UNIVERSITY OF STRATHCLYDE

A Robust and Efficient Key Management for Peer to Peer Mobile Ad Hoc Networks

Hisham Dahshan

A Thesis presented for the degree of
Doctor of Philosophy

Department of Electronic and Electrical Engineering

May 2010

Declaration

I, Hishm Dahshan, hereby declare that this work has not been submitted for any other degrees at this University or any other inistitution and that, except where reference is made to the work of other authors, the material presented is original.

Abstract

A mobile ad hoc network (MANET) is a self-organised wireless network where mobile nodes can communicate with each other without the use of any existing network infrastructure or centralised administration. Trust establishment and management are essential for any security framework of MANETs. However, traditional solutions to key management through accessing trusted authorities or centralised servers are infeasible for MANETs due to the absence of infrastructure, frequent mobility, and wireless link instability. There are many applications in both the military and civil environments, where combinations of authority-based MANETs and self-organised MANETs are needed.

This thesis addresses these problems by introducing a number of novel key management schemes that dynamically switches from a centralized scheme of trust distribution (authority-based MANETs) to a distributed scheme of trust (self-organised MANETs).

In the first part of this thesis, three key management schemes for MANETs have been proposed:

The first scheme represents an authority-based key management scheme for MANETs. This scheme distributes public key certificates and symmetric keys between communicating entities by exploiting the route request (RREQ) and route reply (RREP) messages respectively.

The second proposed scheme represents a robust self-organised public key management for MANETs. It allows each user to create its public key and the corresponding private key, to issue certificates to neighbouring nodes, and to perform public key authentication through at least two independent certificate chains without relying on any centralised authority.

The third proposed scheme represents a combination between an authority-based and self-organised schemes. In this scheme, there are two approaches for key management. The first approach uses threshold cryptography while the second approach provides key management through a web of trust. A mobile node can use one of the two approaches independently and can also use the two approaches together in order to obtain the authenticated public keys of other nodes in the network.

In the second part of this thesis, an elliptic curve threshold key management scheme for MANETs is proposed. In this scheme, an off-line central authority is required in the initialisation phase before network deployment. The central authority creates and preloads the shares matrix for each mobile node in the network initialisation phase. In the network deployment phase, each session member generates its private/public key pair, and the session public key, by collaborating with its trusted neighbours. The generation of the node private/public key pair, and the session public key, is performed without any prior communication between session members.

Acknowledgements

It is a pleasure to thank the many people who made this thesis possible. I would like to begin with thanking my supervisor Dr. James Irvine for giving me an excellent opportunity to carry out a very innovating and challenging research on the security of mobile ad hoc networks. Dr. Irvine provided me endless source of ideas and encouragement. His assistance and instructions during my time at University of Strathclyde has been invaluable.

I am indebted to all my colleagues at University of Strathclyde, especially Dr. Christos Tachtatzis for his feedback and advice. Great thanks to Dr. Swee Goo for her endless support and providing me valuable comments during the course of my research. Special thanks to Dr. Mohammad Haseeb Zafar and Dr. Samir Bendoukha for their support and friendships.

I would like to extend my sincere gratitude to my employer and sponsor The Egyptian Ministry of Defence, Cairo, Egypt for giving me the opportunity to pursue my PhD at the University of Strathclyde.

My deepest and sincere thanks are always to my brothers, parent-in-laws, brother-in-laws for their constant support and to my beloved brother who passed away while I was in my third year of my PhD.

I owe my loving thanks to my wife Aisha and children Mohammed, Malak, and Ahmed without their understanding, endless patience and encouragement it would have been impossible for me to finish this work. I am so lucky and so proud to have such a wonderful family.

Lastly, and most importantly, this thesis is dedicated to those who make the paradise of God under their feet My Father and My Mother.

Publications

Journal Paper

1. H. Dahshan, and J. Irvine, " A robust self-organized public key management for mobile ad hoc networks", Wiley InterScience, Security and Communication Networks Journal, vol. 3, no. 1, pp. 16 – 30, 2009.

2. H. Dahshan, and J. Irvine, " An Elliptic Curve Secret Sharing Key Management Scheme for Mobile Ad Hoc Networks", submitted to Wiley InterScience, Security and Communication Networks Journal, Special Issue on Security in Computer and Cyber-Physical Systems and Networks.

Conference Papers

1. H. Dahshan, and J. Irvine, " A Threshold Key Management Scheme for Mobile Ad Hoc Networks Using Elliptic Curve Dlog-Based Cryptosystem", IEEE CNSR 2010 , Montreal, Canada, May 11–14, 2010.

2. H. Dahshan, and J. Irvine, "An Elliptic Curve Distributed Key Management for Mobile Ad Hoc Networks", IEEE WiVEC 2010, Taipei, Taiwan, May 16–17, 2010.

3. H. Dahshan, and J. Irvine, " A Robust and Redundant Key Management for Mobile Ad Hoc Networks ", IEEE ISWCS'09, Siena, Italy, September 7-10, 2009.

4. H. Dahshan, and J. Irvine, " A Trust Based Threshold Cryptography Key Management For Mobile Ad Hoc Networks ", IEEE VTC 2009 Fall, Anchorage, Alaska, USA, September 20–23, 2009.

5. H. Dahshan, and J. Irvine, " Key Management in Web of Trust for Mobile Ad Hoc Networks “, IEEE AINA 2009, Bradford, UK, May 26-29, 2009.

6. H. Dahshan, and J. Irvine, " On Demand Self-Organized Public Key Management for Mobile Ad Hoc Networks “, IEEE VTC 2009 Spring, Barcelona, Spain, April 26–29, 2009.

7. H. Dahshan, and J. Irvine, "Authenticated Symmetric Key Distribution For Mobile Ad Hoc Networks ", IEEE MASS 2008, Atlanta, Georgia, USA , September, 2008.

8. H. Dahshan, and J. Irvine, "Analysis of Key Distribution in Mobile Ad Hoc Networks Based on Message Relaying", (IEEE WiMob 2008), France, October 2008.

Contents

List of Figures

List of Tables

Abbreviations and Acronyms

Abbreveations

ADV	Adaptive Distance Vector Routing
AHL	Average Hop Length
AKM	Autonomous Key Management
AODV	Ad Hoc On Demand Distance Vector Routing Protocol
ARP	Address Resolution Protocol
ASKD	Authenticated Symmetric Key Distribution
CA	Central Authority
CBR	Constant Bit Rate
CDR	Certificate Delivery Ratio
CEDAR	Core-Extraction Distributed Ad Hoc Routing
CRL	Certificate Revocation List
DCA	Distributed Certificate Authority
DH	Diffie-Hellman Algorithm
DKG	Distributed Key Generation
DLP	Discrete Logarithm Problem

DSA	Digital Signature Algorithm
DSDV	Destination Sequenced Distance Vector Routing Protocol
DSR	Dynamic Source Routing
DSSS	Direct-Sequence Spread Spectrum
ECC	Elliptic Curve Cryptography
ECDLP	Elliptic Curve Discrete Logarithm Problem
EPROM	Erasable Programmable Read-Only Memory
FHSS	Frequency-Hopping Spread Spectrum
FIFO	First-In First-Out
GHS	Gaudry-Hess-Smart
GloMoSim	Global Mobile Information Systems Simulation Library
GSR	Global State Routing
HiperLAN	HIgh-PErformance Radio Local Area Network
HMAC	Hashed Message Authentication Code
IBC	Identity-Based Cryptography
IETF	Internet Engineering Task Force
IFP	Integer Factorisation Problem
IFQ	Interface Queue
ISM	Industrial, Scientific and Medical
KDC	Key Distribution Center
KGC	Key Generation Center
KPD	Key Pre-distribution

LL	Link Layer
MAC	Medium Access Control or Message Authentication Code
MOCA	Mobile Certificate Authority
NAM	Network Animator
NetIF	Network Interface
NS-2	Network Simulator (version 2)
OCSP	On-line Certificate Status Protocol
OFDM	Orthogonal Frequency-Division Multiplexing
OMNeT++	Objective Modular Network Testbed in C++
OpenSSL	Open Secure Socket Layer
OPNET	Optimized Network Engineering Tool
OSI	Open Systems Interconnection
OTcl	Object Tool command language
PAN	Personal Area Network
PDA	Personal Data Assistance
PDR	Packet Delivery Ratio
PGP	Pretty Good Privacy
PHY	Physical
PKI	Public Key Infrastructure
RAM	Random Access Memory
RFC	Request For Comments
ROM	Read-Only Memory

RREP	Route Reply
RREQ	Route Request
RSA	Rivest Shamir Adleman Algorithm
SA	Security Association
SHA	Secure Hash Algorithm
SKD	Symmetric Key Delivery Ratio
SKG	Session Key Generation
SKiMPy	Simple Key Management Protocol for MANETs
SLSR	Secure Link State Routing
TC	Threshold Cryptography
TORA	Temporally Ordered Routing Algorithm
TTP	Trusted Third Party
VSS	Verifiable Secret Sharing
WAP	Wireless Application Protocol
WEP	Wired Equivalent Privacy Protocol
Wi-Fi	Wireless-Fidelity
WiMAX	Worldwide Interoperability for Microwave Access
WLAN	Wireless Local Area Network
WPAN	Wireless Personal Area Network
WRP	Wireless Routing Protocol
ZigBee	Zonal Intercommunication Global-standard, where Battery life was long, which was Economical to deploy, and which exhibited Efficient use of resources

ZRP	Zone Routing Protocol

Symbols

β_{il}	Node u_i contribution of the new member share sent to node u_l
$\mathbb{G}$	A Group of elements in number theory
$\mathscr{O}$	Elliptic Curve point at infinity
$\odot$	Elliptic curve point multiplication by a scalar
B	Number of secrets chosen by the *CA*
Enc_{PK_i}	Encryption using the public key of node u_i
G	Base point of elliptic curve
g	Generator of a Group
$GF(p)$	Galois Field with p elements
$H(x)$	One way hash function
k	Threshold number of nodes required to form a session
lcm	Least common multiple
n	Total number of nodes in the network
pk_i	The public key of node u_i in a session
$PV_l^{(j)}$	Public share generated by node u_j for node u_l
$s_{kh}^{(i)}$	Share of the secret S_{kh} given to node u_i
S_r	A secret of order r chosen by the *CA*
Sig_j	Singnature of node u_j
sk_i	The private key of node u_i in a session
SS_{PK}	Session public key
t	Threshold number of nodes required to recover a shared secret

W	Group of session members
$Z_l^{(j)}$	Secret share generated by node u_j for node u_l
$\sum^{\oplus}$	Point summation under operation $\oplus$
$T_{i,j}$	Trust value between node i and node j

Chapter 1

Introduction

In the past decade, great advances in the wireless network infrastructures, and a rapid growth of wireless and mobile applications technologies, have emerged. Mobile devices are becoming smaller, cheaper, more convenient, and more powerful in their applications. Independent mobile devices may need to be rapidly deployed, especially under the next generation of wireless networks that are developed for environmental or warfare scenarios. Examples of these scenarios include establishing robust, efficient, mobile communication for disaster relief efforts, emergency/rescue operations, sensors, and military networks. In these network scenarios, where no centralised authority available to manage the communication network, mobile nodes can communicate without relying on any fixed infrastructure in what is known as a Mobile Ad hoc Network (MANET).

Exchanging secret information and accessing sensitive services over unprotected wireless links require the deployment of security services in MANETs. While security protocols and standards (e.g. WEP[1] [10], WAP [11], etc) already exist for traditional wireless networks, such as the widely used IEEE 802.11 standard [10] for wireless local area networks (WLANs) and IEEE 802.16 [12] for broadband WLANs, security solutions for ad hoc wireless mobile networks such as MANETs are still inadequate due to their own specific characteristics (e.g., highly dynamic topology and lack of infrastructure). With many security problems still unsolved, this introduces new security challenges

[1] WEP is vulnerable to several types of cryptography attacks due to the misuse of the cryptographic primitives [9]

(e.g. secure routing, preventing of traffic analysis, resistance of captured devices, etc [13]) that require the design of specialised security solutions.

Based on the type of the existing trust relations between mobile nodes, there are two types of MANETs: *authority-based* MANET, and *self-organised* MANET.

In an *authority-based* MANET, there are a-priori trust relations existing between mobile nodes in the network initialisation phase supported by a central authority (CA) or a common organisation. On the other hand, in a *self-organised* MANET, mobile nodes are fully self-organised and do not rely on any CA to support the establishment of trust relations between mobile nodes in the network initialisation phase or in the network deployment phase. When the end users of mobile nodes do not necessary belong to the same organisation or share a single CA, a-priori trust relations between mobile nodes are hard to achieve. On the other hand, for some applications of MANETs, a-priori trust relations between mobile nodes are essential before network deployment due to the high security requirements of the transferred data between network entities.

As opposed to using static long-term keys, fresh cryptographic keys should be used to limit the amount of available ciphertexts in a crypto analysis as well as to reduce the damage of key compromise [14]. The unique features of MANETs envisioned new types of attacks; e.g., attacks on the multi-hop routing protocols. This could occur, by overloading some nodes with excessive traffic to consume their batteries which makes the network disconnected [13]. Hindering these types of attacks may also require the use of cryptographic keys to provide integrity, message authentication, and confidentiality. Many secure routing protocols require the use of shared keys between neighbours of the next hop or between source and destination nodes [15, 16, 17, 18].

1.1 Mobile Ad hoc Networks

In this section, we will discuss the characteristics of MANETs, MANET security, and finally we will present the open problems in MANET research.

1.1.1 Characteristics

MANETs have the following specific characteristics and design constraints [19] :

- **Autonomous terminal:**

  In a MANET, each mobile terminal is an autonomous node, which may function as both a host and a router. Thus, usually endpoints and switches are indistinguishable in a MANET.

- **Distributed operation:**

  Since there is no infrastructure for the central control of the network operations, all network services are distributed among the network nodes. The nodes involved in a MANET should collaborate together to implement functions such as security and routing.

- **Multi-hop routing:**

  Basic types of ad hoc routing algorithms can be single-hop and multi-hop, based on different link layer attributes and routing protocols. When delivering data packets from a source to its destination out of the direct wireless transmission range, the packets should be forwarded via one or more intermediate nodes.

- **Dynamic network topology:**

  Since the nodes are mobile, the network topology may change rapidly and unpredictably. The connectivity among nodes may also vary with time. A MANET should adapt to the traffic and propagation conditions as well as the mobility patterns of the mobile network nodes.

- **Fluctuating link capacity:**

  The nature of high bit-error rates of wireless connection might be more profound in a MANET. One end-to-end path can be shared by several sessions. Terminals may communicate over channels which are noisy, fading and are subject to interference. These usually have less bandwidth than a wired network.

- **Light-weight terminals:**

  In most cases, the MANET nodes are mobile devices with less CPU processing capability, smaller memory size, and lower power storage. Such devices need optimised algorithms and mechanisms that implement the computing and communicating functions using few resources.

1.1.2 MANETs Security

The security of most conventional networks relies on the existence of a specialised network administration that defines the security policy and provides the infrastructure for implementing it. There are two main approaches to provide a MANET with the required security: proactive and reactive [20].

In the proactive approach, various cryptographic techniques are used to thwart security threats. In the reactive approach, the security mechanism seeks to detect threats during the network operation and react accordingly. Each approach has its own features and can address different issues. For instance, most secure routing protocols use the proactive approach to secure the exchanged routing messages between mobile nodes, while the reactive approach is used to secure packets forwarding. In a multi-hop network, packets delivered between mobile nodes should be secured. Therefore, the exchanged routing control messages between the network nodes must follow the security specifications of the routing protocol. Accordingly, secure routing protocols is essential security mechanisms in securing MANETs.

In secure routing protocols (e.g. Ariadne [16], Secure Link State Routing (SLSR) [21], etc.), each mobile node proactively signs its routing control messages using the cryptographic authentication primitives such as digital signature [22] and message authentication codes (MACs) [23]. This way, network nodes can authenticate the exchanged packets and process packets received from legitimate nodes only. Most secure routing protocols assume the preexistence and presharing of symmetric or private/public key pairs and ignore the essential task of key management [2]. In order to prevent any kind of malicious attacks, messages must be integrity protected, which can be achieved by

applying message authentication codes (MACs) [23] and cryptographic hash functions [24]. In order to prevent eavesdropping, all secret messages must be encrypted [14]. Integrity protection, encryption as well as securing the ad hoc routing protocol require cryptographic keys. In fact, many security mechanisms that solve MANETs security issues depend on an efficient and secure key management protocol. Thus, key management is an open research area in MANETs security field [2, 25, 20, 26, 13].

1.1.3 Open Problems

Although a lot of research has been done in the area of MANETs, there have been numerous challenges and research problems still remain open for research. The major open problems are listed as:

- **Autonomous**:

  No centralised administration entity is available in MANETs to manage the operation of the different mobile nodes.

- **Dynamic topology**:

  Nodes are mobile and can be connected dynamically in an arbitrary manner. Links of the network vary over time and are based on the proximity of one node to another node.

- **Device discovery**:

  Identifying relevant nodes which have recently moved to the vicinity and informing neighbours about their existence needs dynamic updates to facilitate automatic optimal route selection.

- **Bandwidth optimisation**:

  Wireless links have to share the radio spectrum and operate within fixed bands.

- **Limited resources:**

  Mobile nodes are usually small, cheap, mobile devices with limited power and storage capacity.

- **Scalability**:

  Scalability can be broadly defined as whether the network is able to provide an acceptable level of service even in the presence of a large number of nodes. Increasing the network size usually increase the computation and communication overhead which needs more resources.

- **Limited physical security**:

  Mobility implies higher security risks such as peer-to- peer network architecture or a shared wireless medium accessible to both legitimate network users and malicious attackers. Eavesdropping, spoofing and denial-of-service attacks should be considered.

- **Lack of infrastructure and self operated**:

  The self healing feature demands a MANET should be able to reorganise itself for any node moving out of its range.

- **Poor Transmission Quality**:

  This is an inherent problem of wireless communication caused by several error sources that result in degradation of the received signal.

- **Network configuration**:

  The whole MANET infrastructure is dynamic and is the reason for dynamic connection and disconnection of the variable links.

- **Topology maintenance**:

  Updating information of dynamic links among nodes in MANETs is a major challenge.

1.2 Motivation

MANET is a temporary network formed by a collection of wireless mobile nodes without the aid of any dedicated centralised authority or predeployed infrastructure used to perform the basic network functions such as routing, i.e. in a MANET, mobile nodes act as both routers and terminals. Originally, MANETs were studied and explored for military and defence applications, such as for establishing instant communication infrastructures during rescue missions in war zones and disaster-affected areas [2, 27, 28], collecting data in hostile environments [29], and self-healing mine fields [25]. Initial investigations in the military sector have also led to suggestions for the deployment of MANETs for countless applications, such as law enforcement [2], virtual classrooms [2], connecting and reading out medical devices in hospitals [30], smart homes [30], wireless personal area networks (WPANs) [31], sharing resources [31], ubiquitous Internet access [32], instant networks for conferences and meetings [33], network games [32]. The security of most conventional networks relies on the existence of a specialised network administration that defines the security policy and provides the infrastructure for implementing it. The lack of any centralised network management, and the poor physical security make MANETs very vulnerable to infiltration, eavesdropping, interference, and so on. Therefore, efficient and robust key management services are central to provide MANETs with security services such as confidentiality, authentication, integrity and non-repudiation.

1.2.1 Authority-Based and Self-Organised Key Management

Due to the very limited resources of the mobile nodes in terms of memory and power, symmetric key cryptography is more suitable to address many of the security challenges in MANETs. The main challenge in using symmetric key based cryptosystems is to find an efficient way of distributing keys and keying materials to mobile nodes. However, the existing key distribution in symmetric schemes, e.g. key predistribution schemes (KPD) [34, 35, 36, 37, 38], poses a major problem because of the absence of an on-line key

distribution center (KDC) in MANETs. Hence, the existing symmetric key distribution schemes for traditional networks are not applicable to MANETs.

There are many applications in both the military and civil environments, where combinations of *authority-based* MANETs and *self-organised* MANETs are needed. In a military environment, according to the situation of the battle, there might be a need for fast deployment of new troops without any a-priori trust relations between the new and the old troops or without any possible access to the CA which can support trust establishment between them. Another example is the disaster rescue operation scenario in which rescue operations and data collection processes at the disaster site are performed. Members at the site are of two categories; specialised responders and non-specialised responders or volunteers. Specialised responders have a-priori trust relations between each other, while volunteers do not have such a-priori trust between each other or between specialised responders and volunteers. In such scenarios, a key management scheme that dynamically switches from a centralised scheme of trust distribution to a distributed scheme is needed.

1.2.2 Elliptic Curve Distributed Key Management

Distributed key generation (DKG) by distributing trust among a group of nodes provides a promising solution for the mentioned problems in Section 1.1.3. In DKG, a set of n servers jointly generate a pair of public and private key in a way that the public key is known to all nodes in the network while the private key is divided between the n servers via a threshold secret sharing scheme such as Shamir's (t, n) threshold cryptography [39]. A model for distributing trusted services (by using threshold cryptography) among a set of servers despite some servers being under control of an attacker has been proposed in [40]. A practical implementation of a DKG on a network of computers has been presented in [41]. A modified version of Pedersen's DKG protocol [42] has been proposed by Gennaro et al. in [43, 44].

Using elliptic curves for cryptographic protocols has been proposed independently by Miller in [45] and Koblitz in [46, 47, 48]. Cryptosystems based on elliptic curve discrete logarithm problem (ECDLP) can use smaller key size than that needed by discrete logarithm problem (DLP) or integer factorisation problem (IFP) based cryptosystems to provide the same level of secrecy [49, 50, 51]. Jian-wei et al. [52] propose an identity-based authentication mechanism based on elliptic curve cryptosystem for wireless sensor networks. This scheme suffers from the weaknesses of ID-based cryptography in which an on-line authority is needed to manage and monitor the identities of the network nodes. In [7], Liu et al. proposed a key management and authentication model for ad hoc network. This scheme is vulnerable to the man-in-the-middle attack. In [53], Wang et al. proposed a key management scheme based on the homomorphism of elliptic curve Paillier scheme. This scheme addresses only the new node joining process and suffers from the delayed authentication process. In [54], an elliptic curve version of Gennaro's DKG protocol [43] has been proposed by Tang et al. This scheme suffers from the heavy communication overhead and computation power required to perform the protocol successfully. The computation power and communication overhead are major concerns in the resource constrains environment such as smart cards and MANETs. Designing an elliptic curve DKG scheme with lower computation power and communication overhead in MANETs while maintaining the same security strength[2] is needed.

1.3 Contribution of the Thesis

The aim of this thesis is to develop key management schemes that provide redundancy and robustness for security association (SA) establishment between pairs of nodes in a MANET. We identify key management as a primary security goal for securing communications in MANETs. Our framework promotes inter-operability between existing architectures and protocols, and provides reasonable security and functionality. The main contributions of this thesis are as follows:

[2] security strength is the amount of work that is required to break a cryptographic algorithm or system [51]

- An authority-based key distribution scheme for MANETs is proposed. The proposed scheme exploits route request messages in distributing public key certificates and the route reply messages in distributing symmetric keys between neighbouring nodes. The proposed scheme breaks the routing-security interdependence cycle [55], by using the localised one-hop communication in distributing public key certificates and symmetric keys.

- A robust self-organised public key management for MANETs is proposed. The proposed scheme exploits the routing infrastructure to discover a certificate chain through a web of trust. A measure of the communication cost of the certificate chain discovery process is also proposed. The first advantage of the proposed scheme is that it has low communication cost. The second advantage is the negligible impact of the proposed scheme on the network performance.

- A robust and redundant key management scheme for MANETs is proposed. The proposed scheme is based on combining the authority-based key management approach and the web of trust approach in one scheme. The proposed scheme exploits the routing process in executing the public key authentication. The proposed scheme provides redundancy since it is operable with and without the existence of the central authority.

- A threshold key management scheme using elliptic curve dlog-based cryptosystem has been proposed. The proposed scheme does not require any prior communications between session members before forming the session. In the network deployment phase, nodes are free to select a group threshold k, which is a major advantage of our scheme that enables mobile nodes to adjust the level of secrecy they require. From the results, the proposed scheme has very low timings compared to the Gennaro's scheme [44], and timing does not vary significantly with changing the key size.

1.4 Outline of the Thesis

The thesis is organised in nine chapters as shown in Figure 1.1.

Chapter 2 provides background information and describes related research efforts in mobile ad hoc networks with particular emphasis on security in MANETs. It also focus on some fundamentals of cryptography, and routing protocols proposed in the literature. In Chapter 3, we present a study of the key management protocols in MANETs

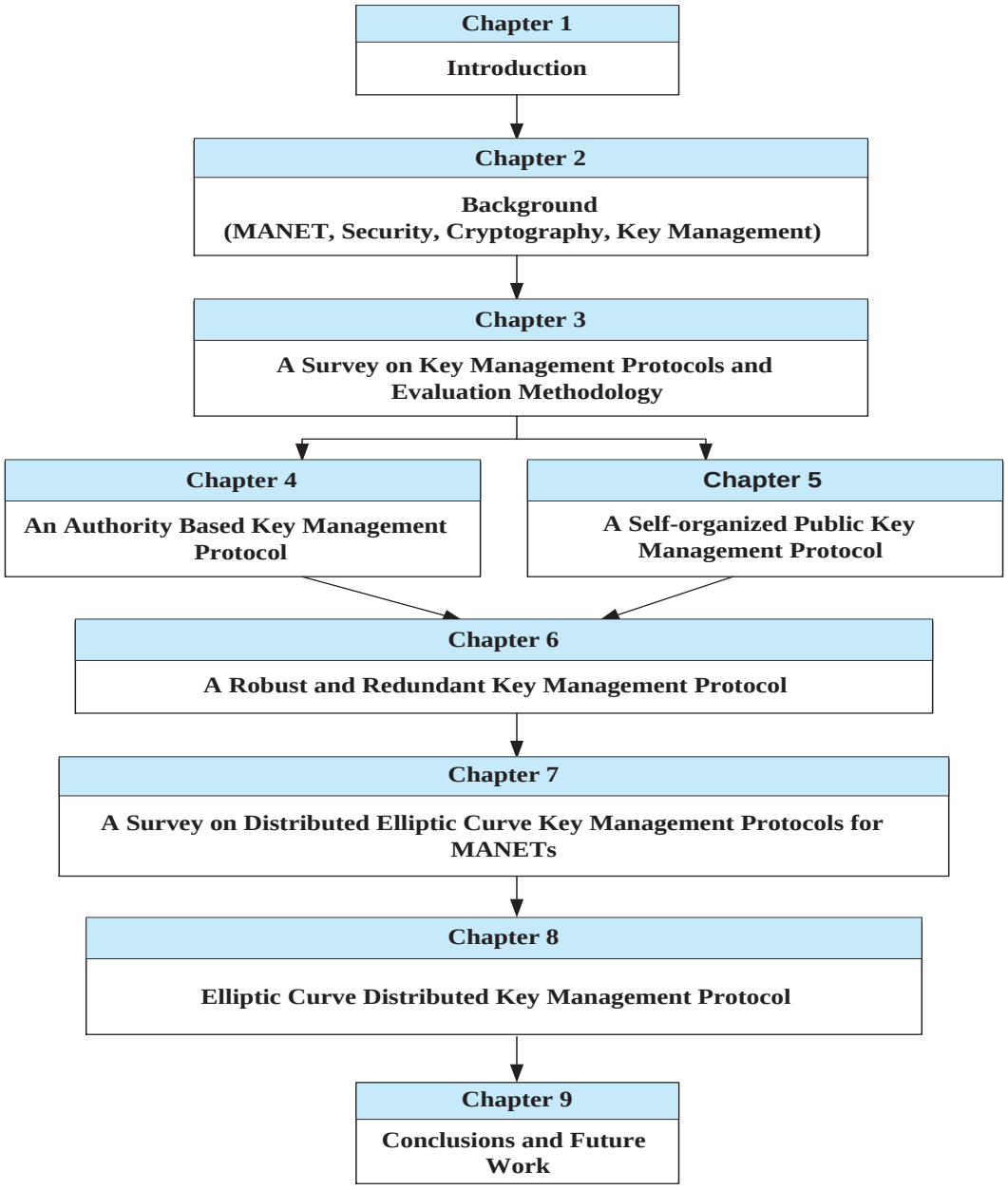

FIGURE 1.1: Thesis Outline

and the evaluation methodology and models adopted in this thesis. In Chapter 4, a key management scheme to distribute public key certificates and symmetric keys in an authority-based MANET is proposed. A robust self-organised public key management for MANETs, and the performance evaluation results using the network simulator (NS-2) [56], are presented in Chapter 5. Chapter 6 provides the scheme description and the performance evaluation results of a key management scheme that inherits the properties of both the authority-based key management scheme and the self-organised key management scheme. Chapter 7 presents a survey on existing elliptic curve threshold key management schemes and highlights key features of elliptic curve-based cryptosystems in MANETs. In Chapter 8, a threshold key management scheme using elliptic curve dlog-based cryptosystem is proposed. Finally, conclusions of the thesis and possible future work are presented in Chapter 9.

Chapter 2

Background

2.1 Introduction

In this chapter, we provide a brief review of background information and concepts about MANETs (Sections 2.2 and 2.3), routing protocols (Section 2.4), security primitives (Section 2.5), and key management (Section 2.6) for an easier understanding of the protocols presented in this thesis.

2.2 MANETs

MANETs are complex distributed systems comprised of wireless nodes that can dynamically self-organise into arbitrary and temporary network topologies. The diversity of MANET applications and research projects, as outlined in Section 1.1, use or emphasise different unique properties of MANETs. However, clear definitions of MANET properties and parameters, e.g. radio propagation models and existing technologies, are still missing. As this thesis is concerned with key management aspects in MANETs, we thus limit our discussions to unique MANET properties and parameters that make implementing key management protocols a challenging task.

2.2.1 Radio Propagation

Radio propagation in open areas free from obstacles is the simplest to treat, but, in real conditions, radio range of a mobile node is subject to many limitations. Most wireless systems are operated in areas with abundance of obstructions such as walls, buildings, and trees as shown in Figure 2.1. Radio propagation mechanism can be categorised into three main attributes [57]:

- **Reflection:**

  It occurs when radio wave propagating in one medium impinges upon another medium with different electromagnetic properties. Part of the radio wave energy may be absorbed or propagated through the reflecting medium, resulting in a reflected wave that is attenuated.

- **Diffraction:**

  It is a phenomenon by which propagating radio waves bend or deviate in the neighbourhood of obstacles. Diffraction results from the propagation of wavelets into a shadowy region caused by obstructions such as walls, buildings, mountains, and so on.

- **Scattering:**

  It occurs when a radio signal hits a rough surface or an object having a size much smaller than or on the order of the signal wavelength. This causes the signal energy to spread out in different directions.

2.2.2 Free Space Propagation Model

The free space propagation model attempts to describe radio wave propagation when the transmitter and receiver have a clear, unobstructed line-of-sight path between them [57]. This model predicts received power decays as a function of the transmitter-receiver separation distance raised to some power, i.e., the decay obeys a power law function. The

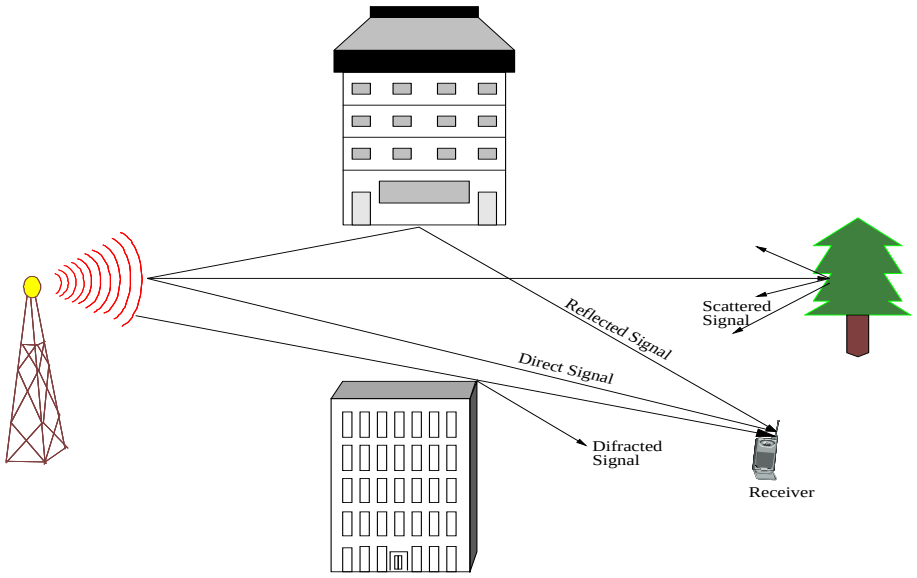

FIGURE 2.1: Reflection, diffraction and scattering of radio wave

free space power received $P_r(d)$ with respect to distance is then given by the following equation:

$$P_r(d) = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^2 L} \tag{2.1}$$

where P_t is the transmitted power, G_t is the transmitter antenna gain, G_r is the receiver antenna gain, λ is the wavelength, d is the distance between the transmitter and the receiver and L is a system loss factor not related to propagation$(L \geq 1.0)$. System losses are usually due to line attenuation, filter losses and antenna losses. Gain of antenna is related to its effective aperture A_e by:

$$G = \frac{4\pi A_e}{\lambda^2} \tag{2.2}$$

The effective aperture is then related to the physical size of the antenna and λ is related to the carrier frequency f by

$$\lambda = \frac{c}{f} \tag{2.3}$$

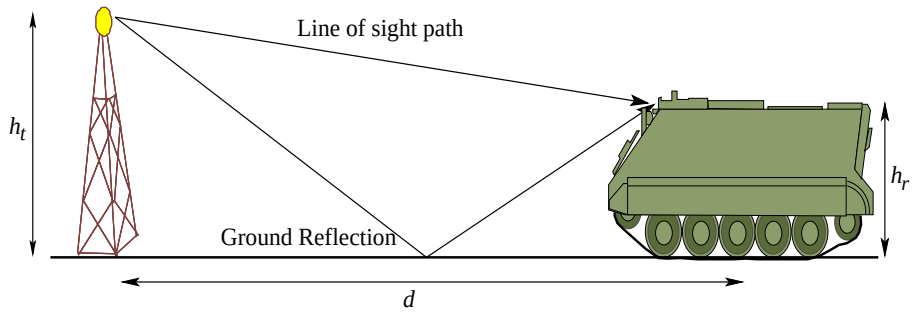

FIGURE 2.2: Two-ray radio propagation model

where c is the speed of light. Equation 2.1 shows that signal strength decays proportionally to the inverse of square of the distance between the transmitter and the receiver.

2.2.3 Two-ray Propagation Model

In a mobile radio channel, a single direct path between the transmitter and receiver is seldom the only physical means for propagation. Hence, the free space propagation model of equation 2.1 is inaccurate in most cases when used alone. The two-ray ground reflection model [58] is a useful propagation model that takes into account both the direct path and a ground reflected propagation path between the transmitter and the receiver. These aspects are depicted in Figure 2.2. The signal strength at the receiver for the two-ray ground model can be expressed as:

$$P_r(d,h_t,h_r) = \frac{P_t G_t G_r h_t^2 h_r^2}{d^4} \tag{2.4}$$

where h_t, h_r are elevations of the transmitter and the receiver, respectively. Thus, the received signal strength for large distances ($d \gg \sqrt{(h_t h_r)}$) decays with the fourth power of the distance d. This is a much faster rate of decay than with the free space propagation model. At large values of d, the received power and path loss become independent of frequency. The two-ray propagation model is the most commonly used propagation model in MANETs research community and will adopted for the network simulation model in this thesis.

2.2.4 MANETs Technologies

Typical wireless communication technologies used in MANETs are:

- **Wi-Fi (IEEE 802.11) [10]:**

  Wireless-Fidelity (Wi-Fi) is a wireless networking technology based on the IEEE802.11 specifications. It has been developed into the following variants:

  1. IEEE802.11a: It uses Orthogonal frequency-division multiplexing (OFDM) modulation technique and operates in the 5 GHz band.
  2. IEEE802.11b: It uses Direct-sequence spread spectrum (DSSS) modulation technique and operates in the 2.4 GHz band.
  3. IEEE802.11g: It uses OFDM/DSSS modulation techniques and operates in the 2.4 GHz band.
  4. IEEE802.11n: It uses OFDM modulation technique and operates in the 2.4 GHz band.
  5. IEEE802.11i: It defines security and authentication mechanisms at the MAC layer.
  6. IEEE802.11h: It is the same as IEEE802.11a with amendment introduced to the IEEE 802.11 standard for Spectrum and Transmit Power Management Extensions.

  All Wi-Fi technologies operate on the 2.4GHz band, except from IEEE802.11a which operates within the 5GHz band.

- **Bluetooth (IEEE 802.15.1) [31]:**

  Bluetooth is a short range wire replacement technology. When Bluetooth devices come within range of each other, they establish contact and form a temporary network called a Personal Area Network (PAN). A multi-hop ad hoc network formed by the interaction of Bluetooth devices is called a Scatternet. Bluetooth uses the Frequency-Hopping Spread Spectrum (FHSS) technique. Bluetooth operates within the 2.4GHz band.

TABLE 2.1: MANETs technologies comparison

Technology	Frequency Band	Data Rate	Comm. Range
Wi-Fi (IEEE 802.11)	2.4 GHz and 5 GHz	2-11 Mbps	~ 100 m
Bluetooth (IEEE 802.15.1)	2.4 GHz	1-3 Mbps	10-100 m
ZigBee (IEEE 802.15.4)	2.4 GHz, 915 MHz, 868 MHz	~ 0.25 Mbps	~ 10 m
WiMAX (IEEE 802.16)	2.5 GHz, 2.3 GHz, 3.65 GHz, and 5.8 GHz	0.5~2 Mbps	5~8 kms

- **ZigBee (IEEE 802.15.4) [59]:**

  ZigBee-enabled devices conform to the IEEE 802.15.4-2003 standard. This standard specifies its lower protocol layers, the physical layer (PHY), and the medium access control (MAC). It targets Low-Rate Wireless Personal Area Network (WPAN). Zigbee operates in the unlicensed 2.4 GHz, 915 MHz and 868 MHz ISM bands. It uses direct-sequence spread spectrum (DSSS) coding.

- **WiMAX (IEEE 802.16) [12]:**

  It stands for a Worldwide Interoperability for Microwave Access. IEEE 802.16 boasts data rates between 500 kbit/s and 2 Mbit/s at a distance of around 5-8 kms.

Other wireless technologies have been used for MANETs, e.g. HIPERLAN **[60]**, but the above represent the main technology options.

A comparison of different MANETs technologies presented in this subsection according to their frequency bands, communication ranges, and data rates are presented in Table 2.1.

Figure 2.3 shows the relationship between the Open System Interconnect (OSI) model and the IEEE 802 reference model. The OSI model presents a detailed and structured communications structure; the IEEE 802 model addresses the lower layer portion, specifically, the Data Link Layer and the Physical Layer.

2.3 MANETs Constraints

In this section, we focus on identifying constraints of MANETs that may affect the implementation of key management mechanisms. We classify constraints as either mobile node constraints or networking constraints.

2.3.1 Mobile Node Constrains

A typical mobile node can be a laptop, PDA, cellular phone or any other mobile wireless device. In MANETs, the network consists of a set of constrained devices with similar specifications without any available access to servers, routers, base stations, or any other powerful entities. In MANETs, most mobile nodes are inexpensive, lightweight, limited-capability, generic, and portable. According to Moore's law [61], the number of

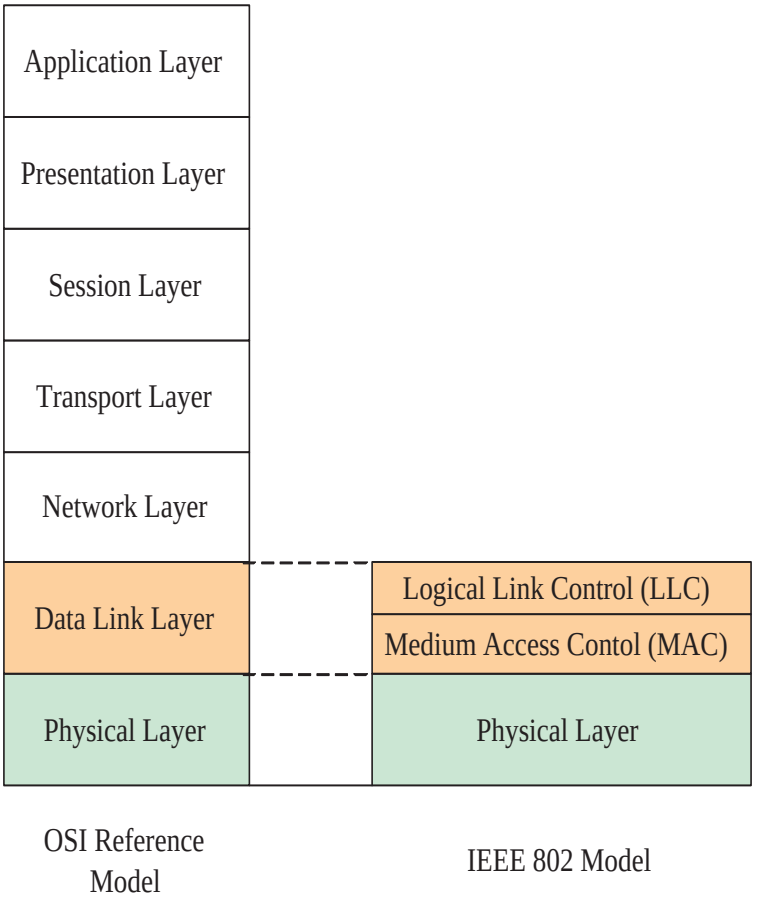

FIGURE 2.3: Relationship of IEEE 802 reference model to the OSI stack [1]

transistors that can be placed inexpensively on an integrated circuit has doubled approximately every two years resulting in faster and smaller CPUs. Furthermore, bandwidth constraints will be less stringent over time due to Gilder's law [62] that states that bandwidth grows at least three times faster than computing power. However, technological advances for extended battery life is comparably much slower. The capabilities and constraints of mobile node hardware will influence the type of security mechanisms that can be implemented on a mobile node. These features of MANET devices lead to several resource constraints, namely:

- **Low computing power:**

  Some computationally intensive operations may not be feasible on a MANET device due to the limited computing power. For instance, public key cryptographic algorithms such as RSA are computationally intensive, executing thousands or even millions of multiplication instructions to perform a single cryptographic operation.

- **Low battery power:**

  Power consumption is perhaps the greatest constraint to mobile node capabilities. We assume that once mobile nodes are deployed in a network, they cannot be recharged. Therefore, the battery charge taken with them to the field must be conserved to extend the life of the individual mobile node. When considering implementing a cryptographic function or protocol within a mobile node, the impact on the mobile node's available energy must be considered.

- **Small memory size:**

  The amount of program storage available for storing security functionality, such as security mechanism implementations, is unlikely to be a constraining factor on security design. Even the most sophisticated cryptographic algorithms can be represented in tens of kilobytes of memory, whereas the amount of program storage available in ROM, EPROM, or other nonvolatile memory is likely to be in the hundreds of kilobytes or megabytes. Likewise, the amount of working memory available for security functionality is unlikely to be a constraining factor.

Most symmetric encryption and hashing functions can be executed in less than one kilobyte of RAM. Even the more memory-consuming public key functions can be executed in just a few kilobytes of RAM.

- **Limited bandwidth:**

  Exchange of key management information varies widely depending on the key management algorithms, protocols, and the number of participating nodes. Key management algorithms based on symmetric or elliptic curve cryptography require the exchange of fewer bits than RSA, thus requiring less bandwidth. Reducing the number of keying relationships to only local neighbours reduces the amount of information exchanged, and thus the amount of required bandwidth.

- **Poor physical security:**

  Mobile nodes may be deployed in a hostile environment where they may be easily compromised or captured by an attacker. Hence, secure information such as cryptographic keys should be stored in an encrypted format in a tamper-resistant storage.

2.3.2 Network Constrains

MANETs have unique features not encountered in more typical wired networks. These features are:

- **Lack of infrastructure:**

  All network services are performed by mobile nodes in order to support a flexible and easily deployable network. The lack of infrastructure limits the amount and type of cryptographic material that should be necessary to deploy a secure MANET.

- **Ad Hoc Networking:**

  Mobile nodes are spontaneously formed for a specific purpose or to offer a certain service. The ad hoc nature limits ability to pre-configure mobile nodes for specific

purposes. Mobile nodes should be able to support various roles in the network to ensure the reliability of the network.

- **Dynamic Topology:**

  Mobile nodes may join or leave the network in an unpredictable manner which results in frequent routing changes. These routing changes can mean that the intermediate nodes processing data for an end-to-end session can change too. Thereby, many security services instead will be provided on a hop-by-hop basis. For instance, cryptographic key establishment will occur with local neighbours in the routing topology. If the routing changes, the set of local neighbours may change and thus cryptographic key establishment may need to occur again.

- **Wireless Network:**

  Wireless links have significantly lower capacity than the hardwired links. In addition, the realised throughput of wireless communications, because of the effects of multiple access, fading, noise, and interference conditions, is often much less than the hardwired communications. This constraint limits the amount and type of cryptographic material that can be transmitted between network nodes.

2.4 MANETs Routing Protocols

Routing is the process responsible for the computation of the routing information, (assigning unique addresses to communicating devices, provides valid source destination routes), and is defined by what is called a routing protocol. The routing protocol prescribes the exchange of information about the network topology with other nodes and the decision process about the routing information a node finally commits to. Note that a routing protocol, although formally located at the network layer, can make use of protocols at higher layers for the sake of communicating with neighbouring nodes about the network topology (for example, it can make interactions with the service discovery agents in the application layer). One can thus distinguish between routing packets, used by the routing processes, and data packets, which denote all other packets sent on the

network. A routing protocol is used in our network simulation model of the proposed key management schemes in order to measure their impact on the network performance.

2.4.1 Characteristics of Ad Hoc Routing Protocols

Despite the different classes in which the several routing protocols can be categorised, in the following we underline the common desirable characteristics [63]:

- **Multi-hop routing capability:**

  To extend the limited transmission range of wireless communications, the routing protocol must be able to exploit store and forward techniques.

- **Dynamic topology maintenance:**

  Since route discovery is an expensive service, the routing protocol should deal with topology changes without wasting the resources already spent for route discovery, namely the topology changes should have only local effects.

- **Loop avoidance:**

  Routing loops occur when the routing protocol selects as next hop, a node already in the built path and, since this implies a loop construct, an efficient avoidance mechanism should be implemented.

- **Minimal control overhead:**

  Control messaging consumes bandwidth, processing resources, and battery power on both the transmitter and receiver side, therefore the routing protocol should be designed to minimise the number of control packets needed to operate.

- **Minimal processing overhead:**

  By minimising the processing cycles, both the computational and power resources can be employed to accomplish the user tasks.

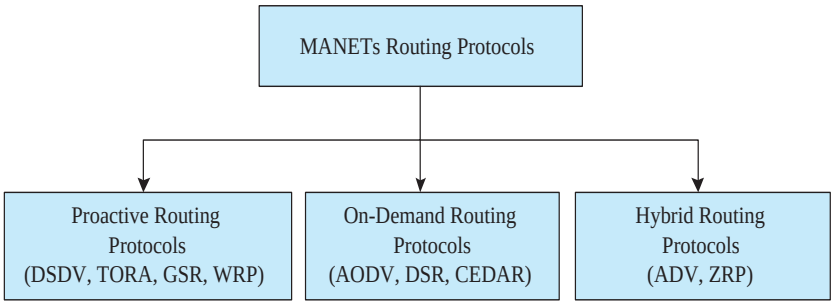

FIGURE 2.4: MANETs routing protocols

Routing protocols in MANETs can be classified into three fundamental approaches: proactive, on-demand (or reactive), or hybrid routing protocols [1]. This classification is depicted in Figure 2.4.

- **Proactive:**

  Each mobile node locally maintains a routing table for storing routing information about any node in the network. Nodes continuously exchange information about the network topology and update their routing tables, so that they are constantly aware of available routes regardless of communication requests. Examples of proactive routing protocols are: Destination Sequenced Distance Vector Routing (DSDV) [64], Temporally Ordered Routing Algorithm (TORA) [65], Global State Routing (GSR) [66] and Wireless Routing Protocol (WRP) [67].

- **On-demand (or Reactive):**

  To minimise the use of control messages, routes are only looked for and established when they are needed which means no unnecessary routing information is maintained. The routing process is divided into two phases: route discovery and route maintenance. The route discovery is initiated when a source node needs a route to a destination node. The route maintenance deletes failed routes and re-initiates route discovery. Examples of reactive routing protocols are: Ad Hoc On Demand Distance Vector (AODV) [68], Dynamic Source Routing (DSR) [69], and Core-Extraction Distributed Ad Hoc Routing (CEDAR) [70].

- **Hybrid:**

  A combination of the proactive and on-demand techniques is provided. Examples of hybrid routing protocols are: Adaptive Distance Vector (ADV) [71] and Zone Routing Protocol (ZRP) [72].

In general, proactive approaches are thought to have shorter latency, as routes are instantly available, whereas on-demand approaches will have lower routing control overheads and therefore achieve higher data throughput.

2.4.2 Ad Hoc On-Demand Distance Vector Routing (AODV)

In this subsection, an overview of the AODV routing protocol, one of the more prominent routing protocols for mobile ad hoc networks, is presented. AODV is being used as the routing protocol in our network simulation model.

The Ad hoc On-demand Distance Vector (AODV) protocol [68] is an extension of DSDV routing protocol in the direction of on-demand behaviour. It has also been standardised by the Internet Engineering Task Force (IETF) as RFC 3561 [73]. It consists of two processes; *route discovery* and *route maintenance*.

2.4.2.1 Route Discovery

When a source node needs to communicate with another node (the destination node), the source node initiates a route discovery process by broadcasting a route request (RREQ) message throughout the entire network via a simple flooding mechanism [73] as follows:

$$S \rightarrow Broadcast : \{RREQ, ID_S, SN_S, ID_D, SN_D, ID_{REQ}, hop_{count}\}$$

The RREQ packet contains the following main fields as shown in Figure 2.5: source IP address ID_S, source sequence number SN_S, destination IP address ID_D, destination sequence number SN_D (created by the destination to be included along with any route

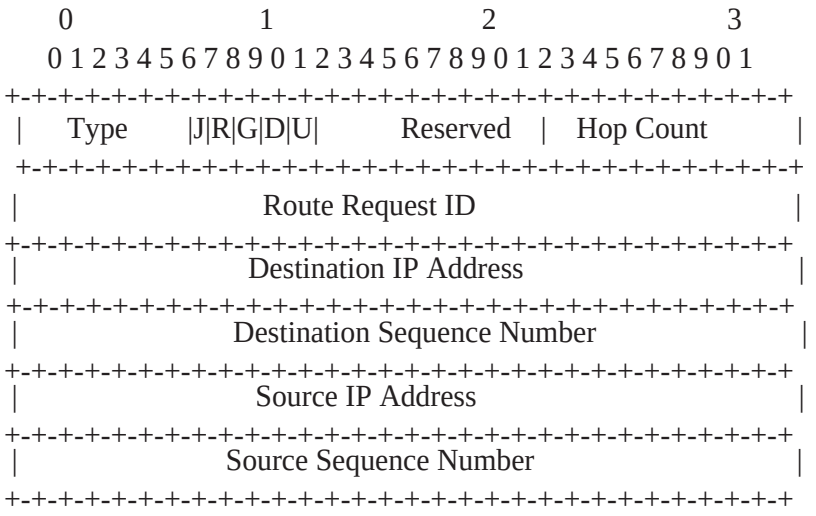

FIGURE 2.5: Route request packet format

information it sends to requesting nodes), route request identifier ID_{REQ} , and hop count. The destination sequence number is used by AODV to ensure that routes are loop-free and contain the most recent route information [64], and the hop count is the number of hops between the source node and the current node. The ID_{REQ} and SN_S are maintained locally by every node and incremented whenever a RREQ is sent in order to identify requests and determine the freshness of routes, respectively. Each node in the network has a routing table as shown in Table 2.2, where each entry has the following fields: destination node identifier, next hop node identifier on the route to this destination, sequence number, and the validity time of this entry.

