Trust- and Clustering-based Authentication Service in MANET

NGAI Cheuk Han

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Master of Philosophy
in
Computer Science and Engineering

June 2004

Abstract of thesis entitled:
Trust- and Clustering-based Authentication Service in MANET
Submitted by NGAI Cheuk Han
for the degree of Master of Philosophy
at The Chinese University of Hong Kong in June 2004

Wireless ad hoc network is a collection of mobile nodes dynamically forming a temporary network without a centralized administration. This kind of network has been applied for both civilian and military purposes. However, security in wireless ad hoc networks is hard to achieve due to the vulnerability of the links, the limited physical protection of the nodes, and the absence of a certification authority or centralized management point. Consequently, novel approaches are necessary to address the security problem and to corporate with the properties of wireless ad hoc network. Similar to other distributed systems, security in wireless ad hoc networks usually relies on the use of different key management mechanisms. Authentication service establishes the valid identities of communicating nodes. The compromise of the authentication service breaks down the whole security system.

In this work, we present a public key authentication service to protect security in the network in the presence of malicious nodes. Nodes originally trust-worthy in the network may be compromised after the attacks. These malicious nodes can harm the authentication service by signing false certificates, so adequate measure is essential to protect the network security. We develop a novel authentication service based on trust and clustering models. It involves a well-defined network model and a trust model. We make use of a clustering formation algorithm and propose some clustering structure maintenance algorithms to keep the network in a balance clustering structure. This structure supports our trust model and allows nodes in the network to monitor and rate each other with an authentication metric. We also propose some security operations, including public key certification, identification of suspicious nodes, and trust value update algorithm. Our authentication service is able to discover and isolate malicious nodes in the network. Finally, we perform security evaluation on the proposed solution through simulations. We simulate the network with malicious

nodes and measure a number of metrics. Comparisons and analyzes are made between our approach and the Pretty Good Privacy with distributed certificate repository to demonstrate the effectiveness of our scheme. In addition, clustering structure formation and maintenance algorithms are implemented to study the network behavior in terms of the node mobility and the balance of clustering structure. Finally, neighbor monitoring and strategies on identification of suspicious nodes are evaluated to illustrate their performance in protecting the network security.

…

Acknowledgement

I would like to thank my supervisor Prof. Michael R. Lyu for his generous guidance and patience given to me in the past two years. His numerous support and encouragement, as well as his inspiring advice are extremely essential and valuable in my research work. Also, I am grateful for his support and advice for my further study.

I am so grateful to Prof. Wong Man Hon and Prof. Liu Jiangchuan for their precious time to serve as my thesis examiners.

I would like to thank all the friends I have made here. It is their friendship and encouragement made my study and life happy. I would also like to show my gratitude to the Department of Computer Science and Engineering, the Chinese University of Hong Kong for the provision of the best equipment and pleasant environment for high quality research.

This work is dedicated to my parents and my sister for their unconditional love and support over the years.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Wireless ad hoc network is a collection of mobile devices forming a network without any supporting infrastructure or prior organization. Nodes in the network should be able to sense and discover with nearby nodes [22]. Due to the limited transmission range of wireless network interfaces, multiple network "hops" may be needed for one node to exchange data with another across the network [13]. There are a number of characteristics in wireless ad-hoc networks, such as the dynamic network topology, roaming of the nodes, limited bandwidth and energy constrain in the network. A crucial difference between ad hoc networks and traditional networks is the lack of central administration or control. This factor leads to a serious problem in network security with the limited physical security on wireless communication. Mobile wireless networks are generally more prone to physical security threats than are fixed-cable nets. The increased possibility of eavesdropping, spoofing, and denial-of-service attacks should be carefully considered [20]. In protecting this vulnerable network from different attacks, the availability of security services is very important [44].

The most common way to protect the network security is done by

encryption and decryption of the messages. Public key cryptography has been recognized as one of the most effective mechanism for providing security service like authentication, digital signature, and encryption. Public key cryptography usually relies on the Certificate Authority (CA) to sign and validate digital certificates. Public key infrastructure (PKI) is deployed in wired network and some infrastructure-based wireless network. Security requirements for CAs are important with an exploration of the wide range of attackers that can be mounted against CAs [45]. Popular network authentication architectures include X.509 standard [37] and Kerberos [46]. Another paper suggests make use of interoperation between many small, independent certificate authorities to build a global-scale public-key infrastructure [25]. However, traditional key distribution schemes are not suitable for wireless ad hoc networks due to its network characteristics. Therefore, new security services are necessary to protect the network security in wireless ad hoc network.

