trans. Inf. Syst. Secur.*, vol. 10, pp. 1-35, 2008.

[66] P. Papadimitratos and Z. J. Haas, "Secure Routing for Mobile Ad Hoc Networks," in *proc. SCS Communication Network and Distributed System Modeling and Simulation Conf. (CNDS'02)*, 2002.

[67] S. Buchegger and J.-Y. L. Boudec, "Performance analysis of the CONFIDANT protocol," in *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking \&amp; computing* Lausanne, Switzerland: ACM, 2002.

[68] T. Grandison, "Trust Management for Internet Applications," Imperial College London, 2003.

[69] M. G. Zapata, "Key management and delayed verification for ad hoc networks," *J. High Speed Netw.*, vol. 15, pp. 93-109, 2006.

[70] G. Theodorakopoulos and J. S. Baras, "On Trust Models and Trust Evaluation Metrics for Ad-Hoc Networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, pp. 318-328, 2006 2006.

[71] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, pp. 498-519, 2001.

[72] M. Mohri, "Semiring frameworks and algorithms for shortest-distance problems," *J. Autom. Lang. Comb.*, vol. 7, pp. 321-350, 2002.

[73] M. Tanabe and M. Aida, "Secure communication method in mobile wireless networks," in *Proceedings of the 1st international conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications* Innsbruck, Austria: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007.

[74] C.-Y. Tseng, P. Balasubramanyam, C. Ko, R. Limprasittiporn, J. Rowe, and K. Levitt, "A specification-based intrusion detection system for AODV," in *Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks* Fairfax, Virginia: ACM, 2003.

[75] C. R. Stephan Eichler, "Challenges of Secure Routing in MANETs: A Simulative Approach using AODV-SEC," in *Mobile Adhoc and Sensor Systems (MASS), 2006 IEEE International Conference on*, 2006.

[76] W. Kiess and M. Mauve, "A survey on real-world implementations of mobile ad-hoc networks," *Ad Hoc Netw.*, vol. 5, pp. 324-339, 2007.

[77] Q. Ke, I. David, D. Maltz, and D. B. Johnson, "Emulation of Multi-Hop Wireless Ad Hoc Networks," in *in The 7th International Workshop on Mobile Multimedia Communications (MoMuC*, 2000.

[78] "The Network Simulator," *ver 2.31, Available at* http://isi.edu/nsnam/ns/, 2007.

[79] X. Zeng, R. Bagrodia, and M. Gerla, "GloMoSim: a Library for Parallel Simulation for Large-scale Wireless Networks," in *proc. 12th Workshop on Parallel and Distributed Simulations (PADS '98)*, 1998.

[80] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," in *Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking* Dallas, Texas, United States: ACM, 1998.

[81] W. Navidi, "Stationary Distributions for the Random Waypoint Mobility Model," *IEEE Transactions on Mobile Computing*, vol. 3, pp. 99-108, 2004.

[82] A. Shamir, "Identity-Based Cryptosystems and Signature Schemes," in *proc. Advances in Cryptology: Crypto'84*, 1984.

[83] A. Weimerskirch and D. Westhoff, "Identity Certified Authentication for Ad-hoc Networks," in *proc. 1st ACM workshop on Security of ad hoc and sensor networks*, 2003.