Reverse Route Setup: When a *RREQ* packet is received by an intermediate node, it checks if it has seen the pair (ID_S, ID_{REQ}) already. According to the check result, it will do one of the following actions: if the intermediate node has seen the pair already, the *RREQ* packet will be dropped in order to prevent processing of duplicates *RREQ*. If the *RREQ* packet is a fresh one, the intermediate node forwards the *RREQ* packet. When an intermediate node forwards an *RREQ* packet, it creates a reverse route back to the source node by inserting the next hop information in its routing table. Once the *RREQ* packet reaches the destination or an intermediate node with a valid route, the destination or intermediate node responds by unicasting a route reply (*RREP*) packet to the source node using the reverse route. The format of the *RREP* packet is shown in

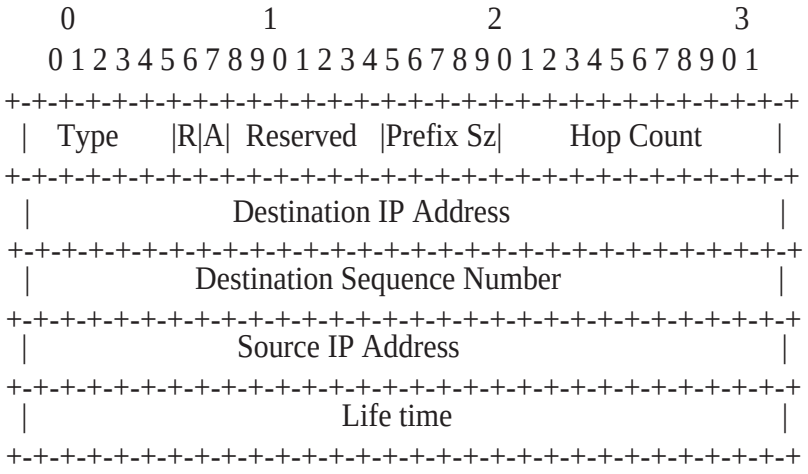

FIGURE 2.6: Route reply packet format

Figure 2.6. The validity of a route at the intermediate nodes is determined by comparing the destination sequence number stored in the intermediate node's routing table with the destination sequence number SN_D exists in the *RREQ* packet.

Forward Route Setup: Each node that forwards the *RREP* packet back to the source node creates a forward route to the destination node by inserting the next hop information in its routing table. Only information about the next hop nodes to the source and destination is required by intermediate nodes along the path from source to destination, while information about other nodes forming the path is not required.

2.4.2.2 Route Maintenance

While the discovered route is being used, each intermediate node along the active route maintains a fresh list of its one-hop neighbours by periodically exchanging *"hello"* control packets with its direct neighbours. If the route becomes inactive, i.e. no data is sent over it, a timer is activated, after the expiration of which the route is considered broken and expires. When a node becomes aware of a link breakage for an active route, a Route Error (*RERR*) packet is generated at the point of breakage. This is then disseminated to the appropriate nodes participating in the route's formation and those nodes actively using the route. The nodes affected by the broken route mark it for

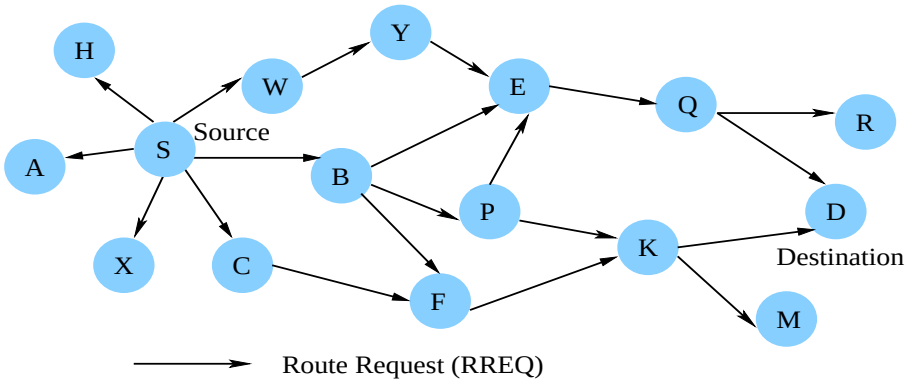

FIGURE 2.7: Route request in AODV

expiration since it is no longer useful. When the source node receives the *RERR* packet, it initiates a new route discovery process.

In the following, a further explanation of the route request, route reply, link failure, and the usage of the routing table is given taking into consideration the example in Figures 2.7, 2.8, 2.9, and in Table 2.2 respectively.

- In Figure 2.7, the source node S initiates a route discovery process by broadcasting an *RREQ* message to be flooded in the network searching for the destination node D. In this example, the source sequence number $SN_S = 3$, the destination sequence number $SN_D = 2$, the route request ID $ID_{REQ} = 1$, and the hop count is 0.

  $S \rightarrow Broadcast : \{RREQ, S, 3, D, 2, 1, 0\}$

- When nodes A, H, X, C, W, B receive the route request, they check if they have already seen the source node identifier ID_S, and the route request identifier ID_{REQ}. If the *RREQ* is not a duplicate one according to the previous check, they search in their routing table for a valid route to the destination node D. If the search results yields that they do not have a valid route to the destination node, they increment the hop count by one and forward the *RREQ*. This process continue till the *RREQ* reach the destination node D.

- Assume that nodes C, and W have routes to the destination node D through routes $C \rightarrow F \rightarrow K \rightarrow D$ and $W \rightarrow Y \rightarrow E \rightarrow Q \rightarrow D$ respectively and the destination

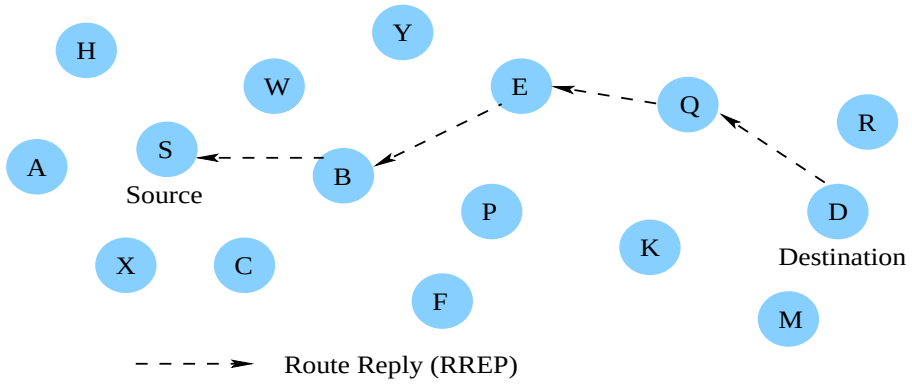

FIGURE 2.8: Route reply in AODV

sequence number SN_D at the intermediate node C is 4 and is 1 at node W, then only node C is allowed to unicast an *RREP* along the route to the source node S. This is because S has an older route to node D compared to the route available to node S, while node C has a more fresh route to the destination node D compared to the route available to node S.

- If eventually the *RREQ* packet reaches the destination node D through the route $B \rightarrow E \rightarrow Q \rightarrow D$, then the destination node D unicasts an *RREP* along the reverse route to S as shown in Figure 2.8. In this case the source node may receive multiple *RREP* packets. The source node S chooses the route which has the minimum number of hops to send data through to the destination node D. All the intermediate nodes receiving an *RREP* update their routes with the latest destination sequence number. They also update the routing information if it leads to the shortest route between the source and the destination nodes.

  Figure 2.9 shows the maintenance process due to a link failure. When a link failure is detected by absence of *"hello"* messages, the source and destination nodes are informed. For instance, when the link between nodes E and Q is broken, both nodes send *RERR* packets to notify the source node S, the intermediate nodes along the path from the source to the destination, and the destination node D about the link failure. These nodes delete the corresponding entries from their routing tables while the source node S reinitiates the route discovery process with a new

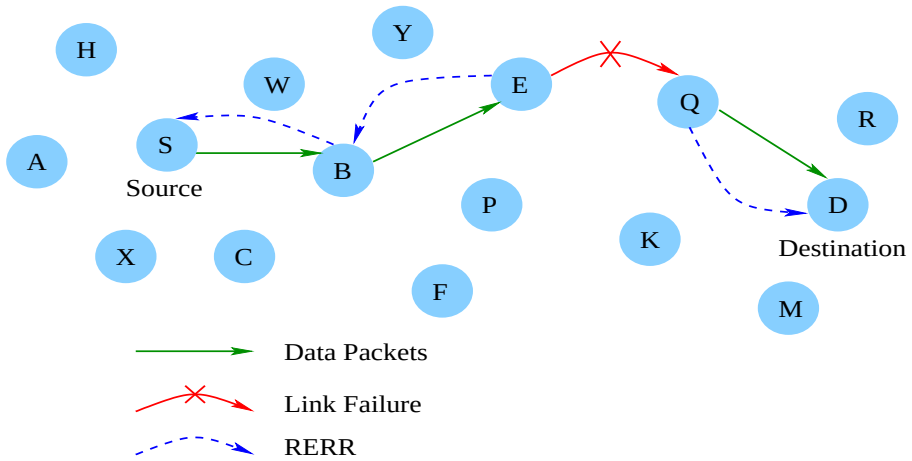

FIGURE 2.9: Link failure in AODV

TABLE 2.2: Routing table in AODV

Index	Next Hop	Destination	Sequence Number	Validity Time
S	B	D	SN_S	T_S
B	E	D	SN_B	T_B
E	Q	D	SN_E	T_E
Q	D	D	SN_Q	T_Q

RREQ packet containing a new ID_{REQ} and the latest destination sequence number SN_D.

2.5 Cryptographic Background

For a general introduction to symmetric and public key cryptography, a brief review of some security concepts that are used in this thesis is presented in this section.

2.5.1 What is a Secure MANET?

It is necessary to define what is meant by being secure before one can determine whether something is secure or not. In MANETs security, it is essential to require confidentiality, integrity, availability, authentication and non-repudiation. These attributes are defined as follows [74, 75]:

- **Confidentiality:**

  Confidentiality is to ensure that secret information is never revealed to unauthorised entities. In an ad hoc network, confidentiality is very important to protect the transmission of secret information over the wireless links which are easily susceptible to eavesdropping.

- **Integrity:**

  Integrity is to ensure that data will not be corrupted or altered in an unauthorised manner during transmission.

- **Availability:**

  Availability is to ensure that network services are available to all entities when required.

- **Authentication:**

  Authentication is the ability to verify the identity of an entity.

- **Non-repudiation:**

  Non-repudiation means that an entity cannot falsely deny that certain actions were performed, e.g. the transmission of data or signing a message.

It is not always possible, or necessary, to require that all five attributes be fulfilled completely, but they should all be taken into consideration when designing a system that involves security [76, 2].

2.5.2 Definitions

2.5.2.1 Security Threats

The types of threats on the security of a communication network are as follows [74]:

- **Interruption:**

  An asset of the system is destroyed or becomes unavailable or unusable. This is an attack on availability.

- **Interception:**

  An unauthorised party gains access to an asset. This is an attack on confidentiality.

- **Modification:**

  An unauthorised party not only gains access to but tampers with an asset. This is an attack on integrity.

- **Fabrication:**

  An unauthorised party inserts counterfeit objects into the system. This is an attack on authenticity.

2.5.2.2 General Types of Attacks

There are many types of attacks on a cipher. They are designed to recover the plaintext and break the key [74].

1. **Ciphertext only:**

   The cryptanalyst has only the ciphertext.

2. **Known-plaintext:**

   The cryptanalyst knows both the ciphertext and its plaintext.

3. **Chosen-plaintext:**

   The cryptanalyst selects the plaintext used to create the ciphertext.

4. **Adaptive-chosen-plaintext:**

   The plaintext is selected based on previous plaintext-ciphertext pairs.

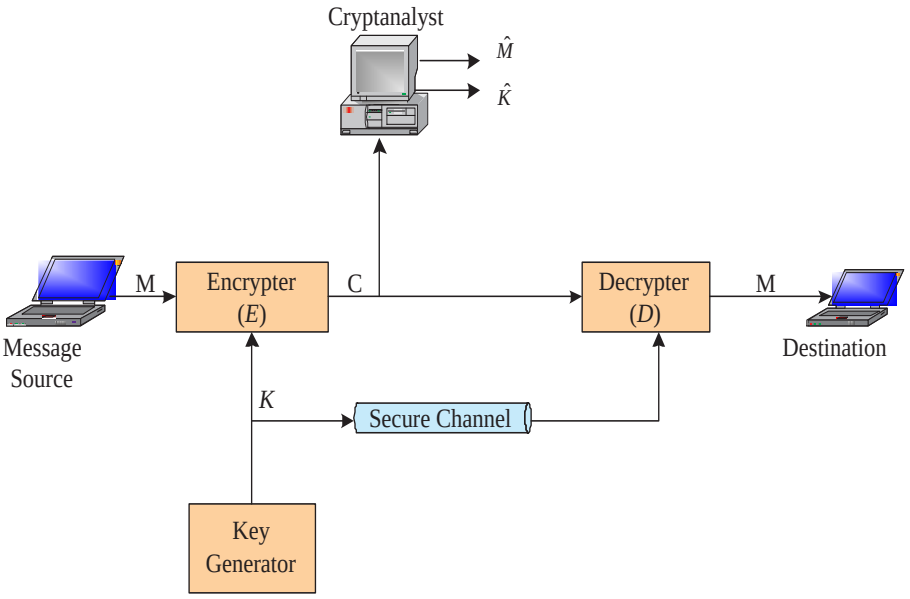

FIGURE 2.10: Symmetric key cryptosystem

5. **Chosen-ciphertext:**

   The cryptanalyst selects the ciphertext to decode and can access the corresponding plaintext.

6. **Man in the Middle Attack:**

   In this attack, the attacker intercepts messages independently from two players in a public key exchange and then relays messages between them, substituting his own public key for the requested one, so that the two original players still appear to be communicating with each other.

2.5.2.3 Symmetric Key Cryptography

Encryption is a major part of a security package. Secure communications play an increasing role in many fields of common life. The basic idea of encryption is to modify the message in such a way that its content can be reconstructed only by a legal recipient. The message might be represented by a sequence of symbols from a finite alphabet. The encryption process, as shown in Figure 2.10, consists of an encryption algorithm E and a key K. The key is a value independent of the plaintext M that controls the algorithm.

The encryption algorithm E will produce a different output depending on the specific key being used at the time.

$$C = E_K(M) \tag{2.5}$$

Changing the key changes the output of the encryption algorithm. Once the ciphertext C is produced, it is transmitted. Upon reception, the ciphertext C can be transformed back to the original plaintext M by using a decryption algorithm D and the same key that was used for encryption as follows:

$$M = D_K(C) \tag{2.6}$$

In practice, C can be understood as an integer.

2.5.2.4 Public (Asymmetric) Key Cryptography

In public key cryptography, such as Rivest, Shamir, Adleman (RSA) algorithm [77], each person gets a pair of keys, called the public key PK and the private key SK. Each person's public key is published while the private key is kept secret. In a real life scenario, Alice usually generates keys PK and SK. She then publishes PK as her public key and keeps SK as her private key. With the public key PK, everyone is able to encrypt a message and send it to Alice. The only key that will result in correct decryption is Alice's private key SK, and she is therefore the only one able to decrypt the messages encrypted with her public key. If Alice wants to send a message to Bob, she will need Bob's public key. If she encrypts with her own private key SK, everyone will be able to decrypt the message, since her public key PK is commonly known. Her private key can, however, be used for digital signatures, because she is the only one who could have encrypted the message. All communications involve only the public key, and no private key is even transmitted. Taking into consideration the example in Figure 2.11, the public key cryptosystem can be summarised as follows:

- Key Generation: The receiver Bob creates his private and public key pair, which we denote by SK_{Bob} and PK_{Bob} respectively.

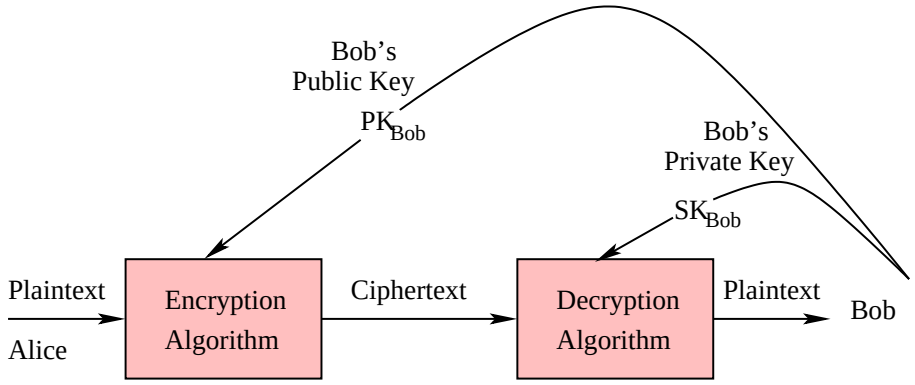

FIGURE 2.11: Public key cryptosystem

- Encryption: Using Bob's public key PK_{Bob}, the sender Alice encrypts her message M, which we call a "plaintext", and obtains a ciphertext C.
- Decryption: Upon receiving the ciphertext C from Alice, Bob decrypts it using his private key SK_{Bob} to recover the plaintext M.

2.5.2.5 Long-term and Short-term Evidence of Trust (Credentials)

Long-term evidences of trust contain authentic information to identify an entity and/or key material and are used over a longer period of time. On the other hand, short-term evidences of trust, may contain the same information but are changed frequently and used for a shorter period of time. Long-term evidences of trust may be used in authentication protocols to prove an entity's identity or in key management protocols to derive session keys. Long-term evidences of trust can be for example, a driver license, passport, employment identity card, date of birth, documentation indicating credit card activity or certified private and public key pairs, whereas short-term evidences of trust are typically session keys or ephemeral private and public key pairs generated for one session. Short-term evidences of trust are typically used to limit the damage of key compromise and reduce the amount of available ciphertext in a cryptanalysis.

2.5.2.6 Hash Functions

Hash functions play an important role in cryptographic applications such as digital signatures, message authentication codes (MACs), and other forms of authentication. The purpose of a hash function is to create a short digest of an arbitrary message by mapping an arbitrarily long message to a message of fixed length. The digest must not have any similarities with the original message, and have a fixed output length (typically 128 to 256 bits). In hash functions, it is easy to compute the output value from the input value, but very difficult "to go back", i.e., compute an input value from the corresponding output value. Because of this feature, hash functions are usually called (cryptographic) one-way functions. In general, a hash function H should have these properties [75, 24]:

- **Preimage resistant:**

  It should be very computationally intensive to find the input from a given output, i.e. given an output y, it should be computationally infeasible to find x such that $H(x) = y$.

- **Weak collision resistance**:

  When given an input x, it should be computationally infeasible to find any second input $x^{'}$ such that $H(x) = H(x^{'})$.

- **Strong collision resistance**:

  It should be computationally infeasible to find any two distinct inputs x and $x^{'}$ such that $H(x) = H(x^{'})$.

Collisions will always exist, because hash functions produce an output with fixed length from an arbitrary input. The number of possible inputs are therefore infinite, whereas the number of possible outputs are finite.

2.5.2.7 Message Authentication Code (MAC)

MACs are used to provide message integrity and authentication. MAC functions take two inputs: an arbitrarily sized message and also a secret key and produce a fixed size

output. A MAC can be constructed from hash functions as keyed hash functions which is referred to as HMAC [23]. MACs are used such that to compute a MAC value for a message and then send the message and its MAC value together to the recipient. MAC functions should have the following security properties [13]:

- **Key non-recovery:**

  It is infeasible to produce the same output without knowledge of the key. This property ensures efficient authentication of the messages while also guarantees data integrity.

- **Computation resistance**:

  Given one or more pairs of messages and the corresponding MAC values, it is computationally infeasible to compute a MAC value for any new input.

2.5.2.8 Digital Signatures

A digital signature is similar to the MAC value: it is attached to a message to ensure its integrity, authenticity and non-repudiation of the message origin. When a message is digitally signed, the signer's private key is used in an asymmetric encryption algorithm, typically RSA. By encrypting with the private key, everyone is able to decrypt, so there is no confidentiality. However, since the private key is only known by the key owner (signer), only he/she could have signed the message. This provides authentication similar to a written signature, which can only be produced by its owner. A digital signature scheme consists of a key generation algorithm, a signing algorithm, and a verification algorithm. The key generation algorithm generates the public/private key pair of the signer. The signing algorithm takes the message to be signed and the private key of the signer, and it produces the digital signature. The verification algorithm takes the digital signature, the public key of the signer, and the message and its output is *accept* if the signature has been generated on the message with the private key corresponding to the published public key; otherwise it outputs *reject*.

2.5.2.9 Identity-Based Cryptography

In 1984, Shamir proposed a concept of identity-based cryptography [78]. In this new paradigm of cryptography, users' identities information such as email or IP addresses or more precisely arbitrary strings instead of digital certificates can be used as public keys for encryption or signature verification. In identity-based cryptography (IBC), the sender Alice can use the receiver's identifier information which is represented by any string, such as email or IP address to encrypt a message. The receiver Bob, having obtained a private key associated with his identifier information from the trusted third party (TTP), can decrypt the ciphertext. Note that all users are able to derive the public keys of any other user in the network from publicly known information without the need to exchange any data. This unique feature of IBC schemes is based on the assumption that the identities of all players in the network are publicly-known to all other players in the network. Due to the predetermination of public keys in IBC schemes, the private keys of users need to be generated and distributed by a TTP. Otherwise, users would derive their private keys from their public keys which would enable all users to compute the private keys of any other user in the network [78]. For that reason, IBC schemes require a TTP that serves as a key generation center (KGC) to generate and distribute private keys.

2.6 Key Management

Cryptographic protocols use keys to authenticate entities and grant access to guarded information to those who exhibit their knowledge of the keys [14]. Therefore, it is necessary that keys be securely generated and distributed to appropriate entities. Secret keys are shared between communicating entities. A secret key can be generated by one player and distributed to another player, either through direct physical contact or a secure channel. In public key cryptography, a public key is made public, while the corresponding private key is kept secret. The existence of secure communication channels is crucial in the mobile ad hoc environment on the account of the use of wireless links

and other characteristics of such networks. In order to make such secure communication possible, it is necessary to provide mobile nodes with the required keying material. This is known as key management process [75]. Public key cryptography is often used to distribute secret keys. In this section, a brief introduction to terms and definitions used in key management is presented.

2.6.1 Public Key Certificate

A public key certificate certifies the binding between a public key and an entity. Certificates are signed bindings by a trusted party whose public key is known to every entity. Public key certificates can be generated and distributed through a central server or a network of nodes that provides such services, or a combination of the two. For example, the certificate contains the node's public key, identity/network address, sequence number, trust value, certificate generation and validity dates.

2.6.2 Trusted Third Party (TTP)

A trusted third party (TTP) is an entity trusted by all users of the network. There are three types of TTPs (as shown in Figure 2.12) which can be categorised depending on the nature of their involvement as follows [75]:

- **In-line TTP**:

  It is an active participant in the communication path between the two users. The in-line TTP is involved in the transmission of data in addition to providing the users with the required keying material.

- **On-line TTP**:

  It is an active participant but only in deciding the keying material without any involvement in the transmission of data between the network users.

- **Off-line TTP**:

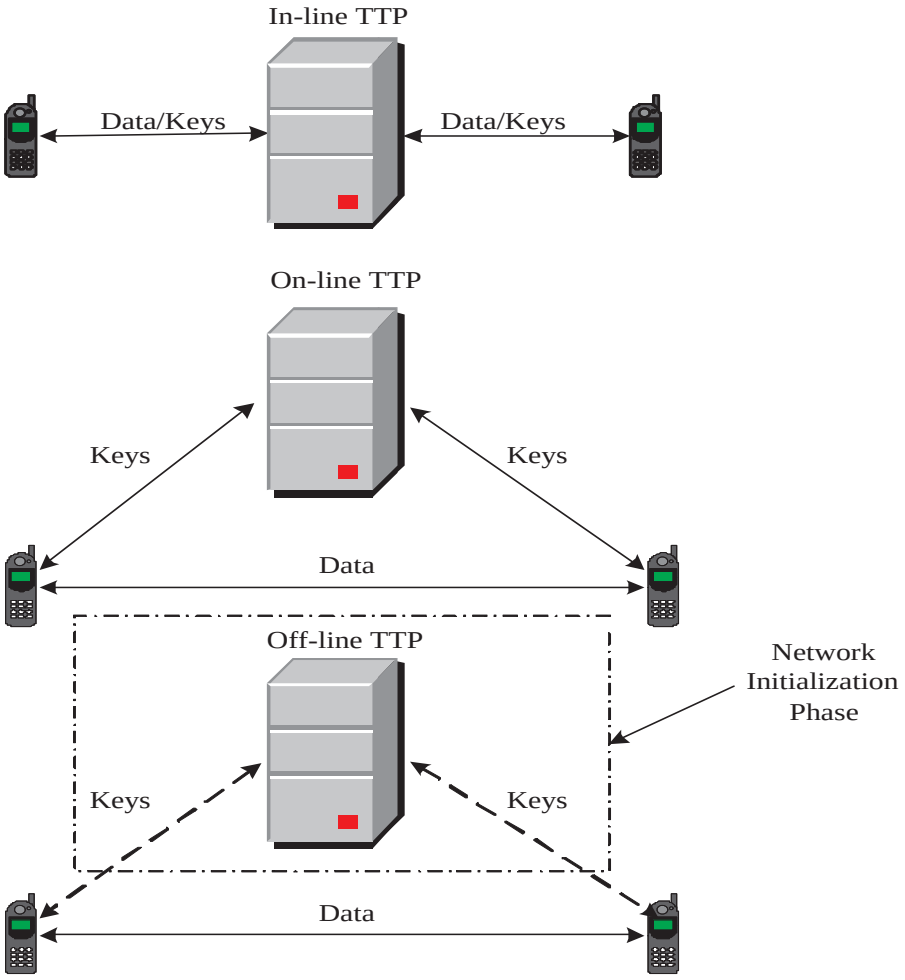

FIGURE 2.12: Classification of trusted third parties (TTPs)

It distributes the keying material to the network users in the network initialisation phase before network deployment. Once the network has been deployed, the off-line TTP is no longer available in the network.

Example of TTPs include the key distribution center (KDC) which is more like an on-line TTP, and the central authority (CA) which is more like an off-line TTP.

2.6.3 Web-of-Trust

In 1991, Phil Zimmerman implemented the first version of Pretty Good Privacy (PGP) [79]. A web of trust is a concept used in PGP, and other compatible systems to establish the authenticity of the binding between a public key and a user. In PGP, when a user

believes that another user is trustworthy according to its trust evidence, they can sign and exchange certificates. A certificate may therefore contain more than one signature. Furthermore, a list of trusted certificates is kept. When a signature of a certificate needs to be verified, the certificate is either included in the mail or retrieved from a public certificate database. To verify the integrity of the signature, the other signatures on the certificate are checked. If any of the signatures matches a certificate in the list of trusted certificates, the new certificate is trusted and added to the list. If not, one of the certificate databases is contacted to see if it can find a trusted path from the received certificate to the users certificate. A trusted path is a path of certificates that have signed each other. The path only needs to be unidirectional, i.e. it must be possible to establish a path from the certificate to the receiver, but not necessarily the reverse way be valid. Each user can specify whether he fully or partially trusts a certificate and how many partially trusted paths he needs to a certificate before it can be trusted. The web of trust scheme is flexible and leaves trust decisions under the control of individual users.

2.6.4 Threshold Key Management

In threshold cryptography (TC) [39], a group of n parties share the ability to perform a cryptographic operation by distributing the trust placed on a single user among a group of n users. In addition, there is a threshold t associated with the threshold cryptosystem such that any t out of the n users can execute the cryptographic operation. Such schemes are referred to as (t,n) TC schemes. In this case, less than t users will not be able to perform the cryptographic operation successfully. In a threshold cryptosystem, if less than t users are compromised, the security of the whole system will not be breached. In threshold key management as shown in Figure 2.13, a set of n servers jointly generate a pair of public and private keys in a way that the public key is known to all nodes in the network while the private key is divided between the n servers via a threshold secret sharing scheme such as Shamir's (t,n) threshold cryptography [39]. Later, in order for a new node to join the network, at least t nodes (among n nodes) need to cooperate and sign a certificate for the new node.

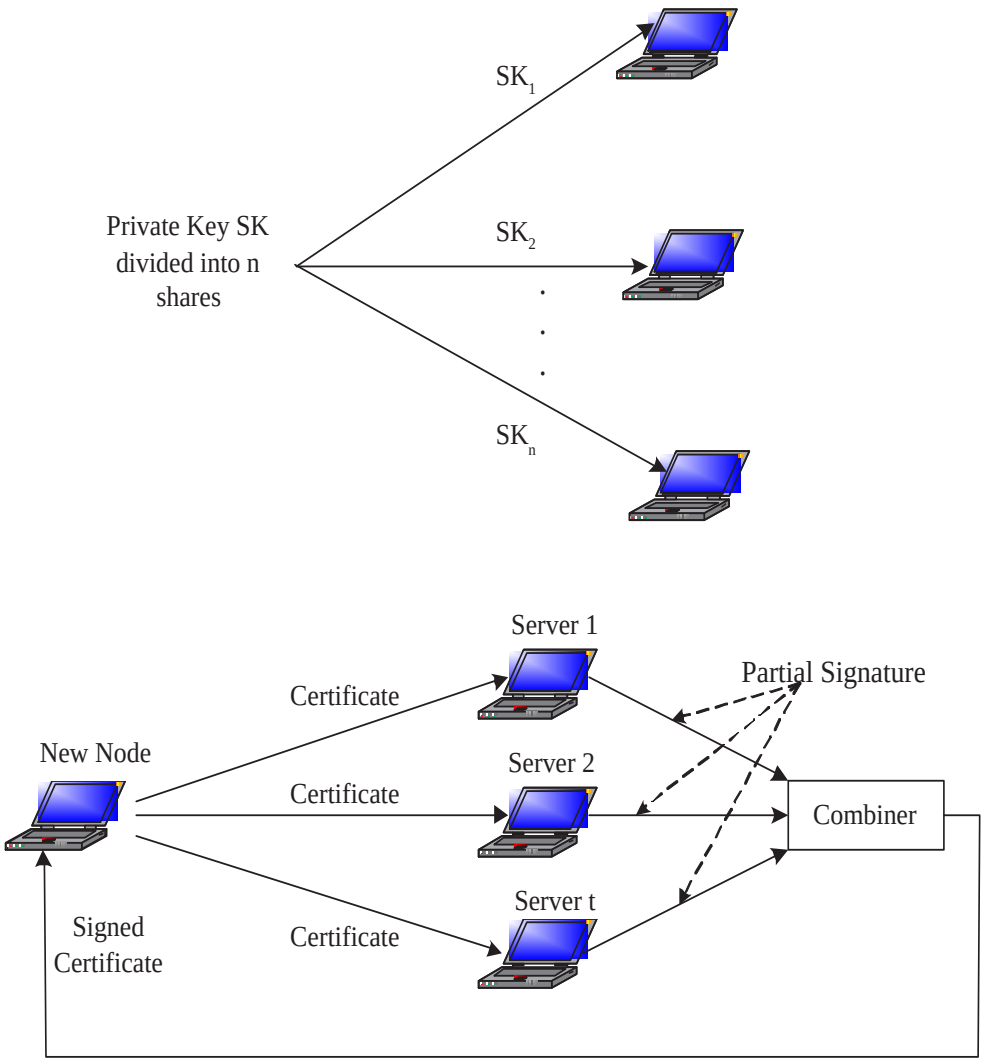

FIGURE 2.13: Threshold key management

2.6.5 Key Revocation

Key revocation is a mechanism to revoke expired or compromised keys and broadcast this information to all network entities. Revocation information is typically provided in form of lists, e.g. a blacklist containing all revoked certificates. Typically these lists are generated by a CA and then either pushed to all nodes from a central point, or pulled from a central point to all nodes. A widely used revocation algorithm for certificates of the public key infrastructure (PKI) standard X.509 makes use of what is called certificate revocation lists (CRLs) [80], which are lists generated by a CA contain the expired and compromised certificates and stored in publicly accessible repositories. Network nodes can then access the repositories to get the latest CRL. Many solutions

were proposed to reduce the size of the CRLs that can grow very large over time. In some proposals, only updated information is provided in the next published CRL, in so called delta CRLs. Another proposal is the On-line Certificate Status Protocols (OCSP) [81], in which network entities request the status of particular certificates and a CA returns the signed status of the requested certificates.

2.7 Summary

In this chapter, the basic concepts and definitions of ad hoc networks were discussed. Radio propagation models which are being used in MANETs have been presented while giving more details about the two-ray propagation model which will be used as the propagation model in our network simulations. Different MANETs technologies have been presented. Mobile nodes constrains and network constrains that affect the design and implementation of security protocols in MANETs have been discussed. The common characteristics and classifications of MANETs routing protocols were provided. The AODV routing protocol [68], one of the more prominent routing protocols for mobile ad hoc networks, was presented as it is being used as the routing protocol in our network simulation model. A cryptographic background and commonly used concepts in MANETs security were provided. Finally, a brief introduction of definitions and concepts being used in MANETs key management protocols were presented. In the next chapter, we will discuss the related published work in key management protocols for MANETs. In addition, the evaluation methodology used in this thesis will be presented.

Chapter 3

A Survey of Key Management Protocols and Evaluation Methodology

3.1 Introduction

The lack of any centralised network management or central authority makes a MANET very vulnerable to infiltration, eavesdropping, interference, and so on. Security in a MANET is an essential component to supply the network with the basic functions such as secure routing and safe packet forwarding.

Key management is a fundamental part of any secure communication. Secure network communications normally involve a key distribution procedure between communication parties, in which the key may be transmitted through insecure channels. For this reason, a framework of trust relationships needs to be built for authentication of key ownership in the key distribution procedure.

In a MANET, key management schemes can be classified into two types. The first type is based on a centralised or distributed Trusted Third Party (TTP). The TTP is responsible for issuing, revoking, renewing, and providing keying material to nodes participating in the network [2, 82, 4] where the key management process is performed using threshold cryptography [39].

The second type of key management is the self-organised key management schemes [83, 84]. Self-organised schemes allow nodes to generate their own keying material, and issue public key certificates to other nodes in the network based on their knowledge. Certificates are stored and distributed by the nodes. Each node maintains a local certificate repository that contains a limited number of certificates selected by the node according to an appropriate algorithm. Public key authentication is performed via chains of certificates.

In this thesis, we propose a number of key management schemes that can work efficiently with and without the existence of a TTP (i.e., authority-based scheme in Chapter 4, self-organised scheme in Chapter 5, and a combination of both types in Chapter 6).

The first part of this chapter explains the key management requirements in MANETs (Section 3.2). It also discusses related published work in key management protocols for MANETs and analyses their suitability for network-layer security (Section 3.3). The second part of the chapter then presents the evaluation methodology adopted in this thesis and is organised as follows. Section 3.4 highlights common evaluation techniques of systems performance. Section 3.5 presents an overview of NS-2 packet level simulation used throughout this work. Section 3.6 illustrates simulation model of AODV and its implementation in NS-2. Section 3.7 discusses methods used during the analysis of the generated results. Section 3.8 demonstrates validation of the routing protocol simulation models. Finally, Section 3.9 concludes the chapter.

3.2 Key Management Requirements in MANETs

Implementing a secure key management protocol in MANETs should meet several requirements in order to be robust and efficient. However, it is not always possible to fulfil all the requirements. The key management protocols employed in MANETs need to consider and incorporate some important security features such as authenticity, confidentiality, integrity, scalability, availability, and flexibility [85, 86, 87, 88]. The reasons are explained as follows:

- **Authenticity:**

  It is important that a mobile node should be able to verify the authenticity of other nodes involved in a shared communication. In other words, the receiver node should recognise the identity of the sender node. Key Authentication is a property whereby a communication entity is assured that only certain authenticated entity may gain access to the keying material.

- **Confidentiality:**

  The key management protocol needs to protect the secrecy of the keying material. An adversary may try to attack the network by capturing the cryptographic keys to reveal the secret data. In this case, the key management protocol has to guarantee that unauthorised parties will not gain any information about the keying material that have been used in securing communications between the network entities.

- **Integrity:**

  Integrity means that data will not be altered in an unauthorised manner during transmission. In a key management protocol, integrity means that only nodes in the network should have access to the cryptographic keys. It also means that only an assigned authority (distributed or centralised) has the privilege to change the keys. Such arrangements would effectively prevent an adversary from obtaining knowledge about the used keys and prevent keys updates.

- **Scalability:**

  Key management operations should be completed in a limited time regardless the variations in number of nodes in the network and node densities. Key management protocols should provide high security features for small and large networks. The communication overhead of the key management process should be kept as low as possible.

- **Availability:**

  In key management, availability ensures that keying material is provided to mobile nodes in the network where and when needed. This prevents degradation of key management services.

- **Flexibility:**

  Flexibility means that the key management protocol should be able to operate in different environments and support dynamic deployment of mobile nodes, i.e., a key management protocol should be useful in multiple applications and allow for adding and removing nodes at any time.

3.3 Related Works

An extensive work on key management schemes have been proposed for MANETs [32][55][4][83] [35][34][89][90][91][92][5][7][93][94][95]. Some of these schemes are based on a centralised or distributed TTP, some are self-organised schemes, and one scheme can be considered as a composite scheme which combines aspects of both. A classification of the state of the art in key management schemes for MANETs is shown in Figure 3.1. This section presents an overview of the most related key management schemes to our work.

3.3.1 Schemes Based on a Centralised or Distributed TTP

This section surveys key management schemes based on a TTP, either centralised or distributed. Schemes based on a centralised or distributed TTP can be further classified into public key schemes, symmetric key schemes, and threshold cryptography schemes. Examples are as follows:

- Partially Distributed CA Scheme [2]
- SKiMPy: A Simple Key Management Protocol for MANETs [3]
- AKM: Autonomous Key Management [96]

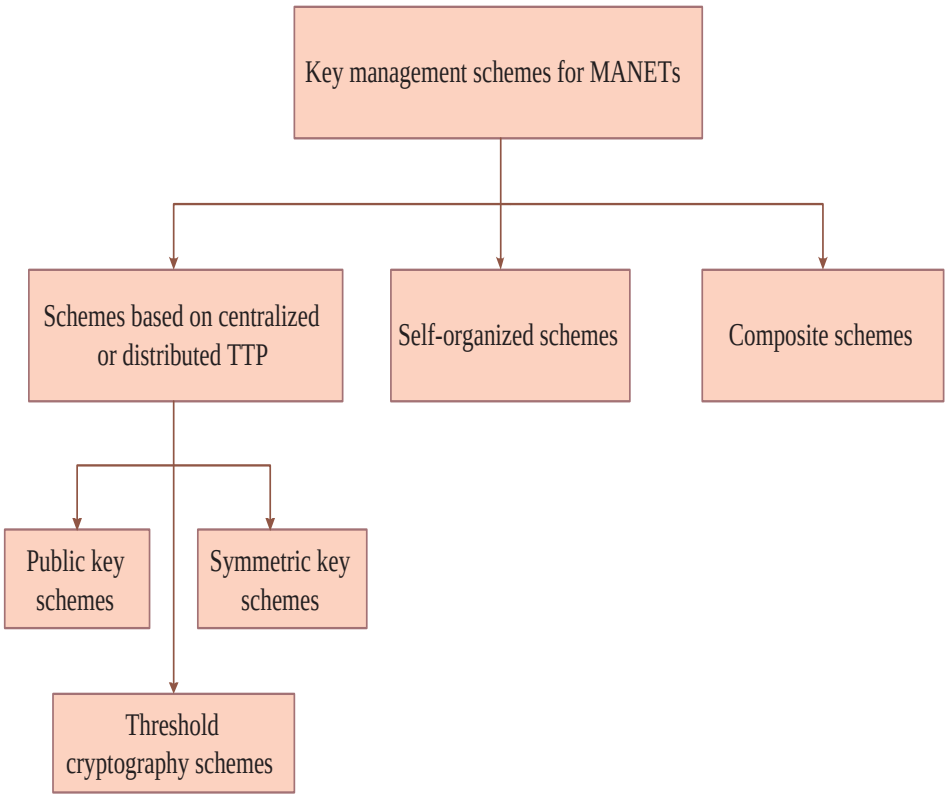

FIGURE 3.1: Classification of key management schemes for MANETs

3.3.1.1 Public Key Schemes

Many key management schemes for MANETs can be classified as public key schemes such as Partially Distributed CA Scheme [2], MOCA: Mobile Certificate Authority [97, 98], Secure and Efficient Key Management (SEKM) [99], and Key Distribution in Mobile Ad Hoc Networks Based on Message Relaying [94]). We chose the Partially Distributed CA Scheme [2] to be analysed as an example of public key schemes.

Partially Distributed CA Scheme [2]: In this scheme, the CA private key k is distributed over a set of n server nodes through a (t,n) secret sharing scheme [39] (where t is the threshold number of nodes required to cooperatively recover the CA private key k). The CA private key is shared among n nodes in such a way that at least t out of n server nodes must cooperate in order to reveal the CA private key k. The distributed certificate authority (DCA), as shown in Figure 3.2, consists of n server nodes which, as a whole, have a public/private key pair K/k. When required, the DCA signs a certificate

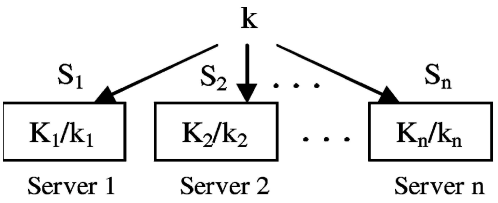

FIGURE 3.2: Key management service K/k configuration [2]

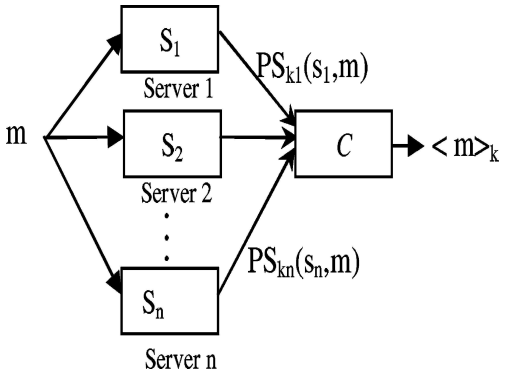

FIGURE 3.3: Threshold signature K/k generation [2]

m in such a way that each server generates a partial signature of the certificate m using its private key share k_i in a threshold signature scheme [100]. Each server then sends its partial signature to the combiner C as shown in Figure 3.3.

A server acting as combiner collects at least t partial signatures to produce a valid signed certificate. In order to prevent mobile adversaries (adversaries that temporarily compromise a server and then attack the next) and adapt to network changes, a proactive key share refreshing [101] has to be implemented. In the proactive key share refreshing, key shares are periodically renewed (without changing the secret key). Thereby, an adversary must compromise more than the threshold t servers within the time interval between key updates in order to attack the system. At least t nodes must cooperate to revoke a certificate. Expired certificates can be renewed by sending a message to any t server nodes.

The off-line CA makes the scheme unsuitable for self-organised MANETs. The fact that all certificates must be known a-priori by the DCA in order to get access to the security services makes the scheme non-scalable.

3.3.1.2 Symmetric Key Schemes

A number of key management schemes for MANETs can be categorised as symmetric key schemes such as SKiMPy [3], and Key infection: Smart Trust for Smart Dust [102]). In the following, a detailed discussion on the SKiMPy scheme [3] is given:

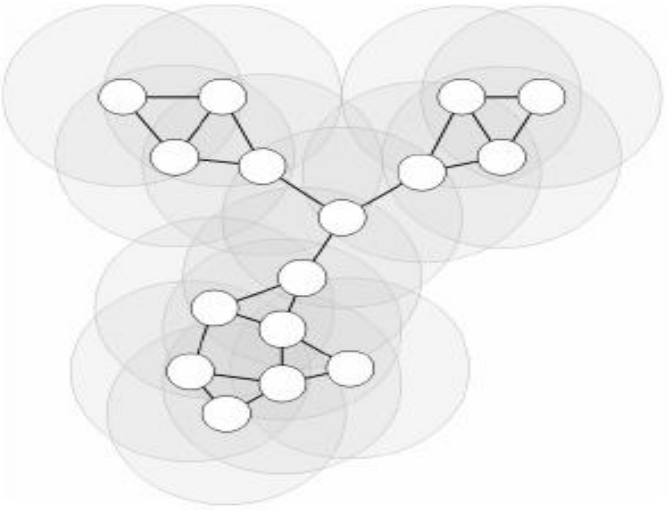

FIGURE 3.4: SkimPy: 3 cells, connected by one node in the middle [3]

A Simple Key Management Protocol for MANETs (SKiMPy) [3]: SKiMPy is a symmetric key management protocol designed for MANETs in emergency and rescue operations. The scheme allows mobile devices carried by the rescue responders to establish a symmetric shared key. The established shared key is used primarily to generate message signatures. In the network initialisation phase, all network nodes generate a random symmetric key and advertise it within their one-hop neighbours through "*hello*" messages. The "*better*" key (i.e., the one with the lowest ID number, freshest timestamp, or a similar parameter), is chosen as the local group key. This "better" key is transferred to the nodes with "worse" keys through a secure channel established with the aid of predistributed certificates. The procedure is repeated until the "better" key has been shared with all nodes in the network as shown in Figure 3.4.

SKiMPy is claimed to be bandwidth efficient in the sense that nodes agree on the *"better*" key locally within their on-hop neighbours. Hence, it does not need for an already running routing protocol. The disadvantage of SKiMPy is as the network is initialised, the symmetric group key is also established. Once the symmetric key has been received, there is no efficient way to expel the node from further participation. The group key (or a key derived from it) now serves as proof of trustworthiness. Thus, SKiMPy adds complexity, but does not increase the security accordingly. This scheme suffers also from the absence of on-line revocation before the network has been initialised.

3.3.1.3 Threshold Cryptography schemes

The Robust and Ubiquitous Security Support scheme [92], and the Autonomous Key Management scheme (AKM) [96] are two examples on the threshold cryptography key management schemes for MANETs. The Autonomous Key Management scheme (AKM) [96] was selected to highlight the main feature of threshold cryptography key management schemes as follows:

Autonomous Key Management scheme (AKM) [96]: In this scheme, Bo et al. [96] provide a fully distributed threshold *CA*. The hierarchical structure of this scheme is a logical tree, in which all the leaf nodes represent real wireless devices, while all the branch nodes only exist logically. Each node receives a share of the *CA* private key. As the number of nodes increases, a hierarchy of key shares is introduced. New nodes then receive a share of the *CA* private key. The root *CA* private/public key pair is bootstrapped by a group of neighbour nodes through distributed verifiable secret sharing [103]. Each of the n neighbours chooses a secret value S_i, and distributes secret shares of this to the other neighbours using a (t,n) secret sharing scheme. The corresponding public key of mobile node i equals to g^{S_i}. The sum of the individual secret values $S = (S_1 + S_2 + S_3 + \ldots + S_n)$ represents the *CA* private key. The corresponding *CA* public key equals:

$$g^S = \prod_{i=1}^{n} g^{S_i} \; mod \; p \tag{3.1}$$

Where p is a prime. The probability of a compromise increases with more nodes holding a share of the *CA* private key. Therefore, when the number of share holders reaches a certain level, nodes split into smaller regional groups and set up a new regional key. The scheme requires mobile nodes to collaborate on changing regions and key hierarchy. Byzantine or faulty nodes may delay such operations. In scenarios like emergency and rescue operations, where *CA* services for issuing initial certificates and revocations are needed, AKM with several regions represents a waste of bandwidth.

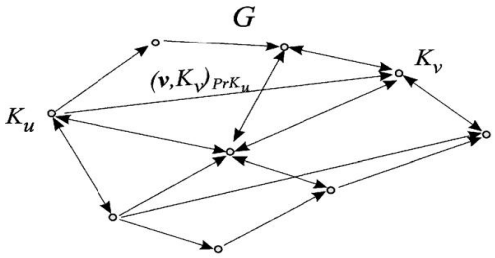

FIGURE 3.5: Step 1: Issuing of public key certificates [4]

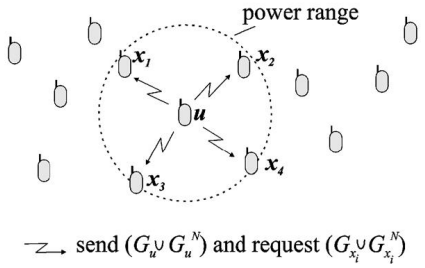

FIGURE 3.6: Step 2: Certificate exchange [4]

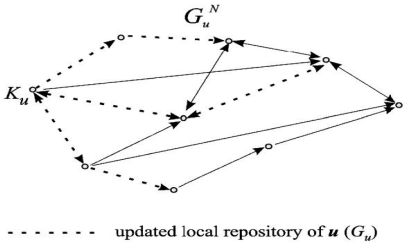

FIGURE 3.7: Step 3: Node u constructs its updated repository from G_u^N [4]

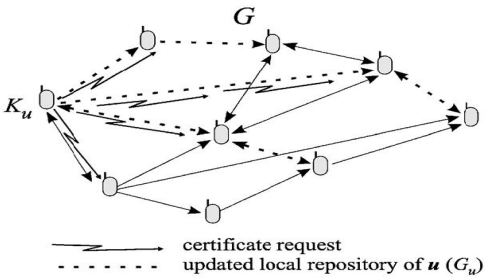

FIGURE 3.8: Step 4: Node u constructs its updated repository by communicating with other nodes [4]

3.3.2 Self-Organised Schemes

A large number of key management self-organised schemes have been proposed due to their suitability for the self-organised characteristic of MANETs. Examples of these schemes are: The Self-Organised Public-Key Management scheme [4], and The Highly Reliable Trust Establishment scheme [104]. In this section, we survey the Self-Organised Public-Key Management scheme [4].

Self-Organised Public Key Management scheme [4]:

In [4], Capkun et al. propose a fully self-organising public key management scheme based on the PGP web of trust [79]. In this scheme, the CA functionality is completely distributed. Each node generates its own private/public key pair and issue certificates to their trusted neighbouring nodes. Each mobile node has a local certificate repository

to store certificates it issues or certificates issued to it from other nodes. This scheme assumes that trust is transitive (i.e., if A trusts B, and B trusts C, then A should also trust C). The nodes merge their certificate repositories, and try to find a verifiable chain of certificates. A summary of the scheme is shown in Figures 3.5, 3.6, 3.7, and 3.8.

An algorithm which is called the Maximum Degree algorithm is suggested to construct a certificate graph with high connectivity even if the sizes of the users' certificate repositories are small due to the small world phenomenon (the hypothesis that everyone in the world can be reached through a short chain of social acquaintances) [105, 106]. Certificates are revoked through revocation messages from their issuer, or implicitly revoked at expiry time. Renewals require contact with the issuer. Certificates are also exchanged periodically between neighbours. Evaluation of expiration times and periodical exchanges requires some sort of synchronisation between the nodes. It is not clear how this synchronisation should be established. The scheme is non scalable and bandwidth consuming because of the periodic certificates exchanges and contact with issuers to update certificates. A compromised node only discloses the keys held by this node. However, a compromised node could be used to issue faked certificates to allow other illegitimate nodes to gain access to the network.