Pretty Good Privacy (PGP) [1, 29] is proposed by following a web-of-trust authentication model. PGP uses digital signatures as its form of introduction. When any user signs for another user's key, he or she becomes an introducer of that key. As this process goes on, a web or trust is established [38]. Its distributed manner in certification is compatible with the characteristics of ad hoc networks. An approach similar to PGP for security in wireless ad hoc networks is proposed in [14, 36]. That paper presents the idea of trust graph and the method of finding a certificate chain from one user to another. However, it assumes that users are honest and do not issue false certificates, though it briefly suggests that this assumption could be relaxed by the introduction of some sort of authentication metric. Although an authentication

metric represents the assurance that a user can obtain the authentic public key of another, it is hard to be estimated accurately. There is still possibility for a node turns from trustable to malicious in a sudden attack. The ability for detecting such misbehavior and the isolation of malicious nodes are important in public key authentication. In this thesis, we provide a secure authentication service that can defend malicious nodes in the network. In addition, we find that it is common to see performance evaluation on new security protocols proposed, but rare to see security evaluation on those works by experiment. Therefore, we carry out a series of simulation to evaluation the security provided by the authentication service we propose. We emulate a network with malicious nodes, which can harm authentication by issuing false certificate. The experiment shows that our authentication service performs well in protecting the authentication even in this hostile environment.

The remaining of this thesis is organized as follows: Section 2 discusses the related work on the current key management systems, clustering techniques and trust valuation methods for ad hoc networks. Section 3 formalizes the system architecture, the network model and the trust model which lay the foundation for our design. In Section 4, we further present the security operations on the public key certification and the update of trust tables. The new solution is evaluated through simulation in Section 5. We fix and vary different parameters in the wireless ad hoc network and estimate its security performance in terms of the successful rate, fail rate, unreachable rate, false-positive error rate, and false-negative error rate. We also study the convergence time, effect of mobility, and make comparison of our security scheme with the PGP approach with distributed certificate repositories. The clustering formation and maintenance algorithms, and various suspi-

cious nodes identification strategies are evaluated. Finally, we conclude the thesis in Section 6.

□ **End of chapter.**

Chapter 2

Background Study

2.1 Mobile Ad Hoc Networks

2.1.1 Definition

Mobile ad hoc network is a collection of wireless mobile nodes dynamically forming a temporary network without the use of any existing network infrastructure or centralized administration. Due to the limited transmission range of wireless network interfaces, multiple network "hops" may be needed for one node to exchange data with another across the network [13].

2.1.2 Characteristics

There are a number of characteristics in mobile ad-hoc networks. One of them is that there are dynamic topologies. Nodes are free to move arbitrarily. Thus, the network topology is typically multi-hop, so may change randomly and rapidly at unpredictable times. Another characteristic is bandwidth-constrained. Wireless links will continue to have significantly lower capacity than their hardwired counterparts. Also, there is energy-constrained in the networks. Some or all of the

nodes in a mobile ad-hoc network may rely on batteries or other exhaustible means for their energy. Finally, there is limited physical security. Mobile wireless networks are generally more prone to physical security threats than are fixed-cable nets. The increased possibility of eavesdropping, spoofing, and denial-of-service attacks should be carefully considered [20].

2.1.3 Applications

Examples of potential practical use of mobile ad-hoc networks may be a group of people with laptop computers at a conference that may wish to exchange files and data without mediation of any additional infrastructure in-between. It may be used in home environment for communication between smart household appliances. Ad-hoc networks are suitable to be used in areas where earthquake or other natural disasters have destroyed communication infrastructures. It perfectly satisfies military needs like battlefield survivability, operation without pre-placed infrastructure and connectivity beyond the line of sight. For monitoring and measuring purposes a large number of small computing devices could be spread over a hostile to form a self-sustained ad-hoc network.

Mobile ad-hoc networks have significant advantages above traditional communication networks. For example, use of ad-hoc networks could increase mobility and flexibility, as ad-hoc networks can be brought up and torn down in very short time. Ad-hoc networks could be more economical in some cases as they eliminate fixed infrastructure costs and reduce power consumption at mobile nodes. They are more robust than conventional wireless networks because of their non-hierarchical distributed control and management mechanisms. Also, radio emis-

sion levels could be kept at low level because of short communication links (node-to-node instead of node to a central base station). This increases spectrum reuse possibility or possibility of using unlicensed bands. Moreover, communication beyond Line Of Sight (LOS) is possible at high frequencies because of multi-hop support in ad-hoc networks.