3.3.3 A Composite Key Management Scheme

In this section we present an overview of the composite key management scheme introduced by Yi and Kravets in [5]. In this scheme, certificates, as proposed in [4], are stored and distributed by nodes in a self-organised nature. The authors showed how a DCA can be used in parallel with certificate chaining to eliminate some of the weaknesses of the self-organised approach as shown in the example in Figure 3.9. In Figure 3.9, nodes with incoming dashed edges, such as K_3 and K_5, are certified by *CA* trusted nodes while nodes with incoming solid edges, such as K_1 and K_2, are certified by the *CA* itself. For example, the dashed edge from K_2 to K_3 represents K_3's certificate signed by K_2 with confidence value 0.5. This trust relation appears on the edge in Figure 3.9 as $Name = "ID_3"/confidence = 0.5$. Mobile nodes that have been trusted by the CA are

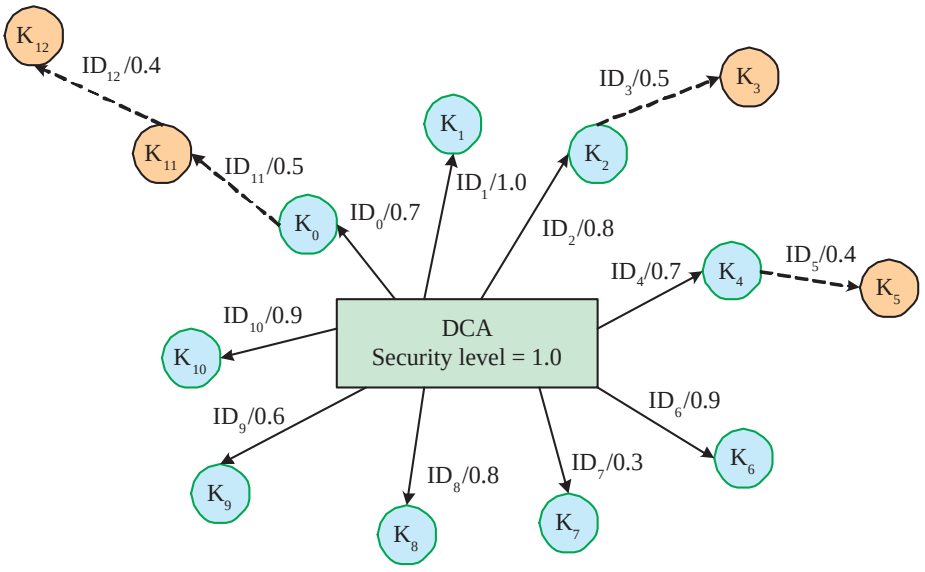

FIGURE 3.9: Example showing DCA composed with one-hop certificate chaining [5]

allowed to issue certificates to others. Mobile nodes requesting a certification service should first try the mobile certificate authority (MOCA) CAs. If this fails, they should search for neighbours that have CA certificates. This scheme increases availability of the key management service since nodes can use either service to obtain keying material. Each certificate in this scheme includes a confidence value. This confidence value reflects the level of confidence the certificate issuer has in the binding between node identity and its public key. This scheme is not suitable for fully self-organised MANETs because issuing certificates in this scheme is limited to nodes that have a *CA* certificate.

3.4 Performance Evaluation Methodology

Performance is often a key issue in the design, development and configuration of systems. It is not always enough to know that systems work properly; they must also work effectively and fairly. Performance analysis studies are conducted to evaluate existing or designed systems, to compare alternative configurations or to find an optimal configuration of a system. The primary step in performance evaluation is to select the right

evaluation technique. In general, there are three techniques used when investigating the protocols for MANETs [8]: measurements, analytical modelling and simulation.

- **Measurements:**

  Measurements involve physically monitoring the actual system. However, such approach incurs high cost or disruptive changes to the system. For example, in this thesis we are measuring the network performance of the key management scheme for a 50 mobile nodes network moving over a 1000×1000 metres flat space, we can deploy devices such as notebooks to measure the protocol performance, but the cost in equipment and manpower is exorbitant. Also, there will be difficulty associated when measuring performance of mobile scenarios in the said field dimensions.

- **Analytical Modelling:**

  This approach has, essentially, two components; first come up with a way to describe a system mathematically with the help of applied mathematical tools such as queueing and probability theories, and then second to apply numerical methods to gain insight from the developed mathematical model. When the system is quite simple and relatively small, analytical modelling would be preferable (over simulation). Here the model needs to be mathematically tractable. The numerical solutions to this model in effect require lightweight computational efforts. If properly employed, analytical modelling can be highly cost-effective and therefore can potentially provide an abstract view of the components interacting with one another in the system. However, if many simplifying assumptions on the system are made during the modelling process, analytical models may not give an accurate representation of the real system.

- **Simulation:**

  Simulation uses a computer to simulate real networks in terms of structures and operations. Compared to analytical modelling, simulation usually requires lower degrees of abstraction in the model (i.e., fewer simplifying assumptions) since

more details of the specifications of the system can be included in the simulation model. When the system is rather large and complex, a simple mathematical formulation may not be feasible. In this case, the simulation approach is usually preferred to the analytical approach. Further, even if a system is available for measurement, a simulation model may be preferred over measurements because it allows the alternatives to be compared under a wider range of workloads and environments. However, a disadvantage of simulation is that direct causal relationship may be more difficult to identify when compared to analytical modelling.

Consequently, a simulation model is used throughout this work, as the network structure being represented is complex.

3.4.1 Types of Simulations

Among the variety of simulation methods described in the literature [8, 107, 108], two important methods are Trace-Driven Simulation and Discrete-Event Simulation.

3.4.1.1 Trace-Driven Simulation

A trace is a time-ordered record of events on a real system. A simulation using a trace as its input is a trace-driven simulation. Trace-driven simulations are common in computer systems since they generally have built-in tracing programs which monitor the activities of the system and the sequences of processes pertinent to the planned simulation. The major advantage of trace-driven simulation is that it yields better accuracy and more flexibility. Another advantage of such approach is that it is relatively easy to validate the model. The main problem with trace-driven approach is that the scope of application is very small. This approach is really only applicable to the performance modelling of computer systems and even then they can only be used when the aim is to make moderate changes to a currently running system, which is not the case for the work carried out in this thesis.

3.4.1.2 Discrete Event Simulation

In discrete event simulation, the process of a system is represented as a chronological sequence of events. Each event occurs at an instant in time and marks a change of state in the system. A discrete event simulation model has an internal source of random numbers. The random numbers derive the components of the simulation model; they are used to determine the occurrence time of between system events, branching probabilities and so on. For example, random number generators are used to generate network topologies, traffic profiles and mobility patterns. The essential feature is that the model is self-contained; it requires no external inputs to operate. All discrete-event simulations have a common structure. The most basic components in a discrete-event simulation are [8, 108, 109, 110]: simulation clock, events, event handler, event list and time-advancing routine. One of the difficulties associated with discrete event simulation derive from its inherent stochastic nature. For example, in a discrete event simulation model inter-arrival times are normally generated from independent, identically distributed, random variables. However, if significant interactive effects between the underlying processes are present, then detailed statistical analysis must be performed in order to discover their characteristics [107]. This information must then be used to appropriately adjust the random variables. Also, verification and validation of discrete event simulation models can be very difficult. Issues such as selecting appropriate levels of detail, determining the modelling assumptions and fixing model parameters will all complicate the task [111, 112].

Despite the problems of performing an accurate simulation, the discrete event simulation method of system analysis is very attractive for the following reasons [107]:

- **Cost-effectiveness:**

  The ability to perform for relatively low cost (compared to other methods).

- **Accessibility:**

  The ability to be performed by non-mathematicians.

- **Power:**

  The ability to apply the difficult problems.

- **Scope:**

  The ability to apply to diverse problems.

- **Simplicity:**

  The ability to understand and evaluate.

The benefits of discrete event simulation outweigh its limitations and, for complex systems, provide the only feasible means for evaluating protocol performance and operation and hence is considered for carrying out this work. As a notational convenience from now on the terms 'model' and 'simulation' should be understood to refer to discrete event model and to discrete event simulation respectively.

3.4.2 Simulation Approach

Researchers generally use simulation tools to analyse network performance prior to physical implementation or to compare multiple alternatives over a range of system parameters. In recent years, several discrete-event network simulation tools have been suggested for performance analysis in MANETs [113]. There are several well-known tools that can be used for the simulation of MANETs such as Network Simulator (NS-2) [56], GloMoSim [114], QualNet [115], OPNET [116] and OMNeT++ [117]. Full comparisons between these tools are presented in [118, 119, 120]. This thesis adopts the NS-2 as a network simulation platform for the following reasons:

- It is widely utilised in the MANETs domain.
- It is Open Source.
- It has no license cost.
- It provides both good manuals and tutorials.

- The source code can be compiled on different platforms, e.g. Unix and Windows.
- Many wireless extensions have been contributed from the UCB Daedelus, the CMU Monarch projects and Sun Microsystems [118].
- New modifications can be easily included.

3.5 NS-2 Overview

Network Simulator 2 (NS-2) [56] developed at UC Berkeley, is an object-oriented, discrete event driven network simulator targeted at networking research. Simulation of wired as well as wireless network functions and protocols (e.g. routing schemes, TCP, UDP) can be done using NS-2. In general, NS-2 provides users with a way of specifying such network protocols and simulating their corresponding behaviours. NS-2 has been significantly improved by the open source community and its current release includes wireless network support provided by the CMU Monarch extensions [121].

The simulator is written in C++ and MIT's object extension to Tool command language (OTcl) [122]. C++ is a powerful programming language, which enables fast execution of applications. However, some modifications may be required in order to perform several simulations, that is, retaining the main structure of the simulation but modifying some parameters, with the purpose of comparing different results. This implies additional time; recompiling C++ code every time a modification is requested. OTcl is an interpreted language, and offers the key advantage that such modifications do not need additional time recompiling but on the other hand, the execution time for an interpreted language is slower than compiled languages. NS-2 makes this unification feasible through tclcl, i.e. OTcl linkage. The main objects of a simulation such as nodes and protocols are implemented using C++ and the configuration of the parameters such as number or position of nodes, time of the simulation, etc. are implemented in OTcl. Hence, the combination of the two languages offers a compromise between performance and ease of use. Implementation and simulation under NS-2 consists of 4 steps:

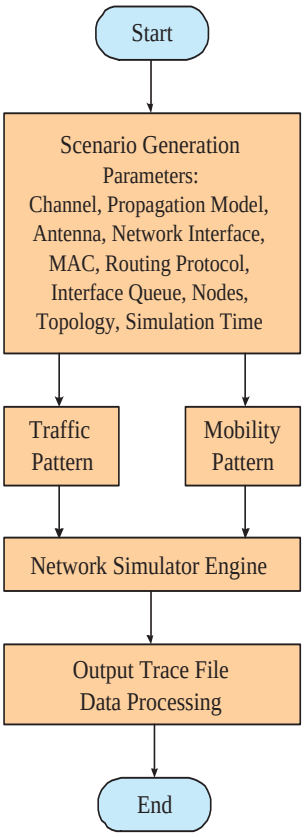

FIGURE 3.10: NS-2 simulation

- Implement the protocol by adding a combination of C++ and OTcl code to NS-2's source base.
- Describe the simulation in an OTcl script.
- Start the NS-2 simulation engine.
- Analyse the generated trace files to give statistics such as packet delivery ratio, or end-to-end delay.

Implementing a new protocol requires adding C++ code to represent the protocol's functionality, as well as updating key NS-2 OTcl configuration files in order for NS-2 to recognise the new protocol and its default parameters. The C++ code also describes which parameters and methods are to be made available for OTcl scripting. The NS-2 simulation process adopted for this work is summarised in Figure 3.10.

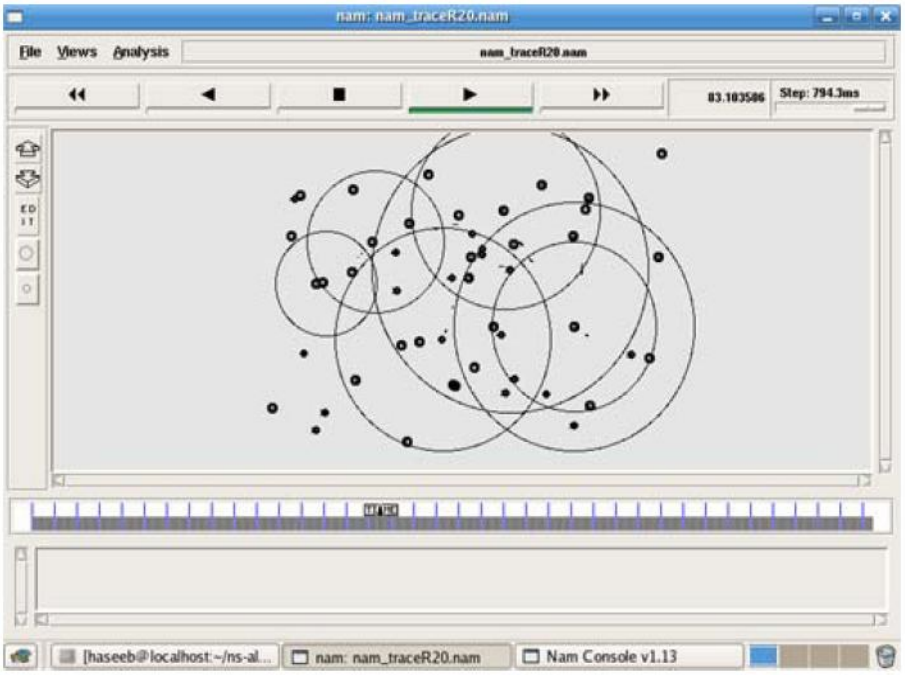

FIGURE 3.11: Network model

3.6 Simulation Implementation

This section provides a detailed descriptions of the network model, mobile node model, AODV routing process model state machine, and mobility process model implemented in NS-2.

3.6.1 Network Model

A screen shot of the network animator (NAM) depicting the network model is shown in Figure 3.11. The scenario contains 50 mobile nodes. Nodes can move around the specified area and communicate over wireless links with a transmission range of 250 m.

3.6.2 Mobile Node Model

Figure 3.12 shows the AODV mobile node model which simulates the protocol [121]. The AODV mobile node model consists of the following modules:

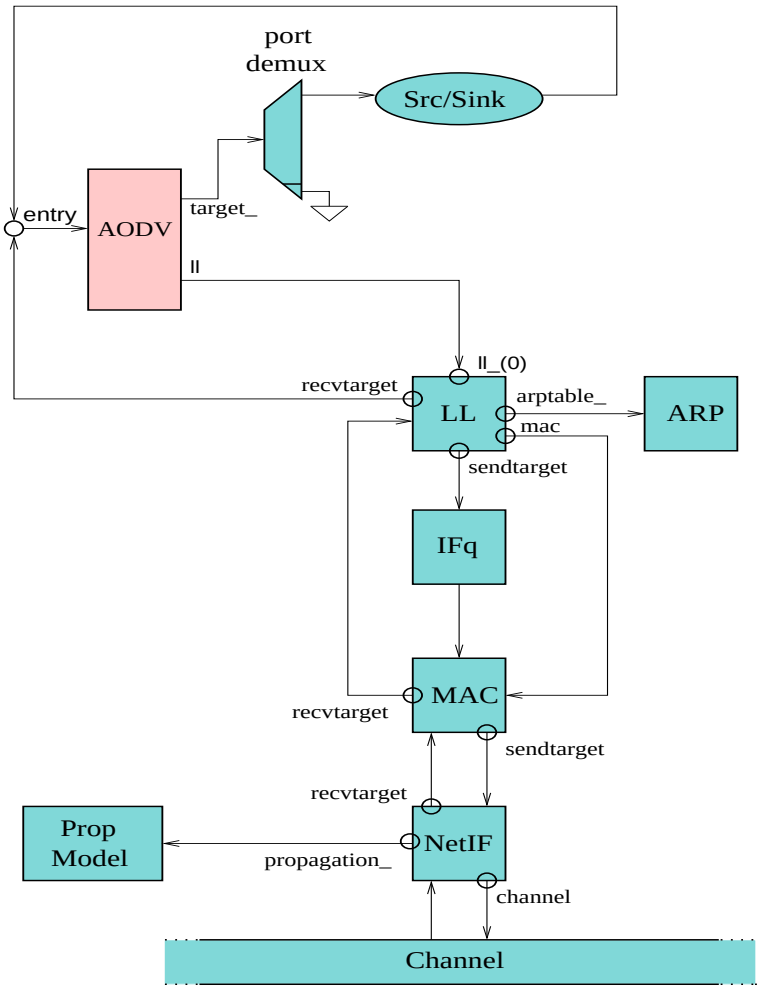

FIGURE 3.12: AODV node model

- **Src/Sink Module:**

  Packets sent by the source (Src) on the mobile node are handled by the mobile node's entry point, which passes them to the AODV agent. Once generated, packets are sent to the immediate lower layers. On the other hand, the sink agent will only receive packets through the port demultiplexer (port demux) if this is the final destination.

- **AODV Module:**

  This module is used to discover and maintain the routing information. Receiving a data packet from the node's entry point, the module first checks its cache. If there is a route path towards the destination node in cache, the module forwards

the data packet to the next node. Otherwise, the module executes the AODV algorithm to discover a route path to destination node.

- **LL Class and ARP Module:**

  The link-layer (LL) class is responsible for simulating the data link protocols. The LL class uses an Address Resolution Protocol (ARP) to determine the hardware addresses of the neighbouring nodes and map IP addresses to their correct interfaces.

- **IFQ Class:**

  An Interface Queue (IFQ) class 'Queue/DropTail/PriQueue' is used to queue all routing and data packets prior to being transmitted by MAC layer. The IFQ has a maximum size of 50 packets, maintaining a queue with two priorities each served in a first-in first-out (FIFO) order. Routing packets are assigned a higher priority than data packets.

- **MAC Module:**

  This module is an implementation of the IEEE 802.11 standard [10] the protocol commonly used in the MANET modelling community, and an appropriate candidate protocol to be used in the proposed applications.

- **Propagation and Antenna Models:**

  The propagation models used in NS-2 are described in section 2.2. The two-ray ground reflection model and a unity gain omni-directional antenna is used in order to make the results obtained from this study comparable with the work of the other researchers who have adopted the same approach. The antennas of the transmitter and the receiver are placed at a height of 1.5 metres above the ground.

- **NetIF Module:**

  The wireless network interface (NetIF) uses the characteristics of the 914MHz Lucent WaveLAN Direct Sequence Spread Spectrum (DSSS) radio [123]. WaveLAN is modelled as a shared-media radio with minimum range of 250m and

nominal bit rate of 2Mbps. The configuration parameters that make the wireless interface Phy/WirelessPhy model the Lucent WaveLAN radio interface is given in Table 3.1.

3.6.3 Mobility Model

In order to evaluate communication protocols for MANETs (such as AODV [68], DSR [69, 124]), it is necessary to develop and use mobility models that realistically represent the movements of mobile nodes that eventually utilise the given protocol. Recently, a number of mobility models which account for different motion patterns have been proposed. For example, Musolesi et al. in [125] propose the community based mobility model which models human mobility within communities and among different communities. T. Camp et al. in [126] introduce the Manhattan mobility model for vehicular mobility on structured roads in a city. In [127], G. Lin et al. propose the Group mobility model to model a motion pattern similar to military combat zones (e.g. the motion of a military infantry commander and his battalion). The random waypoint mobility model [128] is one of the most popular mobility models in MANETs research and widely used

TABLE 3.1: Wireless interface parameters

Parameter	Tcl class variable	Value
Raw bit rate (bps)	$Rb\_$	2×10^6
Power of transmission (W)	$Pt\_$	0.2818
Frequency (Hz)	$freq\_$	914×10^6
System loss factor	$L\_$	1.0
Carrier sense threshold (W): min power required to detect another node's transmission	$CSThresh\_$	1.559×10^{-11}
Receive threshold (W): min power required to receive a packet	$RXThresh\_$	3.652×10^{-10}
Capture threshold (dBm): signal ratio required to maintain receiver capture of incoming packet in face of collision	$CPThresh\_$	10.0

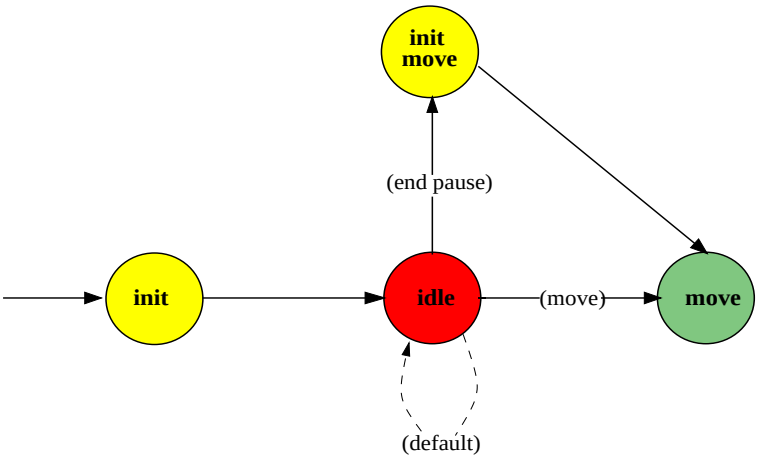

FIGURE 3.13: Random waypoint model

when evaluating MANETs [124, 129, 130, 126, 131, 109, 110] and hence is considered as the mobility model used in this thesis. The operation of the random waypoint mobility model can be summarised as shown in Figure 3.13 as follows:

- **Init State:**

  In the initialisation state, each node picks a random location inside the simulation area.

- **Idle State:**

  This is the default state where the node remains stationary for *pause time* seconds.

- **Init Move State:**

  In this state, the node selects a random destination in the simulation area and a random speed distributed uniformly between 0 and maximum speed.

- **Move State:**

  In this state, the node travels toward the chosen destination at the selected speed. Upon arrival, the node returns to the idle state and pauses for a specified time period before restarting the process.

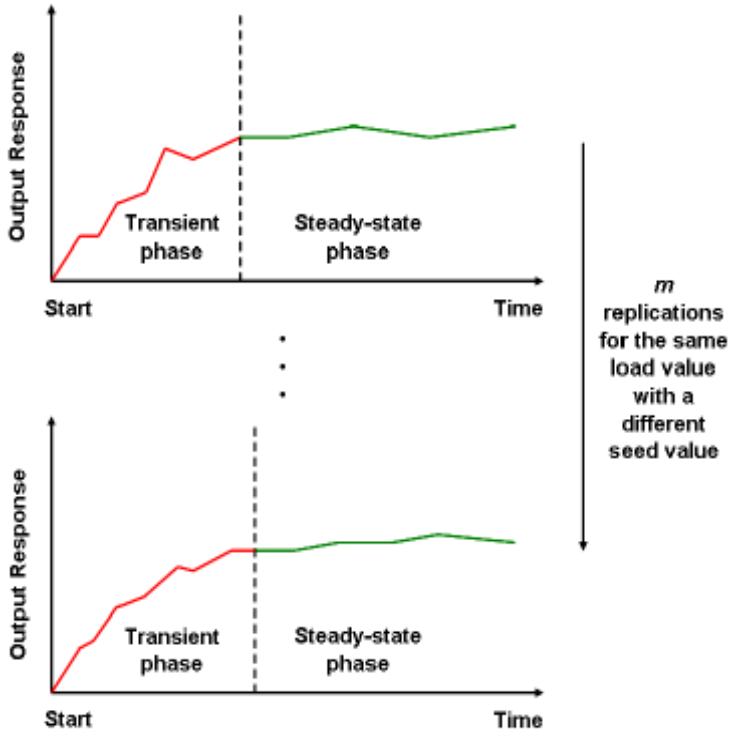

FIGURE 3.14: Recording procedure of the output results

3.7 Output Results Analysis

This section deals with the procedure for obtaining output results in terms of accuracy and repeatability. Figure 3.14 illustrates the recording procedure of the output results.

The simulation involves two phases: firstly, the transient (warm-up) phase whereby the average fluctuation experiences significant variation, and secondly, the steady-state phase in which the average fluctuation tends to remain within a narrow range. The data collection process must begin at the steady-state phase and it is essential that the simulation is repeated several times for the same load value with a different random number generator seed value. There are several issues which must be considered in order to achieve a higher level of accuracy including steady-state phase detection, simulation termination and number of replications. Although there are generally few deterministic methods to calculate these conditions, it is possible to calculate estimated values using some statistical techniques as described in the following sub-sections.

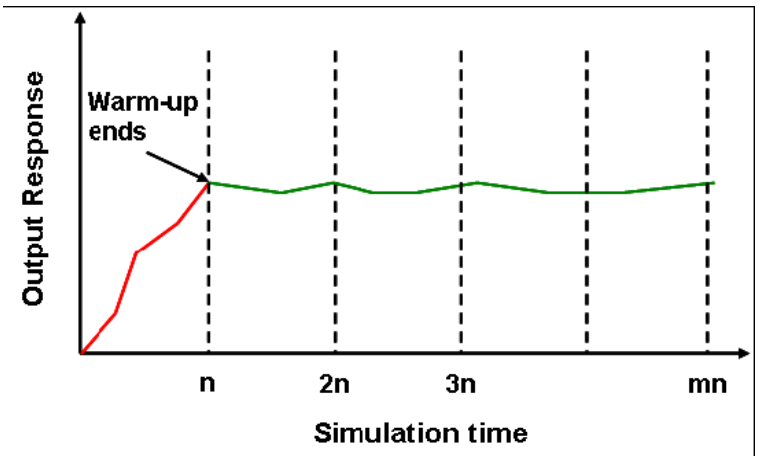

FIGURE 3.15: Batch means technique

3.7.1 Steady-State Detection

In most simulations, only the steady-state performance, that is, the performance after the system has reached a stable state, is of interest. It is important from an efficiency perspective that the onset of steady-state is detected as soon as possible. There are two common techniques used for detecting the steady-state phase: Batch Means and Moving Average of Independent Replications [110, 109, 107, 8].

3.7.1.1 Batch Means

The method of Batch Means requires running very long simulations and then dividing it up into several parts of equal duration. Each part is called a batch or sub-sample. The mean of observations in each batch is called the batch mean. The method requires analysing the variance of these batch means as a function of the batch size. As shown in Figure 3.15, a long run of N observations can be divided into m batches of size n each, where $m = N/n$. Let x_{ij} denote the j^{th} observation in the i^{th} batch, $\bar{x}_i$ is the batch mean and $\bar{\bar{x}}$ is the overall mean. The steps for steady-state detection using batch means are illustrated in Table 3.2 [8].

An advantage of batch mean technique is that only one transient (warm-up) interval needs to be removed during the process of recording observations. On the other hand, the limitation of this approach is that the batches, as illustrated in Figure 3.15, may not

TABLE 3.2: Steady-state detection using batch means [8]

Step	Description	Equation
1)	For each batch, compute a batch mean	$\bar{x_i} = \frac{1}{n}\sum_{j=1}^{n} x_{ij}, \quad i = 1,2,\cdots,m$
2)	Compute an overall mean	$\bar{\bar{x}} = \frac{1}{m}\sum_{i=1}^{m} \bar{x_i}$
3)	Calculate the variance of batch means	$Var(\bar{x}) = \frac{1}{m-1}\sum_{i=1}^{m} (\bar{x_i} - \bar{\bar{x}})^2$
4)	Repeat steps 1 to 3 for increasing batch size n	
5)	Plot the variance as a function of batch size n	
6)	The transient interval is the value of n for which the variance starts decreasing	

really be statistically independent. This would be especially true if the batch sizes are not large enough and will lead to significant correlation between successive batches. Since the confidence estimation measures discussed in section 3.7.2 are based on the independence of the individual simulation runs, this lack of independence between successive batches, if sufficiently serious, may have a significant impact on the accuracy of the confidence estimation procedures.

3.7.1.2 Moving Average of Independent Replications

Replication is the repetition of the simulation. For instance, if the simulation is repeated three times, the simulation is said to have three replications [8]. In the Moving Average of Independent Replications technique, the average is computed over a moving window interval. Multiple independent replications are performed which are subsequently averaged. Each of the independent replications is performed with common load values but different seed values for the random number generation. This results in a smoother

TABLE 3.3: Steady-state detection using moving average of independent replications [8]

Step	Description	Equation
1)	Compute the trajectory mean by averaging across replications	$\bar{x_j} = \frac{1}{m}\sum_{i=1}^{m} x_{ij}, \quad j = 1,2,\cdots,n$
2)	Set $k = 1$ and plot a trajectory of the moving average of successive $2k+1$ values	$\bar{\bar{x}}_j = \frac{1}{2k+1}\sum_{l=-k}^{k} \bar{x}_{j+l}, \quad j = k+1,k+2,\cdots,n-k$
3)	Repeat step 2, with $k = 2,3,\cdots$ until the plot is sufficiently smooth	
4)	Find the knee of the plot, the value j at the knee gives the length of the transient phase	

trajectory mean. Given m replications of size n each, let x_{ij} denote the j^{th} observation in the i^{th} replication, $\bar{x_j}$ is the trajectory mean across m replications and $\bar{\bar{x}}_j$ is the mean of observations within the window of length k. The steps for steady-state detection using moving average of independent replications are illustrated in Table 3.3 [8].

The key advantage of independent replications technique is that it ensures that the samples are independent. On the other hand, the limitation of this approach is that running through the warm-up phase for each replication extends the length of time to perform the replications. Furthermore, there is a possibility that the length of the warm-up period is under estimated, resulting in bias for each instantiation. It is highly desirable in the independent replications technique that the warm-up period is properly detected and removed.

As an example, the average routing overhead (the ratio of the total number of routing packets to data packets transmitted during the simulation) is examined in order to determine the warm-up time of the simulation. When the moving average technique is applied to the average routing overhead, the graph of Figure 3.16 is obtained. The knee of the plot occurs at approximately 200 seconds. In order to ensure that the warm-up period is completely removed, the first 200 seconds of the simulation time are ignored.

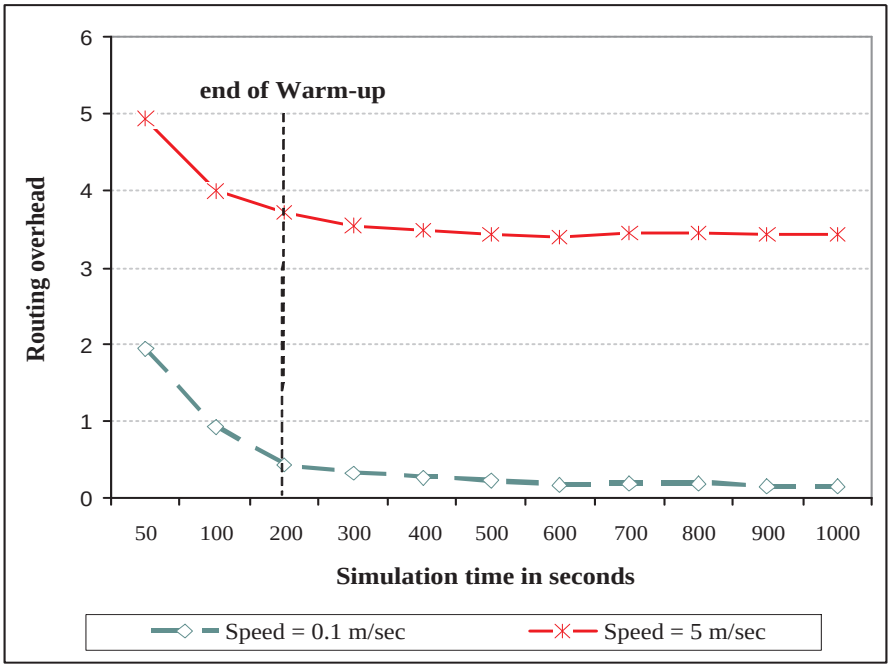

FIGURE 3.16: Moving average applied to the average routing overhead

This is also one of the solutions suggested previously [109, 110] and hence used in this work.

3.7.2 Simulation Time and Confidence Intervals

The simulation time should be properly chosen to guarantee a sufficiently accurate estimate of the final results. If the simulation is too short, the results may be highly variable. On the other hand, if the simulation is too long, computing resources may be unnecessarily wasted. It is better to err towards the latter in that results are free of bias and are stable even though they take a longer time to obtain. The simulation should be repeated with a different random number generator seed value. Ten replications ($m = 10$) are considered in this work of size $n + n_t$ each, where n_t is the length of transient phase. The first n_t observations of each replication are discarded. An overall mean is calculated across the replications. The variance of the replicate means is then calculated and this provides the confidence interval for the mean response. Finally the variance of the replicate means is utilised to find the upper and lower confidence intervals denoted with

TABLE 3.4: Confidence interval for independent replications [91]

Step	Description	Equation
1)	Compute the mean for each replication	$\bar{x_i} = \frac{1}{n} \sum_{j=n_t+1}^{n_t+n} x_{ij}, \quad i = 1, 2, \cdots, m$
2)	Compute the overall mean	$\bar{\bar{x_i}} = \frac{1}{m} \sum_{i=1}^{m} \bar{x_i}$
3)	Calculate the variance of replicate means	$Var(\bar{x}) = \frac{1}{m-1} \sum_{i=1}^{m} (\bar{x_i} - \bar{\bar{x}})^2$
4)	An approximate $100(1-a)\%$ confidence interval for the mean response is:	$C = \bar{\bar{X}} \mp t(\sqrt{\frac{Var(\bar{x})}{m}})$, where $t \equiv t_{m-1,1-\frac{\alpha}{2}}$ is chosen such that: $P\{T_{m-1} \leq t\} = 1 - \frac{\alpha}{2}$ and T_{m-1} is a student-t random variable with $m-1$ degrees of freedom)

vertical bars across the mean values. This is achieved by using the Student's t distribution table since the number of replications is less than thirty [8, 132]. If the number of replications was greater than thirty, a normal distribution could have been used. A confidence interval of 95% (confidence level α of 0.05) is used throughout this thesis. Table 3.4 outlines the steps required to obtain the confidence intervals for the sampled results [132, 109, 110].

3.8 Model Validation

For simulation results to be credible the simulation models in use must undergo verification and validation. O. Balci [133] defines verification as substantiating that a model is built from a problem formulation accurately, where validation is substantiating that the model behaves with satisfactory accuracy within its domain. J.S. Carson [134] and R.G. Sargent [135] define the two terms to be similar and both note that sufficient accuracy is recited when a model can be used instead of a real system for purposes of experimentation and analysis.

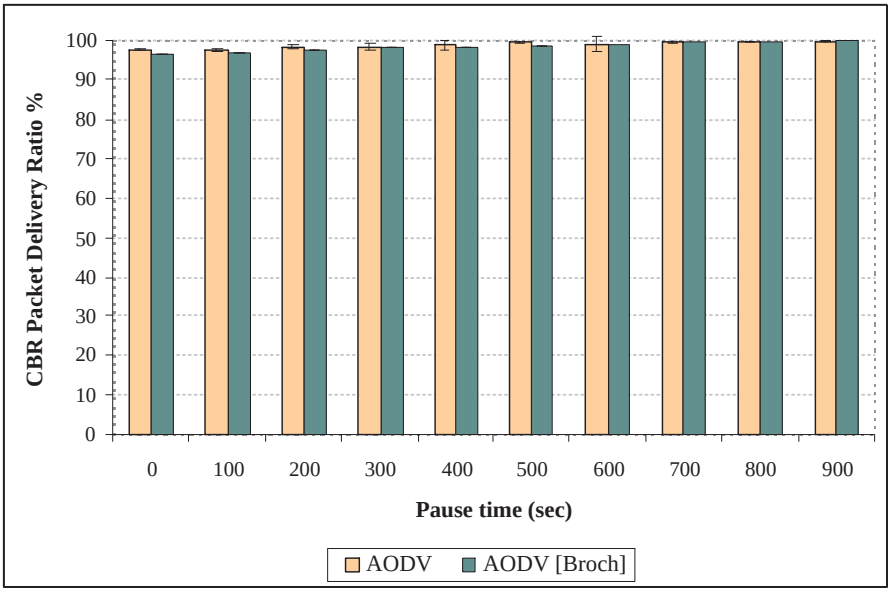

FIGURE 3.17: Packet delivery ratio PDR % vs pause time (sec)

This section presents results of preliminary simulation-based experiments in order to ensure that the implementation of the AODV [68] scheme inside the simulator (NS-2) is faithful to the scheme's specifications. This scheme will be further used for performance comparison with the proposed key management schemes presented in the following chapters. The scheme implementation is validated in two ways. The first validation was performed using the NS-2 "validation test suite", which compares the simulation results of the AODV protocol with some reference simulation results. The second validation test was performed by comparing the simulation results with that of known baseline.

AODV model validation is carried out with the same simulation parameters as stated by J. Broch et al. [136]. The simulation environment consists of 50 wireless nodes forming an ad hoc network, moving according to the random waypoint mobility model within a 1500x300 metres rectangular flat space, which can be setup using a scenario generator script setdest [128]. All simulations are run for 900 seconds of real time. Data traffic is generated using constant bit rate (CBR) using a connection pattern generator script cbrgen.tcl [56]. 20 CBR nodes are chosen randomly from the full set of nodes generating 64 byte data packets at a rate of 4 packets per second.

Figure 3.17 and Figure 3.18 show the packet delivery ratio (the ratio between the number of packets originated by the "*application layer*" CBR sources and the number of

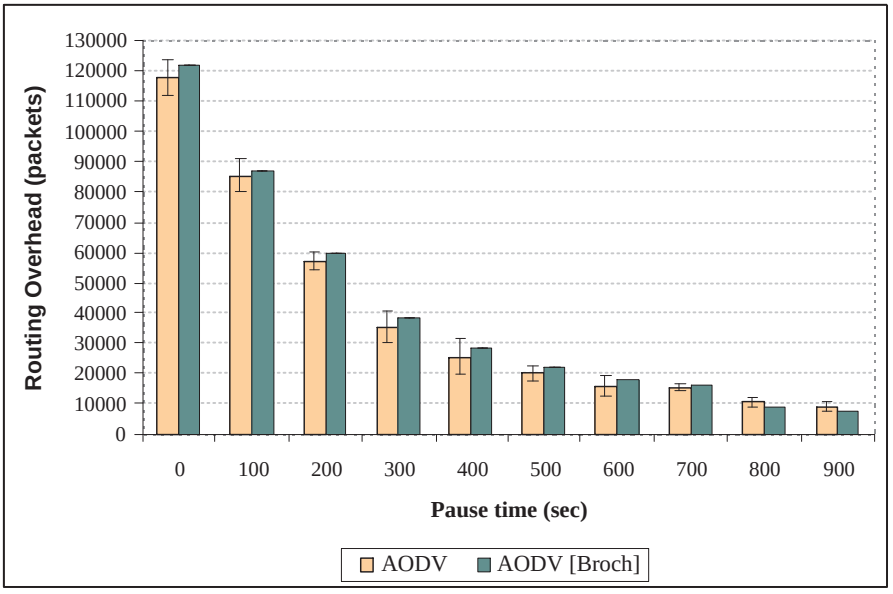

FIGURE 3.18: Routing overhead (packets) vs pause time (sec)

packets received by the CBR sink at the find destination) [136] and routing overhead (the total number of routing packets transmitted during the simulation) [136] as a function of pause time for maximum node movement speed of 20 m/s. Each graph represents the average of 10 random mobility and traffic scenarios for the given pause time, and the error bars represent the corresponding confidence interval of 95%, calculated as described in section 3.7. A pause time of 0 seconds is equivalent to continuous motion, while a pause time of 900 seconds (the length of the simulation) is equivalent to no motion. The results are at the same level when compared to the reported results as both simulations are conducted using same simulation package, i.e. NS-2. The AODV protocol delivers almost all data packets, regardless of pause time, with the packet delivery ratio rising to 100% at pause time 900 (a stationary network). Similarly, the routing overhead approaches that of J. Broch [136] for different values of pause time. Decreasing the pause time causes an increase of the relative mobility of the network nodes which results in an increase of routing packets transmitted during simulations.

3.9 Summary

The first part of this chapter presents the concept of key management with emphasis on the classifications of key management schemes in MANETs. The requirements of key management in MANETs such as authenticity, confidentiality, integrity, scalability, availability, and flexibility are described. A number of key management schemes proposed for MANETs are discussed, aiming at showing various strategies of utilising key management in MANETs. A summary of these schemes is given, highlighting their features, characteristics, and suitability for MANETs. The second part of this chapter explains the evaluation methodology used for evaluation of key management schemes. The simulation model of AODV in NS-2 simulator is described. A number of preliminary simulation-based experiments for AODV are presented to raise confidence in the model behaviour and assumptions.

Chapter 4

An Authority-Based Key Management Protocol

4.1 Introduction

Mobile ad hoc networks (MANETs) are self-configurable, autonomous networks with dynamic topologies and their function heavily depends on the collaboration among ad hoc nodes. Security issues in MANETs have become a central concern and are increasingly important. Symmetric cryptography based, less costly key establishment protocols are more suitable for MANETs applications due to their efficiency and low energy consumption and code storage [137]. In such a case, the main challenge is to find an efficient way of distributing keys and keying materials to mobile nodes.

There are three main techniques for constructing symmetric keys: key distribution using trusted third parties (TTPs), key agreement, and key pre-distribution. TTP-based schemes (for example, Kerberos [138]) rely on a trusted server or servers organised in a hierarchical structure for key agreement between nodes and the TTP owns a database containing the name, the secret key, the expiration time of the secret key and some administrative information of each entity. These schemes may not be practical for large-scale networks, since the deployment of TTP servers is either prohibited due to environmental constraints or uneconomical because of the scalability issue. Key agreement

schemes (such as the Diffie-Hellman key agreement [139]) set up pairwise keys through successive message exchanges in a secure manner by using asymmetric cryptography. These schemes suffer from the limited computation and communication capabilities within mobile nodes. Key pre-distribution schemes (KPD), on the other hand, have attracted much attention in the MANETs research society due to their efficiency and simplicity. In KPD, a trusted authority generates and distributes secret pieces of information that allow some privileged users to recover a private key. A privileged user must be able to compute the secret key individually. A number of KPD protocols have been proposed in the literature [34, 35, 36, 37, 38]. Pre-distribution of secret keys for all pairs of nodes is not viable due to the large amount of memory used when the network size is large. Such an inefficient use of the limited memory in mobile nodes makes the current KPDs scale poorly to very large networks.

In [94], the key distribution scheme which is known as *"Key Distribution in Mobile Ad Hoc Networks based on Message Relaying"* distributes certificates in authority-based MANETs by using an off-line authority to issue each node with its keying material (certificates) before joining the network. The distribution of certificates occurs on a peer-to-peer basis by exploiting the routing infrastructure and chaining peer nodes together. Certificates are transferred along these virtual chains via a message relaying mechanism. Certificates must be distributed on-demand as needed on the network routing layer. The applicability of the Key Distribution in Mobile Ad Hoc Networks based on Message Relaying scheme is mainly constrained by the limited computation and communication capabilities within mobile nodes.

In this chapter, we propose a novel authenticated symmetric key distribution scheme (*ASKD*) for MANETs. In the route request process, the proposed scheme uses Van der Merwe's method [94] for distributing public key certificates. In the route reply process, the public key certificates are used in generating and distributing authenticated symmetric keys between each neighbouring nodes along the path from the destination to the source nodes and between the destination and source nodes. The proposed scheme exploits the routing infrastructure to effectively distribute symmetric keys between mobile nodes. *ASKD* can issue each node in authority-based MANETs with the minimum

amount of keying material sufficient to generate a shared key between this node and any other node in the network. The proposed scheme uses mobility as an aid for bootstrapping the security association between mobile nodes. The *ASKD* is implemented over an on-demand routing protocol. It distributes symmetric keys between mobile nodes in two steps. In the first step, it distributes certificates during the route request process and in the second step, it disseminates symmetric keys during the route reply process. This means that at the end of the route discovery process, there will be a shared key not only between the source node and the destination node but also between any pair of nodes along the route from the source to the destination.

4.2 Overview of the Symmetric Key Distribution Scheme

4.2.1 Network and Security Assumptions

Mobile ad hoc network assumptions are as follows:

- The proposed symmetric key distribution scheme works in support of an on-demand routing protocol;
- We assume that nodes are fully distributed with a moderate node density and a medium access control such as IEEE 802.11 [10];
- Nodes can be stationary or move with low to high mobility (0 m/sec to 20 m/sec).

Security Assumptions are as follows:

- There exists an off-line authority to issue each node with its own certificate before joining the network;
- The certificate must contain the off-line authority's identity, the node's public key, an identity/network address, a unique sequence number, certificate generation date and expiry date.

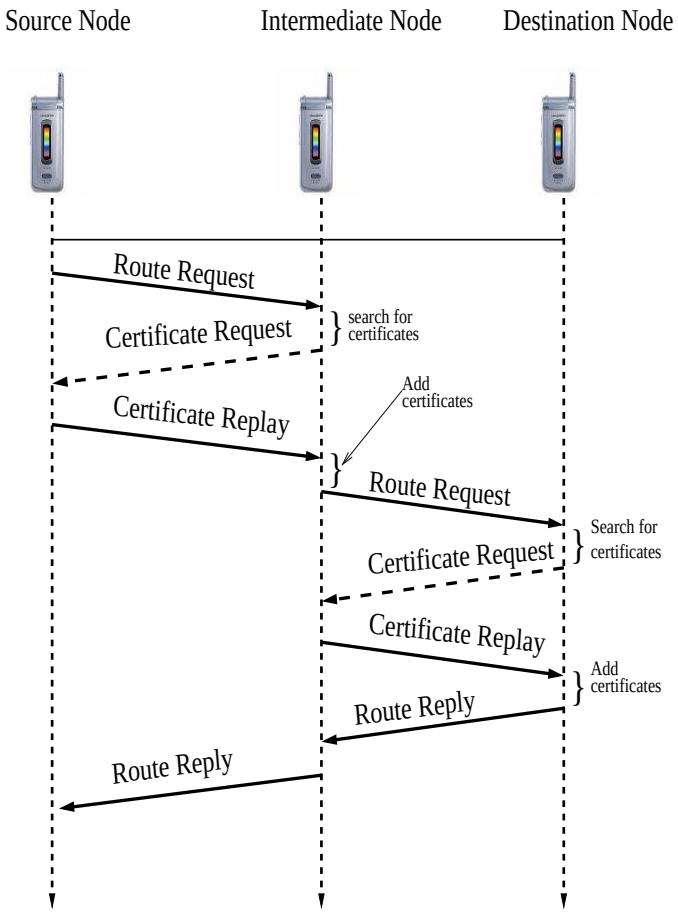

FIGURE 4.1: Certificate distribution main procedure

The randomness of the key generation algorithm should be verified and the generated keys should pass all statistical tests for randomness [140]. The OpenSSL library (version 0.9.8h) [141] is used for generating certificates, symmetric keys and digital signatures [22].

4.2.2 Symmetric Key Distribution Schemes

The proposed symmetric key distribution scheme can be described as follows:

- The routing request message received by an intermediate node acts as a trigger for the node to search in its certificates repository about certificates of the source node and the previous hop node. If certificates are in the repository, the intermediate node will process the route request as normal. If the search finds that certificates

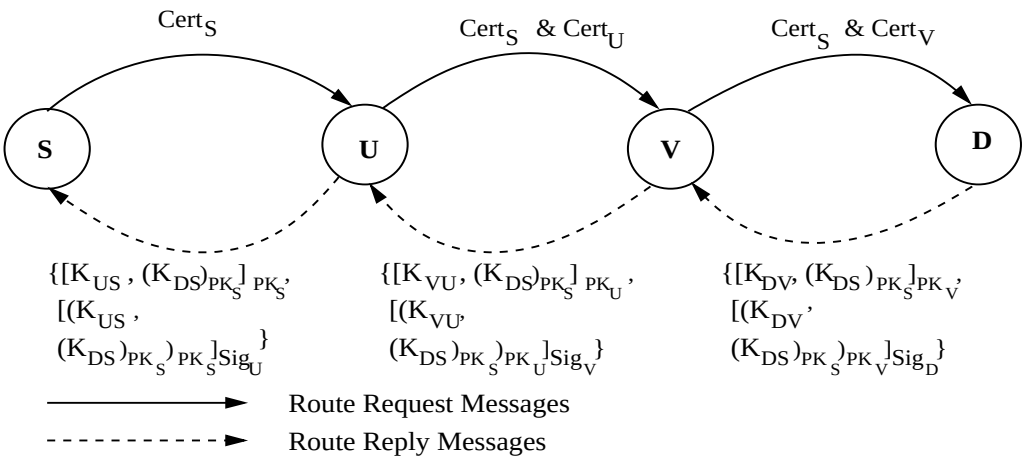

FIGURE 4.2: Symmetric key distribution main procedure

are not in the repository, the node sends a certificate request to the previous hop node requiring certificates of the source node and the previous hop node as shown in Figure 4.1.

- When the intermediate node requests certificates of the source node and the previous hop node, it concatenates its own certificate to the certificate request.
- If the source node is itself the previous hop node, then the source node will reply by sending its own certificate to the intermediate node.
- If the source node and the previous hop node have different addresses (i.e. they are not the same node), the previous hop node replies with its own certificate and the certificate of the source node.
- When the intermediate node receives certificates and after verifying them, signatures on certificates are verified with the issuer's corresponding public key, it adds them to its certificate repository and processes the route request as normal.
- The processes of certificate request and certificate reply will be carried on between all nodes along the route from the source to the destination until the destination node receives certificates of the source node and the previous hop node and adds them to its certificate repository.
- If we suppose the case when the route from the source to the destination consists of four nodes as shown in Figure 4.2, the destination node *D* will generate two

symmetric keys, K_{DS} for encrypting transferred messages between the destination node D and the source node S, and K_{DV} for encrypting transferred messages between node D and the previous hop node V. At this point, the destination node D has certificates of both the source node S ($Cert_S$) and the previous hop node V ($Cert_V$), i.e. it has their public keys, then node D will do the following:

1. Encrypts the generated symmetric key: K_{DS} with node S's public key PK_S.
2. Encrypts the concatenation of the encrypted key generated from the previous step and the generated key K_{DV} with node $V's$ public key PK_V .
3. Signs the result of the previous step with its secret key SK_D.
4. Concatenates the outputs of the last two steps to the route reply message and send them to node V as illustrated in the following message:

   $D \rightarrow V : \{RREP; [K_{DV}, (K_{DS})_{PK_S}]_{PK_V}, [(K_{DV}, (K_{DS})_{PK_S})_{PK_V}]_{Sig_D}\}$

   where Sig is the digital signature using the node's private key.

- When node V receives the route reply message from node D, it will do the following:

  1. The signature of node D will be verified using node D's public key PK_D and the symmetric key (K_{DV}) will be recovered from the encrypted part using node V's secret key SK_V .
  2. Node V will generate a shared key (K_{VU}) to be used in encrypting the forthcoming communications between node V and node U.
  3. Node V encrypts the concatenation of the encrypted key received from node D: $(K_{DS})_{PK_S}$and the generated key K_{VU} with node U's public key PK_U.
  4. Signs the result of the previous step with its secret key SK_V .
  5. Concatenates the outputs of the last two steps to the route reply message and send them to node U as shown below:

$V \rightarrow U : \{RREP, [K_{VU}, (K_{DS})_{PK_S}]_{PK_U}, [(K_{VU}, (K_{DS})_{PK_S})_{PK_U}]_{Sig_V}\}$

- When node U receives the route reply message from node V, it will do the same steps as node V and sends the following message to node S:

  $U \rightarrow S : \{RREP; [K_{US}, (K_{DS})_{PK_S}]_{PK_S}, [(K_{US}, (K_{DS})_{PK_S})_{PK_S}]_{Sig_U}\}$

- Node S receives the route reply message from node U and make the necessary decryption and verification to extract the two shared keys included in the route reply message, K_{US} which will be used for encrypting any forthcoming communication between node S and node U, and K_{DS} which will be used for encrypting the transmitted packets from node S to node D or vice versa.

To save bandwidth and memory of nodes, the number of symmetric keys included in each route reply message is limited to be only two symmetric keys. Taking into consideration the example in Figure 4.2, those keys are K_{DV} and K_{DS} in the first hop from D to V, K_{VU} and K_{DS} in the second hop from U to V, and finally K_{US} and K_{DS} in the last hop from U to S.

4.2.3 On the Security of the Proposed Symmetric Key Distribution Scheme

In this section, we study the security of the proposed *ASKD* protocol for MANETs. In the simplest form of passive attacks, an intruder attempts to obtain the secret key between two particular nodes for example, U and V, without compromising them. To implement a level of security suitable for our purpose, we can choose one-way hash functions such as SHA-1 [142], whose inverse is computationally infeasible for the purpose of key generation. In order to obtain the secret key between nodes V and U (K_{VU}), the intruder must decrypt the keying material included in the route reply message sent from node V to node U, which is computationally infeasible as the intruder does not know the private key of node U (SK_U). Obtaining the secret key between nodes D and S (K_{DS}) from the route reply message transmitted from node V to node U is more difficult, because the intruder should obtain the private keys (SK_U) and (SK_S) of nodes U and S respectively, which is computationally infeasible without compromising node U and node

S. To ensure the authenticity and non-repudiation of the symmetric keys distributed between mobile nodes, all distributed keys are signed by the previous hop node. Forging the digital signature and changing the distributed keys are computationally infeasible without compromising the signing node and obtaining its private key.

4.3 Simulation Environment and Performance Metrics

In this section, we present the simulation environment and performance metrics of our proposed scheme.

4.3.1 Simulation Environment

The simulations were performed using Network Simulator (NS-2) [56], particularly popular in the ad hoc networking community. The MAC layer protocol IEEE 802.11 is used in all simulations. The source-destination pairs are spread randomly over the network. The NS-2 constant bit rate (CBR) traffic generator is used to set up the connection patterns with different random seeds. CBR traffic effectively stresses a network because there are no control mechanisms to consider when flows are delayed or packets lost. Each node has one CBR traffic connection with a single unique destination and can generate at most 10,000 packets. Sources initiation time is uniformly distributed over the first 90 seconds of the simulation time. Every simulation run is 1000 seconds long. The mobgenss [128] mobility scenario generator was used to produce random mobility patterns. The pause time is set to zero. The Ad Hoc On-demand Distance Vector (AODV) routing protocol [68] was chosen for the simulations. The simulation results are the average of 10 runs. The rest of the simulation parameters are summarised in Table 4.1.

TABLE 4.1: Simulation parameters

Parameter	Value
No. of Nodes	50
Area (m2)	1000x1000
Transmission range	250m
Mobility Model	Random waypoint
Propagation Model	Two-Ray Ground
Mean speeds (m/s)	0,5,20
Data Rate	11 Mbs
Load	1-7 packet/s
CBR connections	50
Data packet size	512 bytes
Certificate packet size	512 bytes

4.3.2 Performance Metrics

We have selected the *packet delivery ratio*, *average end-to-end delay*, *certificate delivery ratio*, and the *symmetric key delivery ratio* as metrics during the simulation in order to evaluate the performance of the proposed scheme.

- **Packet delivery ratio (*PDR*):**

  PDR is the ratio of data packets delivered to destinations to those generated by sources. It is a key metric since it shows the packet loss rate which, in turn, affects the throughput of the network. This metric indicates both the completeness and correctness of a routing scheme.

- **Average end-to-end delay:**

  It includes all possible delays caused by buffering during route discovery latency, queueing at the interface queue, retransmission delays at the MAC, and propagation and transfer times.

- **Certificate delivery ratio (*CDR*):**

  CDR is the ratio of received certificates to those transmitted by the nodes. This metric measures the efficiency of the proposed scheme in delivering public key certificates between communication entities.

- **Symmetric key delivery ratio (*SKD*):**

  SKD is the ratio of received symmetric keys to those transmitted by the nodes. This metric indicates the robustness and completeness of the proposed scheme in distributing symmetric keys between the communication entities along the paths from sources to destinations.

4.4 Performance Evaluation

In this section, we measure the impact of our proposed scheme on the performance of the AODV routing protocol [68].

4.4.1 Packet Delivery Ratio (PDR)

Figure 4.3 shows that the results of the constant bit rate (CBR) reference packet delivery ratio (*PDR*) with and without *ASKD* are highly correlated to each other and the impact on network performance is negligible for 0.1 m/sec, 5 m/sec and 20 m/sec mobility, which means that the proposed scheme exploits mobility; as the mobility increases the CBR and CBR with *ASKD* simulations become even more correlated.

4.4.2 End-to-End Delay

Figure 4.4 shows that the *ASKD* scheme does not increase the delay for delivered packets. The end-to-end delay increases with mobility until the end-to-end delay graphs intersect and nodes with higher mobility had lower end-to-end delay than nodes with lower mobility. The end-to-end delay graphs intersect because the average hop length

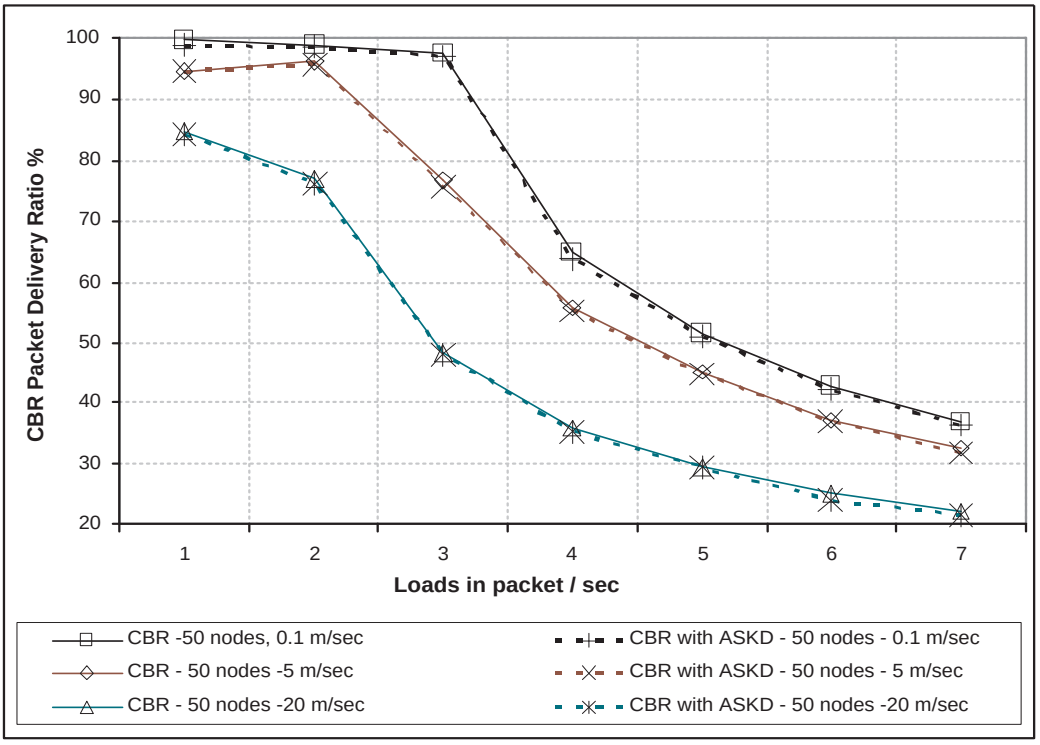

FIGURE 4.3: ASKD's packet delivery ratio PDR % vs load in packet/sec

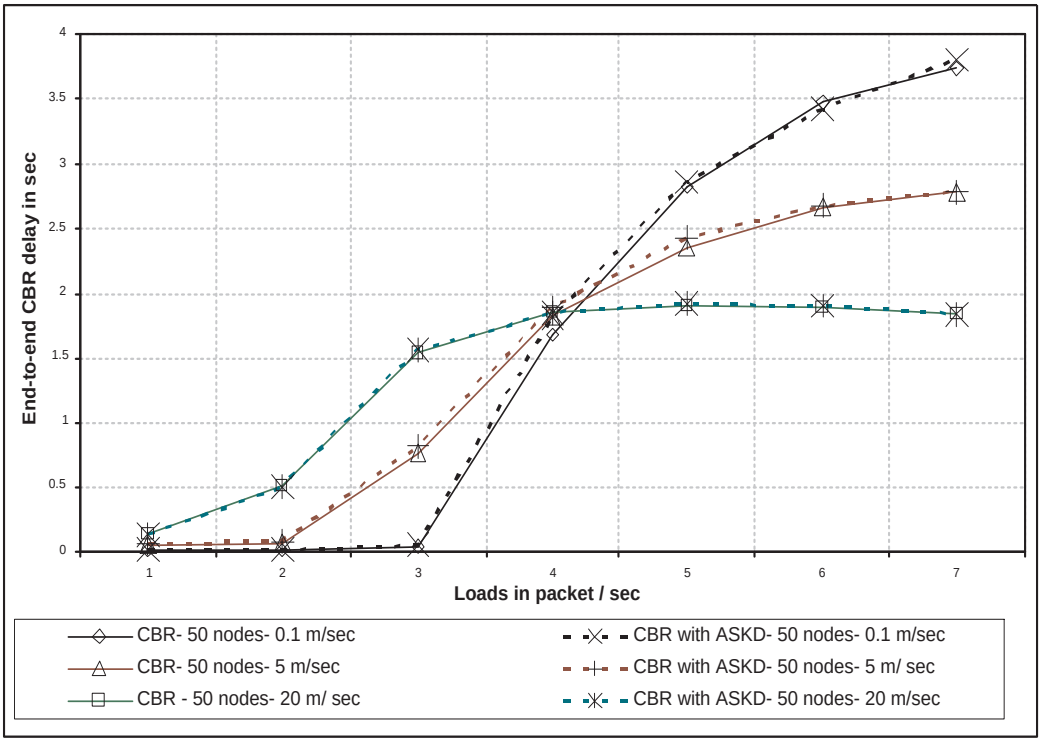

FIGURE 4.4: ASKD end-to-end delay in sec vs load in packet/sec

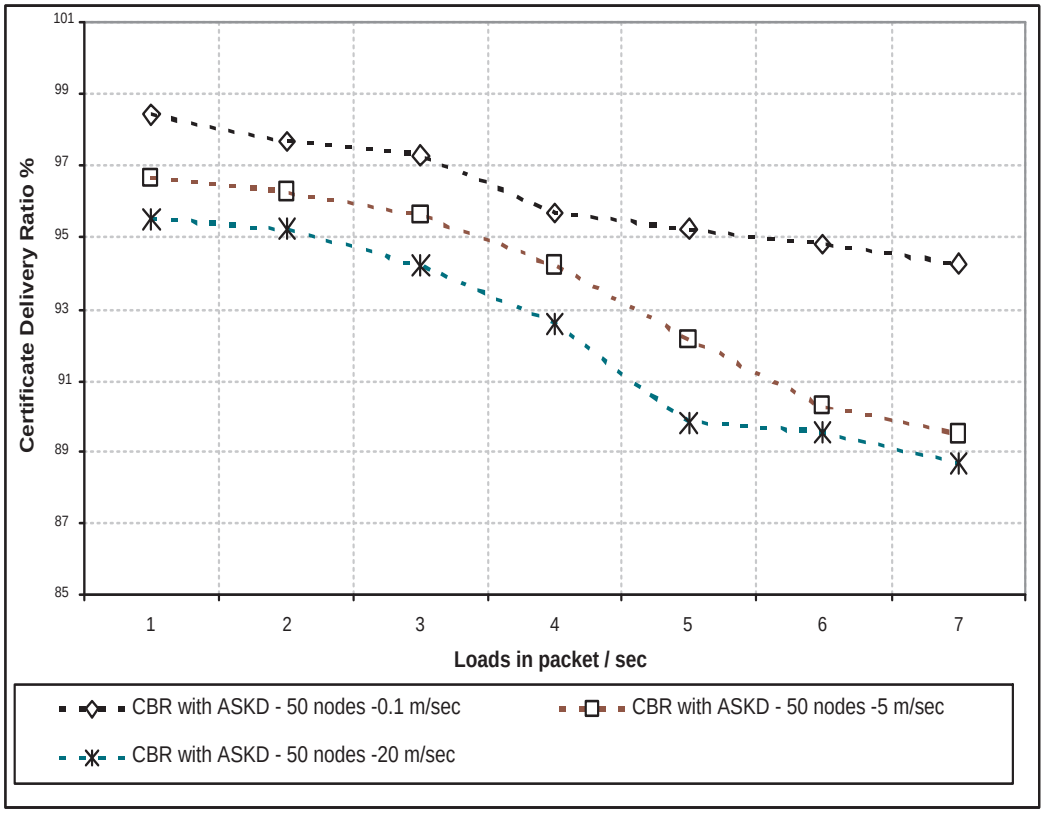

FIGURE 4.5: ASKD certificate delivery ratio vs load in packet/sec

(*AHL*), i.e., the distance between the source and the destination in hops, decreases with increasing node mobility due to increasing the routing failures with increasing node mobility. The *AHL* also decreases with increasing the load in packet/sec due to increasing radio interference and collisions between nodes. At high load, packets from nodes with a higher mobility traverse a lower number of hops than packets belonging to lower mobility nodes. This results in lower end-to-end-delay.

4.4.3 Certificate Delivery Ratio (*CDR*)

Figure 4.5 shows that the certificate delivery ratio (*CDR*) varies between 98% and 88% because of using the localised one-hop communication in distributing certificates between mobile nodes. It can be noticed that the average certificate delivery ratio decreases with an increase in mobility. It also decreases with an increase in load because

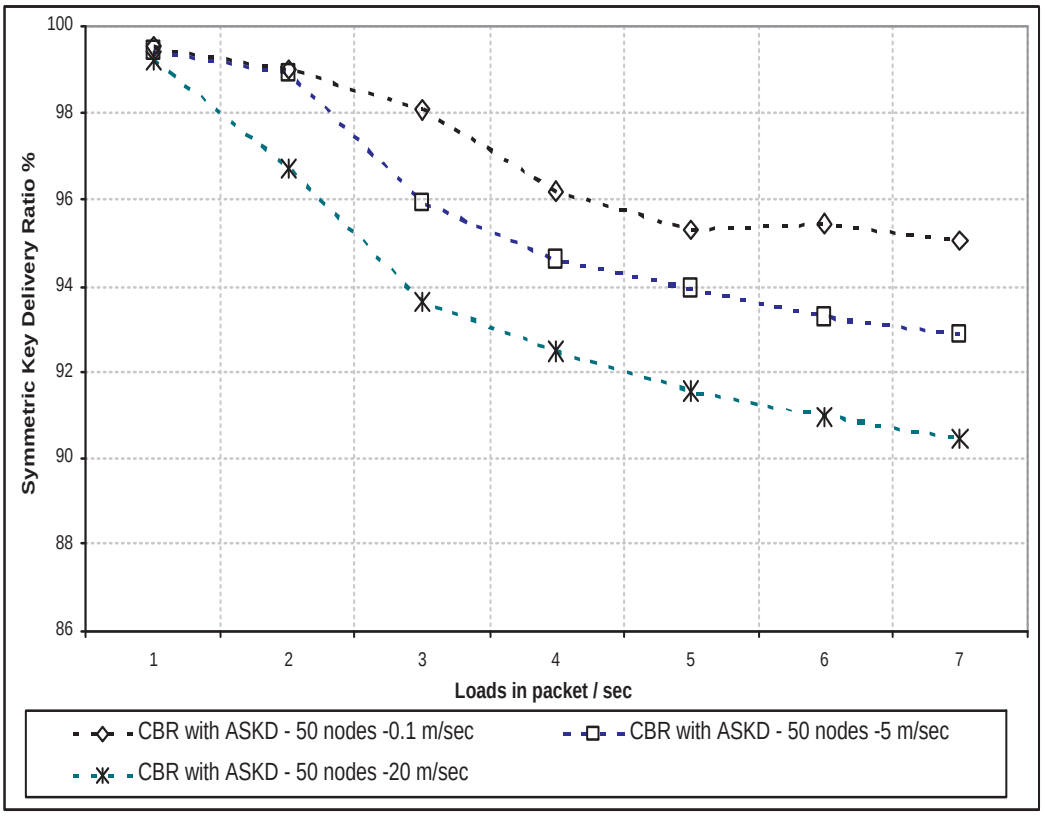

FIGURE 4.6: ASKD symmetric key delivery ratio vs load in packet/sec

this scenario is the worst case scenario where all nodes in the network are transmitting and receiving packets which yield an increase in collisions between nodes.

4.4.4 Symmetric Key Delivery Ratio (SKDR)

Figure 4.6 shows the symmetric key delivery ratio (*SKDR*) as a function of load in packet/sec. *SKDR* varies between 99% and 90% which highlights the success of the proposed scheme in distributing symmetric keys between neighbouring nodes along routes from destination nodes to source nodes. Results show that the proposed scheme avoids dependence on mobility by exploiting the localised one-hop communication in distributing symmetric keys.

4.5 Summary

In this chapter, a new authenticated symmetric key distribution scheme for MANETs has been proposed. The proposed scheme exploits route reply messages in distributing symmetric keys between neighbouring nodes along the reverse route from the destination node to the source node. Inflating the route reply message with a large number of symmetric keys will waste the bandwidth which is of a major concern in MANETs. So, we limited the number of symmetric keys included in the route reply message to be only two symmetric keys. The proposed scheme breaks the routing-security interdependence cycle [55], because it does not depend on the routing infrastructure as it uses the localised one-hop communication in distributing symmetric keys. The confidentiality of the distributed keys is guaranteed by encrypting them with the recipient node's public key. The digital signature is used to guarantee the authenticity of the distributed keys. Simulation results show that the proposed scheme has low communication overhead and negligible impact on the network performance. The proposed scheme is suitable for nodes with low to high mobility (0 m/sec to 20 m/sec) since the symmetric key delivery ratio *SKDR* varies between 99% to 90% for 0.1 m/sec, 5 m/sec and 20 m/sec nodes mobility.

Chapter 5

A Self-Organized Public Key Management Protocol

5.1 Introduction

If a-priori trust relations exist between the nodes in an ad hoc network, entity authentication can be sufficient to assure the correct execution of critical network functions. A-priori trust can only exist in a few special scenarios like military networks and corporate networks, where a common, trusted authority manages the network [143]. With the lack of a-priori trust, classical network security mechanisms can not be used in MANETs. The trust relationships established between network nodes could be used for the provision of higher level security solutions such as key management. Key management is a basic part of any secure communication. Secure network communications normally involve a key distribution procedure between communication parties, in which the key may be transmitted through insecure channels. In MANET, key management can be classified into two kinds; the first one is based on a centralised or distributed trusted third party (TTP). The TTP is responsible for issuing, revoking, renewing, and providing keying material to nodes participating in the network such as [2], [82], and [4] where the key management process is performed using threshold cryptography [39]. In the (n,t) threshold cryptography, a secret key is divided into n shares according to a

random polynomial and kept by n legitimate nodes, which we call share holders. Later, a new node needs to collect t shares from the response of t nodes (among n nodes) based on Lagrange interpolation and generates the original secret key as a legitimate node.

The second kind of key management is the self-organised key management schemes, such as [83], and [84]. Self-organised schemes allow nodes to generate their own keying material, issue public key certificates to other nodes in the network based on their knowledge. Certificates are stored and distributed by the nodes. Each node maintains a local certificate repository that contains a limited number of certificates selected by the node according to an appropriate algorithm. Public key authentication is performed via chains of certificates.

In this chapter, we propose a robust self-organised public key management for MANETs. The proposed scheme is based on the existence of a web of trust between mobile nodes forming the network. Since the use of random graph theory in MANET is limited to quasi-static networks due to constraints and nodes mobility [144], random graph theory is used in the proposed scheme to represent the public keys and certificates of the system in the initialisation phase only. The proposed scheme allows each user to create its public key and the corresponding private key, to issue certificates to neighbouring nodes, and to perform public key authentication without relying on any centralised authority. Each node in the network has a trust table to store the public key certificates and the corresponding trust values. A trust value represents a node's belief that another node is trustworthy. The certificate chain discovery will be performed with the aid of the routing process. A measure of the communication cost is proposed. A comprehensive analysis of the proposed scheme in the mobility environment of MANETs is performed.

The rest of the chapter is organised as follows: Section 5.2 reviews related work. The system description and trust model are presented in Section 5.3. The simulation environment and performance metrics are presented in Section 5.4. In Section 5.5, we illustrate the performance evaluation and security analysis of the proposed scheme. Finally, Section 5.6 concludes the chapter.

5.2 Related Work

In this section a review of related key management schemes for MANETs will be presented. In [2], and [98], threshold cryptography has been proposed to provide a reliable, distributive key management for MANET by exploiting some nodes as trust anchors for the rest of the network. In these schemes, threshold cryptography involves additional computationally intensive modular exponentiation compared to the underlined asymmetric-key cryptographic protocols. Most low-powered wireless nodes do not have the resources to handle such computationally intensive operations. Capkun et al. in [4] proposed a self-organised public key management scheme in which each node issues certificates independently and manages them at its repository. In this scheme, certificates are stored and distributed by the nodes and each node maintains a local certificate repository that contains a limited number of certificates selected by the node according to an appropriate algorithm. Key authentication is performed via chains of certificates. However, this scheme suffers from the delay and the large amount of traffic required to collect certificates. Kitada et al. in [89], and [90] considered the problem of certificate chain discovery by introducing the ad hoc simultaneous nodes search protocol (ASNS) to find a certificate chain. In the proposed scheme by Kitada et al., each node holds in its local repository only certificates issued to it in order to reduce the memory size and collects certificates by broadcasting search packets to chained nodes. The scheme suffers from high communication cost because of broadcasting packets with certificates. Li et al. in [145] proposed a public key management scheme performed by generating a public/private key pair by the node itself, issuing certificates to neighbouring nodes, holding these certificates in its certificate repository. This scheme considers only the updated certificate repository to reduce the number of certificates stored in its certificate repository. Ren et al. in [104] proposed a distributed trust model based on introducing a secret dealer to accomplish trust initialisation in the system bootstrapping phase to overcome the problem of delayed trust establishment in [4]. In this scheme a secret dealer provides each node with a secret short list includes a number of entries, and the number of entries is determined according to the group size n and may vary slightly from node

to node. Each entry contains a binding of node identifier and its corresponding public key: (ID, PK). After receiving the short list, the following conditions should be met:

- Every node in the network receives a secret short list (SL), which contains n semi-randomly selected (ID, PK) pairs;
- The (ID, PK) pairs are distributed symmetrically. If node i gets the (ID, PK) pair of node j, then the (ID, PK) pair of node i is also included in the secret short list SL of node j's, that is, $(ID_i, PK_i) \subset SL_j$, if $(ID_j, PK_j) \subset SL_i$, where SL_i is the short list obtained by node i.

After that, each node starts to issue certificates for the received bindings and stores them locally. The existence of the secret dealer makes the scheme prone to the centralised administration problems. For example, it has a single point of attack because if the secret dealer is compromised during the bootstrapping phase, the security of the whole system will be at risk. The performance of the scheme in the mobility environment of MANETs is low as will be shown in section 5.5 because the certificate graph, which is used to model this web of trust relationship, may not be strongly connected.

5.3 System Description and Trust Model

In this section we present an overview of the trust model and the system description of our proposed scheme.

5.3.1 Trust Model

5.3.1.1 Certificate Chain

The trust model of our proposed scheme is based on the existence of public key certificates as bindings of the public keys and the corresponding user identities IDs. The

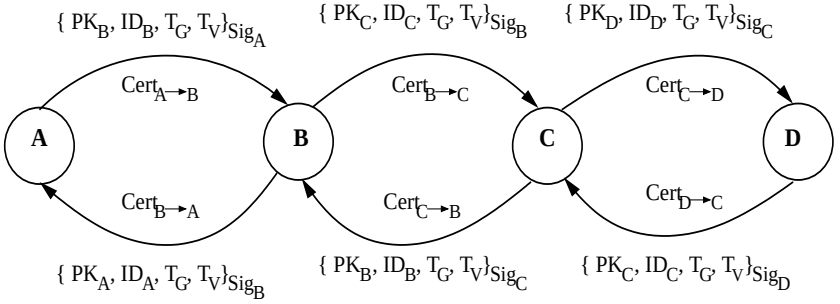

FIGURE 5.1: Certificate chain

certificate should also contain the node's identity/network address, sequence number, trust value, certificate generation and validity dates.

Taking into consideration the example in Figure 5.1, we denote $Cert_{A\rightarrow B}$ as the certificate signed by node A's private key SK_A to represent its assurance in the binding of node B and its public key PK_B. For simplicity we denote Sig_i as the digital signature of node i. A certificate graph $G(V,E)$, represents the public keys and certificates of the system in the initialisation phase, where V and E stand for the set of vertices and the set of edges, respectively. The vertices of the certificate graph represent public keys and the edges represent certificates. A directed edge from node A to node B will exist if there is a certificate signed with the private key of node A that binds node B's identity ID_B and its public key PK_B. In order for a node A to authenticate the public-key of another node D as shown in Figure 5.1, it has to acquire a chain of valid certificates from node A to node D. The first certificate in the chain is a certificate issued by node A, so that it will be verified by node A by using its public key PK_A. Each remaining certificate in the chain will be verified using the public key of the previous certificate in the chain. The last certificate in the chain holds the public key of the target node D. The certificate chain from node A to node D in this example is $\{Cert_{A\rightarrow B}, Cert_{B\rightarrow C}, Cert_{C\rightarrow D}\}$, and the certificate chain from node D to node A is $\{Cert_{D\rightarrow C}, Cert_{C\rightarrow B}, Cert_{B\rightarrow A}\}$.

It is assumed that there exist sparse trust relationships among the nodes so that any node that wishes to join the network can establish independent trust relationship with some of the existing member nodes in the network. For example, a node that wishes to join the network contacts one of the existing network members through a secure side channel and provides its trust evidence. If the existing network member believes that

TABLE 5.1: Trust table

Node ID	Certificate	Trust Value

the requesting node is trustworthy according to its trust evidence, they can sign and exchange certificates. The process is repeated until the joining node gets a sufficient number of certificates.

5.3.1.2 Trust Evaluation

We define a trust value as an authentication metric. This trust value represents the assurance with which a requesting node can obtain the correct public key of a target node. However, the same assurance in the reverse direction need not exist at the same time. In other words, the trust relationship is unidirectional. Each node in the network should have a trust table as shown in Table 5.1 to store the public-key certificates and the corresponding trust values of the nodes it trusts in the network. There are many trust metrics have been proposed to evaluate the trust values, some assume discrete trust values as in PGP [79]. Others assume continuous values for trust [82]. In our trust model, we define the trust value as a continuous value between 0 and 1. A trust value $T_{i,j}$ represents node i's belief that node j is trustworthy. The higher the value of $T_{i,j}$, the more node i trusts node j, and vice versa. The value of $T_{i,j}$ is adjusted according to the strength of the trust evidence presented by node j to node i. Any node in the network can calculate the value of trust $T_{i,j}$ in another node's public key if there exist a certificate chain between the two nodes using the following equation:

$$T_{i,j} = \prod_{k=1}^{h} T_k \tag{5.1}$$

where T_k is the value of trust between two directly trusted nodes along the certificate chain from node i to node j, and h is the number of hops between node i and node j as shown in Figure 5.2.

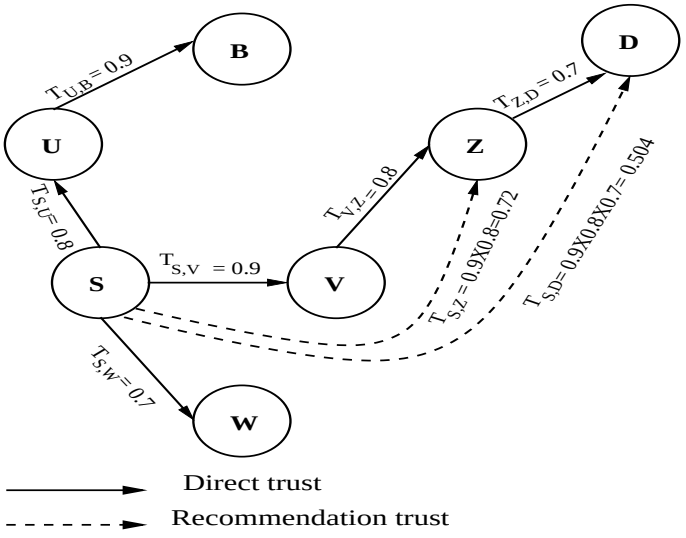

FIGURE 5.2: The trust model

It is obvious from Figure 5.2 that the value that a node trusts in another node's public key fades along the path of recommendation. So, if there is more than one certificate chain between a pair of nodes, the source node should choose the path with higher trust value, which is more probably the path with minimum number of hops.

5.3.1.3 Independent Trusts

The previous work on independent trust evaluation presented by Maurer [146] is extended in our proposed scheme. Dependence between certificate chains leads to inaccurate trust values. For example, if a malicious node is a part of dependent certificate chains, the trust computation result will be completely incorrect. In order for a source node to authenticate the destination node's public key in the adversary environment of MANET, our proposed key management scheme insists on obtaining the destination node's public key through at least two independent certificate chains. In order to introduce the independent certificate chains in computing the trust value of the target node, equation 5.1 will be modified as follows:

$$T_{i,j} = \bigcup_{l=1}^{n} \left(\prod_{k=1}^{h} T_k \right) \tag{5.2}$$

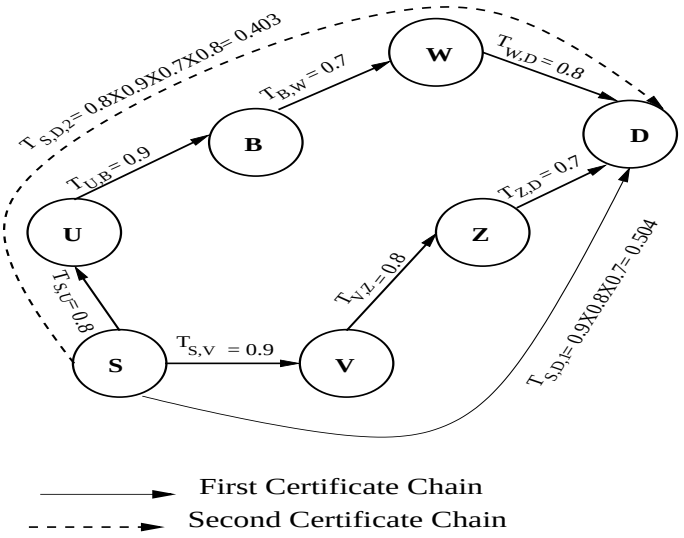

FIGURE 5.3: Independent certificate chains

Where n is the number of independent certificate chains between node i and node j.

From the inclusion-exclusion principle, the probability of the union of two events can be computed by taking the sum of their probabilities, and subtracting the probabilities resulting from intersecting as shown in equation 5.3.

$$p(A \cup B) = p(A) + p(B) - p(A) \times p(B) \tag{5.3}$$

In our proposed scheme, A and B are the trust values computed from the first and the second independent certificate chains respectively.

The presence of independent certificate chains in the authentication process results in an increase in the computed trust value of the target node as shown in the example in Figure 5.3.

The values that the source node S trusts in the destination node D from the first and the second certificate chains are $T_{S,D,1} = 0.504$, and $T_{S,D,2} = 0.403$ respectively. According to equations 5.2, and 5.3, the overall trust value will be computed as follows:

$T_{S,D} = 0.504 + 0.403 - 0.504 \times 0.403 = 0.703$

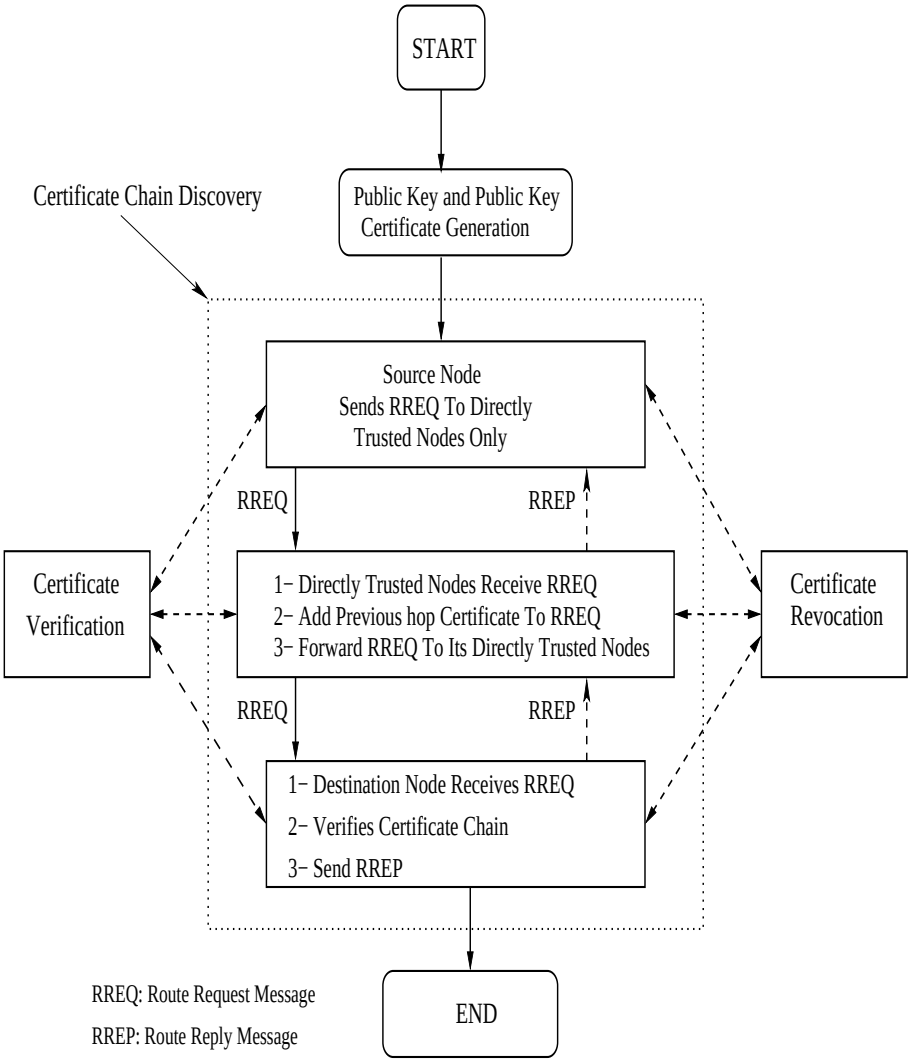

FIGURE 5.4: Scheme description

It is obvious from the above example that the presence of independent certificate chains has increased the trust value more than the trust values computed from a single certificate chain, which is an advantage of our proposed key management scheme.

5.3.2 System Description

The proposed robust self-organised public key management scheme involves four processes as shown in Figure 5.4: public key and public key certificate generation, certificate chain discovery, certificate verification, and certificate revocation. We describe our proposed scheme in detail according to these four processes.

1. Public Key and Public Key Certificate Generation

   In our proposed scheme, each node generates its public key and the corresponding private key locally before joining the network by the node itself. Public key certificates are issued by the nodes based on node information about other nodes in the network. If a node u believes that a public key PK_i belongs to a certain user i, it has to issue a certificate to node i signed by its private key PK_u representing its assurance of the binding of the user's identity ID_i and its corresponding public key PK_i. Issued certificates to or from the node should be stored in its trust table with a validity time.

2. Certificate Chain Discovery

   In this process, the certificate chain discovery is performed by exploiting the routing infrastructure. We assume that a certain number of direct trust relations have been established between each node and its neighbours during the network initialisation. Direct trust relations are usually obtained off-line by visual identification, audio exchange through side channels, and physical contact, but can also be obtained on-line. The number of directly trusted nodes per node is assumed to be uniformly distributed to enable every node in the network to carry out its role in the certificate chain discovery process with equal probability. We assume that a random graph $G(n,p)$ is a graph of n vertices and the probability that a connection exists between any two vertices is p. In [147], Erdos and Renyi showed that there exists, for monotone properties, a value of p such that the property transfers from ''nonexistent'' to ''certainly true'' in a random graph that has a very large number of vertices. However, Frank and Martel have shown by simulation in [148] that these properties are also valid in graphs of moderate size (between 30 to 480 vertices).

   Erdos and Renyi showed that if $p = ln(n)/n + c/n$, where c is a real constant then

$$\lim_{n\to\infty} Pr[G(n,p)\, connected] = e^{-e^{-c}} \tag{5.4}$$

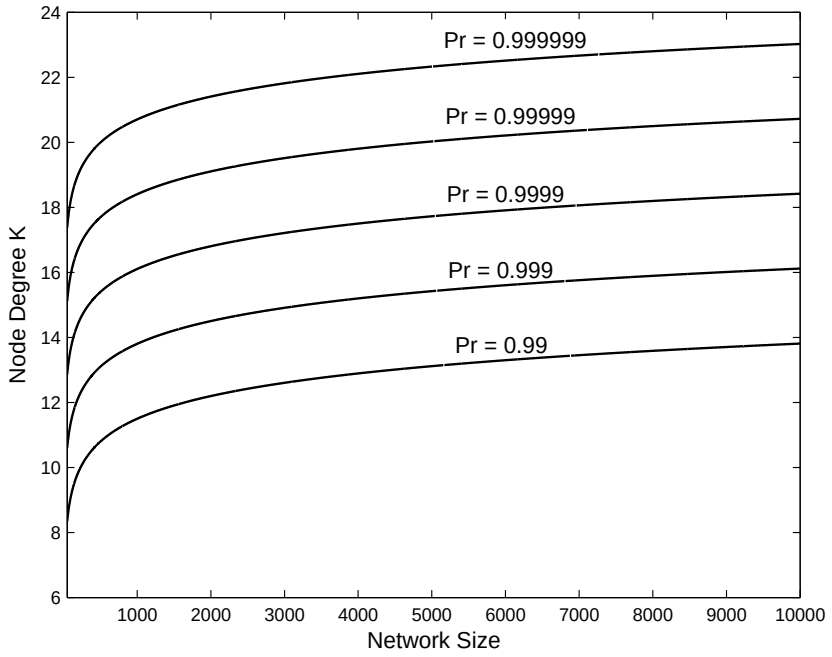

FIGURE 5.5: Degree of node d vs. network size for various $Pr[G(n,p)\,connected]$

Therefore, given the network size n we can find p and the average degree of a node (number of trusted neighbouring nodes) $d = p\,.\,(n-1)$ for which the resulting graph is connected with the desired probability $Pr[G(n,p)\,connected]$. Figure 5.5 illustrates the plot of the expected degree of a node d as a function of the network size n for various values of $Pr[G(n,p)\,connected]$. For example if the desired probability to get a connected graph for 50 nodes network is 0.99, then the average number of neighbouring nodes d should be 6, and for 100 nodes network d should be equal to 10 for the same probability. The proposed robust self-organised public key management scheme works in support of an ad hoc on demand routing protocol (such as AODV [68]) after doing the necessary modifications. The RREQ and the RREP messages format of the AODV routing protocol have been modified as shown in Figures 5.6, and 5.7 respectively in order to implement the proposed key management scheme. The certificate chain discovery process can be explained as follows:

- The source node sends a route request packet to nodes that the source node directly trusts.

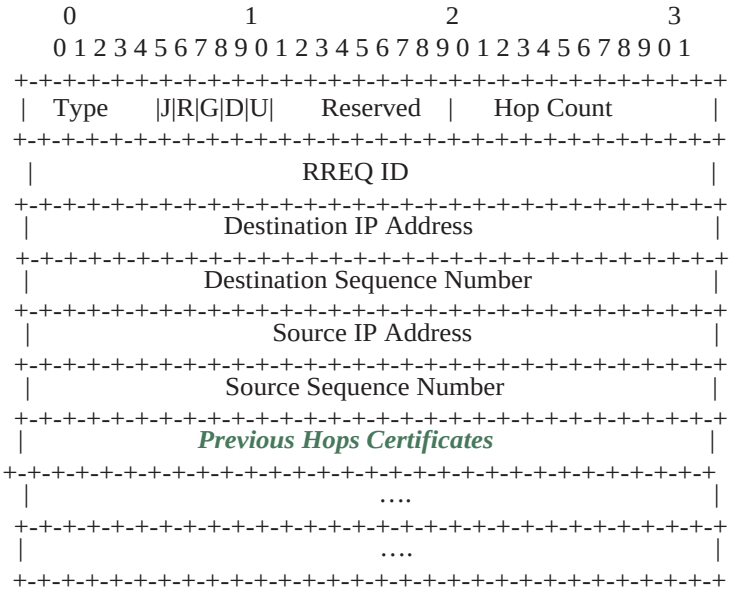

FIGURE 5.6: RREQ message format

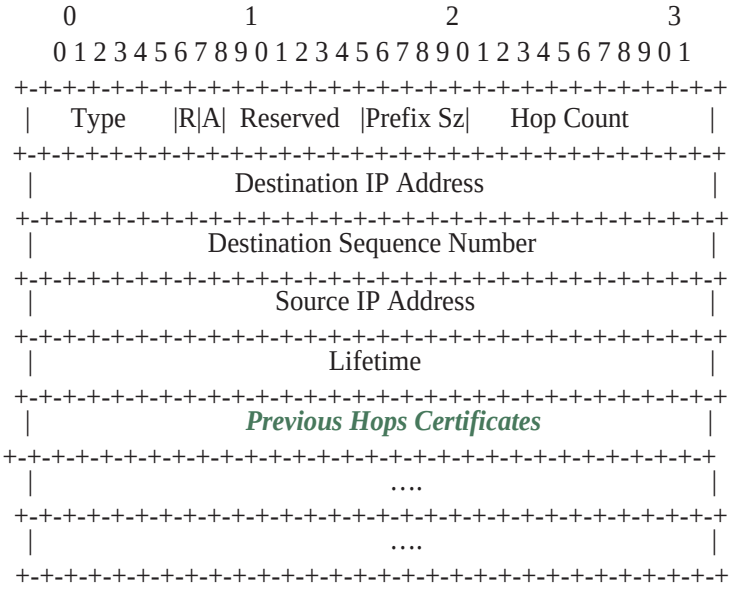

FIGURE 5.7: RREP message format

- When a directly trusted node by the source node receives the route request packet from the source node, it searches for the certificate of the source node signed by this node in its trust table and adds it to the route request packet before forwarding the route request packet to nodes it directly trusts.
- Each node has a route request table and a trust reply table as shown in Tables 5.2 and 5.3, respectively.

- When an intermediate node receives the route request packet from a node it trusts, it sends a route reply to the source node if it has a fresh route to the destination node.
- If the intermediate node does not have a fresh route to the destination node, it searches for the certificate of the sender node in its trust table, and adds it to the route request packet.
- Before forwarding the route request packet to nodes it directly trusts, the intermediate node inserts the ID of the source node, the ID of the destination node, the RREQ ID, and the ID of the sender node in its trust reply table.
- When the destination node receives a route request packet from a node it directly trusts, it verifies the certificates included in the route request packet and inserts the ID of the source node, the RREQ ID, and the ID of the sender node in its trust reply table.
- The destination node waits until it receives at least another route request, repeats the previous step and makes sure that the certificate chain included in the second route request is completely independent from the certificate chain included in the first received one (i.e., there is no intersecting intermediate node in both routes).
- The destination node sends a route reply to the senders of this route request according to its trust reply table.
- When an intermediate node receives the route reply, it adds the certificate of the node from which it received the route reply to the route reply and forwards the route reply to the sender nodes of the corresponding route request according to its trust reply table.
- The process continues until the route reply reaches the source node.
- When the source node receives the route reply, it verifies the certificate chain; computes the trust value of the chain; inserts the RREQ ID, certificate chain entities ID's, and the computed trust value of the chain in the route request table as shown in Table 5.2.

TABLE 5.2: Route request table

Chain #	RREQ ID	Destination ID	1st Hop Node ID	2nd Hop Node ID	...	Last Hop Node ID	Trust Value
1st Chain							
2nd Chain							
...							

TABLE 5.3: Trust reply table

Source ID	Destination ID	RREQ ID	Sender 1	Sender 2	...	Sender n
			Node ID	Node ID	Node ID	Node ID

- The source node waits until it receives a route reply from at least another independent certificate chain, and according to the chain trust value the source node chooses a chain to start data transmission to the destination through it. The source node examines the dependency of the received certificate chain by searching for any intersecting node in the chains stored in its route request table.

Successful communications result in increasing trust between certificate chain entities and signing more certificates for each others. Node mobility through the network results in new trust relations and more certificates are added to the node's trust table.

At the end of the route reply process, the source node S receives two independent certificate chains as shown in the example in Figures 5.8, and 5.9 for the route request and the route reply processes respectively. The first certificate chain contains certificates of nodes, B, E, and Q in addition to the destination node's certificate. The second certificate chain contains certificates of nodes, C, F, and K in addition to the destination node's certificate. According to the computed trust values for each certificate chain, the source node S chooses a chain to be

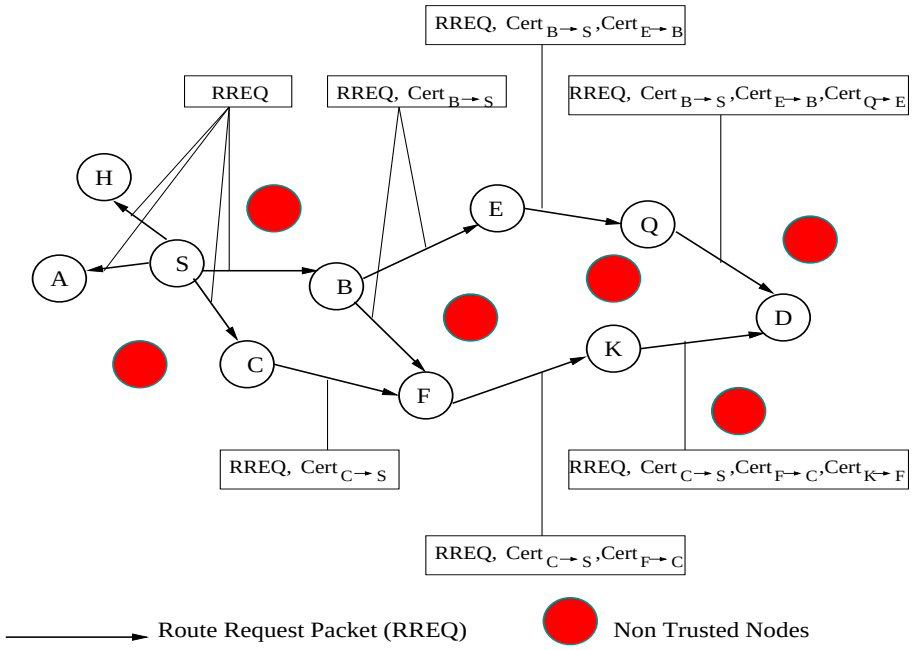

FIGURE 5.8: Route request process

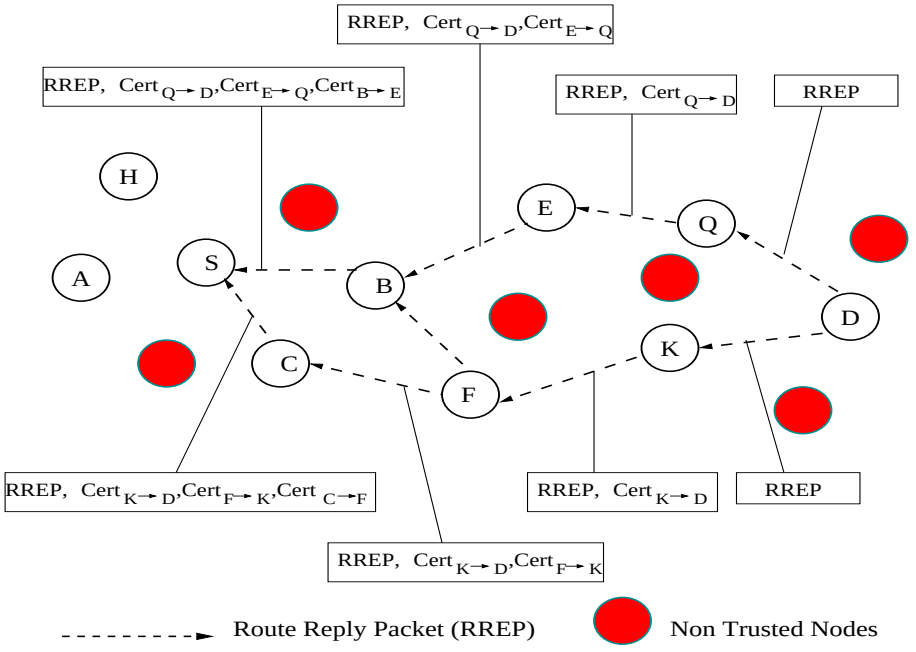

FIGURE 5.9: Route reply process

used in transmitting data packets to the destination node *D*. After receiving the second route reply the route request table of the source node *S* will be as shown in Table 5.4.

TABLE 5.4: Route request table of the source node *S*

Chain #	RREQ ID	Destination ID	1st Hop Node ID	2nd Hop Node ID	Last Hop Node ID	Trust Value
1st Chain	1	D	B	E	Q	Value 1
2nd Chain	1	D	C	F	K	Value 2

3. Public Key Certificate Verification

   The verification of the public key certificate is performed by checking the validity time T_v in the certificates forming the certificate chain. After verifying the certificates of the certificate chain the authenticity of the public keys of both the source and the destination nodes is performed by the certificate chain from the source node to the destination node and vice versa.

4. Public Key Certificate Revocation

   Each node can revoke a certificate it issued if it believes that the binding between the public key and the node's identity is no longer valid or the trust value of the target node is below the trust threshold. Each node can revoke its own certificate if it believes that its private key is compromised. A node can revoke its own certificate by broadcasting a certificate revocation message signed by its private key to nodes it directly trusts includes the revoked certificate and the new one. When a node receives the certificate revocation message, it verifies the revocation message and replaces the revoked certificate by the new one. A node can revoke a certificate it issued by broadcasting a certificate revocation message signed by its private key to nodes it directly trusts. When a node receives a certificate revocation message, it verifies the signature of the revocation message, searches in its trust table for the trust values of the sending node and that of the node its certificate is claimed to be revoked. According to the search results, it will do the following:

- Step 1: if the search result yields that the trust value of the sending node is greater than the trust threshold (i.e., the sending node is trustworthy), it multiplies the trust value of the sending node by -1.
- Step 2: adds the calculated value from the previous step to the trust value of the node the certificate of which is claimed to be revoked (let the trust value of this node equal to 0.7).
- Step 3: if the sum from the previous step is below the trust threshold, it deletes the revoked certificate from its trust table. Otherwise if the sum is still greater than the trust threshold, the accused certificate is still to be used.

Otherwise, if the search result yields that the trust value of the sending node is below the trust threshold (i.e., the sending node is not trustworthy), it ignores the revocation message.

For example, let the trust value of the sending node equal to 0.85 and the trust threshold is 0.1 then the output from step 1 is -0.85. The output from step 2 is: $-0.85+0.7=-0.15$. The output from step 3 is $-0.15<0.1$ and the certificate should be revoked.

5.3.3 Communication Cost

The communication cost of our proposed scheme consists of two phases; the route request phase and the route reply phase. In the route request phase, the source node sends the route request packet to nodes it directly trusts without inserting its own certificate in the route request packet as this certificate is stored in the trust table of nodes it directly trusts which saves bandwidth in the first hop. The communication cost of the certificate chain discovery during the route request phase can be calculated as follows:

$$C_{req}(h)=\sum_{i=2}^{h} d^i \times (i-1) \times cert\_size \tag{5.5}$$

Where h is the average number of hops between the source and the destination, d is the average number of trusted nodes per node, and $cert\_size$ is the size of the certificate in bits.

In the route reply phase, the destination node does not insert its own certificate in the first hop because this certificate is stored in the trust table of nodes it directly trusts which saves bandwidth in the first hop of the route reply. The destination node sends a route reply message to all sender nodes it has received the route request from according to its trust reply table as shown in Table 5.3.

The upper limit of the communication cost in this phase can be calculated as follows:

$$C_{rep}(h) = \sum_{i=2}^{h} (i-1) \times d \times cert\_size \tag{5.6}$$

The total communication cost of the certificate chain discovery process will be calculated as follows:

$$C_{total}(h) = C_{req}(h) + C_{rep}(h) \tag{5.7}$$

$$C_{total}(h) = \sum_{i=2}^{h} (d + d^i) \times (i-1) \times cert\_size \tag{5.8}$$

5.4 Simulation Environment and Performance Metrics

5.4.1 Simulation Environment

Simulations were performed using Network Simulator (NS-2) [56]. On a desktop PC with an Intel Core 2 Duo 2.6 GHz processor and 1 GB memory. The OpenSSL library (version 0.9.8h) [141] is used for generating certificates, and digital signatures [22].

In order to complete a certificate chain with probability equal to 0.999, the average number of trusted nodes per node need not to be less than 6 as discussed in subsection 5.3.2. The MAC layer protocol IEEE 802.11 is used in all simulations. The source-destination pairs are spread randomly over the network. The NS-2 constant bit-rate (CBR) traffic generator is used to set up the connection patterns with different random seeds. Each node has one CBR traffic connection with a single unique destination and can generate at most 10,000 packets. Sources initiation time is uniformly distributed over the first 90 seconds of the simulation time. Every simulation run is 1000 seconds long. The mobgenss [128] mobility scenario generator was used to produce random waypoint mobility patterns [136]. The pause time is set to zero. The Ad Hoc On-demand Distance Vector (AODV) routing protocol [68] was chosen for the simulations. The simulation results are the average of 10 runs. The rest of the simulation parameters are summarised in Table 5.5. These scenarios are the worst case scenarios because all nodes in the network are transmitting and receiving packets starting from the time of joining the network till the end of the simulation. It is generally accepted that data encrypted using a key size of 2048-bits should be safe. Therefore, the public key would require 256 bytes, a signature consists of 128 bytes value, node's identity is of 16 bytes, network address is of 16 byte, sequence number is of 16 bytes, trust value is of 16 bytes, and the certificate expiration time can be encoded in 2 bytes. In the simulation, the certificate is constructed as follows: $< ID; NetworkAdd; PK;$ $Sequencenumeber; Expirationtime; trustvalue; signature; 62bytesreserved >$ which is a total of 512 bytes in length.