Despite the mentioned advantages and potential application possibilities, ad-hoc networks are yet far from being deployed on large-scale commercial basis. Some fundamental ad-hoc networking problems remain unsolved or need optimized solutions. Although various routing protocols are suggested and tested for mobile ad-hoc networks, performance metrics like throughput, delay and protocol overhead in relation to successfully transmitted data need better optimization. This optimization would probably also depend on application type and desire to maximize the throughput or minimize the delay. One single protocol will probably not be able to work efficiently across entire range of design parameters and operating conditions. An additional complexity factor in ad-hoc network design is that different layers of the system are highly interdependent.

2.1.4 Standards

IEEE 802.11

IEEE 802.11 is a digital wireless data transmission standard in the 2.4 GHz ISM band aimed at providing a wireless LAN between portable computers and between portable computers and a fixed network infrastructure. This standard defines a physical layer and a MAC layer. The most popular technology is the direct sequence spread spectrum and

can offer a bit rate of up to 11 Mbps in the 2.4 GHz band, and in the future, up to 54Mbps in the 5GHz band. The basic access method in the IEEE 802.11 MAC protocol is the Distributed Coordination Function which is a Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) MAC protocol. However, the 802.11 standard cannot do multi-hop networking as it is. The development of a number of protocols is required. The maximum data rate of IEEE 802.11 is 11Mbps. Its range is 100 meters [32].

Bluetooth

Bluetooth is a digital wireless data transmission standard operating in the 2.4 GHz Industrial, Scientific, and Medicine (ISM) band aimed at providing a short range wireless link between laptops, cellular phones and other devices. In this band are defines 79 different Radio Frequency (RF) channels that are spaced of 1MHz. The main aim of the Bluetooth Specification is to guarantee the interoperability between different applications that may run over different protocol stacks. However, in order to implement a wireless multi-hop network over Bluetooth, either or both a packet switch layer and a circuit switch layer need to be defines on top of the Bluetooth data link layer protocol. The maximum data rate of Bluetooth is 1Mbps. Its range is 10 meters. Bluetooth has lower power consumption than IEEE 802.11. Also, Bluetooth support both voice and data packet types while IEEE 802.11 just support data packet type [32].

2.1.5 Routing Protocols

There are a number of routing protocols have been developed for mobile ad hoc networks. They can be divided into two categories, which

the table-driven protocols and the source-initiated on-demand protocols. DSDV belongs to the table-driven protocols. The most popular protocols nowadays are the AODV and DSR protocols. Both of them belong to the source-initiated on-demand protocols. We will briefly describe DSDV, AODV and DSR protocols in the following sections.

DSDV

DSDV stands for Destination-Sequenced Distance-Vector Routing. It is a table-driven algorithm based on the classical Bellman-Ford routing mechanism. Table-driven routing protocols attempt to maintain consistent, up-to-date routing information from each node to every other node in the network. These protocols require each node to maintain one or more tables to store routing information, and they respond to changes in network topology by propagating updates throughout the network in order to maintain a consistent network view [21].

AODV

AODV stands for Ad Hoc On-Demand Distance Vector Routing. It builds on the DSDV algorithm. It is an improvement on DSDV because it typically minimizes the number of required broadcasts by creating routes on a demand basis, as opposed to maintaining a complete list of routes as in DSDV algorithm. AODV is classified as a pure on-demand route acquisition system, since nodes that are not on a selected path do not maintain routing information or participate in routing table exchanges. The Figure 2.1 shows how the AODV route request and route reply message flow [21].

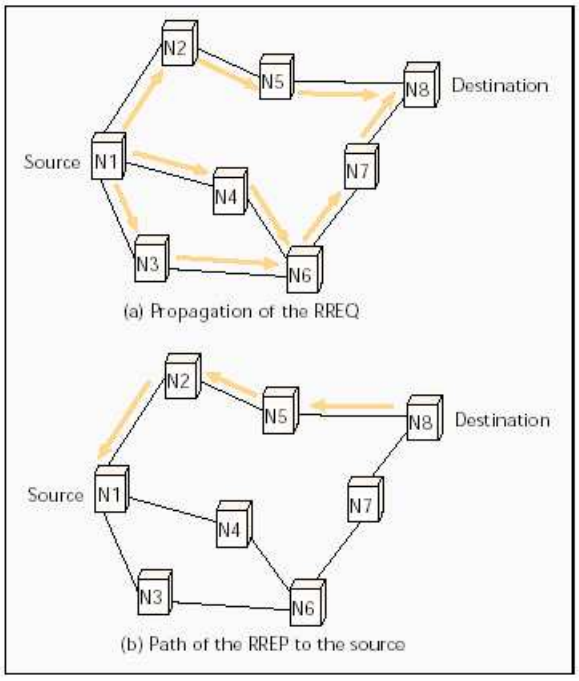

Figure 2.1: The AODV routing protocol

DSR

DSR stands for Dynamic Source Routing. It is an on-demand routing protocol that is based on the concept of source routing Mobile nodes are required to maintain route caches that contain the source routes of which the mobile is aware. Entries in the route cache are continually updated as new routes are learned. The protocol consists of two major phases, which are the route discovery and route maintenance. The route discovery was initiates by broadcasting a route request packet if a node does not have a route to the destination. Route maintenance is accomplished through the use of route error packets and acknowledgements. Route error packets are generated at a node when the data link layer encounters a fatal transmission problem. The Figure 2.2 shows how the DSR route request and route reply message flow [21].

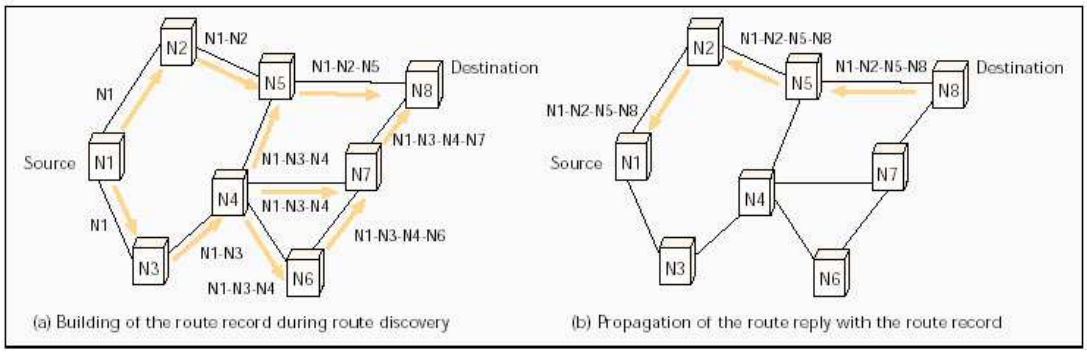

Figure 2.2: The DSR routing protocol

2.2 Security in Mobile Ad Hoc Networks

2.2.1 Vulnerabilities

Due the characteristics of mobile ad hoc networks that we described in the previous section, there are a number of vulnerabilities of the networks. One characteristic is that mobile ad hoc networks have open medium, and lack of clear line of defence. The use of wireless links renders a wireless ad-hoc network susceptible to attacks ranging from passive eavesdropping to active impersonating, message replay, and message distortion. Active attacks might allow the adversary to delete messages, to inject erroneous messages, to modify messages, to impersonate a node, thus violating availability, integrity, authentication, and non-repudiation. All these mean that a wireless ad-hoc network will not have a clear line of defence, and every node must be prepared for encounters with an adversary directly or indirectly.

Another characteristic is that there is dynamic changing topology. Mobile nodes are autonomous units that are capable of roaming in-

dependently. Nodes roaming in a hostile environment with relatively poor physical protection, have non-negligible probability of being compromised. Therefore, not just external attacks should be considered, but attacks launched inside the network by compromised nodes should also be dealt with. It means that nodes with inadequate physical protection are receptive to being captured, compromised, and hijacked. It is easy to attach and hard to detect, so any node in a wireless ad-hoc network must be prepared to operate in a mode that trusts no peer.

Moreover, mobile ad hoc network has decentralized management. There is lack of centralized monitoring and management point. Decision-making in ad-hoc networks is usually decentralized and many ad-hoc network algorithms rely on the cooperative participation of all nodes. Ad hoc network are supposed to operate independently of any fixed infrastructure. This makes the classical security solutions based on certification authorities and on-line servers inapplicable [50].

2.2.2 Motivation for the Attacks

From the above description, it is clear to notice that mobile ad hoc networks are easy to be attacked. However, it may still be interesting to know what is the motivation for attacking the mobile ad hoc networks. Some reason is that is it possible to gain various advantages by malicious behavior. For example, a node can get better service than cooperating nodes, gain monetary benefits by exploiting incentive measures or trading confidential information, save power by selfish behavior, extract data to get confidential information, and so on [11].