5.4.2 Performance Metrics

In addition to the *packet delivery ratio (PDR)* and the *average end-to-end delay* described in Subsection 4.3.2, we have selected the *number of certificates delivered through the network*, *certificate chain completion ratio*, *average hop length (AHL)*, and the *routing overhead* as metrics during the simulation in order to evaluate the performance of the proposed scheme.

TABLE 5.5: Simulation parameters

Parameter	Value
No. of Nodes	50
Area (m2)	1000x1000
Transmission range	250m
Mobility Model	Random waypoint
Propagation Model	Two-Ray Ground
Mean speeds (m/s)	0, 5, 20
Data Rate	11 Mbps
Load	4 packet/s
CBR connections	50
Data packet size	512 bytes

- **Number of certificates delivered through the network:**

  It is the total number of certificates delivered through the network during the certificate chain discovery processes.

- **Certificate chain completion ratio:**

  It is the ratio of the average number of independent certificate chains received by a node to the number of transmitted route requests.

- **Average hop length (AHL):**

  AHL is the average distance between the source node and the destination node.

- **Routing Overhead (RO):**

  Routing overhead is the ratio of the total number of routing packets to data packets transmitted during the simulation. It is an important metric, as it measures the efficiency of a routing protocol; the degree to which it will function in congested or low-bandwidth environments.

5.5 Performance Evaluation

In this section, the simulation results will be presented. The performance evaluation of the proposed scheme will be performed in two steps, in the first step, a comparison of the performance of the proposed scheme, the performance of Ren et al. web of trust scheme [104], and the CBR reference which is the simulation of the AODV [68] without any modifications are presented. The performances of the three mentioned schemes are measured for different numbers of trusted nodes per node. In the second step the dynamical characteristic of the proposed scheme will be presented by measuring the performance of the proposed scheme in a realistic environment for different numbers of nodes joining and leaving the network.

5.5.1 Performance of the Proposed Scheme

In this Subsection, we present the PDR, end-to-end delay, number of certificate delivered, AHL, and certificate chain completion ratio of our proposed scheme.

5.5.1.1 Packet Delivery Ratio (PDR)

Figure 5.10 shows that the packet delivery ratio increases with increasing the number of trusted nodes per node d. In our proposed scheme, discovering a certificate chain is performed through trusted nodes only. However, increasing the number of trusted neighbours d increases the probability for a node to discover a certificate chain which results in an increase in the packet delivery ratio. When d is less than 3, it is very difficult for a node to discover two independent certificate chains through less than 3 trusted neighbours. This authentication failure results in a very low PDR as shown in Figure 5.10. Increasing the number of trusted nodes per node d to 6 at the network initialisation phase (as expected in our setting) makes the packet delivery ratio approaches that of the AODV reference. It shows also that Ren et al. scheme is not robust in the mobility environment as increasing the node's speed from 0.1 to 5 m/s and 20 m/s degrades the packet delivery ratio too much.

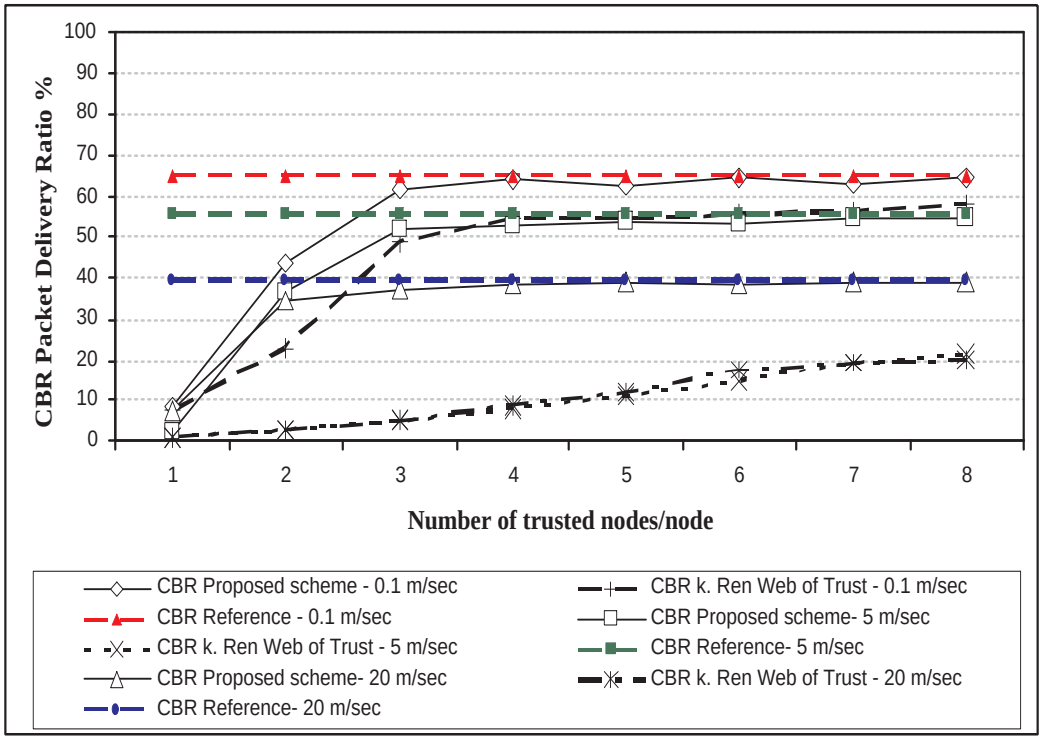

FIGURE 5.10: Packet delivery ratio PDR % vs number of trusted nodes/node

5.5.1.2 End-to-End Delay and Average Hop Length (AHL)

Figure 5.11 shows that the end-to-end delay of our proposed scheme for d equal to 6 approaches the end-to-end delay of AODV reference, and the end-to-end delay of Ren et al. scheme is far away from the end-to-end delay of AODV reference because of the successive failures to find certificate chains. Figure 5.11 shows also that the end-to-end delay in our proposed scheme increases with increasing the number of trusted nodes per node d. The end-to-end delay is calculated for all data and control packets, i.e., those dropped before reaching the destination and those succeed to reach their destination. In order to understand why the end-to-end delay increases with increasing d we should put Figure 5.12 into context which shows the AHL as a function of the number of trusted nodes per node d. It is obvious that the AHL increases with increasing d as the probability that a mobile node can discover a route increases with increasing the number of its trusted neighbours d through whom a route can be discovered. For lower values of d, a higher percentage of the data and control packets are dropped before reaching

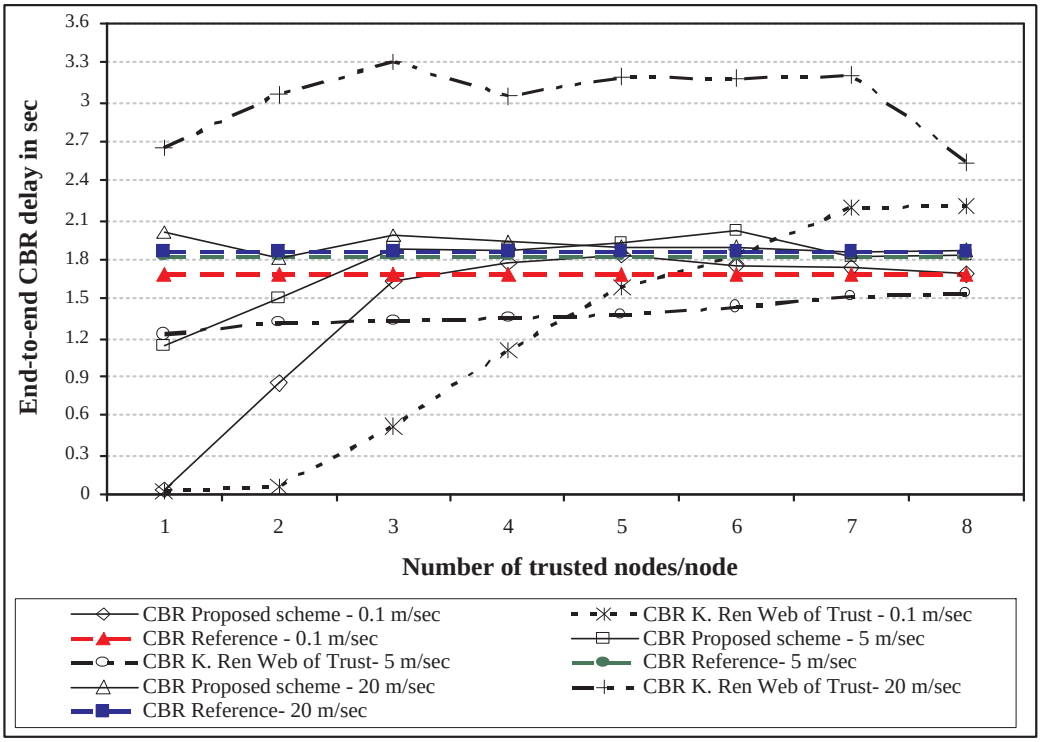

FIGURE 5.11: End-to-end delay in sec vs number of trusted nodes/node

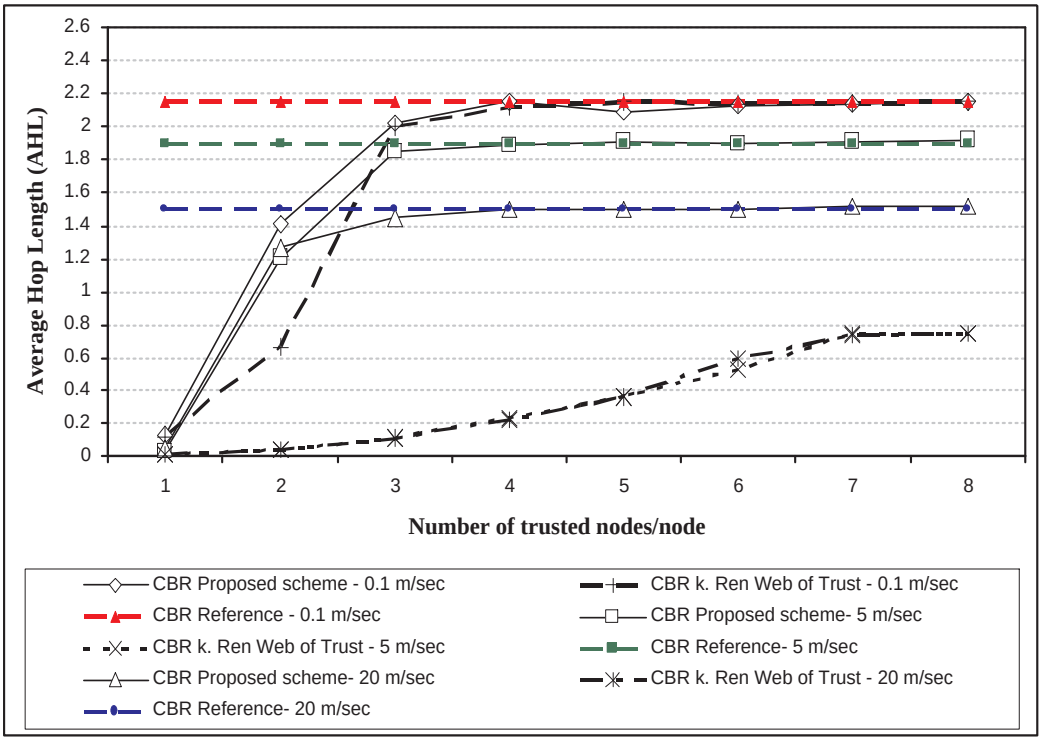

FIGURE 5.12: Average hop length

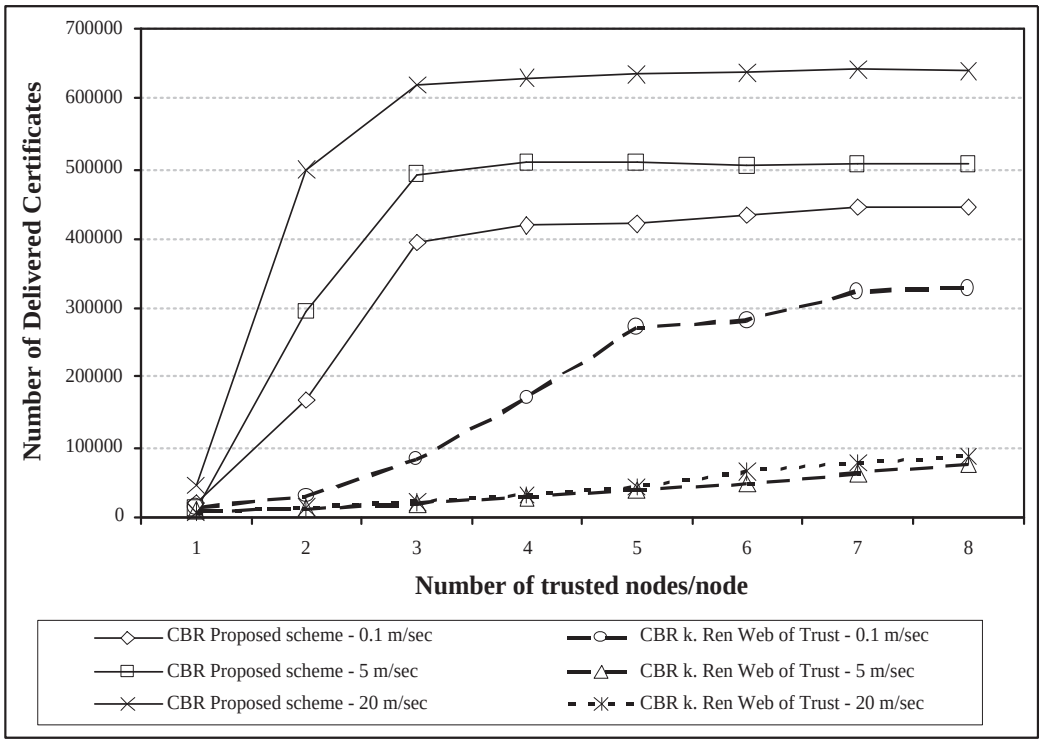

FIGURE 5.13: Number of certificates delivered through the network

the destination due to routing failures as shown in Figure 5.10, i.e., the dropped data and control packets traverse a lower number of hops which results in lower end-to-end delay. By increasing d, the probability of discovering a route increases and hence the percentage of data and control packets that traverse higher number of hops due to their success to reach the destinations becomes higher than that in case of lower d which results in higher end-to-end-delay.

5.5.1.3 Number of Certificates Delivered

Figure 5.13 shows that the number of delivered certificates through the network increases as d increases as expected. Increasing d increases the probability for a node to discover a certificate chain which results in an increase of the number of certificates delivered through the network. It also shows that the number of delivered certificates of Ren et al. scheme is lower than that of our proposed scheme due to the failures to discover certificate chains in Ren et al. scheme [104].

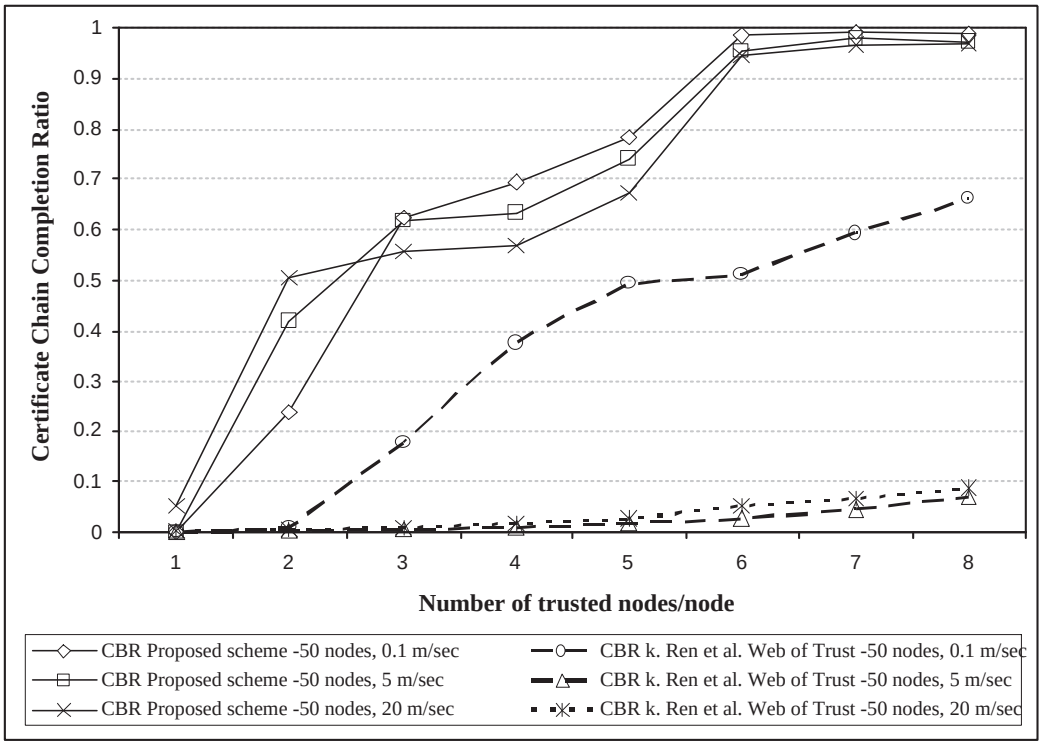

FIGURE 5.14: Certificate chain completion ratio

5.5.1.4 Certificate Chain Completion Ratio

Figure 5.14 shows the certificate chain completion ratio as a function of the number of trusted nodes per node. Increasing the value of d increases the certificate chain completion probability. For d less than 3 the certificate chain completion ratio for 20 m/sec node mobility is higher than that for 5 m/sec node mobility which in turn is higher than that for 0.1 m/sec node mobility. The reason is due to the fact that the length of the certificate chain for higher node mobility comprises of lower number of nodes than that for lower node mobility as shown in Figure 5.12 which makes completing a certificate chain easier. When d is set between 3 and 5, the certificate chain completion ratio for lower node mobility exceeds that for higher node mobility. For d equal to 6 (as expected in our setting), the certificate chain completion ratio approaches the target which is 0.999. For d equal to 8, nodes can certainly complete a certificate chain. The success ratio for completing a certificate chain in Ren et al. scheme for 0.1 m/s node mobility does not exceed 0.68 even at d equal to 8, and becomes worse with increasing node mobility.

It is obvious from Figures 5.10, 5.11, 5.12, 5.13, and 5.14 that when d is less than three it is very difficult for a node to get two independent certificate chains as a reply to its request which causes a degradation on the network performance.

5.5.2 The Dynamic Characteristics of the Proposed Scheme

In this subsection, the performance of the proposed scheme is evaluated with regard to different numbers of nodes joining and leaving the network which is known as the dynamical characteristics of the network. Nodes joining and leaving the network have been simulated in NS-2 [56] by changing the initial energy of the Mobile node in its energy model.

The processes of joining and leaving the network occur simultaneously while keeping the network size unchanged. The performance of the network is measured after each pair of nodes leaves or joins the network. In order to simulate a realistic network, the simulation environment used in Section 5.4.1 will be the same except that the number of trusted nodes per node is set to 6 in all simulations presented in this subsection and the number of connections is set to 25 rather than 50. The number of connections is the number of source and hence destination nodes in the network. We chose the PDR, end-to-end delay, and the number of certificates delivered as metrics to measure the dynamic characteristics of the proposed scheme.

5.5.2.1 Packet Delivery Ratio (PDR)

Figure 5.15 shows that the packet delivery ratio of our proposed scheme has changed little even after the number of nodes leaving and joining the network approaches 10, i.e., 20% of the network size. The packet delivery ratio in this scenario is higher than the packet delivery ratio in Figure 5.10 due to decreasing the number of connections to 25 rather than 50. Decreasing the number of connections result in an increase in the packet delivery ratio as a result of decreasing the radio interference and collisions between nodes caused by the hidden/exposed terminals

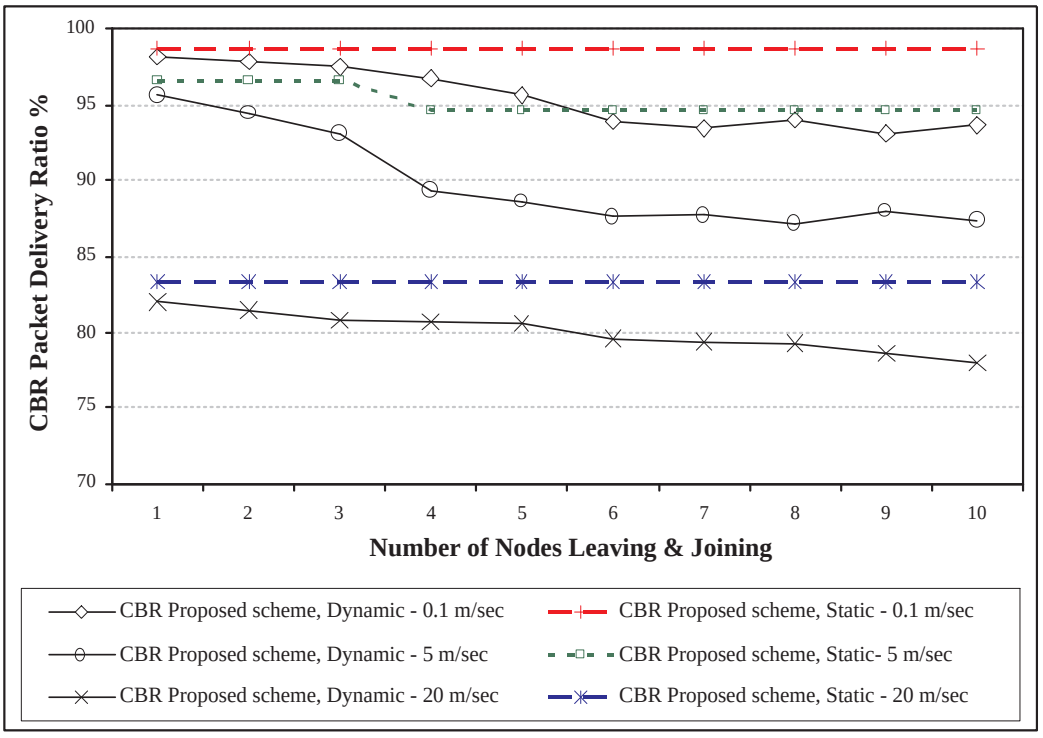

FIGURE 5.15: Packet delivery ratio PDR % vs number of nodes leaving and joining

problems.

5.5.2.2 End-to-End Delay

Figure 5.16 shows that the change in the end-to-end delay of our proposed scheme with increasing the number of nodes leaving and joining the network was low even for 20% of nodes leaving and joining the network. For 5 m/sec node mobility, the end-to-end delay increases with increasing the number of nodes leaving and joining the network due to increasing the routing failures. For 20 m/sec node mobility, the end-to-end delay decreases with increasing the number of nodes leaving and joining the network because at high mobility (e.g., 20 m/sec), the AHL is low as shown in Figure 5.12 and the impact of increasing the number of nodes leaving and joining the network on the routing process is minimal.

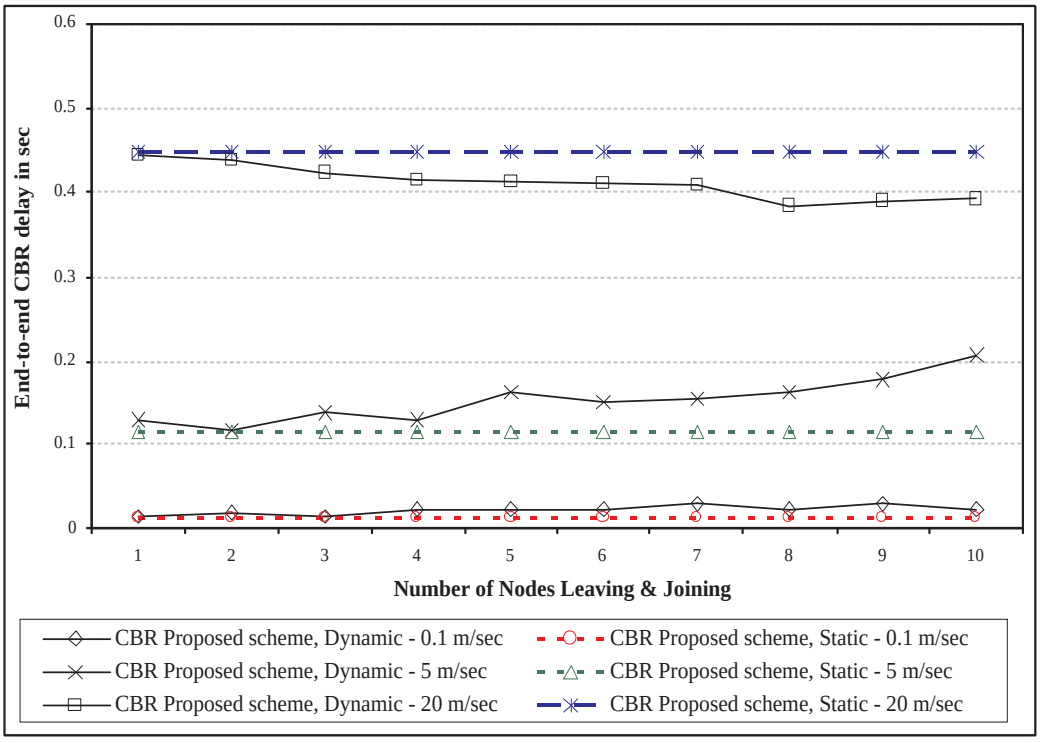

FIGURE 5.16: End-to-end delay in sec vs number of nodes leaving and joining

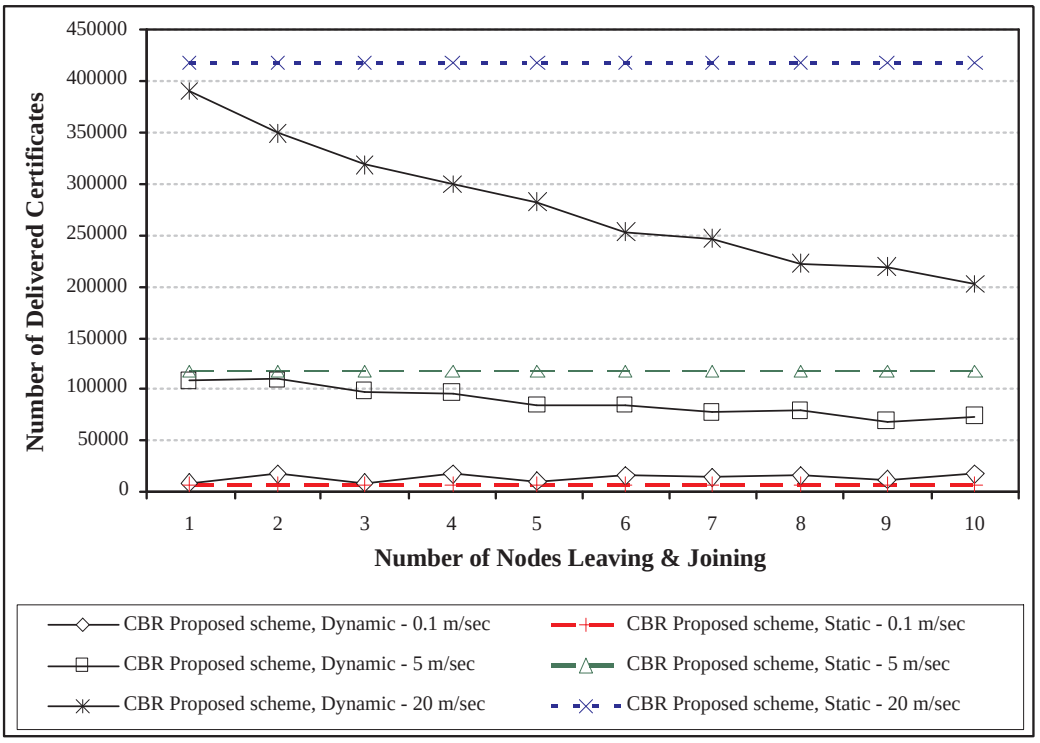

FIGURE 5.17: Number of certificates delivered through the network vs number of nodes leaving and joining

5.5.2.3 Number of Certificates Delivered

Figure 5.17 shows that the number of certificates delivered through the network decreases with increasing the number of nodes leaving and joining the network. The reason is that increasing the number of nodes leaving and joining the network increases the routing failures which results in increasing the number of dropped certificates.

5.5.3 Security Analysis

In this subsection, we study the security of the proposed key management scheme. In the network initialisation phase, an attacker can compromise a mobile node and try to deceive other nodes by making them believe in false certificates. First, the malicious node can insert false certificates in the certificate chain during the certificate chain discovery process. This can be done by binding a public key PK_A to a user B instead of to a user A. It can also be done by binding a false key $PK_A^{'}$ to a user A. Second, the malicious node can delete certificates from the certificate chain to thwart the certificate chain discovery process. Our proposed key management scheme prevents these attacks by allowing mobile nodes to detect false certificates and to perform the public key authentication process with high probability. If a malicious node inserted false certificates in the certificate chain, or deleted correct certificates from the certificate chain during the certificate chain discovery process, the authentication process will not be affected because the source node waits until it receives two independent certificate chains without any intersecting node to perform the authentication process. If a certificate received by a mobile node that binds the node A's ID to a public key PK_A and later on received a certificate that binds the node A's ID to a public key $PK_A^{'}$, it can easily detect the duplication. The duplication can be resolved by authenticating the (ID, PK_A) pair through another two independent certificate chains. The existence of the trust value along with the certificate enhances the security of our proposed key management scheme by enabling mobile nodes to choose the certificate chain that has the highest trust value. The certificate chain completion ratio and the routing overhead were chosen as metrics to

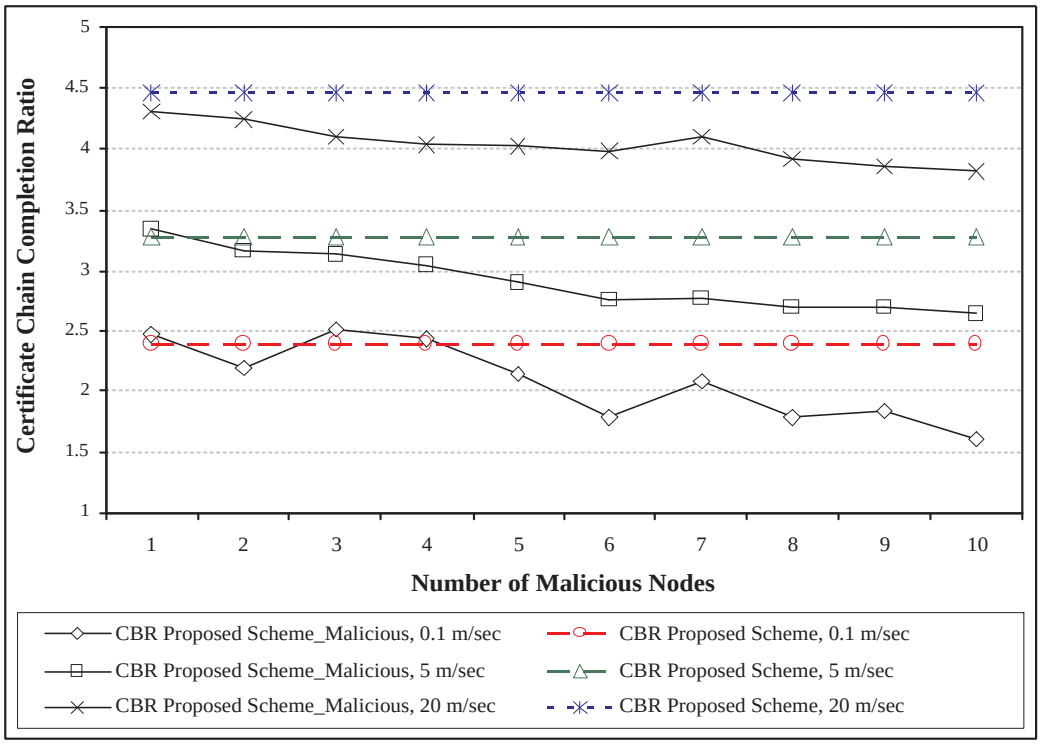

FIGURE 5.18: Certificate chain completion ratio

justify the robustness of the proposed key management scheme against malicious nodes attacks.

5.5.3.1 Certificate Chain Completion Ratio

In Figure 5.18, the certificate chain completion ratio is calculated for different number of malicious nodes in the network. It shows that the certificate chain completion ratio increases with increasing node's mobility due to decreasing the average hop length with increasing the node's mobility as shown in Figure 5.12, i.e., the number of certificates in the certificate chain is lower during higher mobility than that in lower mobility which makes the certificate chain discovery process easier. Figure 5.18 shows also that the certificate chain completion ratio decreases with increasing the number of malicious nodes in the network as a result of inserting false certificates or deleting correct ones. It is obvious from Figure 5.18 that mobile nodes can perform the certificate chain discovery process and subsequently the public key authentication with high probability even

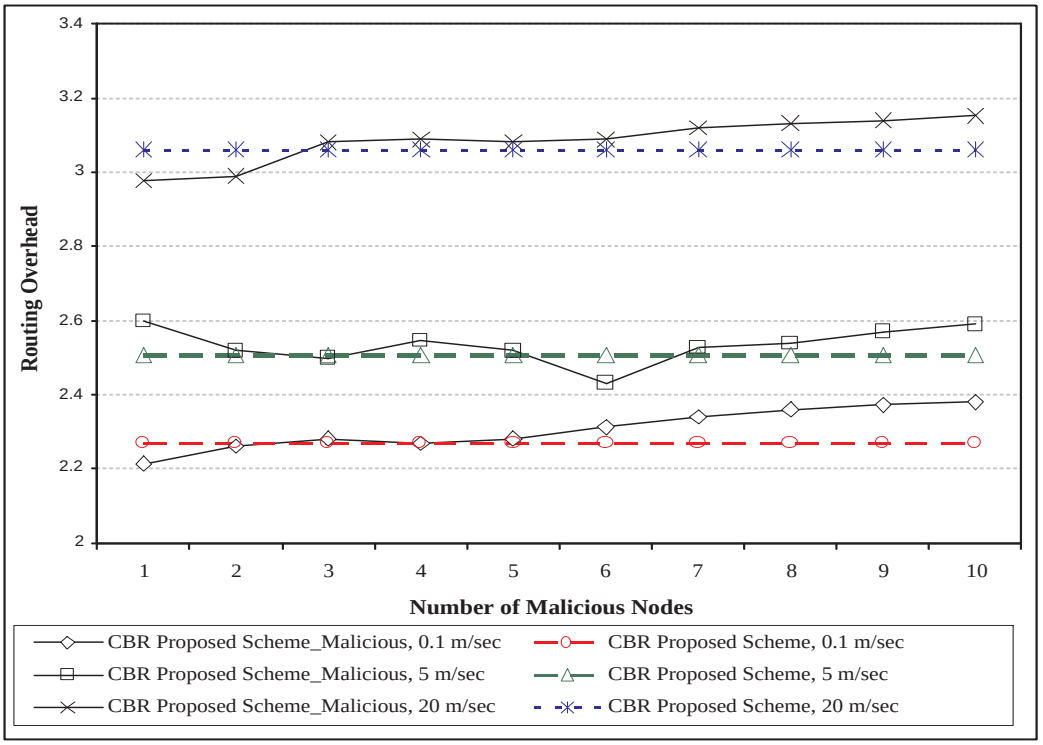

FIGURE 5.19: Routing overhead

if the number of malicious nodes in a 50-nodes network reaches 10, i.e., the number of malicious nodes is 20% of the network size.

5.5.3.2 Routing Overhead

Figure 5.19 shows the routing overhead for different number of malicious nodes in the network. Increasing the number of malicious nodes in the network results in increasing the failures in the certificate chain discovery process and an increase in broadcasting certificate revocation messages through the network. It is obvious from Figure 5.19 that the induced routing overhead due to increasing the number of malicious nodes in the network is negligible for different node's mobility.

5.6 Summary

In this chapter, we presented a robust self-organised public key management scheme. The proposed scheme exploits the routing infrastructure to discover a certificate chain through a web of trust. A measure of the communication cost of the certificate chain discovery process has been proposed. The first contribution of this chapter is that our proposed scheme has low communication cost because each node limits its search for the certificate chain to its directly trusted nodes only. The second contribution is that the use of trust values along with the public key certificates in at least two independent certificate chains enhances the authentication of the proposed key management scheme. The third contribution is concluded from the performance evaluation of Ren et al. scheme that shows that random graph theory is not suitable for managing trust in the mobility environment of MANET. Random graph theory is suitable for trust management during the network initialisation phase or for stationary networks only. The results have shown that our proposed scheme has negligible impact on the network performance and the scheme is suitable for stationary networks and networks with low to high mobility. Simulation results show also that our proposed scheme is robust in both the dynamic and the static networks and against malicious nodes attacks.

Chapter 6

A Robust and Redundant Key Management Protocol

6.1 Introduction

Due to node mobility and geometric constrains, the connectivity of the network graph is hard to achieve in the web of trust key management schemes. Consequently, the process of public key authentication fails. In addition, in some situations (such as in the battlefield), there is a need for a level of trust between the network entities higher than that of the web of trust approach.

In this chapter we propose a robust and redundant key management scheme for MANETs. In the proposed scheme, we have two kinds of mobile nodes. The first kind is called *CA* nodes which are the share holder nodes that are directly trusted by the *CA*. The second kind is the self-organised nodes that join the network after the network formation. A self-organised node generates its private/public key pair locally by the node itself before joining the network. A self-organised node can join the network by contacting as many nodes as possible and establish trust relations with these nodes to get certificates from these nodes. Self-organised nodes will not have a share of the *CA* private key. When a self-organised node is trusted by at least t *CA* nodes (i.e. it has t number of certificates from share holders in its certificate repository), it can combine the partial signatures of

these t nodes and generate a certificate signed by the CA private key SK and henceforth can be verified by every node in the network that has the CA public key. The rest of this chapter is organised as follows: Section 6.2 provides the system and adversary model of our proposed scheme. The simulation model is described in Section 6.3 and then the results are presented in Section 6.4. Finally Section 6.5 summarises the chapter.

6.2 System and Adversary Model

In this section we present an overview of the system and the adversary model of our proposed scheme.

6.2.1 Adversary Model

This proposed scheme has taken into account the following four types of adversaries:

- **Eavesdropper:**

  This adversary passively monitors the channel, and listens to all public information.

- **Denial of service adversary:**

  This adversary may not respond to a message either deliberately or due to stop-fail type of failure during a protocol run.

- **Static Adversary:**

  This adversary chooses the corrupted node at the beginning of the protocol. Its decision about the corrupted node can not be changed by exploiting the obtained information during the protocol run time.

- **Replay adversary:**

  This adversary buffers the received messages and retransmits them whenever necessary to impersonate a honest node.

6.2.2 Trust Model

In the proposed scheme, there is an off-line central authority *CA* to issue each node with a system certificate before the network formation. The system certificate includes the node public key PK_i, node identity u_i, a unique sequence number, certificate generation and expiry dates. Each node from this kind is configured also with a share SK_i of the *CA* private key SK, and the system public key PK. We denote $Cert_{ca_i}$ as the certificate of the *CA* trusted node u_i signed by the *CA* private key SK. This certificate can be verified by any node that has the *CA* public key PK. We denote mobile nodes belong to this type as *CA* trusted nodes. Each node from this type holds a polynomial share $f_i(u_i)$, where

$$f(v_i) = SK + \sum_{j=1}^{t-1} a_j (v_i)^j \tag{6.1}$$

where $a_1, ..., a_{t-1}$ are uniformly distributed over a finite field.

The other type of mobile nodes in our proposed scheme are mobile nodes that join the network after network formation without any previous access to the *CA*. We denote mobile nodes that belong to this type as self-organised nodes. A self-organised node generates its private key SK_i and the corresponding public key Pk_{i_i} locally before joining the network by the node itself.

In order for a node u_i from this type to join the network, it has to send a certificate request to its neighbouring nodes. The certificate request should include the trust evidence that the requesting node holds along with the requesting node's certificate $cert_i$. According to the validity and strength of the trust evidence presented in the certificate request, a neighbouring node decides whether to issue a partial certificate to the requesting node or not. When a neighbouring node u_j decides to serve the certificate request, it generates and returns a partial certificate $Cert_{ca_j \rightarrow s_i}$ to the requesting node u_i by applying its polynomial share on the requesting node's certificate $cert_i$. When a neighbouring node returns a partial certificate to a requesting node it also sends the *CA* public key PK to the requesting node. Upon receiving at least t partial certificates from its neighbouring nodes, node u_i constructs its system certificate $Cert_{ca_i}$. A self-organised node u_i can issue a certificate to another self-organised node u_j by signing node u_j's certificate by

its private key SK_i. We denote $Cert_{s_i \rightarrow s_j}$ as the certificate signed by the self-organised node u_i's private key SK_i to represent its assurance in the binding of the self-organised node u_j and its public key PK_j. For simplicity we denote Sig_i as the digital signature of node u_i.

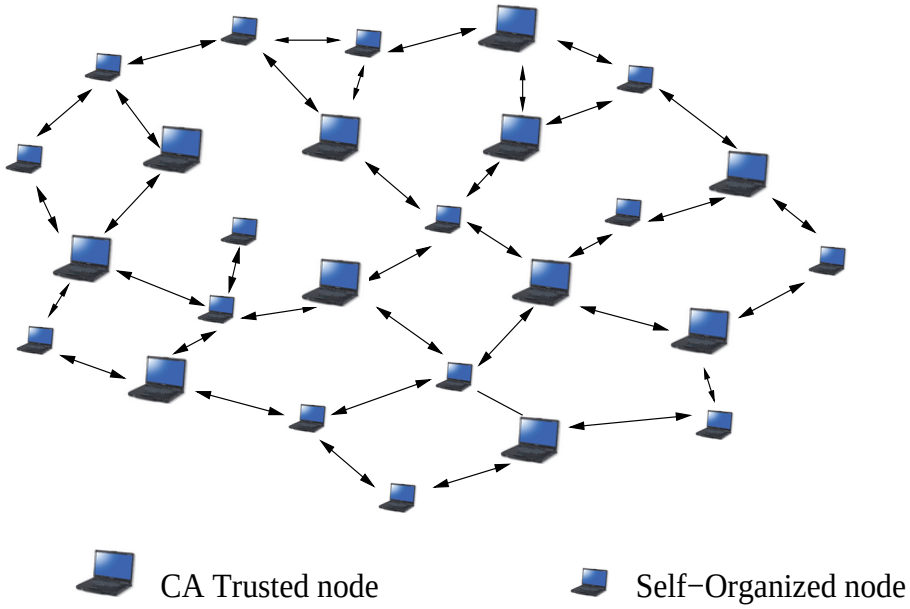

FIGURE 6.1: Mobile nodes in the proposed scheme

Each node in the network has a certificate repository to store valid certificates it issued or certificates issued to it by other entities. In order for a node u_i to authenticate the public key of another node u_j, it has to acquire a chain of valid certificates from node u_i to node u_j. Certificates of *CA* trusted nodes will be verified by the *CA* public key PK. Each remaining certificate in the chain will be verified using the public key of the previous certificate of the chain. The last certificate in the chain holds the public key of the target node k. Interaction between *CA* trusted nodes and self-organised nodes are shown in Figure 6.1.

In order for a self-organised node to acquire a *CA* certificate, it has to be trusted by at least t *CA* trusted nodes. In other words, its certificate has to be partially signed by these nodes, so that it can combine these partial certificates to form a *CA* certificate.

6.2.3 Basic Operation

In this section, we briefly describe the basic operation of our proposed scheme. We assume that every *CA* trusted node is configured with the master public key *PK* and a share SK_i of the master private key *SK* from the *CA* before joining the network. No single *CA* trusted node knows or can deduce the *CA* master private key from the piece it holds. Only a threshold number of *CA* trusted nodes can deduce the *CA* master private key *SK*. Our proposed scheme involves the following processes:

6.2.3.1 Public Key Certificate Generation

A *CA* trusted node has a *CA* certificate signed by the *CA* master private key *SK*. Any mobile node in the network can verify a *CA* certificate by the *CA* public key *PK*. Each self-organised node generates its public/private key pair locally itself. Public key certificates are issued by the nodes based on its knowledge about other nodes in the network. If a node u_j believes that a public key PK_i belongs to a certain user u_i, it has to issue a certificate to node u_i signed by its private key SK_j representing its assurance of the binding of the node u_i's identity and its corresponding public key PK_i. Issued certificates to or from the node are stored in its certificate repository with a validity time. Certificates are issued from *CA* trusted nodes to self-organised nodes or from a self-organised node to another self-organised node. There is no certificate issuing to *CA* trusted nodes as they hold *CA* certificates before joining the network. A self-organised node can generate a *CA* certificate only when it holds at least t partial certificates from *CA* trusted nodes in its certificate repository.

6.2.3.2 Public Key Authentication

Authentication of public keys is performed by exploiting the routing infrastructure. The process of public key authentication is performed as shown in the examples in Figures 6.2, and 6.3 and in the flowchart in Figure 6.4.

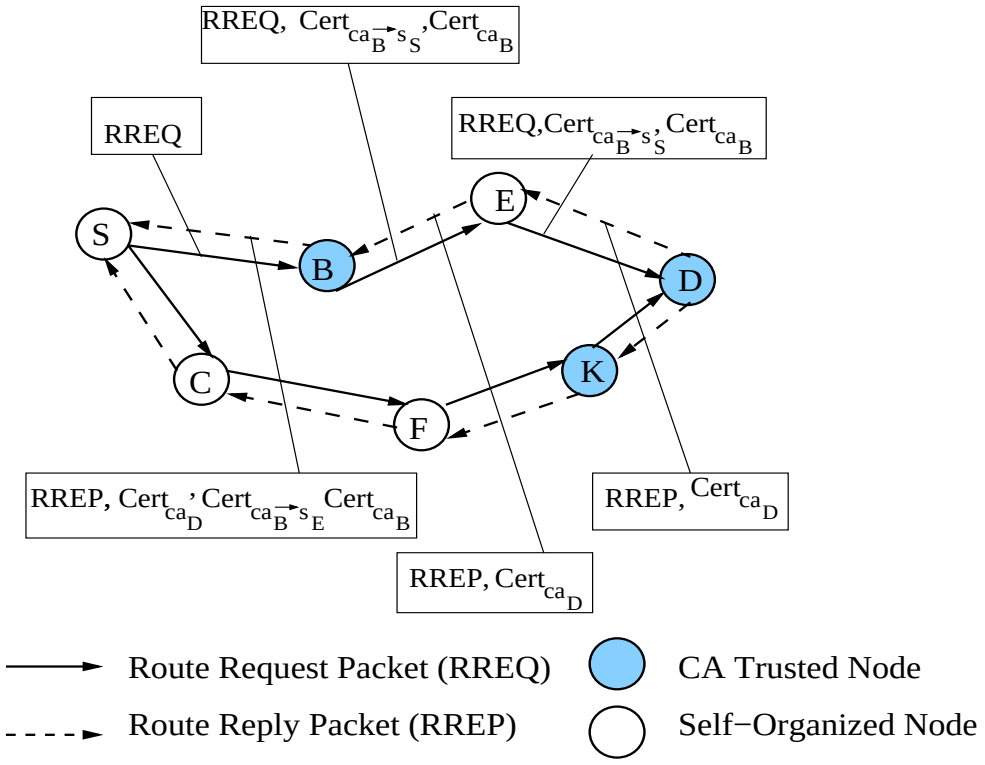

FIGURE 6.2: First certificate chain between *S* and *D*

A *CA* certificate can be verified by any node in the network as all network nodes have the *CA* public key. In the proposed scheme, the public key authentication process is performed through trusted nodes only and the received route requests from untrusted senders are discarded. Having a *CA* certificate enables a mobile node to store in its certificate repository certificates that belong to the self-organised nodes only. This results in reducing the required memory size of a mobile node. The advantage of having a *CA* certificate is obvious in Figure 6.2, and Figure 6.3 as the received requests from nodes *B* and *K* are processed without any direct trust relations between *B* and *K* and the recipients of the requests (in the example nodes *E* and *D*). On the other hand, a request received from a self-organised node will be processed only if there is a direct trust relation exists between the self-organised node and the recipients of the requests.

6.2.3.3 Public Key Certificate Verification and Revocation

CA certificates are verified using the *CA* public key which is feasible for all nodes participating in the network. All other certificates can be verified only by the issuer of the

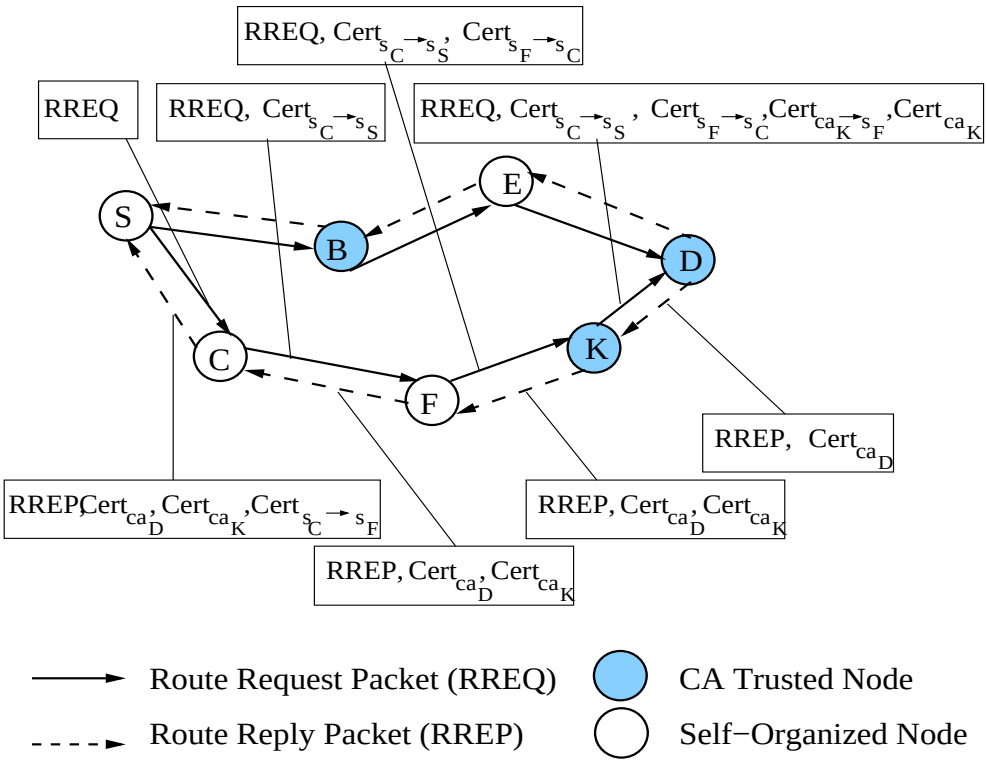

FIGURE 6.3: Second certificate chain between S and D

certificate. Each node can revoke a certificate it issued if it believes that the binding between the public key and the node's identity is no longer valid. Each node can revoke its own certificate if it believes that its private key is compromised. The revoked certificate is put onto the certificate revocation list CRL. A mobile node u_i can revoke a *CA* certificate in two cases: in the first case, when by direct monitoring determines that one of its neighbouring node u_j is compromised or misbehaving. In the other case, when it receives an accusation against any node. In this case, it firstly checks the CRL to find out if the accuser is trusted or not. If the accuser is not a trusted node, the accusation will be dropped but if the accuser is a trusted node, it waits until it receives at least t accusations against the same node to revoke the accused node's certificate. The CRL is broadcast to entire network periodically.