2.2.3 Types of Attacks

There are many different types of attacks can be occurred in mobile ad hoc networks. One of them is the passive denial-of-service attacks. Under this kind of attacks, the misbehaving providers simply do not perform the requested function. For example, it may not participate to the Route Discovery phase of the protocol. Another type of attack is the active denial-of-service attacks. Under this kind of attacks, the malicious node prevent other providers from serving a request by communicating bogus information on reputation ratings for legitimate nodes, by performing traffic subversion or by using the security mechanism itself causing explicit Denial of Service. There are many other kinds of attacks. Most common attacks are those against routing and forwarding, such as the no forwarding or incorrect forwarding attacks, setting incorrect metrics on route for priority and remaining time in the cache, frequent route updates, and so on.

2.3 Cryptography

2.3.1 Cryptographic goals

The fundamental goal of cryptography is to address the confidentiality, data integrity, authentication, and non-repudiation in information security. Confidentiality is a services used to keep the content of information from all but those authorized to have it. Data integrity is a service, which addresses the unauthorized alteration of data. Authentication is a service related to identification. Non-repudiation is a service, which prevents an entity from denying previous comments or actions. These services can be used to prevent and detect cheating and other malicious activities [54]. In the following subsections, we

will present some popular cryptographic techniques, like symmetric-key encryption, asymmetric-key encryption, digital signatures, and digital certificates.

2.3.2 Symmetric-key encryption

Symmetric key encryption involves using a single key to encrypt and decrypt data. A plain text message can be encrypted using a shared key to generate the cipher text. The plain text message can be received by decryption the cipher text with the same key. It should be noted that the key for encryption and decryption are the same in symmetric key encryption. Generally speaking, symmetric key encryption is fast and secure. However, the shared key must be distributed over a secure communication channel. The problem is that the physical medium you're sending the packets across is insecure. If it were secure, there would be no reason to encrypt the message in the first place. Anyone who might be monitoring the network could steal the encrypted packets and the key necessary for decrypting them.

2.3.3 Asymmetric-key encryption

Asymmetric-key encryption also called as public key encryption. Unlike asymmetric encryption schemes that involved parties share a common encryption or decryption key, public key encryption depends on tow different but mathematically related keys. The two different keys are a public key that's sent along with the message and a private key that is always in the possession of the recipient. The private key is based on a derivative of the public key and only the two keys working together can decrypt the packets. The public key encryption is more secure because it only requires an authenticated channel as opposed to a secure channel

that is required for the distribution of symmetric encryption keys. The down side of public key encryption is that it tends to be very slow and resource intensive. This makes it difficult to send large amounts of data using public key encryption. It is typically only used to encrypt small amount of data, like digital signatures.

2.3.4 Digital Signatures

A digital signature is an electronic signature that can be used to authenticate the identity of the sender of a message or the signer of a document, and possibly to ensure that the original content of the message or document that has been sent is unchanged. Digital signatures are easily transportable, cannot be imitated by someone else, and can be automatically time-stamped. The purpose of a digital signature is to provide a means for an entity to bind its identity to a piece of information held by the entity into a tag called a signature.

2.3.5 Digital Certificates

Digital certificate is an attachment to an electronic message used for security purposes. The most common use of a digital certificate is to verify that a user sending a message is who he or she claims to be, and to provide the receiver with the means to encode a reply. An individual wishing to send an encrypted message applies for a digital certificate from a Certificate Authority (CA). The CA issues an encrypted digital certificate containing the applicant's public key and a variety of other identification information. The CA makes its own public key readily available through print publicity or perhaps on the Internet. The recipient of an encrypted message uses the CA's public key to decode the digital certificate attached to the message, verifies it as issued by the

CA and then obtains the sender's public key and identification information held within the certificate. With this information, the recipient can send an encrypted reply. The most widely used standard for digital certificates is X.509.