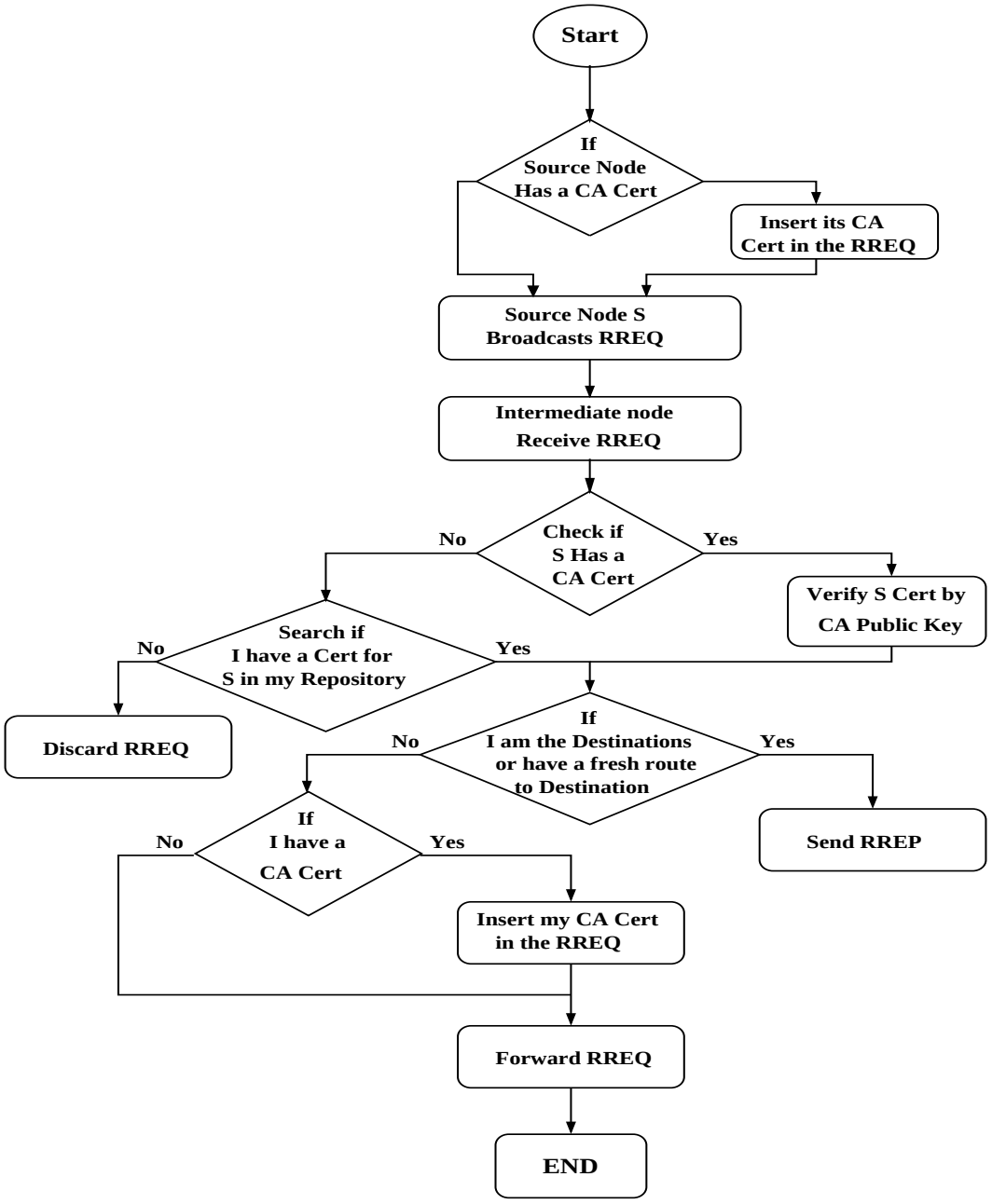

FIGURE 6.4: Public key authentication process

6.3 Simulation Model

Simulations were performed using Network Simulator (NS-2) [56]. The MAC layer protocol IEEE 802.11 is used in all simulations with the data rate 11 Mbps. The source-destination pairs are spread randomly over the network. The NS-2 constant bit-rate (CBR) traffic generator is used to set up the connection patterns with different random seeds. Every simulation run is 1000 seconds long. The mobgenss [128] mobility scenario generator was used to produce random waypoint mobility patterns (refer to Subsection 3.6.3). The pause time is set to zero. A total number of nodes is set to 50 nodes in 1000m x 1000m network area. The transmission range is set to 250m. The traffic loading is set to four CBR packet/sec. The propagation model is Two-Ray Ground [58]. The source-destination pairs are spread randomly over the network. The data packet size is 512 bytes and the size of certificates is also set to 512 bytes. The total number of connections in the network is set to 25 connections. The Ad Hoc On-demand Distance Vector (AODV) routing protocol [68] has been modified to implement the proposed key management scheme. The simulation results are the average of 10 runs. The number of *CA* trusted nodes (t) required to retrieve the CA's private key is set to 5 in the simulation.

6.3.1 Performance Metrics

In addition to the *packet delivery ratio (PDR)* and the *average end-to-end delay* described in Subsection 4.3.2, the *public key authentication ratio* has been selected as a metric during the simulation in order to evaluate the performance of the proposed scheme.

Public Key Authentication Ratio:

It is the ratio of the average number of verified certificate chains received by a node to the number of transmitted route requests.

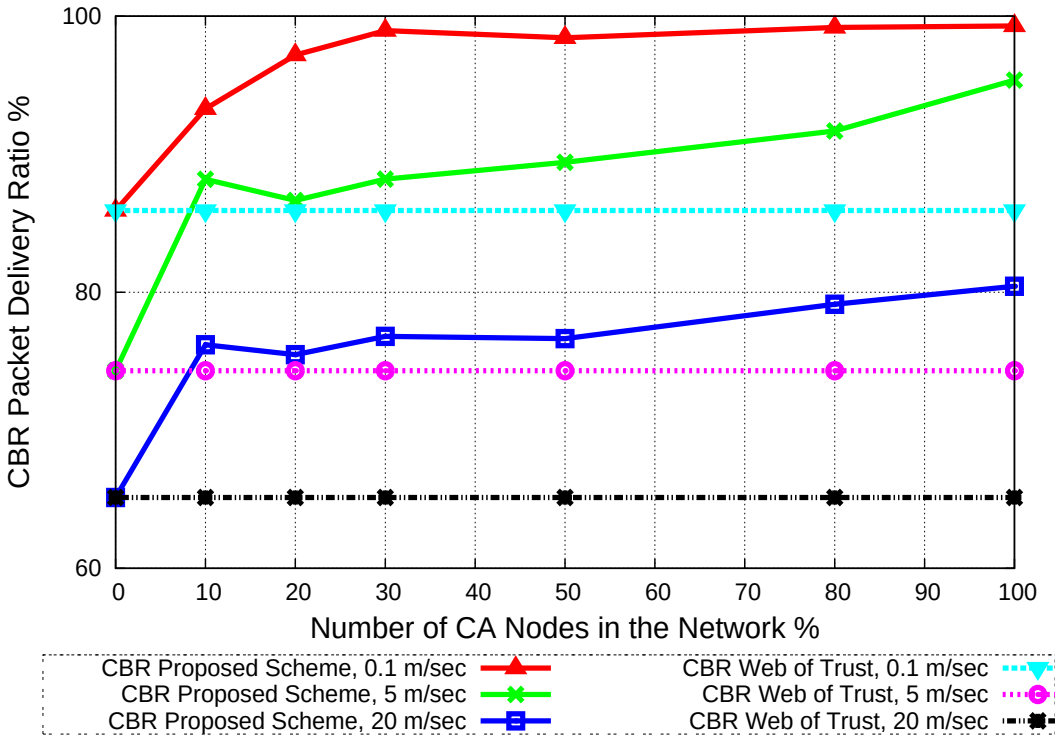

FIGURE 6.5: Packet delivery ratio PDR % vs percentage of *CA* nodes

6.4 Results

In this section, the simulation results of the performance evaluation and security analysis of the proposed scheme are discussed.

6.4.1 Performance Evaluation

In this subsection, we present the main simulation results. In order to evaluate the performance of our proposed scheme, we compare the performance results of our proposed scheme and that of the web of trust scheme.

6.4.1.1 Packet Delivery Ratio (PDR)

Figure 6.5 shows that the packet delivery ratio of our proposed scheme for node mobility 0.1 m/sec, 5 m/sec, and 20 m/sec becomes higher than that of the web of trust approach

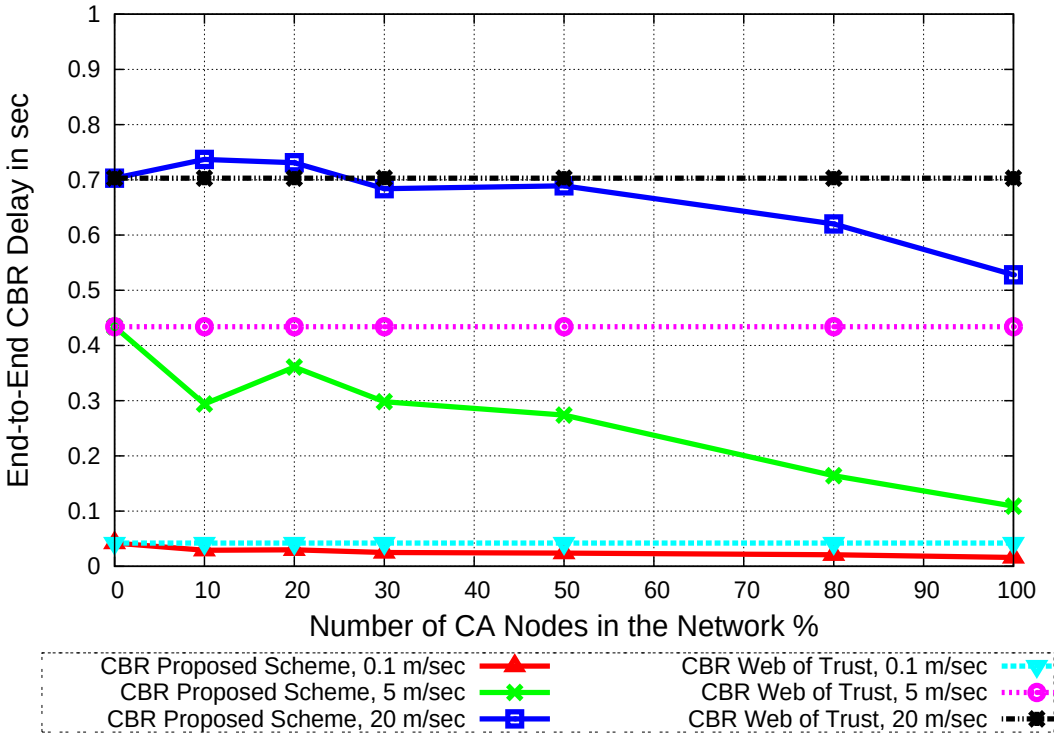

FIGURE 6.6: End-to-end delay in sec vs percentage of *CA* nodes

with increasing the number of nodes that have *CA* certificates in the network. The results in Figure 6.5 matches with the description introduced in Section 6.2 that received requests from *CA* nodes are processed without the need for the existence of direct trust relations between the *CA* nodes and the request recipients.

6.4.1.2 End-to-End Delay

Figure 6.6 shows the end-to-end delay versus the number of *CA* nodes in the network. The end-to-end delay of our proposed scheme decreases with increasing the number of *CA* nodes in the network due to the decrease of retransmissions of routing requests sent by mobile nodes in case of the lack of routes.

6.4.1.3 Public Key Authentication Ratio

Figure 6.7 shows that the public key authentication ratio of our proposed scheme is higher than that of the web of trust approach and becomes higher with increasing the

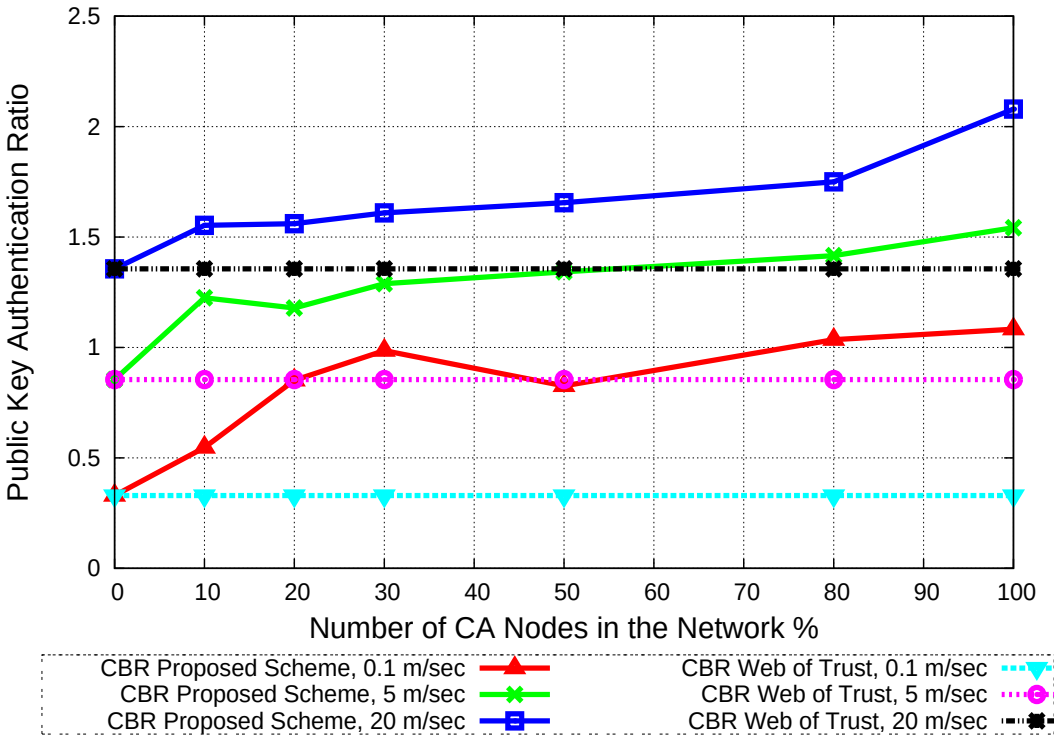

FIGURE 6.7: Public key authentication ratio vs percentage of *CA* nodes

percentage of *CA* nodes in the network. Increasing the number of *CA* nodes in the network increases the number of verified certificate chains received by destination and source nodes in the route request and route reply respectively.

6.4.2 Security Analysis

In this subsection, we study the security of the proposed key management scheme. In the network initialisation phase, an attacker can compromise a mobile node and try to deceive other nodes by making them believe in false certificates. First, the malicious node can insert false certificates in the certificate chain during the public key authentication process. Second, the malicious node can delete certificates from the certificate chain to thwart the public key authentication process. The existence of *CA* nodes along with self-organised nodes in the certificate chains enhances the resistance of the proposed scheme against these types of attacks. Figure 6.8 shows that our proposed scheme enables the source and the destination nodes to certainly perform the public key authentication process if the number of malicious nodes is less than 10% for 0.1 m/sec node

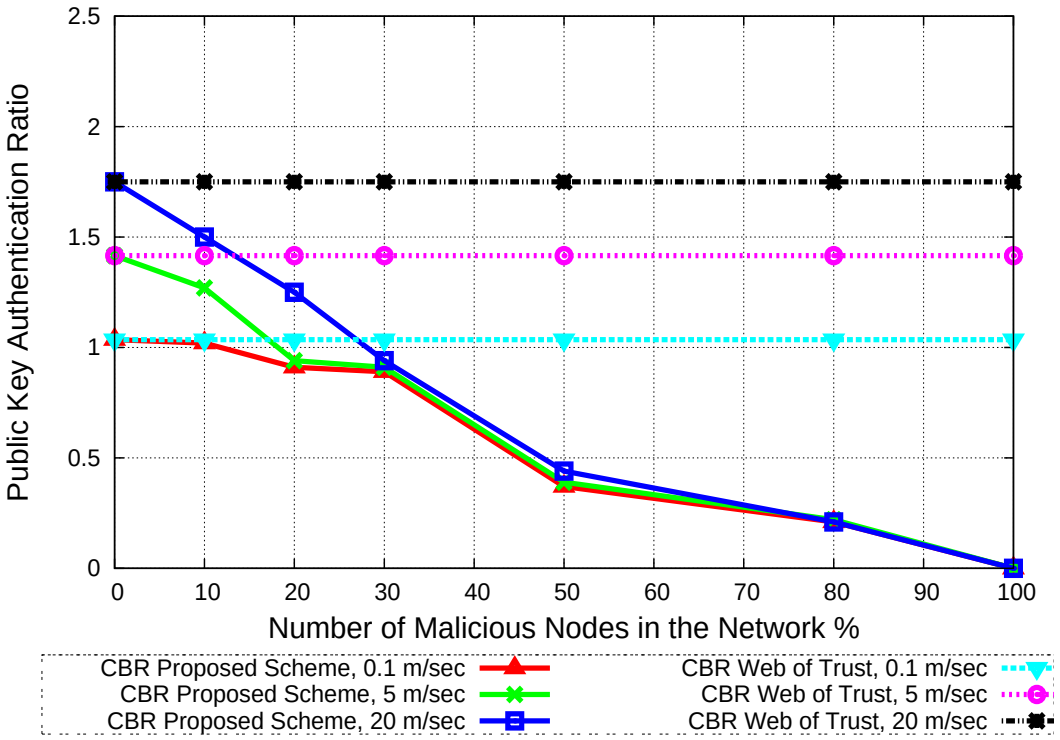

FIGURE 6.8: Public key authentication ratio vs percentage of malicious nodes

mobility, less than 18% for 5 m/sec node mobility, and less than 28% for 20 m/sec node mobility. It also shows that the source and the destination nodes can perform the public key authentication process with probability higher than 0.5 for different node mobility even if the number of malicious nodes in the network exceeds 40%. The probability of achieving public key authentication increases with increasing the node mobility due to decreasing the average number of hops between the source and the destination node and accordingly a large number of verified certificate chains are received.

6.5 Summary

In this chapter, a robust and redundant key management scheme for MANETs has been proposed. In the proposed scheme, there are two approaches for key management. The first approach uses threshold cryptography to establish the distributed central authority among mobile nodes. The second approach is the key management through a web of trust. A mobile node can use one of the two approaches independently and also can use

the two approaches together in order to obtain the authenticated public keys of other nodes in the network. The proposed scheme has been evaluated using the network simulator NS-2 and simulation results show the effectiveness of the proposed scheme. Security analysis shows that the proposed scheme is robust against malicious nodes attacks.

Chapter 7

A Survey of Distributed Elliptic Curve Key Management Protocols for MANETs

7.1 Introduction

The theory of elliptic curves is a classical topic in many branches of number theory and algebra, but in the last few decades it is receiving more attention in cryptographic applications. One class of these curves is elliptic curves over finite fields, which is also called Galois fields. These elliptic curves are finite groups with special structures, which can play the roles of the modulus groups in the discrete logarithm problems (DLPs).

Using elliptic curves for cryptographic protocols was proposed independently by V. S. Miller [45] and N. Koblitz [46, 47, 48] in 1985. Cryptosystems based on elliptic curve discrete logarithm problem (ECDLP) can use a smaller key size than that needed by DLP or integer factorisation problem (IFP) based cryptosystems to provide the same level of secrecy [49, 50, 51]. Reducing the key size while maintaining the same security strength saves memory, computation power, and communication overheads which are major concerns in the resource constrained environment such as smart cards and MANETs.

TABLE 7.1: Key sizes in bits for equivalent security levels

Security (Bits)	Symmetric Key Algorithms	ECC	DH/DSA/RSA
80	Skipjack [149]	160	1024
112	3DES [14]	224	2048
128	AES-128 [150]	256	3072
192	AES-192 [150]	384	7680
256	AES-256 [150]	512	15360

Table 7.1 gives approximate comparable key sizes for symmetric systems, Elliptic Curve Cryptography (ECC) systems, and public key systems such as Diffie-Hellman (DH), Digital Signature Algorithm (DSA), and Rivest Shamir Adleman Algorithm (RSA) [49, 50, 51]. For example, when securing a 192-bit symmetric key, it is prudent to use either 409-bit ECC or 7680-bit DH/DSA/RSA.

This chapter provides a brief review of background information and concepts about elliptic curve cryptography. It also discusses related published work in distributed elliptic curve key management protocols for MANETs and analyses their applicability for MANETs environment. The chapter is organised as follows: Section 7.2 presents a brief background information about elliptic curve cryptography. Section 7.3 surveys elliptic curve key management protocols for MANETs. Finally, the chapter is concluded in section 7.4.

7.2 Elliptic Curve Cryptography

In this section, we present a brief introduction which only covers the background information essential for our cryptographic purposes. A good arithmetic introduction to

elliptic curves can be found in [151], and excellent references for the cryptographic issues related to elliptic curves can be found in [152, 153].

7.2.1 Definitions

- Group:

  $\mathbb{G}$ is used to denote a group which is a set with some binary operation that combines any two of its elements to form a third element.

- Group Order:

  The number of elements in $\mathbb{G}$, denoted $\mid \mathbb{G} \mid$, is called the order of $\mathbb{G}$. A group $\mathbb{G}$ is finite if $\mid \mathbb{G} \mid$ is finite.

- Cyclic Group:

  A group $\mathbb{G}$ is cyclic if there is an element $g \in \mathbb{G}$ such that for each $a \in \mathbb{G}$ there is an integer i where $a = g^i$. Such an element g is called a generator of $\mathbb{G}$.

- Finite Field:

  A finite field or Galois field $GF(q)$ is a field that contains only q elements.

- Integer Factorisation Problem (IFP) [14]:

  IFP is the breaking down of a composite number n into smaller non-trivial primes, which when multiplied together equal the original integer. For instance, given a positive integer n, find its prime factorisation; that is, write $n = p_1^{e_1} p_{2\ldots}^{e_2} p_k^{e_k}$ where the p_i are pairwise distinct primes and each $e_i \geq 1$.

- Discrete Logarithm Problem (DLP):

  Let $\mathbb{G}$ be a finite cyclic group of order n and let g be a generator of $\mathbb{G}$. The discrete logarithm problem (DLP) in $\mathbb{G}$ is as follows: Given (g, g^a) , determine the unique integer a.

- Elliptic Curve:

  An elliptic curve E over a field $GF(q)$ is commonly given by an affine Weierstrass equation of the form

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \tag{7.1}$$

  where $a_i \in GF(q)$ for $i = 1,2,3,4,6$. The elliptic curve $E(GF(q))$ is defined to be the set of points $(x,y) \in GF(q) \times GF(q)$ that satisfy this equation, along with a point at infinity denoted as $\mathscr{O}$.

- Order of Point on Elliptic Curve:

  Given a point $P = (x,y)$ and a positive integer n we define $[n]P = P + P + ... + P$ (n times). The order of a point $P = (x,y)$ is the smallest positive integer n such that $[n]P = \mathscr{O}$

- Elliptic Curve Discrete Logarithm Problem (ECDLP):

  Let $E(GF(q))$ be an elliptic curve defined over $GF(q)$, where q is a prime or $q = p^m$ for some integer m. Suppose that $P \in GF(q)$, and that $Q \in GF(q)$ such that $Q \in$ the subgroup generated by P. Determine the unique integer n, such that $Q = [n]P$.

7.2.2 Elliptic Curve for Cryptographic Applications

There are two groups of elliptic curves in cryptographic applications: prime curves defined over $GF(p)$ and binary curves defined over $GF(2^p)$.

7.2.2.1 Prime Elliptic Curve

Prime curves are best for software applications where it takes relatively few logic gates to create a powerful, fast cryptosystem [154]. Assuming that the prime elliptic curve equation is E over $GF(p)$, defined modulo a prime p, is the set of solutions (x,y) as shown in equation 7.2:

$$E : y^2 = (x^3 + ax + b) \, mod \, p \tag{7.2}$$

where a and $b \in GF(p)$ for p a prime and satisfy

$$4a^3 + 27b^2 \equiv 0 \, (mod \, p) \tag{7.3}$$

The elliptic curve group includes also the point at infinity $\mathscr{O}$ which is the third point of intersection of any straight line with the curve. E is an abelian group with the point $\mathscr{O}$ serving as the identity. For the prime curve the addition operation can be defined as follows: Let $P = (x_1, y_1) \in E$; then $-P = (x_1, -y_1)$. If $Q = (x_2, y_2) \in E$, $Q \neq -P$, then $P + Q = (x_3, y_3)$, where

$$x_3 = \lambda^2 - x_1 - x_2, \tag{7.4}$$

$$y_3 = \lambda (x_1 - x_3) - y_1, \tag{7.5}$$

and

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & \text{if } P \neq Q \\ \dfrac{3x_1^2 + a}{2y_1} & \text{if } P = Q \end{cases} \tag{7.6}$$

Multiplication by a scalar is defined by repeated addition; for example, $3P = P + P + P$. An example of adding two points on a prime elliptic curve is shown in Figure 7.1. Another example of doubling a point on a prime elliptic curve is shown in Figure 7.2.

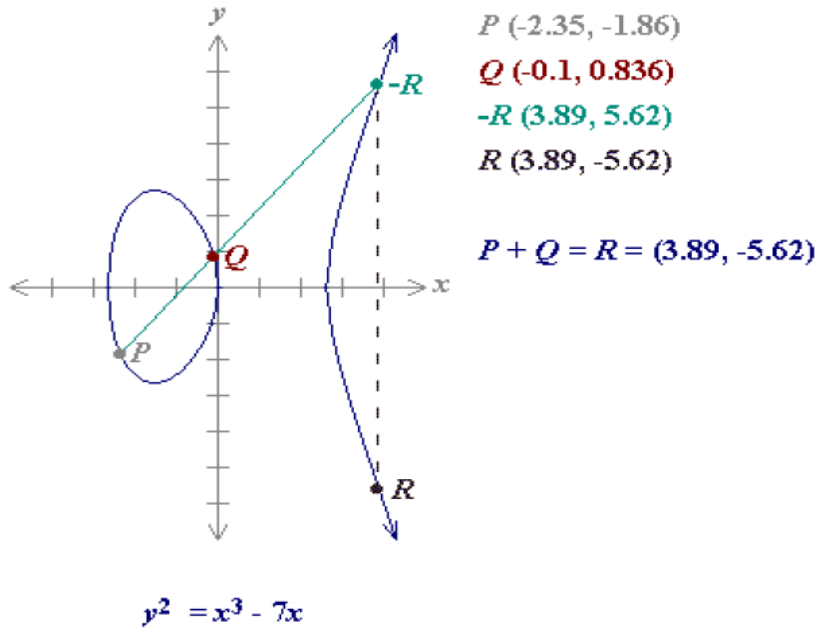

FIGURE 7.1: An example of adding two points on an elliptic curve [6]

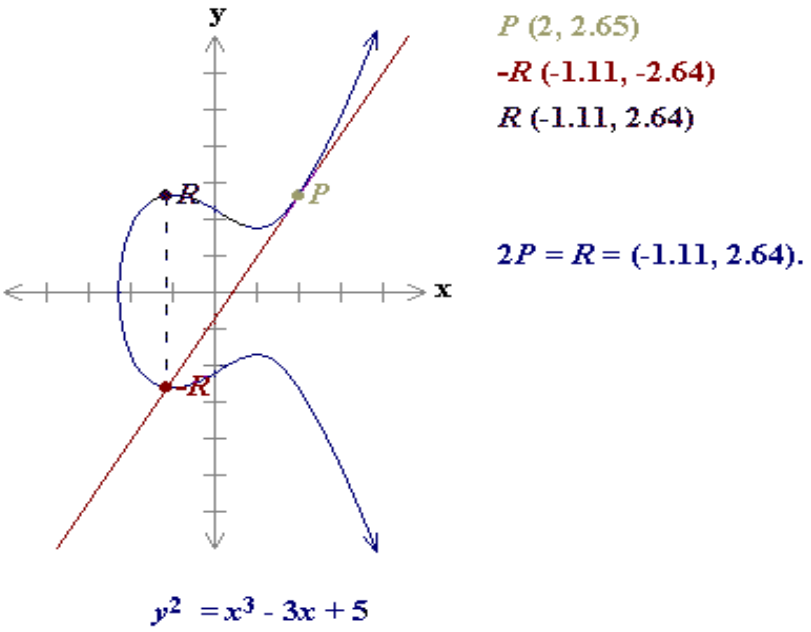

FIGURE 7.2: An example of doubling a point on an elliptic curve [6]

7.2.2.2 Binary Elliptic Curve

A binary curve defined over $GF(2^p)$ can be represented as shown in equation 7.7.

$$E : y^2 + xy = x^3 + ax^2 + b \tag{7.7}$$

where a and $b \in GF(2^p)$ for p a prime, and $b \neq 0$, together with the point at infinity $\mathcal{O}$. E is an abelian group with the point $\mathcal{O}$ serving as the identity. For the binary curve the addition operation can be defined as follows:

Let $P = (x_1, y_1) \in E$; then $-P = (x_1, y_1 + x_1)$. If $Q = (x_2, y_2) \in E$, and $Q \neq -P$, then $P + Q = (x_3, y_3)$, where

$$x_3 = \begin{cases} (\frac{y_1 + y_2}{x_1 + x_2})^2 + \frac{y_1 + y_2}{x_1 + x_2} + x_1 + x_2 + a & \text{if } P \neq Q \\ x_1^2 + \frac{b}{x_1^2} & \text{if } P = Q \end{cases} \tag{7.8}$$

and

$$y_3 = \begin{cases} (\frac{y_1 + y_2}{x_1 + x_2})(x_1 + x_3) + x_3 + y_1 & \text{if } P \neq Q \\ x_1^2 + (x_1 + \frac{y_1}{x_1}) x_3 + x_3 & \text{if } P = Q \end{cases} \tag{7.9}$$

Multiplication is the same as in prime fields, where multiplication by a scalar is defined by repeated addition; for example, $3P = P + P + P$.

7.2.3 Elliptic Curve Cryptography Domain Parameters

The operation of any cryptographic scheme involves arithmetic operations on an elliptic curve over a finite field determined by some elliptic curve domain parameters. If E is an elliptic curve over a finite field $GF(q)$, then let $E(GF(q))$ denote the points in E including the point at infinity $\mathcal{O}$. The parameters of the elliptic curve E are: $\{q, FR, a, b, G, p, h\}$, where q is a prime power ($q = p$ or $q = 2^p$), an indication FR of the method used for representing field elements, a and b are the coefficients of the elliptic

curve equation. G is the base point of the elliptic curve $E(F_q)$, denoted as $G = (x_G, y_G)$, a prime p which is the order of G, and finally the cofactor $h = (\#E(GF(q))/p)$, where $\#E(GF(q))$ denote the number of points on elliptic curve E.

7.2.4 Attacks on ECDLP

The following known attacks are specified to ECDLP:

- **Prime-field anomalous curve:**

  An elliptic curve is said to be anomalous if the trace of the Frobenius map [155] is equal to 1. In this case, the order of the elliptic curve E over $GF(p)$ is equal to p, and ECDLP can be reduced to DLP in an additive group [156, 157]. Thus, we must avoid the use of prime curves for which the group order is equal to the order of the finite field.

- **Curves over field** $GF(2^m)$:

  Gaudry-Hess-Smart (GHS) Weil descent attack [158] can be used to solve the ECDLP over characteristic two finite fields of composite degree (i.e. m is composite). Weil descent reduces ECDLP in $GF(2^m)$ to a DLP in hyperelliptic curve, then one can use algorithms for the hyperelliptic curve DLP. Algorithms for hyperelliptic curve DLP are significantly faster than the best available ones for the ECDLP. However, Menezes et al. proved in [159] that the Weil descent GHS attack is infeasible for all elliptic curves defined over $GF(2^p)$ for primes $p \in [160, 600]$.

7.3 Elliptic Curve Key Management Schemes for MANETs

In this section we survey the related published work on elliptic curve key management schemes for MANETs. We focus our survey on schemes that are based on threshold cryptography [39] such as ECDKG [54], Key Management Based on Elliptic Curve Paillier Scheme [53], and A Key Management and Authentication Model [7].

7.3.1 ECDKG :

Tang et al. [54] propose a distributed key generation protocol based on the elliptic curve discrete logarithm problem for low power devices. The scheme consists of the following five algorithms: the key distribution (KD) algorithm, the key verification (KV) algorithm, the key check (KC) algorithm, the key nonces collection (KN) algorithm, and the key generation (KG) algorithm.

1. The Key Distribution (KD) Algorithm:

   - Step 1: each mobile node u_i chooses two random polynomials $f_i(z)$, $f_i^{'}(z)$ of degree t as follows:

   $$f_i(z) = a_{i0} + a_{i1}z + \cdots + a_{it}z^t \tag{7.10}$$

   $$f_i^{'}(z) = b_{i0} + b_{i1}z + \cdots + b_{it}z^t \tag{7.11}$$

   where $a_{il}, b_{il} \in GF(q)$, and $(0 \leq l \leq t)$.

   Let $SK_i = a_{i0} = f_i(0)$ is the private key of the server node u_i.

   - Step 2: compute the shares $s_{ij} = f_i(u_j)\, mod\, p$, $s_{ij}^{'} = f_i^{'}(u_j)\, mod\, p$ for $j = 1, \cdots, n$ and sends s_{ij}, $s_{ij}^{'}$ to u_j using the private channel.
   - Step 3: compute and broadcast the following public values:

   $$PV_{il} = (a_{il} \odot G) \oplus (b_{il} \odot G^{'}) \tag{7.12}$$

   Where $(0 \leq l \leq t)$, G is the base point, and $G^{'}$ is another point the discrete logarithm of which with respect to G is not known.

2. The Key Verification (KV) Algorithm:

   - Step 1: each node that receives s_{ji} and $s_{ji}^{'}$ from node u_j verifies that:

   $$(s_{ji} \odot G) \oplus (s_{ji}^{'} \odot G^{'}) = \sum_{l=0}^{t}{}^{\oplus} ((u_i)^l \odot PV_{il}) \tag{7.13}$$

- Step 2: u_i broadcasts a complaint against u_j if formula 7.13 fails.
- Step 3: u_i broadcasts s_{ij} and s'_{ij} that satisfy formula 7.13 if u_i receives a complaint to him from u_j.

3. The Key Check Algorithm:

   Step 1: node u_i removes node u_j from its trusted group if it received more that t complaints against u_j , or if the reply of the node u_j in step 3 of the key verification algorithm does not satisfy formula 7.13.

4. The Key Nonces Collection (KN) Algorithm:

   Each mobile node in the session broadcasts a nonce containing its identity to countermeasure the static attack presented in [103]. In this attack, two faulty players collude to make a bias on a given bit of the public key which causes the generated public key to be non uniformly distributed over the given field.

5. The Key Generation (KG) Algorithm:

   - Step 1: each node u_i computes its public key $PK_i = SK_i \odot G$ and broadcasts PK_i.
   - Step 2: each node u_i receives PK_j $(j = 1, \cdots, n)$ and computes the session public key

$$SS_{PK} = \sum_{j=1}^{n} {}^{\oplus} PK_j \tag{7.14}$$

The generated public key follows a uniform distribution in the elliptic curve additive group and the scheme is invulnerable to the static attack [103]. Each node in the network has to communicate with all other nodes in the network through a secure channel. The requirement of an additional secure channel for transferring secret shares is a waste of bandwidth. Due to the characteristics and constrains of MANETs presented in section 1.1.1 and section 2.3 respectively, it can not be assumed that each node will be able to communicate successfully with all other nodes in the network in order to generate session keys. This gives rise to the low protocol success ratio as will be shown in section 8.5.2.

7.3.2 Key Management Based on Elliptic Curve Paillier Scheme

Wang et al. [53] propose a key management based on the elliptic curve paillier scheme for ad hoc networks. The scheme assumes that the group has been created and each member in the group has a shared secret provided to it in the network initialisation phase. When a new node wants to join the group, it must be initiated by receiving a sub-key corresponding with the system key. Once the new node received its sub-key, it can participate into the network activities or authenticate the other new nodes. The scheme is based on the homomorphism of elliptic curve Paillier scheme : if C_1 is an encryption of m_1 and C_2 is an encryption of m_2, then $(C_1 + C_2)$ is an encryption of $(m_1 + m_2)$. The protocol can be summarised in the following steps:

- Step 1: each node chooses the following public parameters [160]: a modulus $N = pq$ as a product of two odd primes p and q, and a random elliptic curve $E : y^2z = (x^3 + axz^2 + bz^3)$, where $a, b \in GF(N)$.
- Step 2: each node chooses a secret key [160]:

$$M = lcm(\mid E(GF(p)) \mid, \mid E(GF(q)) \mid) \tag{7.15}$$

  Where *lcm* is the least common multiple.

- Step 3: the CA picks a random polynomial $f(z)$ of degree $t-1$ such that:

$$f(z) = a_0 + a_1 z + \cdots + a_t z^{t-1} \tag{7.16}$$

  where $a_l \in GF(q)$, and $(0 \leq l \leq t)$.

  Let $a_0 = f_i(0)$ is the system private key and the total number of nodes in the network $n < q$.

- Step 4: the CA compute the sub-key $s_j = f(u_j) \, mod \, p$ for $j = 1, \cdots, n$ and sends s_j to u_j. After all the n nodes in the network get their own sub-keys, the polynomial $f(z)$ and the system private key a_0 are destroyed.

- Step 5: when a node wants to join the network it chooses the public parameters and the secret key as mentioned in step 1 and step 2 respectively.

- Step 6: the new node u_{new} sends a request to join the network to u_1 (one of t honest neighbour nodes $(u_1, u_2, \cdots, u_t)$).

- Step 7: u_1 generates a partial sub-key for the new node, encrypts the generated partial sub-key with the new node public key, and forwards the request along with the encrypted sub-key to the neighbour node u_2. The process continues with the authentication sequence $u_1 \rightarrow u_2 \rightarrow \cdots \rightarrow u_t$ and each node encrypts its generated partial sub-key and adds it to the encrypted messages that received from the previous nodes before forwarding the message to the following node in the authentication group.

- Step 8: u_{new} receives the reply from the last node in the authentication group u_t, decrypts the reply with its secret key M, and computes its sub-key. The new node's sub-key allows the new node to participate in the network security services including issuing sub-keys for new nodes.

The scheme suffers from the delay in the authentication process of the new node because the authentication is performed in a serial way ($u_1 \rightarrow u_2 \rightarrow \cdots \rightarrow u_t$) which causes each node in the authentication group to wait until the previous node finishes its computation of the new node's partial sub-key. Each encryption process includes a number of point multiplications which is computationally expensive operations and can add further delay to the authentication process. The serial mechanism of the authentication process is not suitable in the mobility environment of MANETs because the neighbouring nodes may change unpredictably while the authentication process is still running. Furthermore, the scheme addresses only the process of new nodes joining the network and ignores how existing nodes can renew their sub-keys.

7.3.3 A Key Management and Authentication Model

Lou et al. in [7] propose a key management and authentication model for ad hoc network. This scheme is based on elliptic curve combined public key scheme and threshold cryptography. An off-line central authority (CA) is needed in the initialisation phase only. In the network deployment phase, all key management processes are self-organised. There are two types of mobile nodes in the network: *general nodes* denoted as *GN*, and *server nodes* denoted as *SN*. The protocol can be summarised as follows:

- Step 1: in the initialisation phase, CA chooses the scheme security parameters such as:

  a) An elliptic curve E over a finite field $GF(q)$;

  b) h map functions;

  c) A private seed key $m \times h$ matrix

$$SSK = \begin{bmatrix} r_{11} & r_{12} \cdots & r_{1h} \\ r_{21} & r_{22} \cdots & r_{2h} \\ \cdots & \cdots & \cdots \\ r_{m1} & r_{m2} & r_{mh} \end{bmatrix}_{m \times h} \tag{7.17}$$

  d) A public seed key $m \times h$ matrix

$$PSK = \begin{bmatrix} (x_{11}, y_{11}) & (x_{12}, y_{12}) \cdots & (x_{1h}, y_{1h}) \\ (x_{21}, y_{21}) & (x_{22}, y_{22}) \cdots & (x_{2h}, y_{2h}) \\ \cdots & \cdots & \cdots \\ (x_{m1}, y_{m1}) & (x_{m2}, y_{m2}) & (x_{mh}, y_{mh}) \end{bmatrix}_{m \times h} \tag{7.18}$$

  where $(x_{ij}, y_{ij}) = r_{ij} \odot G$ and G is the base point of the elliptic curve $E(F_q)$, denoted as $G = (x_G, y_G)$ of order p.

  The private seed key matrix *SSK* is kept secret while the other parameters can be public.

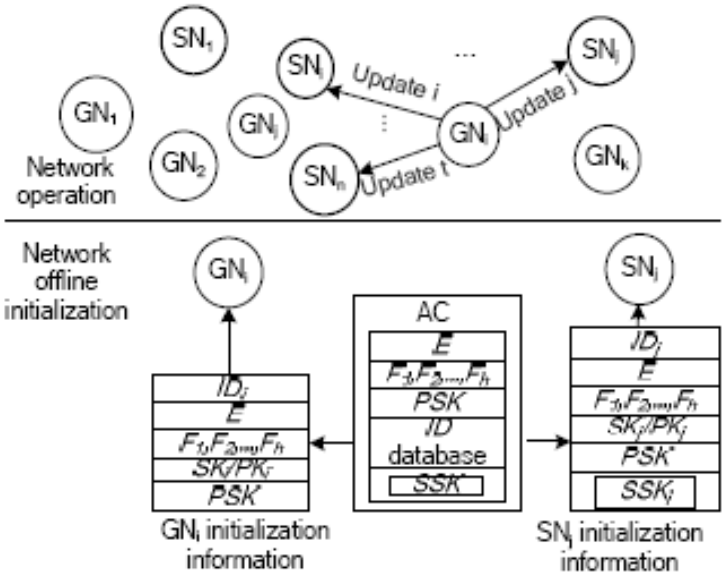

FIGURE 7.3: An overview of Lou et al. scheme [7]

- Step 2: the CA maps the node identity into h values: m_1, m_2, $\cdots$, m_h as row numbers. Then the CA picks out elements from each column of SSK matrix according to the computed row numbers. The private/public key pair SK_i/PK_i of node u_i can be computed as follows:

$$SK_i = (r_{m_1 1} + r_{m_2 2} + \cdots + r_{m_h h}) \, mod \, p \tag{7.19}$$

$$PK_i = ((x_{m_1 1}, y_{m_1 1}) \oplus (x_{m_2 2}, y_{m_2 2}) \oplus \cdots \oplus (x_{m_h h}, y_{m_h h})) \, mod \, q \tag{7.20}$$

  The initialisations of general nodes are now completed and the following step is performed for server nodes only.

- Step 3: the CA splits the SSK matrix into n shared seed key matrices using Shamir (t,n) threshold scheme [39]. This n shared matrices $SSK_{d(d=1,\cdots,n)}$ are sent to n server nodes as shown in Figure 7.3. The n shared seed matrices $SSK_{d(d=1,\cdots,n)}$ are used to update the private keys of network nodes as shown in Figure 7.3 and can be used to issue a private key to a new node that requests to join the network.

This scheme inherits the weaknesses of ID-based cryptography. The major problem with ID-based cryptographic schemes is that they yield only level 1 trust [161], where

the private key of a node depends on its identity. Another problem is how to authenticate the identity of a node before sending the shares of the private key corresponding to its identity. The scheme is vulnerable to the man-in-the-middle attack on nodes joining the network [162]. The scheme does not specify how server nodes can securely send the private key shares to the requesting node. It is also not clear how a couple of nodes will perform the authentication process in the presence of an adversary.

7.4 Summary

This chapter presents a background information and concepts of elliptic curve cryptography. It also reviews the different kinds of elliptic curves employed for cryptographic applications. Two families of elliptic curves are used for cryptographic applications: prime curves defined over $GF(p)$ and binary curves defined over $GF(2^p)$. Prime curves are best for software applications while binary curves are best for hardware applications [154]. Different types of attacks on ECDLP have also been presented. A number of distributed key management schemes based on ECDLP have been studied putting some light on their suitability for the mobility environment of MANETs.

Chapter 8

Elliptic Curve Distributed Key Management Protocol

8.1 Introduction

Traditional key management service is based on a central authority (CA) or a trusted third party (TTP) to issue public key certificates to all nodes in the network. However, providing key management by relying on a single TTP or CA is not applicable within the pervasive environment of MANETs. In the mobility environment of MANETs, only distributive key management schemes can work efficiently.

In this chapter, a threshold key management protocol for MANETs using elliptic curve dlog-based cryptosystem is proposed. In the proposed scheme, an off-line authority is required in the network initialisation phase before network deployment. The off-line authority constructs the shares matrix for each node in the network by distributing a large number of secrets to all network nodes according to a verifiable secret sharing scheme [163, 164]. After network deployment, the off-line authority has no role in the network. When a group of nodes wish to establish a secure session, a node from the session broadcasts a session formation request. The request includes the identities of the session members, the required threshold of the session, and a nonce. Each session member collaborates with its neighbouring nodes to generate its private/public key pair,

other session members public keys, and the session public key. Neighbouring nodes provide session members with the required shares even if the neighbouring nodes are not session members. The degree of the secret sharing polynomial used by the off-line authority to generate the shares matrix for each node determines the number of neighbouring nodes needed to provide the required shares to session members.

The rest of the chapter is organised as follows: Section 8.2 reviews related work. Section 8.3 provides the scheme description of our proposed scheme. The security analysis of the proposed scheme is then presented in Section 8.4. The performance evaluation of the proposed scheme is presented in Section 8.5. Finally Section 8.6 concludes the chapter.

8.2 Related Work

In this section a review of related key management schemes are presented. In [2], and [98], threshold cryptography has been proposed to provide a reliable, distributive key management for MANETs by exploiting some nodes as a trust anchor for the rest of the network. A model for distributing trusted services (by using threshold cryptography [39]) among a set of servers despite some servers being under control of an attacker has been proposed by Cachin et al. in [40].

In [164], a non interactive verifiable secret sharing was presented by P. Feldman. The scheme is optimal in that it tolerates up to $(n-1)/2$ bad players, which is provably bitwise secure, assuming the intractability of taking discrete logarithms. This scheme shows also that the verifiable secret sharing can be used as a subroutine to simulate a simultaneous broadcast network, which makes protocols such as secret bidding as efficient as the best verifiable secret sharing.

In [42], the first DKG scheme was proposed by Pedersen in 1991. The basic idea in Pedersen's DKG protocol [42] is to have n parallel runs of Feldman's verifiable secret sharing (VSS) protocol [164] in which each player P_i acts as a dealer and selects a random secret $z_i \in GF(q)$. The secret value x is the sum of the shared z_i's. The

public value y is taken to be the product of the y_i's that correspond to the shared z_i's since Feldman's VSS has the additional property of revealing $y_i = g^{z_i}$. Chronopoulos et al. proposed a practical implementation of DKG on a network of computers in [41]. In this scheme, secure sockets are used to create the private channels over the Internet. A modified version of Pedersen's DKG protocol [42] has been proposed in [44]. The modified DKG protocol [44] is proved to satisfy the security requirements from DKG protocols and, in particular, it ensures a uniform distribution of the generated keys. The modified DKG tolerates up to t halting players for $n \geq 2t+1$ and t eavesdropping players for $n \geq t+1$ and t static adversary for $n \geq 3t+1$. The protocol can be used as a secure replacement for the many applications of Pedersen's protocol.

In [54], an elliptic curve distributed key generation protocol has been proposed by Tang et al. In this protocol, a player can reveal its public key share only when all other players in their local non-disqualified sets have agreed not to change their local non-disqualified sets. A nonce is used as a timestamp to emulate the timeout and serves as a confirmation to other players that this player is ready to extract its public key. However, this scheme suffers from the heavy communication overhead required to perform the protocol successfully as shown in Section 7.3.1. In [53], Wang et al. proposed a key management scheme based on the homomorphism of elliptic curve Paillier scheme. This scheme addresses only the new node joining process. In addition, the generation of the new node's sub-key is performed in a serial manner among the new node's honest neighbours. Hence, the authentication process of the new node is time consuming as shown in Section 7.3.2. In [7], Liu et al. proposed a key management and authentication model for ad hoc network. This scheme is based on the combination between elliptic curve and ID-based cryptography. This scheme is vulnerable to the man-in-the-middle attack. In addition, it inherits the weakness of ID-based cryptography. For further details about this algorithm refer to Section 7.3.3.

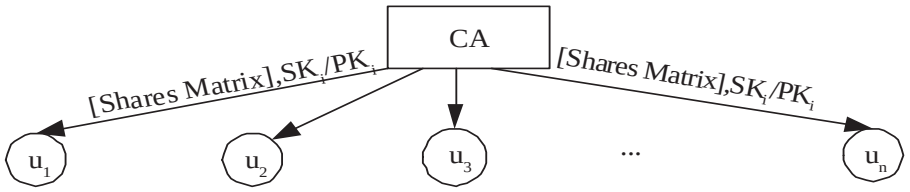

FIGURE 8.1: Network initialisation phase

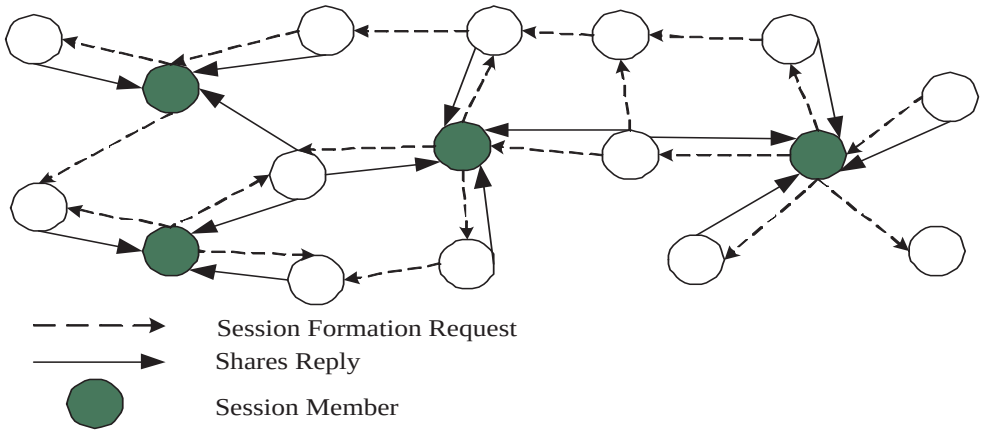

FIGURE 8.2: Network deployment phase

8.3 Scheme Description

The network initialisation phase and network deployment phase of our proposed scheme are shown in Figures 8.1 and 8.2 respectively. In the proposed scheme, a central authority (*CA*) is required in the initialisation phase only as shown in Figure 8.1. *CA* selects an appropriate elliptic curve and determines the elliptic curve cryptography domain parameters. We make the following assumptions in the proposed scheme:

- Each user in the network has a unique identification number ID in $GF(q)$ where q is a prime or $q = 2^p$.
- Each node u_i is assumed to have a long term public/private key pair PK_i/SK_i, where the public key $PK_i = SK_i \odot G$ is known to all mobile nodes in the network (where $\odot$ is the point multiplication by a scalar operation; an example of point multiplication by 2 is shown in Figure 7.2).

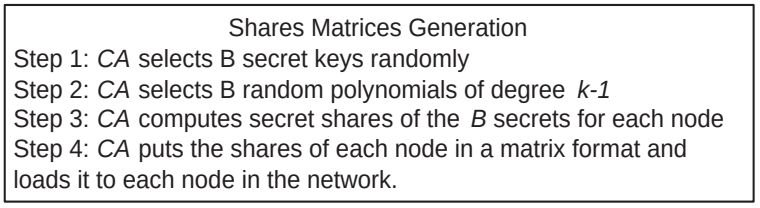

FIGURE 8.3: Overview of the shares matrices generation

- The proposed key management scheme works in support of an ad hoc on demand routing protocol (such as AODV [68])
- The maximum number of nodes that can be compromised in the network is $t-1$ nodes.
- The computation power capabilities of mobile nodes are sufficient to perform the required computations for the proposed scheme.

8.3.1 Shares Matrices Generation:

An overview of the shares matrices generation process is shown in Figure 8.3.

Let U be the set of nodes in the network initialisation phase, where

$$U = \{u_i,\ 1 \leq i \leq n\}$$

The set of nodes U has a cardinality equal to n and nodes that $\in U$ are identified by their unique identities $(u_1, u_2, \cdots, u_n)$, where $u_i \in GF(q)$. Shares matrices generation phase includes the following steps:

- Step 1: *CA* selects B secret keys S, where

$$S = \{S_1, S_2, \cdots, S_B\}$$

- Step 2: *CA* rearranges the selected secret keys from the previous step and constructs the following secret keys matrix:

$$\begin{bmatrix} S_{11} & \cdots & S_{1d} \\ \vdots & \vdots & \vdots \\ S_{m1} & \cdots & S_{md} \end{bmatrix}_{m \times d} \tag{8.1}$$

  where $B = m \times d$.