2.3.6 Certificate Authority

As mentioned in the previous sub-section, a certificate authority is a trusted third-party organization or company that issues digital certificates used to create digital signatures and public-private key pairs. The role of the CA in this process is to guarantee that the individual granted the unique certificate is, in fact, who he or she claims to be. Usually, this means that the CA has an arrangement with a financial institution, such as a credit card company, which provides it with information to confirm an individual's claimed identity. CAs are a critical component in data security and electronic commerce because they guarantee that the two parties exchanging information are really who they claim to be. Even though the public-key encryption looks ideal, it is possible for an adversary to defeat the system without breaking the encryption system. For example, an adversary can impersonates a communication by sending an entity an incorrect public key. It can then intercepts encrypted messages, decrypts with its private and re-encrypt the message with the correct public key of the receiver, and send it. This shows that authenticate public keys is necessary even in public-key encryption system. A public-key certificate consists of a data part and a signature part. The data part consists of the name of an entity, the public key corresponding to that entity, validity period, etc. The signature part consists of the signature of a trusted third party over the data part. A trust third party must take appropriate measures to verify the identity

of A and ensure the public key to be certificated actually belongs to A in order to create a public-key certificate for A. In order for an entity B to verify the authenticity of the public key of A, B must have an authentic copy of the public signature verification function of the trust third part. In this way, entity can gain trust in the authenticity of another party's public key by acquiring and verifying the certificate [54].

2.4 Literature Review

Traditional network authentication solutions rely on physically present, trust third-party servers, or called certificate authorities. Popular network authentication architectures include X.509 standard [37] and Kerberos [41]. There is some model on hierarchical CAs and CA delegations [60] have been proposed, but it does not address issues like service availability and robustness. However, ad hoc network is infrastructureless, there is no centralized server for key managements. There is also SPKI is a more flexible and less hierarchical security infrastructure solution [23]. However, it is devised primarily for Internet, and does not meet the requirements of mobile ad hoc network.

Pretty Good Privacy (PGP) [29, 1] is proposed following a web-of-trust authentication model, but it is unable to scale beyond a relatively small community of trust individuals. Also, the members may be unable to reach consensus on who is trusted and who is not, since independent "communities of trust webs" may be formed as a by-product. Another active research area is security function sharing [63, 39, 33, 28], a popular method is using threshold secret sharing [66]. The basic idea is distributing the functionality of the centralized CA server among a fixed group of servers. Proactive secret sharing is proposed to improve

robustness by updating the secret keys periodically [27, 26, 30].

The paper written by Zhou and Hass [73] proposed the partially distributed certificate authority that makes use of a (k, n) threshold scheme to distribute the services of the certificate authority to a set of specialized server nodes. Each of these specialized server node can generate a partial certificate using their share of certificate signing key. A valid certificate can be obtained only be combining k such partial certificates. This approach basically assumes there is rich network connectivity among this small group of server nodes. Also, the server nodes better to have a multicast address because a client node needs to locate any k of the n server nodes for the certificate renewal. It may not be true that ad hoc network support multicast traffic, then the client node needs to broadcast its request and will generate a large amount of network traffic. Similar to the partially distributed CA, the fully distributed certificate authority was proposed by Luo and Lu [52, 47, 53].

The fully distributed certificate authority approach extends the idea of the partially distributed approach by distributing the certificate services to every nodes and a threshold number (k, n) of neighboring nodes can collaboratively act as a server to provide certification services for other nodes. It minimizes the effort and complexity for mobile nodes to locate and contact the service providers in a dynamic multi-hop wireless network. However, this approach assumes that there are k neighbors of every node, which may not be always true. Our scheme is inspired by these proposals, but extends the 1-hop neighboring nodes taking part in certificate renewal to nodes that farther away, such as 2-hop or even 3-hop neighbors. However, the monitoring schemes on ad hoc networks usually can just detect the misbehavior of their 1-hop

neighbors, so we make use of the trust level concept for judging a node trustable or not by calculating the values on the trust chain. Therefore, nodes can decide other nodes, which are 2-hop or farther distance can be trusted or not. This makes it possible for k nodes, not direct neighbors to the requesting node to take part in the certificate renewal. It reduces the problem of not enough neighboring nodes for certificate renewal.

Another public key infrastructure service called MOCA (Mobile Certificate Authority) was proposed. It employs threshold cryptography to distribute the CA functionality over specially selected nodes based on the security and the physical characteristic of nodes [70, 69].

Other solutions include the self-issued certificates proposed by Hubaux et al [36, 15]. It issues certificates by users themselves without the involvement of any certificate authority. In this algorithm, each user can build its own local certificate repositories for storing the certificates that they have issued. Any pair of users can find certificate chains to each other using only their certificate repositories. This solution does not require any form of infrastructure, but it lacks a certificate revocation mechanism. Also, it has problems if the number of certificates issued did not reach certain amount because it is possible that a trust chain does not exist.