- Step 3: *CA* picks $m \times d$ secret polynomials $f_{11}(x), \cdots, f_{md}(x)$:

$$f_{kh}(x) = (S_{kh} + \sum_{i=1}^{t-1} a_i^{(kh)} x^i) \, mod \, p, \tag{8.2}$$

  where $1 \leq k \leq m$, $1 \leq h \leq d$.

- Step 4: using the B secret polynomials form the previous step, the *CA* computes secret polynomial shares for each node such that:

  $s_{kh}^{(i)} = f_{kh}(u_i)$, where $1 \leq k \leq m$, $1 \leq h \leq d$, and $1 \leq i \leq n$.

- Step 5: *CA* establishes, and preloads the following $m \times d$ shares matrix to node u_i before network deployment:

$$\begin{bmatrix} s_{11}^{(i)} & \cdots & s_{1d}^{(i)} \\ \vdots & \vdots & \vdots \\ s_{m1}^{(i)} & \cdots & s_{md}^{(i)} \end{bmatrix}_{m \times d} \tag{8.3}$$

Shares of the same indices in the shares matrices of every mobile node represent shares for the same secret S_r. Any group of t nodes can collaborate to generate a CA secret such that:

$$\sum_{i=1}^{t} \prod_{\substack{j=1 \\ j \neq i}}^{t} \frac{j}{j-i} s_{kh}^{(i)} = S_{kh} \tag{8.4}$$

where $1 \leq k \leq m$, $1 \leq h \leq d$.

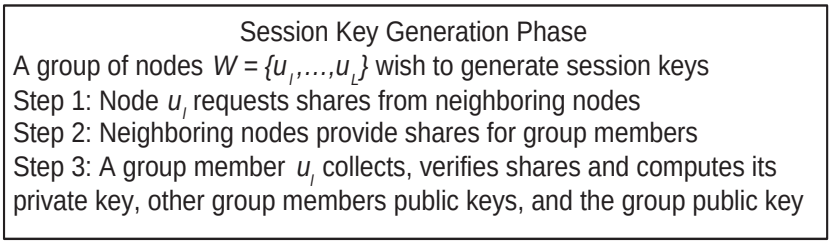

FIGURE 8.4: An overview of the session key generation

The parameter t chosen by the *CA* determines the number of neighbouring nodes needed to collaborate with a session member u_l in order to construct its private/public key pairs, session members public keys, and the session public key. This parameter can not be changed after the system is set up.

8.3.2 Session Key Generation

An overview of the session key generation process is shown in Figure 8.4

When a group of nodes $W = \{u_l,\ 1 \leq l \leq L\}$ wish to generate session keys to be used in securing communications for this session (for example generate a signature for a message m), they follow the following steps:

- Step 1: the mobile node u_l which has the largest index (identity number) over all other session members, broadcasts a session formation request as follows:

$$u_l \rightarrow Broadcast : \{REQ, Nonce, k, u_1, \cdots, u_L\}_{Sig_l}$$

  where k is the number of shares to be picked from the shares matrix by each node in order to construct its secret sharing polynomial, and Nonce is a large random number used to randomise the session keys generation process.

- Step 2: when a neighbouring node u_j (which is not necessarily a session member and not limited to the one-hop neighbours) receives the request, it authenticates the origin of the request by verifying the signature of the sender node u_l using the mobile node u_l's long term public key PK_l.

- Step 3: when the verification succeeds, mobile node u_j maps the session member *IDs* and the Nonce by using a k recursive hash functions $H^k(x)$, where $H^2(x) = H(H(x))$, to obtain shares indices to be picked from its shares matrix as follows:

  a) Calculate k row numbers as follows:

$$R_i = H^i(u_1|u_2|\cdots|u_L|Nonce)\, mod\, m \tag{8.5}$$

  Where $1 \leq i \leq k$

  b) Calculate k column numbers as follows:

$$C_i = H^i(Nonce)\, mod\, d \tag{8.6}$$

  Where $1 \leq i \leq k$

- Step 4: mobile node u_j picks k shares from its shares matrix (refer to equation 8.3) according to the rows and columns indices obtained from the previous step such that $s^{(j)}_{R_1C_1}, s^{(j)}_{R_2C_2}, \cdots, s^{(j)}_{R_tC_t}$, and constructs its secret sharing polynomial as follows:

$$f^{(j)}(x) = (a^{(j)}_1 + \sum_{i=2}^{k} a^{(j)}_i x^{i-1})\, mod\, p \tag{8.7}$$

  Such that, for $(1 \leq i \leq k)$:

$$a^{(j)}_i = s^{(j)}_{R_iC_i} \tag{8.8}$$

- Step 5: using its secret sharing polynomial, node u_j calculates a share for each session member $u_l \in W$ as follows:

$$Z^{(j)}_l = f^{(j)}(u_l) = a^{(j)}_1 + \sum_{i=2}^{k} (a^{(j)}_i (u_l)^{i-1})\, mod\, p \tag{8.9}$$

  Where $1 \leq l \leq L$

- Step 6: node u_j calculates the public shares $\left\{PV^{(j)}_1, \cdots, PV^{(j)}_L\right\}$, where

$$PV^{(j)}_l = Z^{(j)}_l \odot G \tag{8.10}$$

and sends out the following message to the session member node u_l:

$$u_j \rightarrow u_l : \left\{ REP, Enc_{PK_l}\left[Z_l^{(j)}\right], PV_1^{(j)}, \cdots, PV_L^{(j)} \right\}_{Sig_j}$$

Where $Enc_{PK_l}\left[Z_l^{(j)}\right]$ refers to the encryption of the share $Z_l^{(j)}$ using the public key PK_l of node u_l and sig_j is the digital signature of node u_j using its long term private key SK_j.

- Step 7: mobile node u_j computes and broadcasts the following public values:

$$Pval_i^{(j)} = (a_i^{(j)} \odot G) \quad \text{where } (1 \leq i \leq k) \tag{8.11}$$

- Step 8: mobile node u_l collects at least t replies from its neighbouring nodes in addition to its own calculations from the previous steps to reconstruct its private/public key pair, session members public keys and the session public key as follows:

  a) The private key of the mobile node u_l in this session is calculated as follows:

$$sk_l = \sum_{j=1}^{t} Z_l^{(j)} \prod_{\substack{i=1 \\ i \neq j}}^{t} \frac{i}{i-j} \; mod \; p \tag{8.12}$$

  b) The public key of the mobile node u_l in this session is:

$$pk_l = sk_l \odot G \tag{8.13}$$

  c) The session member u_l calculates the public key for other session members u_h, where $(1 \leq h \leq L, h \neq l)$ as follows:

$$pk_h = \sum_{j=1}^{t}{}^{\oplus} PV_h^{(j)} \prod_{\substack{i=1 \\ i \neq j}}^{t} \frac{i}{i-j} \; mod \; p \tag{8.14}$$

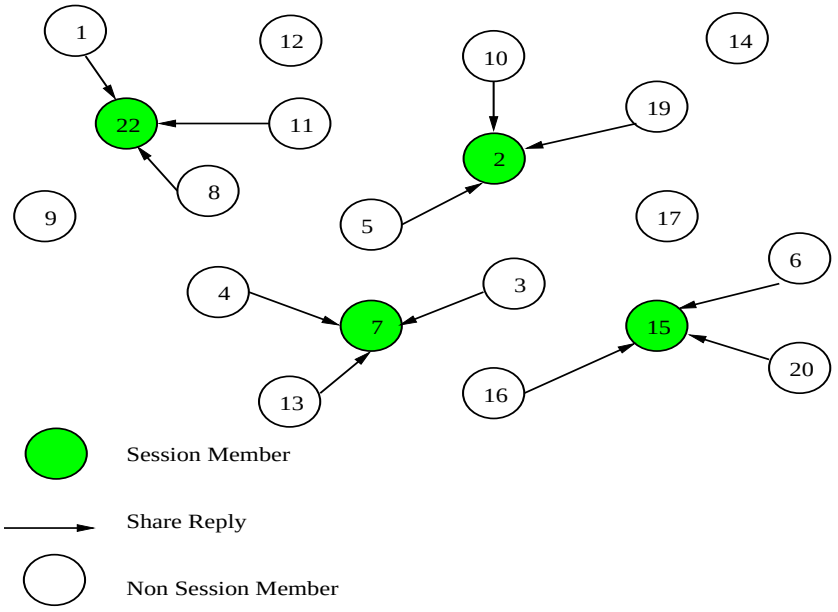

FIGURE 8.5: Session key generation example

d) The session member u_l calculates the session public key by Lagrange interpolation of public keys of at least k session members as follows:

$$SS_{PK} = \sum_{l=1}^{k}{}^{\oplus} pk_l \prod_{\substack{i=1 \\ i \neq l}}^{k} \frac{i}{i-l} \; mod \; p \tag{8.15}$$

where k is the threshold number included in the session formation request which determines the minimum number of mobile nodes required to form the session.

The advantage of the key generation method described above is that shares matrices of quite small sizes can form a very large key space. If the size of the shares matrix is $m \times d$, and the threshold is k, then the key space will be equal to

$$\frac{(m \times d)!}{[(m \times d) - k]!} \tag{8.16}$$

For example if each element in the shares matrix is 256 bits, then a shares matrix of 12 K bytes size (which is small enough to be stored in a mobile ad hoc node with limited memory size) with the threshold k equal to 5 can produce 10^{14} session keys.

We now explain the session key generation process with the help of the example in Figure 8.5. We assume the values of t (threshold number of nodes required to recover a shared secret) and k (threshold number of nodes required to form a session) are both equal to 3 in this example. Let nodes 2, 7, 15, and 22 wish to establish session keys to be used in securing communications inside the group. Node 22 which has the largest index among all other session members, broadcasts the following session formation request (refer to step 1 in Subsection 8.3.2):

$$Node\,22 \rightarrow Broadcast: \{REQ, Nonce, 3, (2,7,15,22)\}_{Sig_{22}}$$

For simplicity let Nonce equal to 451 and the size of the shares matrix of each node is 24×16. When receiving the request, neighbouring nodes of the session members calculate the indices of the shares to be picked from their shares matrices as follows:

TABLE 8.1: Calculated shares by neighbours of a session member

Session Member	Calculated Shares
Node 2	$Z_2^{(j)} = f_j(2) = s_{57}^{(j)} + s_{31}^{(j)}(2) + s_{42}^{(j)}(2)^2$
Node 7	$Z_7^{(j)} = f_j(7) = s_{57}^{(j)} + s_{31}^{(j)}(7) + s_{42}^{(j)}(7)^2$
Node 15	$Z_{15}^{(j)} = f_j(15) = s_{57}^{(j)} + s_{31}^{(j)}(15) + s_{42}^{(j)}(15)^2$
Node 22	$Z_{22}^{(j)} = f_j(22) = s_{57}^{(j)} + s_{31}^{(j)}(22) + s_{42}^{(j)}(22)^2$

1. Using equation 8.5, calculate rows indices as follows:

$$R_1 = H(2 \mid 7 \mid 15 \mid 22 \mid 451)\, mod(24) = 5$$

$$R_2 = H^2(2 \mid 7 \mid 15 \mid 22 \mid 451)\, mod(24) = 3$$

$$R_3 = H^3(2 \mid 7 \mid 15 \mid 22 \mid 451)\, mod(24) = 4$$

2. Using equation 8.6, calculate column indices as follows:

$$C_1 = H(451)\, mod(16) = 7$$

$$C_2 = H^2(451)\, mod(16) = 1$$

$$C_3 = H^3(451)\, mod(16) = 2$$

3. According to the indices calculation, node u_j which is a neighbour of a session member picks the corresponding shares from its shares matrix as follows: $s_{57}^{(j)}$, $s_{31}^{(j)}$, and $s_{42}^{(j)}$.

4. Each neighbour u_j builds its secret sharing polynomial using the shares he picked in the previous step as follows:

   $f^{(j)}(x) = s_{57}^{(j)} + s_{31}^{(j)}x + s_{42}^{(j)}x^2$. For example, taking equation 8.3 into consideration, s_{57}^{j} refers to the share with row number equal to 5 and column number equal to 7 in the shares matrix of node u_j.

5. Each neighbour u_j calculates a share $Z_i^{(j)}$ for each session member u_i as shown in Table 8.1 and completes steps 6, 7, and 8 in the session key generation process.

8.3.3 Shares Verification

There are two verification methods a session member can perform to prevent malicious nodes from inserting faked shares during the computations of the session keys. The first verification method can be performed as follows:

- Step 1: each session member u_l (where $1 \leq l \leq L$) verifies the shares it received from its neighbouring nodes. Node u_l checks if

$$PV_l^{(j)} = \sum_{i=0}^{t}{}^{\oplus} (Pval_i^{(j)} \odot u_l^{\dot{l}}) \tag{8.17}$$

  Where $Pval_i^{(j)}$ is defined in equation 8.11.

  If the verification fails, u_l broadcasts a complaint against u_j.

- Step 2: each mobile node u_j that received a complaint from another mobile node u_l, broadcasts the values $PV_l^{(j)}$ that satisfy equation 8.17.

- Step 3: mobile node u_l deletes the shares it received from the mobile node u_j if received more that t complaints against u_j in step 1, or if the reply of the mobile node u_j in step 2 does not satisfy equation 8.17.

- Step 4: each session member u_l verifies its session secret key share it received from another node u_j by checking if

$$Z_l^{(j)} \odot G = \sum_{i=0}^{t}{}^{\oplus} (Pval_i^{(j)} \odot u_l^i) \tag{8.18}$$

If the verification fails, u_l broadcasts a complaint against u_j and repeat step 2 and step 3.

The second verification method can be performed as follows: as the session member node u_l collects at least t replies from its neighbouring nodes, it then tries the different t combinations of using $t-1$ out of the received t shares in addition to its own calculation in steps 2 to 6 in Subsection 8.3.2 when calculating the session keys. If all the t combinations are identical, then there is no malicious node among the t replies senders. Otherwise, if there is a combination that results in different session keys, the excluded shares from this combination is considered faked and u_l broadcasts a complain against the mobile node where the faked shares were sent from. This method is useful only if there is no more than one malicious node among the senders of the replies as it can detect one fake reply only.

8.3.4 Shares Matrix Refreshing

Each mobile node protects its shares matrix from being compromised by storing it in an encrypted format in a tamper-resistant storage. However, there is still some risk that a mobile node is compromised and its shares matrix revealed. During the network life time, an adversary may compromise t mobile nodes, hence disclosing the system secrets. In order to prevent these types of attacks, a periodical share refreshing scheme should be performed without changing the system secrets. The share refreshing scheme is based on the following property: when a secret S is distributed to n nodes by using

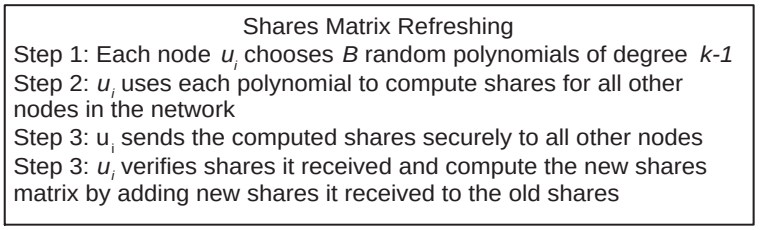

FIGURE 8.6: Shares matrix refreshing

the polynomial

$$f(x) = (S + \sum_{i=1}^{t-1} a_i x^i) \, mod \, p \tag{8.19}$$

Where $f(0) = S$ such that $\{s_1, \cdots, s_n\}$ are shares of the secret S, and another secret $S^{'}$ is distributed to n nodes by using the polynomial

$$f^{'}(x) = (S^{'} + \sum_{i=1}^{t-1} a_i x^i) \, mod \, p \tag{8.20}$$

Where $f^{'}(0) = S^{'}$ such that $\left\{s_1^{'}, \cdots, s_n^{'}\right\}$ are shares of the secret $S^{'}$ then according to the linearity property $\left\{s_1 + s_1^{'}, \cdots, s_n + s_n^{'}\right\}$ are shares of the secret $S + S^{'}$ by using the polynomial $f(x) + f^{'}(x)$, where $f(0) + f^{'}(0) = S + S^{'}$. The aim of our share refreshing scheme is to meet the following:

- Share refreshing process does not reveal any additional information than the key generation process.
- The key management scheme should be available for usage during the share refreshing process.

An overview of the shares refreshing process is shown in Figure 8.6.

For simplicity, the share refreshing process for one element $s_{kh}^{(i)}$ $(1 \leq k \leq m, \, 1 \leq h \leq d)$ in the shares matrix of each node u_i can be described as follows:

- Step 1: each mobile node $u_i (1 \leq i \leq n)$, chooses a random polynomial $\tilde{f}^{(i)}(z)$ of degree $t-1$ as follows:

$$\tilde{f}^{(i)}(z) = \tilde{a}_1^{(i)} z + \cdots + \tilde{a}_{t-1}^{(i)} z^{t-1} \tag{8.21}$$

where $\tilde{a}_0^{(i)} = 0$, and $\tilde{a}_l^{(i)} \in GF(q)$ $(1 \leq l \leq t-1)$.

- Step 2: u_i computes $\hat{s}_j^{(i)} = \hat{f}^{(i)}(u_j) \, mod \, p$, $(j = 1, \cdots, n)$ and sends the following message to u_j:

  $u_i \rightarrow u_j$: $\left\{ Enc_{PK_j} \left[\hat{s}_j^{(i)} \right] \right\}$

- Step 3: u_i computes and broadcasts the following $t-1$ public values:

$$\hat{A}_l^{(i)} = \hat{a}_l^{(i)} \odot G \tag{8.22}$$

  Where $(1 \leq l \leq t-1)$.

- Step 4: each node in the network u_j decrypts the received message from node u_i in Step 2 and verifies the share it received as follows:

$$\hat{s}_j^{(i)} \odot G = \sum_{l=1}^{t-1}{}^{\oplus} (u_j^l \hat{A}_l^{(i)}) \tag{8.23}$$

  If the verification fails, u_j broadcasts a complain against u_i.

- Step 5: each node u_i that received a complaint from another node u_j broadcasts the value $\hat{s}_j^{(i)}$ that satisfy equation 8.23.

- Step 6: node u_i removes node u_j from its trusted group if it received more that t complaints against u_j in step 4, or if the reply of the node u_j in step 5 does not satisfy equation 8.23.

- Step 7: node u_i computes the new share $\tilde{s}_{kh}^{(i)}$ in its shares matrix as follows:

$$\tilde{s}_{kh}^{(i)} = s_{kh}^{(i)} + \sum_{l=1}^{n} \hat{s}_i^{(l)} \tag{8.24}$$

Network nodes repeat these steps to complete the shares refreshing process for all elements in their shares matrices.

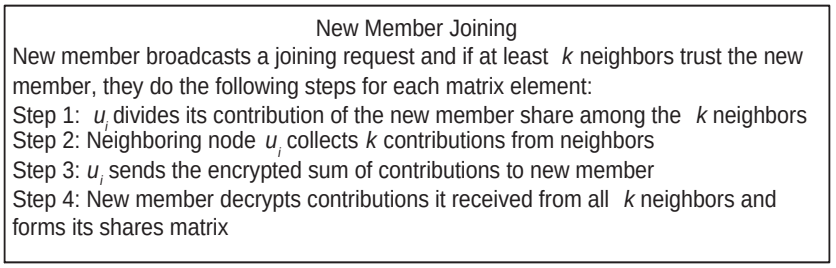

FIGURE 8.7: New Member Joining

8.3.5 New Member Joining

In most ad hoc network applications, the number of nodes is not fixed and the network should be flexible to add new members according to different situations. The existence of the *CA* to issue shares matrix for each new member wishing to join the network after network deployment is not applicable and inconvenient. Although the process of joining a new member to the network is an important issue for a mobile ad hoc network, it should not be at the expense of changing the shares matrices of the old members. Taking into consideration the previous requirements, the process of joining a new member to the mobile ad hoc network in our proposed scheme can be summarised as follows: the new member node u_j broadcasts a joining request contains its identity and the trust evidence to its neighbouring nodes as follows:

$u_j \rightarrow Broadcast : \{u_j, Trust\,Evidence\}$, where *Trust Evidence* is a credential that existing nodes can verify. If a group of at least t old members (We call this group henceforth the issuing group *IG*) believe that the requesting node is trustworthy according to its trust evidence, they cooperate to issue a shares matrix for the requesting node by using the Lagrange interpolation rule on every share of their shares matrices.

An overview of the new member joining process is shown in Figure 8.7

The new member joining process can be summarised as follows:

- Step 1: each node $u_i \in IG$ calculates a contribution $Q_i(j)s_{kh}^{(i)}$ by using the shares in its shares matrix (refer to equation 8.3) Where $1 \leq k \leq m$, $1 \leq h \leq d$ and the

Lagrange coefficient $Q_i(j)$ in this case can be calculated as follows:

$$Q_i(j) = \prod_{\substack{l=1 \\ l \neq i}}^{t} \frac{j-l}{i-l} \tag{8.25}$$

- Step 2: node $u_i \in IG$ divides randomly its contribution $(Q_i(j)s_{kh}^{(i)} \; mod \; p)$ among the t members of the IG group and sends an encrypted message includes β_{il} (where $l = 1, \cdots, t$) to each IG member such that:

$$\sum_{l=1}^{t} \beta_{il} = Q_i(j)s_{kh}^{(i)} \; mod \; p \tag{8.26}$$

- Step 3: node u_i broadcasts the public value $(Q_i(j)s_{kh}^{(i)}) \odot G$.

- Step 4: node $u_i \in IG$ collects $t-1$ contributions from the other members of the IG group in addition to its own contribution, encrypts them with its private key and sends the following message to the new member:

$$u_i \rightarrow u_j : \left\{ Enc_{PK_i} \left[\sum_{l=1}^{t} \beta_{li} \right] \right\}$$

- Step 5: the new member node u_j collects and decrypts the received t messages from the IG group to construct its share $s_{kh}^{(j)}$ as follows:

$$s_{kh}^{(j)} = \sum_{i=1}^{t} \sum_{l=1}^{t} \beta_{li} = \sum_{l=1}^{t} \sum_{i=1}^{t} \beta_{li} = \sum_{l=1}^{t} Q_l(j)s_{kh}^{(l)} \; mod \; p \tag{8.27}$$

- Step 6: the new member u_j verifies its share $s_{kh}^{(j)}$.

  The new member u_j checks if

$$s_{kh}^{(j)} \odot G = \sum_{i=1}^{t}{}^{\oplus} (Q_i(j)s_{kh}^{(i)}) \odot G \tag{8.28}$$

  If the verification fails, then some nodes in the IG group are sending incorrect shares. In order to ensure that the shares matrix of the new member are constructed from correct shares only, the new member node requests from each node

$u_i \in IG$ to broadcast the public values $\beta_{il} \odot G$ and checks if

$$\sum_{l=1}^{t}{}^{\oplus} \beta_{il} \odot G = (Q_i(j) s_{kh}^{(i)}) \odot G \tag{8.29}$$

If the verification fails, the new member u_j broadcasts a complaint against u_i.

8.4 Security Analysis

The proposed threshold key management scheme achieves the correctness and secrecy requirements necessary to provide a distributed key generation protocol based on ECDLP as mentioned in [44, 54]. In this section, a security analysis of the proposed scheme in the initialisation phase, and in the network deployment phase is presented.

8.4.1 Initialisation Phase

8.4.1.1 Correctness

The *CA* secret shares S_r (where $1 \leq r \leq B$) are uniformly distributed in $GF(q)$, and the generated shares matrices are uniformly distributed in $GF(q)$. Any subset shares with the same matrix indices from any honest t mobile nodes define the same secret S_r.

8.4.1.2 Secrecy

No subset of less than t mobile nodes can recover a *CA* secret S_r. In order to increase the secrecy of the preloaded shares matrices, each matrix element is generated from a different secret polynomial chosen by the *CA*. The Nonce is used in the session formation request to randomise the process of session keys generation. Therefore, if the same session members wish to generate another session after the expiration of the previous one, the indices of the shares matrices elements (refer to equations 8.5 and 8.6) required

to construct the secret sharing polynomial for each replying node will be different from the previous session which results in completely different session keys.

8.4.2 Network Deployment Phase

8.4.2.1 Correctness

The private session key of each session member is uniformly distributed in $GF(q)$, and the corresponding public key is uniformly distributed in $GF(p)$ since the determination of whether the neighbouring nodes participating in the session keys generation process are honest or not depends on public broadcast information as in step 7 in Subsection 8.3.2.

Theorem 1:

All subset secret shares $Z_l^{(j)}$ provided by any honest t neighbouring nodes u_j $(1 \leq j \leq t)$ define the same secret key sk_l for mobile node u_l $(1 \leq l \leq L)$ if the t neighbouring nodes picked shares with the same indices (R_iC_i) from their shares matrices, where $(1 \leq i \leq k$).

Proof:

Without loss of generality, by rearranging equation 8.12, a group of t neighbouring nodes provides shares to the node u_l to generate its private key of node u_l such that:

$$sk_l = \sum_{j=1}^{t} \prod_{\substack{i=1 \\ i \neq j}}^{t} \frac{i}{i-j} Z_l^{(j)} \ mod\ p \tag{8.30}$$

by substituting equation 8.9 in equation 8.30:

$$sk_l = \sum_{j=1}^{t} \prod_{\substack{i=1 \\ i \neq j}}^{t} \frac{i}{i-j} \left(a_1^{(j)} + a_2^{(j)}(u_l) + \cdots + a_k^{(j)}(u_l)^{k-1} \right) \ mod\ p \tag{8.31}$$

$$= \left(\sum_{j=1}^{t} \prod_{\substack{i=1 \\ i \neq j}}^{t} \frac{i}{i-j} a_1^{(j)} \right) + \left(\sum_{j=1}^{t} \prod_{\substack{i=1 \\ i \neq j}}^{t} \frac{i}{i-j} a_2^{(j)}(u_l) \right) + \cdots$$

$$+ \left(\sum_{j=1}^{t} \prod_{\substack{i=1 \\ i \neq j}}^{t} \frac{i}{i-j} a_k^{(j)}(u_l)^{k-1} \right) \ mod\ p \tag{8.32}$$

by substituting equation 8.8 in equation 8.32:

$$sk_l = \left(\sum_{j=1}^{t} \prod_{\substack{i=1 \\ i \neq j}}^{t} \frac{i}{i-j} s_{R_1C_1}^{(j)} \right) + \left(\sum_{j=1}^{t} \prod_{\substack{i=1 \\ i \neq j}}^{t} \frac{i}{i-j} s_{R_2C_2}^{(j)}(u_l) \right) + \cdots$$

$$+ \left(\sum_{j=1}^{t} \prod_{\substack{i=1 \\ i \neq j}}^{t} \frac{i}{i-j} s_{R_kC_k}^{(j)}(u_l)^{k-1} \right) \ mod\ p \tag{8.33}$$

If all the t nodes picked the shares corresponding to the same indices R_iC_i ($1 \leq i \leq k$) from their shares matrices (which are shares for the same secret as shown in subsection 8.3.1) then by substituting equation 8.4 in equation 8.33 we get:

$$sk_l = S_{R_1C_1} + \sum_{i=2}^{k} S_{R_iC_i}(u_l)^{i-1} \tag{8.34}$$

which can be constructed from the replies of any group of t nodes if they picked shares with the same indices from their shares matrices.

Theorem 2:

All subset public shares $PV_h^{(j)}$ provided by any honest t neighbouring nodes u_j ($1 \leq j \leq t$) define the same public key pk_h for mobile node u_h ($1 \leq h \leq L$) if the t neighbouring nodes picked shares with the same indices (R_iC_i) from their shares matrices, where ($1 \leq i \leq k$).

Proof:

By rearranging equation 8.12, from the replies of any group of t neighbouring nodes, the mobile node u_h can generate its public key pk_h such that:

$$pk_h = \sum_{j=1}^{t}{}^{\oplus} \prod_{\substack{i=1 \\ i\neq j}}^{t} \frac{i}{i-j} PV_h^{(j)} \ mod\ p \tag{8.35}$$

by substituting equation 8.10 in equation 8.35:

$$= \sum_{j=1}^{t}{}^{\oplus} \prod_{\substack{i=1 \\ i\neq j}}^{t} \frac{i}{i-j} \left(Z_h^{(j)} \odot G \right) \ mod\ p \tag{8.36}$$

by substituting equation 8.9 in equation 8.36 and rearranging:

$$= G \odot \left[\sum_{j=1}^{t} \prod_{\substack{i=1 \\ i\neq j}}^{t} \frac{i}{i-j} \left(a_1^{(j)} + a_2^{(j)}(u_h) + \cdots + a_k^{(j)}(u_h)^{k-1} \right) \right] \ mod\ p \tag{8.37}$$

$$= G \odot \left[\left(\sum_{j=1}^{t} \prod_{\substack{i=1 \\ i\neq j}}^{t} \frac{i}{i-j} a_1^{(j)} \right) + \left(\sum_{j=1}^{t} \prod_{\substack{i=1 \\ i\neq j}}^{t} \frac{i}{i-j} a_2^{(j)}(u_h) \right) + \cdots \right.$$

$$\left. + \left(\sum_{j=1}^{t} \prod_{\substack{i=1 \\ i\neq j}}^{t} \frac{i}{i-j} a_k^{(j)}(u_h)^{k-1} \right) \right] \ mod\ p \tag{8.38}$$

by substituting equation 8.8 in equation 8.38:

$$= G \odot \left[\left(\sum_{j=1}^{t} \prod_{\substack{i=1 \\ i\neq j}}^{t} \frac{i}{i-j} s_{R_1C_1}^{(j)} \right) + \left(\sum_{j=1}^{t} \prod_{\substack{i=1 \\ i\neq j}}^{t} \frac{i}{i-j} s_{R_2C_2}^{(j)}(u_h) \right) + \cdots \right.$$

$$\left. + \left(\sum_{j=1}^{t} \prod_{\substack{i=1 \\ i\neq j}}^{t} \frac{i}{i-j} s_{R_kC_k}^{(j)}(u_h)^{k-1} \right) \right] \ mod\ p \tag{8.39}$$

With all the t nodes picking the shares corresponding to the same indices R_iC_i from their shares matrices ($1 \leq i \leq k$), and by substituting equation 8.4 in equation 8.39 we get:

$$pk_h = G \odot \left[S_{R_1C_1} + \sum_{i=2}^{k} S_{R_iC_i}(u_h)^{i-1} \right] \ mod\ p \tag{8.40}$$

$$= G \odot sk_h \; mod \; p \tag{8.41}$$

which can be constructed from the replies of any group of t nodes if they picked shares with the same indices from their shares matrices.

Theorem 3:

Each session member u_l generates the same session public key SS_{PK} from the replies it received from any group of t neighbouring nodes (refer to equation 8.15) if they picked shares with the same indices (R_iC_i) from their shares matrices, where ($1 \leq i \leq k$).

Proof:

Taking equation 8.15 into consideration, this theorem is a direct result of theorem 2.

8.4.2.2 Secrecy

At least k session members need to cooperate in signing a message m, since a partial signature for a message m can be generated only by using a session member's private key sk_i. No subset of less than k session members can recover the session secret key. When a session member u_l receives its private key share $Z_l^{(j)}$, it can verify the received secret share by checking if it satisfies equation 8.18.

8.5 Performance Evaluation

In this Section, the performance evaluation of our proposed threshold key management scheme is presented. We measure the computation complexity of our proposed scheme by presenting its timing results, and we measure its communication overhead for different node mobility.

8.5.1 Computation Overhead

We measure the computation overhead of the proposed scheme by presenting its timing results in comparison with that of Gennaro et al. scheme [44].

8.5.1.1 An Overview of Gennaro et al. scheme

In Gennaro et al. scheme, a group of mobile nodes u_i $(i = 1,2,\cdots,n)$ generate session keys to be used in securing communications between network entities. The scheme includes the following steps:

- Step 1: each mobile node u_i chooses two random polynomials $f_i(z)$, $f_i^{'}(z)$ of degree t as follows:

$$f_i(z) = a_{i0} + a_{i1}z + \cdots + a_{it}z^t \tag{8.42}$$

$$f_i^{'}(z) = b_{i0} + b_{i1}z + \cdots + b_{it}z^t \tag{8.43}$$

  where a_{il}, $b_{il} \in GF(q)$, and $(0 \leq l \leq t)$.

  Let $SK_i = a_{i0} = f_i(0)$ is the private key of the server node u_i.

- Step 2: compute the shares $s_{ij} = f_i(u_j)\, mod\, p$, $s_{ij}^{'}(u_j)\, mod\, p$ for $j = 1,\cdots,n$ and sends s_{ij}, $s_{ij}^{'}$ to u_j using the private channel.

- Step 3: compute and broadcast the following public values:

$$PV_{il} = (a_{il} \odot G) \oplus (b_{il} \odot G^{'}) \tag{8.44}$$

  Where $(0 \leq l \leq t)$, G is the base point, and $G^{'}$ is another point the discrete logarithm of which with respect to G is not known.

- Step 4: each server node u_j verifies the shares it received from other server nodes. u_j checks the following verification for $i = 1, \cdots, n$:

$$(s_{ij} \odot G) \oplus (s'_{ij} \odot G') = \sum_{l=0}^{t} {}^{\oplus} (u_i^l \odot PV_{il}) \tag{8.45}$$

  If the verification fails, u_j broadcasts a complaint against u_i.

- Step 5: each server node u_i that received a complaint from another server node u_j broadcasts the values s_{ij}, s'_{ij} that satisfy equation 8.45.

- Step 6: server node u_i removes server node u_j from its trusted group if it received more that t complaints against u_j in step 4, or if the reply of the server node u_j in step 5 does not satisfy equation 8.45.

- Step 7: each server node u_i computes its public key $PK_i = SK_i \odot G$ and broadcasts PK_i.

- Step 8: each server node u_i receives PK_j $(j = 1, \cdots, n)$ and computes the session public key

$$SS_{PK} = \sum_{j=1}^{n} {}^{\oplus} PK_j \tag{8.46}$$

8.5.1.2 Timing Results

The growing difference in key bit length for equivalent security levels as shown in Table 7.1 accounts for the performance advantages to be obtained from using ECC in key management protocols for symmetric key cryptography. ECC also gives promising results for substituting RSA/DH/DSA in public key cryptographic protocols. One of the major factors that affect the performance of ECDLP-based cryptographic protocols is the domain parameters (refer to Subsection 7.2.3) and the elliptic curve selected for implementation (refer to Subsection 7.2.2). The fundamental operation underlying ECC is point multiplication, which is defined over finite field operations. Elliptic curves are defined over either prime fields $GF(p)$ or binary fields $GF(2^p)$. In environments in which an arithmetic processor is already available, the performance of $GF(p)$ can be improved

so that in some cases it exceeds the performance of $GF(2^p)$. In the performance evaluation of our proposed scheme, we consider only prime fields $GF(p)$ since binary field arithmetic, is insufficiently supported in PARI/GP [165] and would thus lead to lower performance. On a desktop PC with an Intel Core 2 Duo 2.6 GHz processor and 1GB memory, PARI/GP [165] is used to evaluate the performance of our proposed scheme. The prime elliptic curve over $GF(p)$ is defined by the equation

$$y^2 = (x^3 + ax + b) \, mod \, p \tag{8.47}$$

where a and $b \in GF(p)$ for p a prime and satisfy

$$4a^3 + 27b^2 \equiv 0 \, (mod \, p) \tag{8.48}$$

The domain parameters (refer to Subsection 7.2.3) used in our implementation are as follows:

A field size q which defines the underlying finite field $GF(q)$, where $q > 3$ is a prime number; two field elements a and b in $GF(q)$ which define the equation of the elliptic curve $E : y^2 = (x^3 + ax + b) \, mod \, p$; two field elements x_G and y_G in $GF(q)$, which define a point $G = (x_G, y_G)$ of prime order on E ; the order p of the point G (it must be the case that $p > 2^{160}$; and the cofactor h =(number of points on elliptic curve/p). The performance evaluation of the proposed scheme will be given in terms of the two thresholds t and k (out of the total n mobile nodes). The performance of the proposed scheme is evaluated for three different key sizes: 192 bits, 239 bits, and 256 bits [166]. The total number of mobile nodes in the network is set to 30 nodes. Values that remain constant between different scheme runs (for example, the inner parts of the Lagrange coefficients) can be precomputed and are therefore not included in the evaluation.

a) Session Key Generation (SKG) Timing:

Figures 8.8, 8.9, and 8.10 show the session key generation (SKG) timing of our proposed scheme compared with that of Gennaro et al. scheme [44] for key sizes 192 bits, 239 bits, and 256 bits, respectively. The SKG timing is illustrated with respect to the *CA* threshold t and the group threshold k. The SKG timing presented

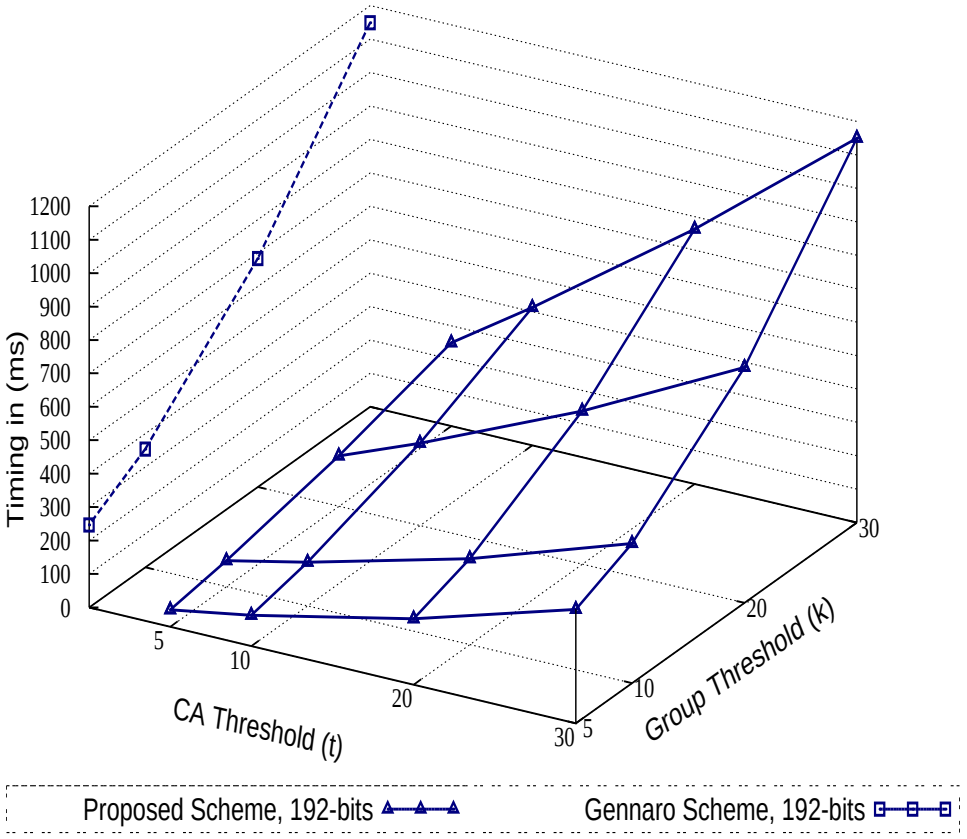

FIGURE 8.8: Session key generation timing in (ms) for key size 192 bits

in these figures are the total time of both the generation and verification processes described in Sections 8.3.2 and 8.3.3, respectively. For a group threshold k equal to 5, the SKG timing of our proposed scheme is 70 to 80% less than that of Gennaro's scheme for the same values of t. For k equal to 10, the SKG timing of the proposed scheme is 60 to 70% less than that of Gennaro's scheme for the same values of t. For k equal to 20, the SKG timing of the proposed scheme is 30 to 40% less than that of Gennaro's scheme.

When k approaches n (the total number of nodes), the session key generation timing of our proposed scheme approaches that of Gennaro's scheme. A comparison of the SKG timing of our proposed scheme to that of Gennaro's scheme is summarised in Table 8.2. The gain in the computation overhead of our proposed scheme compared to Gennaro's scheme comes from the fact that the computation

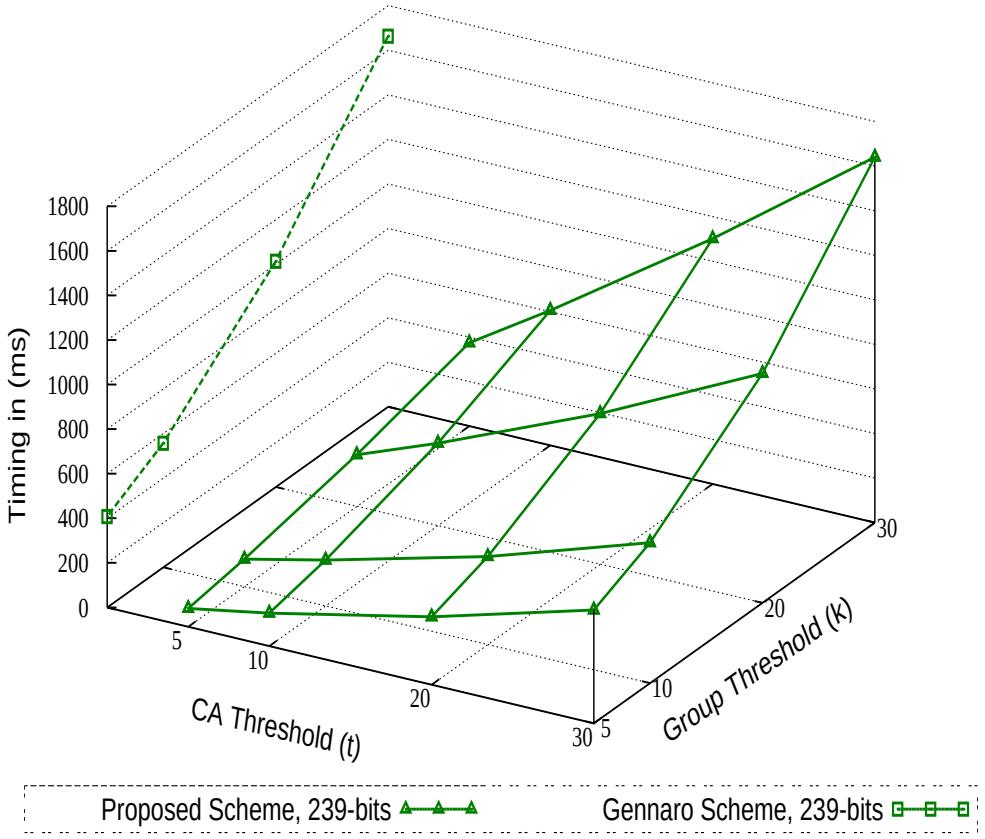

FIGURE 8.9: Session key generation timing in (ms) for key size 239 bits

in Gennaro's scheme depends mainly on the total number of nodes in the session n and the threshold t while in our proposed scheme, the major part of the computation (session members private/public key pairs) depends on the parameter k which determines the threshold number of nodes necessary to form a session of n nodes and can be adjusted accordingly. When k equal to n for the same value of t,

TABLE 8.2: SKG timing comparison

Group threshold (k)	CA threshold (t)	Proposed scheme compared to Gennaro's scheme
5	5~30	80~70% less
10	5~30	70~60% less
20	5~30	40~30% less
30	5~30	Equal timing

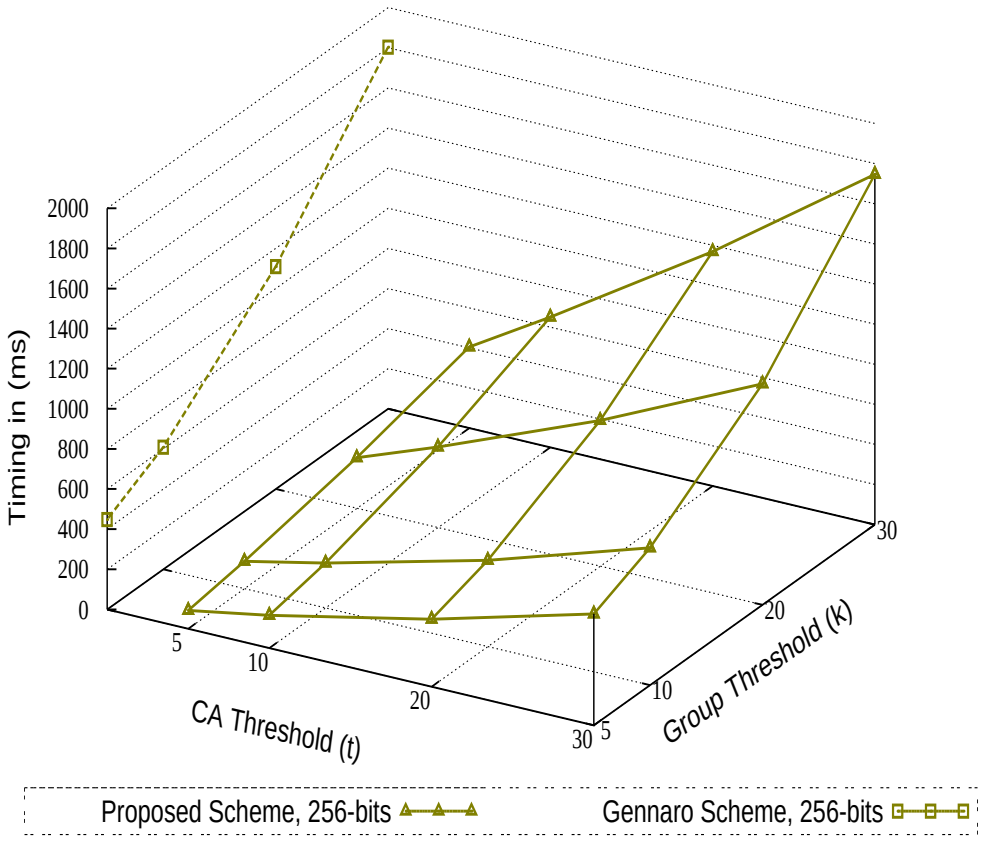

FIGURE 8.10: Session key generation timing in (ms) for key size 256 bits

the timing results of our proposed scheme approaches that of Gennaro's scheme because at this point each session member needs to obtain shares from all nodes in the session (which is the same case as in Gennaro's scheme). The session key generation time increases with increasing the threshold k as a result of increasing the point multiplication operations in the computation of the session public key SS_{PK} as shown in equation 8.15. Increasing the key length from 192 bits to 239 bits and to 256 has a minor effect on the timing of the session key generation process.

b) Shares Matrix Refreshing Timing:

Taking into consideration the example of the shares matrix memory size of 12 k-bytes presented in Subsection 8.3.2 (which can be of a dimension 24×16),

TABLE 8.3: Proposed scheme share refreshing timing in (ms)

CA Threshold (t)	192-bits Curve		239-bits Curve		256-bits Curve	
	Generation	Verification	Generation	Verification	Generation	Verification
5	4642	37380	7672	60879	8586	67912
10	10483	40836	17225	65311	19358	72664
20	22223	47563	36408	74009	47249	81582
30	34586	54971	55540	82717	62206	90633

TABLE 8.4: Gennaro's scheme share refreshing timing in (ms)

CA Threshold (t)	192-bits Curve		239-bits Curve		256-bits Curve	
	Generation	Verification	Generation	Verification	Generation	Verification
5	15	115	25	184	28	203
10	30.90	178.71	49.90	269.20	56.03	292.08
20	59.86	418.48	99.77	594.92	112.02	630.31
30	89.94	811.72	149.68	1129	168.58	1186

TABLE 8.5: Proposed scheme share refreshing timing for one column in (ms)

CA Threshold (t)	192-bits Curve		239-bits Curve		256-bits Curve	
	Generation	Verification	Generation	Verification	Generation	Verification
5	193	1558	320	2537	358	2829
10	437	1701	718	2721	807	3028
20	926	1982	1517	3084	1969	3399
30	1441	2290	2314	3447	2592	3776

Table 8.3 shows the share refreshing generation and verification time of our proposed scheme for a shares matrix of dimension 24×16 for different CA threshold (t). From Table 8.3, we conclude that the share refreshing process is computationally expensive compared to the session key generation process. This results from the fact that all network nodes are involved in the computations of updating shares of a large number of secrets (24×16 secrets in this case) in addition to the heavy

TABLE 8.6: New member joining timing in (ms)

CA Threshold (t)	192-bits Curve		239-bits Curve		256-bits Curve	
	Proposed scheme	Gennaro's scheme	Proposed scheme	Gennaro's scheme	Proposed scheme	Gennaro's scheme
5	121.40	32.64	198.60	53.60	220.80	60.16
10	129.00	77.92	203.00	124.18	225.00	137.78
20	154.43	304.13	233.43	447.80	254.82	482
30	179.68	902	266.30	1278	289.28	1355

computations of verifying each updated share. Table 8.4 presents the share refreshing timing of Gennaro's scheme for different *CA* threshold (t). The share refreshing process in Gennaro's scheme is much less computationally expensive than our proposed scheme. However, in Gennaro's scheme, the share refreshing process updates only one session key, but the share refreshing process in our proposed scheme updates shares that can produce a key space of 10^{14} possible keys. If all elements of the shares matrix are updated at each share refreshing process, the computation overhead will be expensive as shown in Table 8.3. Thus, in our proposed scheme, we suggest that mobile nodes update only one column of their shares matrices in each share refreshing process. The timing results of updating one column of the shares matrix (16 elements) are shown in Table 8.5.

c) New Member Joining Timing:

In Table 8.6, we illustrate the timing results of the new member joining process of our proposed scheme in comparison with that of Gennaro's scheme as presented by Tang et al. in [54]. Results are presented for key sizes 192 bits, 239 bits, and 256 bits. Results show that for small *CA* threshold t (5,10), the new member joining process in our proposed scheme takes more time than that in Gennaro's scheme. With increasing the *CA* threshold t (20,30), the timing results of the new member joining process in Gennaro's scheme exceed the timing results of the same process in our scheme. Table 8.6 shows that the timing of the new member joining process increases slightly with increasing the *CA* threshold t. It shows also that the timing of this process increases slightly with increasing the

key sizes. This reflects the suitability of the proposed scheme to the dynamical nature of MANETs where the number of network nodes changes frequently. The majority of the computations in the new member joining process are consumed in verifying the shares of the shares matrix of the new member. The majority of computations in the verifications of the new member shares matrix are performed in the point multiplication and addition operations in step 5 of Subsection 8.3.5 as illustrated in equations 8.28 and 8.29.

8.5.2 Communication Overhead

In order to evaluate the communication overhead of the proposed scheme, simulations are performed using Network Simulator (NS-2) [56]. The MAC layer protocol IEEE 802.11 is used in all simulations. The source-destination pairs are spread randomly over the network. The NS-2 constant bit-rate (CBR) traffic generator is used to set up the connection patterns with different random seeds. Every simulation run is 1000 seconds long. The mobgenss [128] mobility scenario generator was used to produce random mobility patterns. The pause time is set to zero. The total number of mobile nodes in the simulation is set to 50, while the number of mobile nodes which are trusted by the *CA* and have shares matrices is set to 30 out of the total 50 mobile nodes. The Ad Hoc On-demand Distance Vector (AODV) routing protocol [68] was chosen for the simulations. The share reply packet size is set to 512 bytes. The simulation results are the average of 10 runs. The rest of the simulation parameters are summarised in Table 8.7.

8.5.2.1 Performance Metrics

We have selected the *Session Key Generation Success Ratio* as a metric to evaluate the performance of the proposed scheme.

TABLE 8.7: Simulation parameters

Parameter	Value
No. of Nodes	50
No. of SKG Nodes	30
Area(m^2)	1000x1000
Transmission range	250m
Mobility Model	Random waypoint
Propagation Model	Two-Ray Ground
Mean speeds (m/s)	0.1, 5, 20
Data Rate	11 Mbps
Load	4 packet/s
Share reply packet size	512 bytes

Session Key Generation Success Ratio:

It is the ratio of the number of completed session key generation process to the number of session formation requests sent (refer to step 1 in Subsection 8.3.2).

8.5.2.2 Simulation Results

The communication overhead of our proposed scheme is measured for node mobility 0.1 m/sec, 5 m/sec, and 20 m/sec.

Figure 8.11 shows the session key generation success ratio with respect to the group threshold k and the *CA* threshold t in comparison with that of Gennaro's scheme for mobility 0.1 m/sec. It shows that the session key generation success ratio varies from 98% to 35% with increasing the *CA* threshold t from 5 to 30 but on the other hand the effect of the group threshold k on the session key generation success ratio is negligible. The session key generation success ratio in Gennaro's scheme does not exceed 35% for node mobility 0.1 m/sec.

Figure 8.12 shows that the session key generation success ratio of our proposed scheme for node mobility 5 m/sec decreases from 96% to 21% with increasing the *CA* threshold

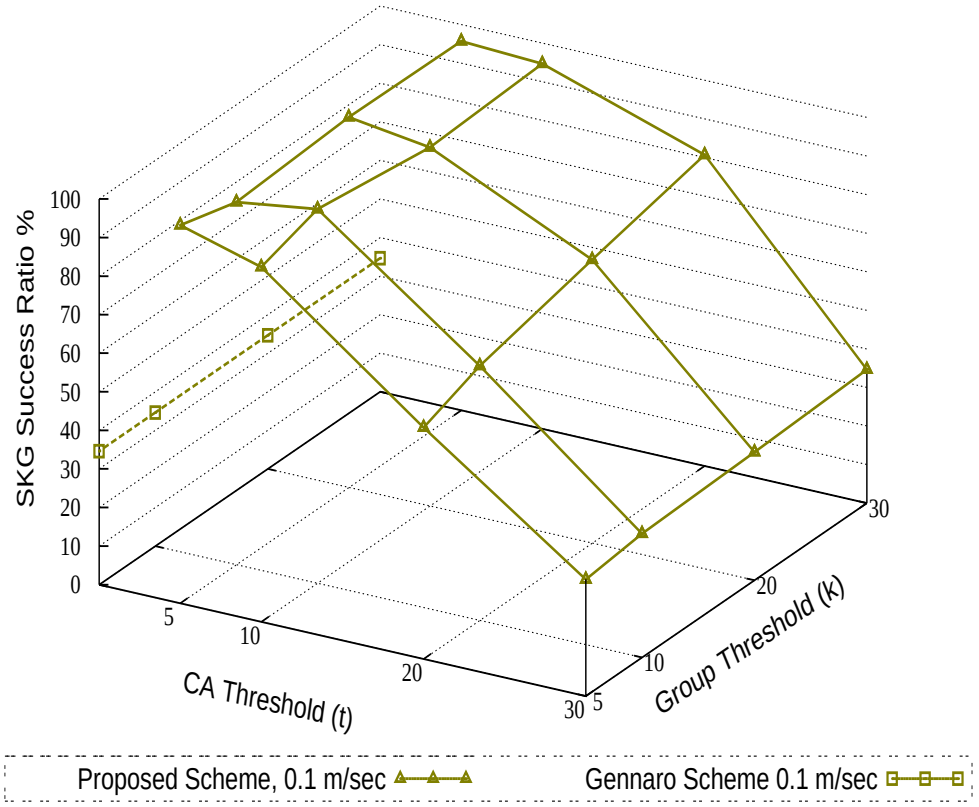

FIGURE 8.11: Session key generation success ratio for 0.1 m/sec mobility

t from 5 to 30. In Gennaro's scheme, the session key generation success ratio for node mobility 5 m/sec is 21%.

Figure 8.13 shows that the session key generation success ratio decreases from 96% to 12% with increasing the *CA* threshold *t* from 5 to 30 for node mobility 20 m/sec. The session key generation success ratio in Gennaro's scheme for node mobility 20 m/sec is 12%. Increasing the CA threshold in Figures 8.11, 8.12, and 8.13 results in decreasing the session key generation success ratio. This is due to the fact that increasing the *CA* threshold *t* increases the number of nodes a session member needs to successfully communicate with in order to get their shares contributions in the session key generation process (refer to step 6 in Subsection 8.3.2). On the other hand the effect of the group threshold *k* on the session key generation success ratio is negligible because increasing the group threshold *k* has no effect on the number of nodes needed to provide their shares

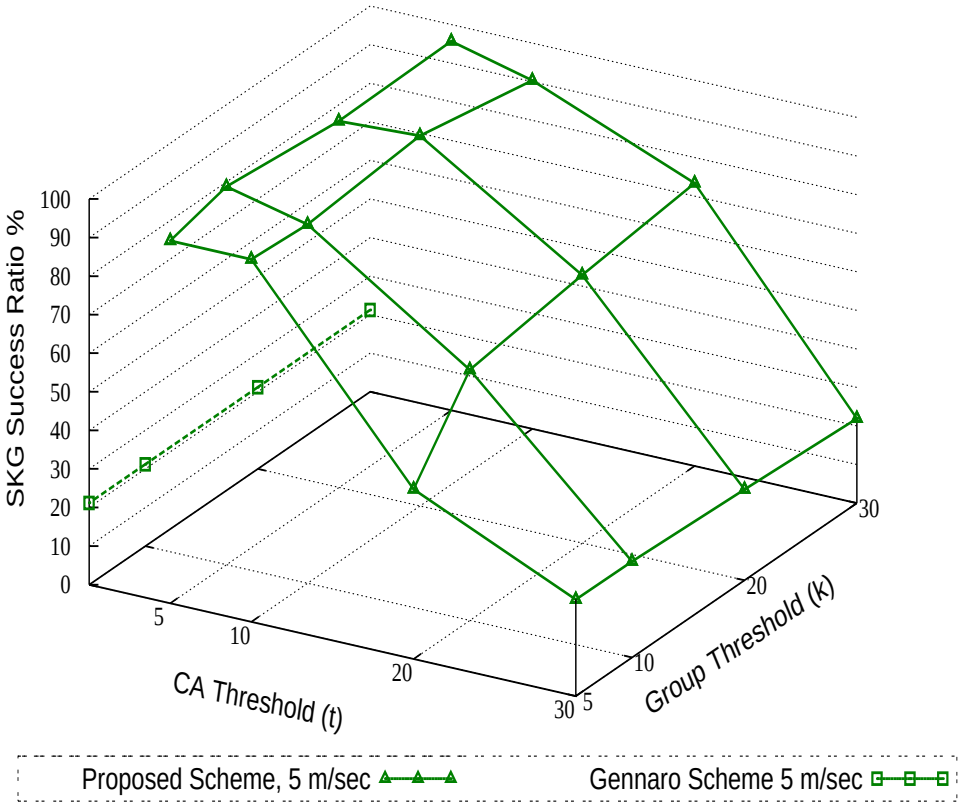

FIGURE 8.12: Session key generation success ratio for 5 m/sec mobility

to a session member in the session key generation process. The session key generation success ratio of our proposed scheme exceeds that of Gennaro's scheme because in our proposed scheme, each session member needs to obtain shares from only t neighbouring nodes while in Gennaro's scheme, each session member needs to obtain shares from all other session members which is often hard to achieve in the mobility environment of MANETs. A comparison of the session key generation success ratio for different node mobility is summarised in Table 8.8.

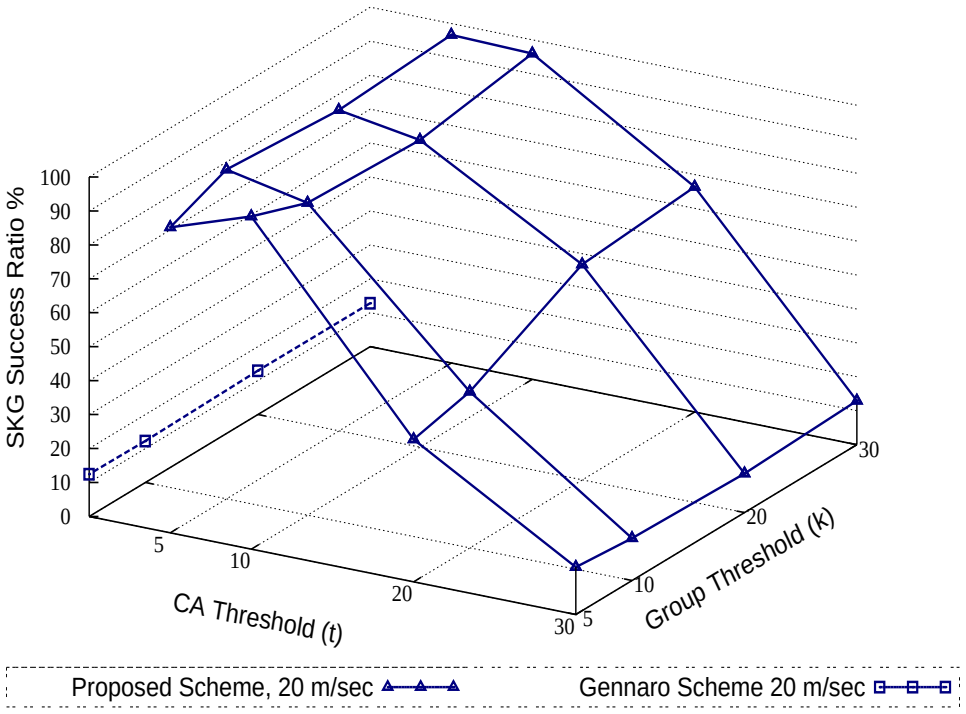

FIGURE 8.13: Session key generation success ratio % for 20 m/sec mobility

TABLE 8.8: A comparison of session key generation success ratio

CA threshold (t)	SKG success ratio		
	0.1 m/sec mobility	5 m/sec mobility	20 m/sec mobility
5	98%	96%	96%
10	92~97%	90~96%	90~94%
20	60~80%	44~73%	42~66%
30	35%	21%	12%

8.6 Summary

In this chapter, a threshold key management scheme using elliptic curve dlog-based cryptosystem has been proposed. The proposed scheme does not require any prior communications between session members before forming the session. In the network deployment phase, nodes are free to select the group threshold k, which is a major advantage of our scheme that enables mobile nodes to adjust the level of secrecy they require.

From the timing results, our proposed scheme has very low timings compared to Gennaro et al. scheme [44] as shown in Tables 8.2, 8.5, and 8.6. Results show also that computations timing in our proposed scheme does not vary significantly with changing the key size which reflects the suitability of the proposed scheme for applications where devices are resource constrained such as in the mobile ad hoc environments. Our proposed scheme has a very low communication overhead compared to that of Gennaro's scheme. As a result, the SKG success ratio of our proposed scheme exceeds that of Gennaro's scheme for different node mobility when the number of *CA* threshold t is less than the number of SKG nodes in the network as shown in Table 8.8. When the number of *CA* threshold t is equal to the number of SKG nodes in the network (30 nodes in Table 8.8), the SKG success ratio of our proposed scheme approaches that of Gennaro's scheme.

Chapter 9

Conclusions and Future Work

Mobile Ad hoc Networks (MANETs) are a collection of wireless mobile nodes that dynamically form a temporary wireless network without relying on any infrastructure or central authority (CA). MANETs have attracted a growing attention among the research community over recent years. This has been motivated by recent advances in wireless technology and mobile computing devices and the enormous applications that could be realised using MANETs. MANETs offer unique benefits and greater flexibility for a range of applications ranging from commercial applications to the high-risk emergency services and military operations. Although the nodes in MANETs share many of the properties of their counterparts in the traditional wired and wireless networks, they present certain unique challenges arising mainly from the absence of fixed infrastructure, capacity constrained operation and random mobility patterns of nodes, which are unique to such networks.

Despite recent research efforts in the development of security algorithms and protocols specifically designed to address the challenge of key management in MANETs, a large research space remains open for further exploration.

In this thesis, a number of novel key management protocols for MANETs have been presented. These key management protocols provide redundancy and robustness for security association (SA) establishment between mobile nodes in MANETs.

9.1 Summary of Contributions

The major focus of this research has been the design and analysis of new key management schemes for MANETs that can provide redundancy and robustness for security association (SA) establishment between pairs of nodes. The proposed schemes can dynamically switch from a centralised scheme of trust distribution to a distributed scheme. The major contributions made in this research study are summarised below.

- *An Analysis of Key Management Schemes and Evaluation Methodology*: A detailed analysis of the most recent and relevant key management protocols for MANETs has been presented in Chapter 3. In this analysis, several key management schemes for MANETs have been analysed in terms of computation and communication overheads. The presented key management schemes have been classified into schemes based on centralised or distributed TTP, self-organised schemes, and a composite key management scheme. These key management schemes have been analysed and the weaknesses in them were identified and avoided in our key management schemes presented in Chapters 4, 5, and 6. The evaluation methodology and models adopted in this thesis to carry out the performance evaluation of the key management schemes with accurate results and conclusions have been discussed in the second part of Chapter 3. A number of simulation-based experiments for Ad Hoc On-Demand Distance Vector Routing (AODV) protocol were presented to validate the model behaviour and assumptions. This work ensured that such an implementation mirrored as far as possible the operation and performance of different models.

- *Authority-Based Key Distribution Scheme*: We presented a novel symmetric key distribution scheme that effectively distributes authenticated symmetric keys between mobile nodes. The proposed scheme distributes symmetric keys between mobile nodes in two steps. In the first step, it distributes certificates during the route request process and in the second step, it disseminates symmetric keys during the route reply process. The proposed scheme breaks the routing-security interdependence cycle [55] because it uses the localised one-hop communication

in distributing symmetric keys, and hence it does not require a routing infrastructure. Simulation results presented in Section 4.4 show that the proposed scheme has a negligible impact on the network performance and can be easily implemented because of its low complexity.

- *Self-Organised Key Management Scheme*: We presented a robust self-organised, public key management scheme for MANETs in Chapter 5. The scheme relies on establishing a small number of trust relations between neighbouring nodes during the network initialisation phase. The proposed scheme exploits the routing infrastructure to discover a certificate chain through a web of trust. Experiences gained as a result of successful communications and node mobility through the network enhance the formation of a web of trust between mobile nodes. The proposed scheme allows each user to create its public key and the corresponding private key, to issue certificates to neighbouring nodes, and to perform public key authentication through at least two independent certificate chains without relying on any centralised authority (CA). A measure of the communications cost of the key distribution process has been presented in Section 5.3.3. The proposed scheme has low communication cost because each node limits its search for the certificate chain to its directly trusted nodes only. A performance comparison of our proposed scheme with Ren et al. scheme [104] for different node mobility was presented in Section 5.5.1. The comparison shows that random graph theory is not suitable for managing trust in the mobility environment of MANET. Random graph theory is suitable for trust management during the network initialisation phase or for stationary networks only. Simulation results presented in Section 5.5 show that the proposed scheme is robust and efficient in the mobility environment of MANET and against malicious nodes attacks.

- *Authority-Based and Self-Organised Key Management*: While central authorities perform the key management process in traditional wired networks, it is better to distribute the burden of the key management process in a MANET among multiple nodes. In Chapter 6, we proposed a robust and redundant key management scheme for MANETs. In the proposed scheme, there are two approaches for key

management. The first approach uses threshold cryptography to establish the distributed central authority among mobile nodes. The second approach is the key management through a web of trust. The first approach represents an authority-based MANET, while the second approach represents a self-organised MANET. A mobile node can use one of the two approaches independently and can also use the two approaches together in order to obtain the authenticated public keys of other nodes in the network. The proposed scheme has been evaluated in Section 6.4 using the network simulator NS-2 [56]. In the performance evaluation of our proposed scheme, we compare the network performance of our proposed scheme (where mobile nodes use both approaches together) with the network performance of the web of trust approach only. Simulation results show the effectiveness of our proposed scheme for different node mobility compared to the web of trust approach. Security analysis show that our proposed scheme is robust against malicious nodes attack. For instance, mobile nodes can certainly perform the public key authentication process if the number of malicious nodes is less than 10% for 0.1 m/sec node mobility, less than 18% for 5 m/sec node mobility, and less than 28% for 20 m/sec node mobility.

- *An Analysis of Elliptic Curve Distributed Key Management Schemes*: We presented in Chapter 7 an overview of the background information and concepts of elliptic curve cryptography. The main types of elliptic curves used for cryptographic applications such as prime curves over $GF(p)$ and binary curves over $GF(2^p)$ have been studied. Furthermore, we presented the attacks on elliptic curve discrete logarithm problem (ECDLP) for both prime and binary curves. Prime curves are best for software applications while binary curves are best for hardware applications [154]. An analyses of the most recent and relevant distributed elliptic curve key management schemes for MANETs has been performed in Section 7.3. Our analyses highlighted the suitability of the analysed schemes to the mobility environment of MANETs in terms of their computation and communication overheads. The analyses show that the ECDKG scheme presented in Section 7.3.1 have a very high communication overhead. The scheme presented

in Section 7.3.2 suffers from the delayed authentication process since the authentication process is performed in a serial manner among the neighbours of the new node. Finally, the scheme presented in Section 7.3.3 does not specify how server nodes can securely send the private key shares to the requesting node. Furthermore, this scheme is vulnerable to the man-in-the-middle attack.

- *Elliptic Curve Distributed Key Management*: Cryptosystems based on elliptic curve discrete logarithm problem (ECDLP) can use smaller key size than that is needed by discrete logarithm problem (DLP) or integer factorisation problem (IFP) based cryptosystems to provide the same level of secrecy. In Chapter 8, we presented a distributed elliptic curve key management scheme for MANETs. The proposed scheme is implemented using elliptic curve dlog-based cryptosystem. In this scheme, an off-line central authority is required in the initialisation phase before network deployment. The *CA* creates and preloads the shares matrix for each mobile node in the network initialisation phase. Each session member generates its private/public key pair, and the session public key by collaborating with its trusted neighbours. In order to generate the required keying material, each session member needs to receive a threshold number of shares from the shares matrices of its neighbours. The generation of the node private/public key pair, and the session public key is performed without any prior communication between session members. In the network deployment phase, nodes are free to select the group threshold k, which is a major advantage of our scheme that enables mobile nodes to adjust the level of secrecy they require. Timing results show that the session key generation (SKG) timing in our proposed scheme is 70 to 80% less than that of Gennaro's scheme [44] when the group threshold k equal to 5 out of 30 nodes ($\simeq$ 16% of the total number of nodes; the average is 10%). Results show also that timing does not vary significantly with changing the key size which reflects the suitability of the proposed scheme for applications where devices are resource constrained such as in the mobile ad hoc environments. The network performance evaluation of our proposed scheme for different node mobility is presented in Section 8.5.2. The performance evaluation shows that, when the *CA* threshold t is 10% of the total number of nodes in the network (5 in a 50

nodes network) which is commonly accepted, the session key generation success ratio in our proposed scheme is 98% for 0.1 m/sec node mobility, 96% for 5 m/sec node mobility, and 96% for 20 m/sec node mobility. In Gennaro's scheme [44], the session key generation success ratio for 0.1 m/sec node mobility is 35%, 21% for 5 m/sec node mobility, and 12% for 20 m/sec node mobility for the same *CA* threshold (t), which reflects the efficiency and robustness of our proposed scheme.

9.2 Future Work

Several security issues and unsolved problems that require further research have emerged in the course of this thesis. In this section we describe some of the possible directions for future work.

- Our proposed key management protocols have been analysed and proved a large number of security properties and the resistance to most common attacks. This could be supplemented by a formal security proof. We believe that the formal security proof is a good choice for formally proving the security of the proposed key management protocols because the design of these protocols rely on security models which have been proved to be secure such as MAC, digital signature and public key encryption [167, 168].

- Key revocation is one of the most important and challenging issues for the key management in MANETs. Conventional techniques for key revocation use certificate revocation lists (CRLs). Due to the features of MANETs such as the absence of an on-line CA and a centralised repository, performing key revocation by a central authority (CA) is hard to achieve. Two kinds of solutions have been proposed for the key revocation in MANETs. The first solution uses threshold cryptography and some network nodes collaborate to revoke keys of malicious nodes, whereas the second solution is fully self-organised, in which each node has its own view about the network and decides whether keys of other nodes should be revoked based on its own observations and the information collected

from peer nodes in the network. The propagation delay of accusation messages in multi-hop neighbourhood should be studied. Furthermore, the total time from observing a malicious behaviour to the actual revocation of a key should be analysed. Some misbehaviour may just happen accidentally and last only for a short time. Therefore, it is more reasonable to keep monitoring the behaviour of the accused node instead of immediately revoking its keying material and excluding it from the network.

- Although the advantages of using public key cryptosystems in securing MANETs, symmetric key cryptosystems are more suitable for MANETs because of their computation efficiency and nodes resources constraints. The presented elliptic curve distributed key management scheme in Chapter 8 can be used as a first step in a key management framework. Based on the security associations (SAs) established between session members in the first step, in the second step, each pair of nodes in a session can cooperatively generate a symmetric key to be used in encrypting and decrypting their exchanged messages.

- Providing authentication for the exchanged messages between distributed nodes in MANETs is challenging. Due to the wireless nature of MANETs, malicious nodes can easily inject fake messages or alter the legitimate messages during multi-hop forwarding. In such cases, MANETs applications need to rely on authentication mechanisms to assert the integrity of the legitimate data messages. Thus, an efficient and robust digital signature mechanism is essential to provide MANETs with the required security services. The off-line/on-line digital signature [169, 170, 171, 172] is a promising solution for the above problems. The basic idea of this solution is to split the signature generation algorithm into two phases: the off-line phase and the on-line phase. For efficient performance, the majority of signature computation is performed off-line independently of the message to be signed. The results of the off-line computation are stored to be used during the on-line phase. The generation of the required signature is performed efficiently during the on-line phase by using the stored computation of the off-line phase. The existing works in this topic suffer from the heavy computation

requirements in the off-line phase in addition to the large size of the generated signature. Thus, more research need to be done in order to minimise the computations overhead and to decrease the signature size, and hence makes the off-line/on-line digital signature suitable and efficient in MANETs environment.

Bibliography

[1] M. Ilyas, ed., *The Handbook of Ad Hoc Wireless Networks*. CRC Press LLC, 2002.

[2] L. Zhou and Z. J. Hass, "Securing Ad Hoc Network," *IEEE network*, vol. 13, no. 6, pp. 24–30, 1999.

[3] M. Puzar, J. Andersson, T. Plagemann, and Y. Roudier, "SKiMPy: A Simple Key Management Protocol for MANETs in Emergency and Rescue Operations," in *Second European Workshop for Security and Privacy in Ad-hoc and Sensor Networks, ESAS 2005*, 2005.

[4] S. Capkun, L. Buttyan, and J.-P. Hubaux, "Self-Organized Public-Key Management for Mobile Ad Hoc Networks," *IEEE Transactions on Mobile Computing*, vol. 2, no. 1, pp. 52 – 64, 2003.

[5] J. H. S. Yi and R. Kravets, "Composite key management for ad hoc networks," in *First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'04)*, pp. 52– 61, 2004.

[6] Certicom.com, "ECC Tutorial." http://www.certicom.com/.

[7] J. Liu, C. Liu, and K. Guo, "A Key Management and Authentication Model for Ad hoc Network," in *IEEE 18th International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC 2007.*, pp. 1–5, September 2007.

[8] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modelling*. New York, NY: Wiley- Interscience, April 1991.

[9] N. Borisov, I. Goldberg, and D. Wagner, "Intercepting Mobile Communications: The Insecurity of 802.11," in *MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking*, (New York, NY, USA), pp. 180–189, ACM, 2001.

[10] "IEEE standard 802.11-1999," September 1999.

[11] "Wireless Application Protocol Forum." http://www. wapforum. org/, 1999.

[12] "IEEE Standard 802.16-2004," October 2004.

[13] L. Buttyan and J.-P. Hubaux, *Security and Cooperation in Wireless Networks*. Cambridge: Cambridge University Press, 2008.

[14] A. Menezes, P. von Orschot, , and S. Vanstone, *Handbook of Applied Cryptography*. CRC press LLC, 1997.

[15] Y.-C. Hu, D. B. Johnson, and A. Perrig, "SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad Hoc Networks," in *IEEE Workshop on Mobile Computing Systems and Applications (WMCSA '02)*, pp. 3–13, 2002.

[16] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Ariadne: a secure on-demand routing protocol for ad hoc networks," *Wirel. Netw.*, vol. 11, no. 1-2, pp. 21–38, 2005.

[17] 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)*, January 2002.

[18] M. G. Zapata, "Secure ad hoc on-demand distance vector routing," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 6, no. 3, pp. 106–107, 2002.

[19] J.-Z. Sun, "Mobile Ad Hoc Networking: An Essential Technology for Pervasive Computing," tech. rep.

[20] H. Yang, H. Luo, F. Ye, S. Lu, and L. Zhang, "Security in mobile ad hoc networks: challenges and solutions," *IEEE Wireless Communications*, vol. 11, no. 1, pp. 38–47, 2004.

[21] 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), SAINT-W '03*, (Washington, DC, USA), IEEE Computer Society, 2003.

[22] A. S. R. L. Rivest and L. M. Adleman., "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Communications of the ACM*, vol. 21, pp. 120–126, February 1978.

[23] US Federal Information Processing Standard, The Keyed-Hash Message Authentication Code (HMAC), *FIPS 198*, March 2002.

[24] US Federal Information Processing Standard, *FIPS 180-3, Secure Hash Standard*, February 2004.

[25] J.-P. Hubaux, L. Buttyan, and S. Capkun, "The Quest for Security in Mobile Ad Hoc Networks," in *ACM Symposium on Mobile Networking and Computing-MobiHOC 2001*, 2001.

[26] M. Carvalho, "Security in Mobile Ad Hoc Networks," *IEEE Security and Privacy*, vol. 6, pp. 72–75, 2008.

[27] B. Leiner, R. Ruther, and A. Sastry, "Goals and Challenges of the DARPA GloMo program," *IEEE Personal Communications*, vol. 3, pp. 34–43, December 1996.

[28] R. Ruppe, S. Griswald, P. Walsh, and R. Martin, "Near Term Digital Radio (NTDR) System," in *IEEE Military Communications Conference (MILCOM)*, pp. 1282–1287, November 1997.

[29] F. Stajano, B. Christianson, B. Crispo, and M. Roe, "The resurrecting duckling-what next," in *8th International Workshop on Security Protocols, Lecture Notes in Computer Science*, Springer-Verlag Berlin Heidelberg, April 2000.

[30] F. Stajano and R. Anderson, "The resurrecting duckling: Security issues for adhoc wireless networks," in *7th International Workshop on Security Protocols, Lecture Notes in Computer Science*, Springer-Verlag Berlin Heidelberg, 1999.

[31] Bluetooth, *Specification of the Bluetooth System*. Version 1.1, Available at https://www.bluetooth.com, February 2001.

[32] D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong, "Talking to strangers: Authentication in ad-hoc wireless networks," in *The ISOC 2002 Network and Distribution Systems Security Symposium*, 2002.

[33] A. Salako, "Authentication in Ad hoc Networking," in *Proceedings of London Communications Symposium 2002*, 2002.

[34] L. Eschenauer and V. D. Gligor, "A Key-Management Scheme for Distributed Sensor Networks," in *The 9th ACM conference on Computer and communications security*, pp. 41–47, 2002.

[35] H. Chan, A. Perrig, and D. Song, "Random Key Predistribution Schemes for Sensor Networks," in *IEEE Symposium on Security and Privacy*, (Berkeley, California), pp. 197–213, May 2003.

[36] W. Du, J. Deng, Y. S. Han, and P. K. Varshney, "A Pairwise Key Predistribution Scheme for Wireless Sensor Networks," in *10th ACM Conference on Computer and Communication Security (CCS 03)*, pp. 42–51, 2003.

[37] D. Liu and P. N. W. Du, "Group-Based Key Pre-Distribution in Wireless Sensor Networks," in *Fourth ACM Workshop Wireless Security (WiSe 05)*, pp. 11–20, 2005.

[38] L. Zhou, J. Ni, and C. V. Ravishankar, "Efficient Key Establishment for Group-Based Wireless Sensor Deployments," in *Fourth ACM Workshop Wireless Security (WiSe 05)*, pp. 1–10, 2005.

[39] A. Shamir, "How to Share a Secret," *Communication of the ACM*, vol. 22, pp. 612–613, 1979.

[40] C. Cachin, "Distributing Trust on the Internet," in *Proceeding of the 2001 International Conference on Dependable Systems and Networks*, pp. 183–192, 2001.

[41] A. T. Chronopoulos, F. Balbi, D. Veljkovic, and N. Kolani, "Implementation of Distributed Key Generation Algorithms using Secure Sockets," in *Third IEEE International Symposium of the Network Computing and Applications*, pp. 393–398, IEEE Computer Society, 2004.

[42] T. P. Pedersen, "A Threshold Cryptosystem without a Trusted Party," *EUROCRYPT'91*, vol. 547, pp. 522–526, 1991.

[43] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Revisiting the Distributed Key Generation for Discrete-Log Based Cryptosystems," in *RSA Security'3*, 2003.

[44] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Secure Distributed Key Generation for Discrete-Log Based Cryptosystems," *Journal of Cryptology*, vol. 20, pp. 51–83, January 2007.

[45] V. S. Miller, "Use of elliptic curves in cryptography," *Lecture notes in computer sciences; Advances in cryptology*, pp. 417–426, 1986.

[46] N. Koblitz, "Elliptic Curve Cryptosystems," *Mathematics of Computation*, vol. 48, pp. 203–209, Jan 1987.

[47] N. Koblitz, "CM-Curves with Good Cryptographic Properties," in *the 11th Annual International Cryptology Conference on Advances in Cryptology*, vol. 576, pp. 279–287, LNCS, Springer-Verlag London, UK, 1991.

[48] N. Koblitz, A. Menezes, and S. Vanstone, "The State of Elliptic Curve Cryptography," *Designs, Codes and Cryptography*, vol. 19, pp. 173–193, March 2000.

[49] K. Lauter, "The Advantages of Elliptic Curve Cryptography for Wireless Security," *IEEE Wireless Communications*, vol. 11, pp. 62–67, Feb 2004.

[50] M. AYDIN and G. AYDIN, "A Survey Of Elliptic Curve Cryptography," *Journal of Electrical & Electronics Engineering, Istanbul University*, vol. 6, no. 2, pp. 211–221, 2006.

[51] National Institute of Standard and Technology (NIST), *Recommendation for Key Management*. Special Publication 800-57 Part 1, March 2007.

[52] J. Jian-wei and L. Jian-hui, "Research on key management scheme for wsn based on elliptic curve," in *First International Conference on Networked Digital Technologies, 2009. NDT '09.*, pp. 536–540, 2009.

[53] S. Z. Huaqun Wang and G. Jiang, "Key Management Based on Elliptic Curve Paillier Scheme in Ad Hoc Networks," in *Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*, 2007.

[54] C. Tang, A. T. Chronopoulos, and C. S. Raghavendra, "Soft-Timeout Distributed Key Generation for Digital Signature based on Elliptic Curve D-log for Low-Power Devices," in *First International Conference on Security and Privacy for Emerging Areas in Communications Networks, SecureComm 2005.*, 2005.

[55] R. B. Bobba, L. Eschenauer, V. Gligor, and W. Arbaugh, "Bootstrapping Security Associations for Routing in Mobile Ad-Hoc Networks," in *IEEE Global Telecommunications Conference*, 2003.

[56] K. Fall and K. Vardhan, "The Network Simulator (NS-2)," Available at: http://www.isi.edu/nsnam/ns.

[57] V. Garg, *Wireless Communications and Networking*. Elsevier, 2007.

[58] S. Loyka and A. Kouki, "Using two ray multipath model for microwave link budget analysis," *IEEE Antennas and Propagation Magazine*, vol. 43, pp. 31–36, October 2001.

[59] Institute of Electrical and Electronics Engineers, Standard for Local and metropolitan area networks, *ZigBee, IEEE 802.15.4-2003*, 2003.

[60] European Telecommunications Standards Institute (ETSI), *HiperLAN (High Performance Radio LAN)*, 1996.

[61] G. E. MOORE, "Cramming More Components Onto Integrated Circuits," *Electronics Magazine*, vol. 38, pp. 114–117, April 1965.

[62] G. Gilder, *Telecosm: How Infinite Bandwidth Will Revolutionize Our World*. KurzweilAI.net, February 2001.

[63] J. P. Macker and M. S. Corson, *Mobile ad hoc networks (MANETs): routing technology for dynamic wireless networking*, ch. Chapter 10, pp. 275–300. IEEE Press Wiley, 2003.

[64] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance vector routing (dsdv) for mobile computers," in *Proceedings of ACM SIGCOMM'94*, pp. 234–244, 1994.

[65] V. Park and S. Corson, *Temporally-Ordered Routing Algorithm (TORA) Version 1*. Internet Draft <draft-ietf-manet-tora-spec-04.txt>, June 2001.

[66] T.-W. Chen and M. Gerla, "Global State Routing: A New Routing Scheme for Ad-hoc Wireless Networks," in *Proc. IEEE ICC'98*, pp. 171–175, 1998.

[67] S. Murthy and J. J. Garcia-Luna-Aceves, "An efficient routing protocol for wireless networks," *Mob. Netw. Appl.*, vol. 1, no. 2, pp. 183–197, 1996.

[68] C. E. Perkins and E. M. Royer, "Ad-hoc On-Demand Distance Vector Routing," in *2nd IEEE Workshop on Selected Areas in Communication*, (New Orleans, LA), pp. 24–30, 1999.

[69] D. B. Johnson, D. A. Maltz, and J. Broch, *DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks*. The CMU Monarch Project, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA.

[70] R. Sivakumar, P. Sinha, and V. Bharghavan, "CEDAR: A Core-Extraction Distributed Ad hoc Routing algorithm," *IEEE Journal on Selected Areas in Communications*, vol. 17, pp. 1454–1465, 1999.

[71] R. V. Boppana and S. P. Konduru, "An adaptive distance vector routing algorithm for mobile, ad hoc networks," in *Proceedings of IEEE Twentieth Annual Joint*

Conference of the IEEE Computer and Communications Societies. INFOCOM 2001., pp. 1753–1762, 2001.

[72] Z. Haas, "A New Routing Protocol For The Reconfigurable Wireless Networks," in *Proceedings of IEEE 6th International Conference on Universal Personal Communications 97*, pp. 562–566, 1997.

[73] C. E. Perkins and E. M. Belding-Royer, *Ad hoc ondemand distance vector (AODV) routing*. IETF Mobile Ad Hoc Networking Working Group INTERNET DRAFT, RFC 3561, http://www.ietf.org/rfc/rfc3561.txt, July 2003.

[74] W. Stallings, *Network and Internetwork Security Priciples and Practice*. Prentice Hall, Englewood Cliffs, New Jersey,, 1995.

[75] F. Anjum and P. Mouchtaris, *Security for Wireless Ad Hoc Networks*. John Wiley & Sons, 2007.

[76] C. P. Pfleeger and S. L. Pfleeger, *Security in Computing (4th Edition)*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2006.

[77] R. L. Rivest, *Handbook of Theoretical Computer Science*, ch. 13, pp. 717–755. Elsevier, 1990.

[78] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proceedings of CRYPTO 84 on Advances in cryptology*, (New York, NY, USA), pp. 47–53, Springer-Verlag New York, Inc., 1985.

[79] P. Zimmermann, *The Official PGP User's Guide*. Cambridge, Massachusetts, 1995.

[80] R. Housley, W. Ford, W. Polk, and D. Solo, *RFC 2459. Internet X.509 Public Key Infrastructure Certificate and CRL Profile*, January 1999.

[81] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams, *RFC 2560. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP*, June 1999.

[82] E. Ngai, M. Lyu, and R. Chin, "An Authentication Service Against Dishonest Users in Mobile Ad Hoc Networks," in *IEEE Aerospace Conference Proceedings*, vol. 2, pp. 1275–1285, 2004.

[83] 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.

[84] M. G. Zapata, "Key management and delayed verification for ad hoc networks," *Journal of High Speed Networks*, vol. 15, pp. 93–109, 2006.

[85] B. Aziz, E. Nourdine, and E. K. Mohamed, "A Recent Survey on Key Management Schemes in MANET," in *3rd International Conference on Information and Communication Technologies: From Theory to Applications, ICTTA 2008*, 2008.

[86] A. M. Hegland, E. Winjum, S. F. Mjolsnes, C. Rong, O. Kure, and P. Spilling, "A survey of key management in ad hoc networks," *IEEE Communications Surveys and Tutorials*, vol. 8, no. 1-4, pp. 48–66, 2006.

[87] 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, no. 1, p. 1, 2007.

[88] Y. Xiao, V. K. Rayi, B. Sun, X. Du, F. Hu, and M. Galloway, "A survey of key management schemes in wireless sensor networks," *Comput. Commun.*, vol. 30, no. 11-12, pp. 2314–2341, 2007.

[89] Y. Kitada, Y. Arakawa, K. Takemori, A. Watanabe, and I. Sasase., "On demand distributed public key management using routing information for wirelss ad hoc netwoks.," *IEICE Transactions on Information and Systems*, vol. J88 D1, pp. 1571–1583, October 2005.

[90] Y. Kitada, A. Watanabe, K. Takemori, and I. Sasase, "On demand distributed public key management for wireless ad hoc networks," in *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PacRim)*, 2005.

[91] Y. Kitada, A. Watanabe, K. Takemori, , and I. Sasase, "On demand distributed public key management without considering routing tables for wireless ad hoc

networks," in *Asia Pacific Symposium on Information Technology (APSITT)*, pp. 375–381, 2005.

[92] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang, "Providing robust and ubiquitous security support for mobile ad-hoc networks," in *Proceedings of the Ninth International Conference on Network Protocols*, 2001.

[93] C. Ma and R. Cheng, "Key Management Based on Hierarchical Secret Sharing in Ad-Hoc Networks," in *Information Security and Cryptology: Third SKLOIS Conference, Inscrypt 2007*, pp. 182–191, Lecture Notes In Computer Science, Springer-Verlag Berlin Heidelberg, 2007.

[94] J. van der Merwe, D. Dawoud, and S. McDonald, "Key Distribution in Mobile Ad Hoc Networks based on Message Relaying," in *Fourth European Workshop on Security and Privacy in Ad hoc and Sensor Networks (ESAS'07)*, (Cambridge, UK), July 2-3, 2007.

[95] H. Mohri, I. Yasuda, Y. Takata, and H. Seki, "Certificate Chain Discovery in Web of Trust for Ad Hoc Networks," in *Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops*, vol. 2, pp. 479–485, IEEE Computer Society, 2007.

[96] B. Zhu, F. Bao, R. H. Deng, M. S. Kankanhalli, and G. Wang, "Efficient and robust key management for large mobile ad hoc networks," *Comput. Netw.*, vol. 48, no. 4, pp. 657–682, 2005.

[97] S. Yi and R. Kravets, "Key Management for Heterogeneous Ad Hoc Wireless Networks," in *ICNP '02: Proceedings of the 10th IEEE International Conference on Network Protocols*, (Washington, DC, USA), pp. 202–205, IEEE Computer Society, 2002.

[98] S. Yi and R. Kravets, "MOCA: Mobile Certificate Authority for Wireless Ad Hoc Networks," in *Proceedings of the 2nd Annual PKI Research Workshop (PKI 2003)*, 2003.

[99] B. Wu, J. Wu, E. B. Fernandez, M. Ilyas, and S. Magliveras, "Secure and efficient key management in mobile ad hoc networks," *J. Netw. Comput. Appl.*, vol. 30, no. 3, pp. 937–954, 2007.

[100] Y. Desmedt and Y. Frankel, "Threshold cryptosystems," in *Advances in Cryptology-Crypto 1989, LNCS*, vol. 435, pp. 307–315, Springer-Verlag, 1990.

[101] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung, "Proactive Secret Sharing Or: How to Cope With Perpetual Leakage," in *Proceedings of the Conference on Advances in Cryptology (CRYPTO'95)*, pp. 339–352, Springer-Verlag, 1995.

[102] R. Anderson, H. Chan, and A. Perrig, "Key Infection: Smart Trust for Smart Dust," in *Proceedings of the 12th IEEE International Conference on Network Protocols, ICNP '04*, (Washington, DC, USA), pp. 206–215, IEEE Computer Society, 2004.

[103] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Secure Distributed Key Generation for Discrete-Log Based Cryptosystems," in *EUROCRYPT'99*, 1999.

[104] K. Ren, T. Lib, Z. Wanb, F. Baob, R. H. Dengb, and K. Kima, "Highly reliable trust establishment scheme in ad hoc networks," *The International Journal of Computer and Telecommunications Networking, ElSEVIER*, vol. 45, no. 6, pp. 687–699, 2004.

[105] S. Milgram, "The Small World Problem," *Psychology Today*, vol. 61, 1967.

[106] J. Travers, S. Milgram, J. Travers, and S. Milgram, "An Experimental Study of the Small World Problem," *Sociometry*, vol. 32, pp. 425–443, 1969.

[107] K. Watkins, *Discrete Event Simulation in C*. New York, NY, USA: McGraw-Hill, Inc., 1993.

[108] A. M. Law and D. M. Kelton, *Simulation Modeling and Analysis*. McGraw-Hill Higher Education, 1999.

[109] C. Tachtatzis, *Load distribution and energy awareness in MANETs using multipath routing*. PhD thesis, Department of Electrical & Electronic Engineering, University of Strathclyde, Glasgow, UK, April 2008.

[110] M. H. Zafar, *Multipath Routing and Quality of Service Support for Mobile Ad-hoc Networks*. PhD thesis, Department of Electrical & Electronic Engineering, University of Strathclyde, Glasgow, UK, May 2009.

[111] T. R. Andel and A. Yasinac, "On the Credibility of MANET Simulations," *Computer*, vol. 39, no. 7, pp. 48–54, 2006.

[112] I. Stojmenovic, "Simulations in wireless sensor and ad hoc networks: matching and advancing models, metrics, and solutions," *Communications Magazine, IEEE*, vol. 46, no. 12, pp. 102–107, 2008.

[113] S. Kurkowski, T. Camp, and M. Colagrosso, "MANET simulation studies: the incredibles," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 9, no. 4, pp. 50–61, 2005.

[114] *GloMoSim (Global Mobile Information Systems Simulation Library)*. http://pcl.cs.ucla.edu/projects/glomosim/.

[115] *QualNet, Scalable Network Technologies*. http://www.scalable-networks.com/.

[116] *OPNET (Optimized Network Engineering Tool)*. http://www.opnet.com/.

[117] *OMNeT++ (Objective Modular Network Testbed in C++) Community Site*. http://www.omnetpp.org/.

[118] G. DiCaro, "Analysis of simulation environments for mobile ad-hoc networks," tech. rep., Dalle Molle Institute for Artificial Intelligence, Manno, Switzerland, December 2003.

[119] M. Halvardsson and P. Lindberg, "Reliable Group Communication in Military Ad-hoc Network," tech. rep., School of Mathematics and Systems Engineering, Vaxjo University, 2004.

[120] L. Begg, W. Liu, K. Pawlikowski, S. Perera, and H. Sirisena, "Survey of Simulators of Next Generation Networks for Studying Service Availability and Resilience," tech. rep., University of Canterbury, New Zealand, February 2006.

[121] The CMU Monarch Project, "The cmu monarch project's wireless and mobility extensions to ns." http://www.monarch.cs.rice.edu/cmu-ns.html, 1998.

[122] *OTcl - Object Tcl Extensions*.

[123] B. Tuch, "Development of WaveLAN, an ISM Band Wireless LAN," *AT&T Technical Journal*, vol. 72, no. 4, pp. 27–33, 1993.

[124] C. Perkins, *Ad Hoc Networking*. Addison-Wesley, 2001.

[125] M. Musolesi and C. Mascolo, "A community based mobility model for ad hoc network research," in *Proceedings of the 2nd international workshop on Multi-hop ad hoc networks: from theory to reality, REALMAN '06*, (New York, NY, USA), pp. 31–38, ACM, 2006.

[126] T. Camp, J. Boleng, and V. Davies, "A Survey of Mobility Models for Ad Hoc Network Research," *Wireless Communication & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, vol. 2, pp. 483–502, 2002.

[127] G. Lin, G. Noubir, and R. Rajaraman, "Mobility Models for Ad hoc Network Simulation," in *Proceedings of 23rd Conference of the IEEE Communications Society (INFOCOM 2003)*, vol. 1, pp. 454–463, March 2004.

[128] W. Navidi and T. Camp, "Stationary Distributions for the Random Waypoint Mobility Model," *IEEE Transactions on Mobile Computing*, vol. 3, no. 1, pp. 99–108, 2004.

[129] E. Royer and C. Toh, "A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks," *IEEE Personal Communications*, vol. 6, pp. 46–55, April 1999.

[130] S. R. Das, C. E. Perkins, and E. M. Royer, "Performance Comparison of Two On-demand Routing Protocols for Ad Hoc Networks," in *INFOCOM 2000*, (Tel-Aviv, Israel), March 2000.

[131] L. Hogiea, P. Bouvry, and F. Guinand, "An Overview of MANETs Simulation," *Electronic Notes in Theoretical Computer Science*, vol. 150, pp. 81–101, March 2006.

[132] M. K. Nakayama, "Statistical analysis of simulation output," in *WSC '08: Proceedings of the 40th Conference on Winter Simulation*, pp. 62–72, Winter Simulation Conference, 2008.

[133] O. Balci, "Verification validation and accreditation of simulation models," in *WSC '97: Proceedings of the 29th conference on Winter simulation*, (Washington, DC, USA), pp. 135–141, IEEE Computer Society, 1997.

[134] J. S.Carson, "Model verification and validation," in *WSC '02: Proceedings of the 34th conference on Winter simulation*, pp. 52–58, Winter Simulation Conference, 2002.

[135] R. G. Sargent, "Model verification and validation," in *WSC '03: Proceedings of the Winter simulation conference*, pp. 37–48, Winter Simulation Conference, 2003.

[136] J. Broch, D. Maltz, D. Johnson, Y. Hu, and J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols," in *the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pp. 85–97, ACM, Dallas, TX. ACM, October 1998.

[137] D. W. Carman, P. S. Kruus, and B. J. Matt, "Constraints and Approaches for Distributed Sensor Security," tech. rep., NAI Labs, The Security Research Division, September 2000.

[138] B. Neuman and T. Tso, "Kerberos: An Authentication Service for Computer Networks," *IEEE Communication Magazine*, vol. 32, no. 9, pp. 33–38, 1994.

[139] W. Diffie and M. E. Hellman, "New Directions in Cryptography," *IEEE Transaction on Information Theory*, vol. IT-22, no. 6, pp. 644–654, 1976.

[140] C. S. Division, "The NIST Statistical Tests Suite For Randomness." http://csrc.nist.gov/groups/ST/toolkit/rng/documentationsoftware.html.

[141] *OpenSSL cryptography library*. Available online at www.openssl.org.

[142] R. Safavi-Naini and H. Wang, "New Constructions for Multicast Re-keying Schemes using Perfect Hash Families," in *The Seventh ACM Conference on Computer and Communications Security,*, pp. 228–234, 2000.

[143] C. de Morais Cordeiro and D. P. Agrawal, *Ad Hoc And Sensor Networks Theory and Applications*. World Scientific, 2006.

[144] J. HÃ€rri, C. Bonnet, and F. Filali, "Kinetic mobility management applied to vehicular ad hoc network protocols," *Computer Communications*, vol. 31, pp. 2907–2924, July 2008.

[145] H. K. R. Li, J. Li and P. Liu, "Localized Public-Key Management for Mobile Ad Hoc Networks," in *IEEE Global Telecommunications Conference (Globecom)*, pp. 1284–1289, 2004.

[146] U. Maurer, "Modelling a Public-Key Infrastructure," in *Proceedings of the 4th European Symposium on Research in Computer Security*, vol. 1146, (Lecture Notes In Computer Science, Berlin), pp. 325–350, Springer-Verlag, 1996.

[147] J. Spencer, *The Strange Logic of Random Graphs, Algorithms and Combinatorics*. Springer, Berlin, 2002.

[148] J. Frank and C. U. Martel, "Phase Transitions in the Properties of Random Graphs," *Principles and Practice of Constraint Programming (CP-95), Cassis, France*, 1995.

[149] *SKIPJACK and KEA Algorithm Specification*. Available from http://csrc.nist.gov/encryption/skipjack-kea.htm, May 1998.

[150] J. Daemen and V. Rijmen, *The Design of Rijndael: AES - The Advanced Encryption Standard*. Springer, 2002.

[151] J. Silverman, *The arithmetic of elliptic curves*. Springer GTM 106, 1986.

[152] I. F. Blake, G. Seroussi, and N. P. Smart, *Elliptic Curves in Cryptography*. Cambridge University Press, Cambridge, 1999.

[153] I. F. Blake, G. Seroussi, and N. P. Smart, *Advances in Elliptic Curve Cryptography*. Cambridge University Press, 2004.

[154] A. D. Fernandes, "Elliptic curve cryptography," *Dr. Dobb's Journal*, vol. 24, no. 12, pp. 56–62, 1999.

[155] L. C. Washington, *Elliptic Curves Number Theory and Cryptography, Second Edition*. CRC Press, 2008.

[156] I. A. Semaev, "Evaluation of Discrete Logarithms in a Group of p-Torsion Points of an Elliptic Curve in Characteristic p," *Mathematics of Computation*, vol. 67, pp. 353–356, 1998.

[157] N. P. Smart and N. P. Smart, "The Discrete Logarithm Problem On Elliptic Curves of Trace One," *Journal of Cryptology*, vol. 12, pp. 193–196, 1999.

[158] M. Maurer, A. Menezes, and E. Teske, "Analysis of the GHS Weil Descent Attack on the ECDLP over Characteristic Two Finite Fields of Composite Degree," *LMS Journal of Computation and Mathematics*, vol. 5, pp. 127–174, 2001.

[159] A. Menezes and M. Qu, "Analysis of the Weil Descent Attack of Gaudry, Hess and Smart," in *CT-RSA 2001: Proceedings of the 2001 Conference on Topics in Cryptology*, (London, UK), pp. 308–318, Springer-Verlag, 2001.

[160] S. D. Galbraith, "Elliptic Curve Paillier Schemes," *Journal of Cryptology*, vol. 15, pp. 129–138, December 2002.

[161] H. Petersen and P. Horster, "Self-certified keys-Concepts and Applications," in *In Proceedings of the Third Conference on Communication and Multimedia Security*, 1997.

[162] A. Khalili, J. Katz, and W. A. Arbaugh, "Toward Secure Key Distribution in Truly Ad-Hoc Networks," in *SAINT-W '03: Proceedings of the 2003 Symposium on Applications and the Internet Workshops (SAINT'03 Workshops)*, (Washington, DC, USA), p. 342, IEEE Computer Society, 2003.

[163] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch, "Verifiable secret sharing and achieving simultaneity in the presence of faults," in *SFCS '85: Proceedings of the 26th Annual Symposium on Foundations of Computer Science*, (Washington, DC, USA), pp. 383–395, IEEE Computer Society, 1985.

[164] P. Feldman, "A Practical Scheme for Non-interactive Verifiable Secret Sharing," in *28th IEEE Annual Symposium on Foundations of Computer Science*, pp. 427–438, 1987.

[165] The PARI Group, Bordeaux, *PARI/GP, version 2.4.3*, 2008. available from http://pari.math.u-bordeaux.fr.

[166] *ANSI X9.62, Public Key Cryptography For The Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)*.

[167] M. Bellare, R. Canetti, and H. Krawczyk, "A modular approach to the design and analysis of authentication and key exchange protocols (extended abstract)," in *STOC '98: Proceedings of the thirtieth annual ACM symposium on Theory of computing*, (New York, NY, USA), pp. 419–428, ACM, 1998.

[168] R. Canetti and H. Krawczyk, "Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels," in *Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques, EUROCRYPT '01*, (London, UK), pp. 453–474, Springer-Verlag, 2001.

[169] S. Even, O. Goldreich, and S. Micali, "On-line/off-line digital signatures," in *Proceedings on Advances in cryptology : Crypto'89*, pp. 263–275, Springer-Verlag New York, 1989.

[170] S. Even, O. Goldreich, and S. Micali, "On-Line/Off-Line Digital Signatures," *Journal of Cryptology*, vol. 9, no. 1, pp. 35–67, 1996.

[171] A. Shamir and Y. Tauman, "Improved online/offline signature schemes," in *Proceedings of Advances in Cryptology: Crypto '01*, vol. 2139/2001, pp. 355–367, Springer Berlin / Heidelberg, 2001.

[172] C. Crutchfield, D. Molnar, D. Turner, and D. Wagner, "Generic On-line/Off-line Threshold Signatures," in *Public Key Cryptography - PKC 2006, LNCS 3958*, 2006.