Apart from public-key encryption system, distributed key management system based on symmetric encryption is also proposed [9]. This solution is suitable for nodes with low performance that are unable to perform public key encryption. The solution proposed by Balfanz et al [7] presents a mechanism for bootstrapping trust relationship in local ad hoc networks where the network nodes have no prior relationship with each other. However, it requires the nodes to be in short distance

during the bootstrapping phase, so it is unsuitable for distributed ad hoc networks. The paper from Asokan and Ginborg [6] describes a password authenticated group key agreement protocol that is an extension to the Hypercube protocol. It considers a collaborative network where a group of people wish to set up a secure wireless network during a meeting. It assume that a password can be chosen and shared within the room, then this password can be used in the password authenticated hypercube protocol for sharing a strong secret. However, this protocol assumes the participating nodes are arranged in hypercube and it is only suitable for very small ad hoc networks. Another paper [34] overviews several existing Diffie-Hellman based protocols for group key establishment. It found that none of these protocols were found suitable for all types of ad-hoc networks mainly because they demand the network topology to follow a predestined structure.

Some related security solutions for mobile ad hoc networks also include system imprinting, tamper resistance, intrusion detection, mitigation of routing misbehavior. System imprinting is done at node initialization to make a devices know who is its master. A paper presents the resurrecting duckling security policy model [64], which describes secure transient association of a device with multiple serialized owners. A number of papers proposed mechanisms on detecting the misbehavior of nodes. A paper develops a viable intrusion detection system for wireless ad-hoc networks. It proposed that each node is responsible for detecting signs of intrusion locally an independently, but neighboring nodes can collaboratively investigate in a broader range [50]. Another paper presented that trust relationships and routing decisions are made based on experienced, observed, or reported routing and forlwing behavior of other nodes nodes. It proposed new routing protocol extensions to

detect and isolate misbehaving nodes, so make it unattractive to deny cooperation [11]. A similar paper also proposed to install watchdog and pathrater in the network to detect and mitigate routing misbehavior [48]. A paper proposed the components of CONFIDANT, assumed to be present in every node. CONFIDANT consists of the components, which are the monitor, the reputation system, the path manager, and the trust manager [12]. The self-organized feature of the solution is provided through fully localized design. The proposed security solution composes of four components. They are the neighbor verification, security enhanced routing protocol, neighbor monitoring, and intrusion reaction [67]. The papers from Peitro Michiardi and Rdfik Molva pointed out three possible roles that nodes can assume: the requestor, the provider and the peer role. It proposed a security mechanism that solves the problems due to misbehaving nodes. It incorporates a reputation mechanism that provides an automatic method for the social mechanisms of reputation [56, 55].

Recently, there are a number of secure routing protocols proposed. Most of them are built on the existing routing protocols in mobile ad hoc networks, such as the DSDV, DSR and the AODV protocols. A paper proposed a protocol that can be applied to several existing routing protocols. It presented a route discovery protocol that mitigates the detrimental effects of malicious behavior, as to provide correct connectivity information [59]. To deal with external attacks, standard schemes such as digital signatures to protect information authenticity and integrity have been considered. The use of a keyed one-way hash function with a windowed sequence number for data integrity in point-to-point communication and the use of digital signature to protect messages sent to multiple destinations was proposed [73]. A paper

proposed a protocol called Ariadne. Ariadne was built on DSR and TESLA, and relies on efficient symmetric cryptography. It prevents attackers or compromised nodes from tampering with uncompromising routes consisting of uncompromising nodes, and also prevents a large number of types of Denial-of-Service attacks [42]. Another paper proposed SEAD as a secure ad hoc network routing protocol based on the design of the Destination-Sequenced Distance-Vector routing protocol (DSDV). In order to support use with nodes of limited CPU processing capability, and to guard against Denial-of-Service (DoS) attacks in which an attacker attempts to cause other nodes to consume excess network bandwidth or processing time, SEAD uses efficient one-way hash functions and do not use asymmetric cryptographic operations in the protocol. This protocol can be used with any suitable authentication and key distribution schemes, but it is not straightforward [35]. One more paper looks at AODV in detail and develops a security mechanism to protect its routing information. In this paper, it assumes that there is a key management sub-system that makes it possible for each ad hoc node to obtain public keys from the other nodes of the network. Further, each ad hoc node is capable of securely verifying the association between the identity of a given ad hoc node and the public key of that node [44]. A survey paper gives an overview of potential vulnerability and requirement of ad-hoc network, and the proposed prevention, detection and reaction mechanisms for cooperative routing and thwarting attacks [71].

□ **End of chapter.**

Chapter 3

Related Work

In this thesis, we suggest an authentication service that is different from the above protocols. The public key authentication service we propose involves a well-defined trust model and network model. It follows the "web of trust" model proposed in PGP [29] with our own contribution. In addition, it adopts a clustering-based network model in the meantime. One class of existing clustering algorithm in wireless ad hoc network is based on independent dominating sets of graphs. Weighted based clustering algorithms, on the other hand, are proposed in [31]. These algorithms define a vertex with optimal weight within its neighborhood is a clusterhead, and the neighborhood of a clusterhead is a cluster. The above definitions are used as they agree with the network model of our authentication service. In our model, the network is divided into several clusters. Each cluster involves a clusterhead and its neighborhoods. The clusterhead is responsible for managing the join and leave of the cluster members, and the merge and division with other clusters. The weight idea is generalized in [8] such that any meaningful parameter can be used as the weight to best exploit the network properties. Recent work is also performed on clus-

ter formation such that a node is either a clusterhead or is at most d hops away from a clusterhead [5]. Weakly-connected dominating set is proposed for clustering ad hoc networks in [17]. A zonal algorithm for clustering ad hoc networks is proposed in [18] to divide the network into different regions and make adjustments along the borders of the regions to produce a weakly-connected dominating set of the entire graph. Moreover, a Group-based Distance Measurement Service (GDMS) is also proposed. Nodes in GDMS are self-organized into Measurement Groups (Mgroups) to form a hierarchical structure. A set of algorithms is proposed to handle network dynamics and optimize the group organization [51]. Another paper surveys several clustering algorithms based on graph domination [19]. It also describes results that show building clustered hierarchies is affordable and clustering algorithms can be used to build virtual backbones.

Regarding to the authentication in ad hoc network, it generally depends on a trust chain formed by trusted intermediaries. To evaluate the trusts from the recommendation of other reliable entities, the relying node needs to estimate their trustworthiness. It is a well-known technique for authenticating entities in a large-scale system. Some work has extended this technique to include multiple paths to strengthen authentication, but it has to handle intersecting paths, ambiguities in the meaning of certificates, and interdependencies in the use of different keys. A paper develop a set of guiding principles for the design of a satisfactory metric of authentication [61]. Different metrics have been proposed to evaluate the confidence afforded by the paths. A paper proposed a metric that represents a set of trust relationship by a directed graph [10]. It introduces the semantics of direct trust values differ from that of recommendation trust values. It shows that different values can

be combined to a single value by considering the opinions from the respective recommending entities. The metric in PGP has three levels of trust, including the Complete trust, Marginal trust, and Notrust [74]. This approach requires one Completely trusted signature or two Marginally trusted signature to established a key as valid [65]. Another paper explores the use of multiple paths to redundantly authenticate a channel and focuses on two notions of path independence. They are the disjoint paths and connective paths that seem to increase assurance in the authentication [62]. Besides, a trust management method is proposed in [4] to address the problem of reputation-based trust management. It allows assessing trust by computing an agents reputation from its former interactions with other agents and manage data in decentralized way with P-Grid [3]. Moreover, a paper presents a distributed and secure method to compute global trust values, based on Power iteration. This algorithm improve the reputation management in P2P networks [43]. The distributed trust model is proposed based on recommendations [2]. In this model, discrete levels of trust are used and it develops an algorithm for calculating trust and using values in recommendations. Furthermore, a distributed scheme for trust inference in peer-to-peer networks. It describes a technique for efficiently storing user reputation information in a completely decentralized manner, and show how this information can be used to efficiently identify non-cooperative users [49]. Finally, a paper solves the problem of users who claim multiple, false identities, or who possess multiple keys, and whose that conflicting certificate information can be exploited to improve trustworthiness [40].

□ **End of chapter.**

Chapter 4

Architecture and Models

In this section, we describe the architecture, network model, and trust model of the authentication service we propose for wireless ad hoc network.

4.1 Architecture of the Authentication Service

The authentication service we propose aims at providing secure public key certification despite the presence of malicious nodes in the network. Malicious nodes in authentication may issue false certificates to the others. To deal with the problem, we propose a novel authentication service which is clustering- and trust-based. The reason is that the clustering-based network model gives advantages on the behavior monitoring among the nodes. The monitoring power of the nodes in wireless mobile ad hoc network is usually limited to its neighboring nodes, so nodes in the same cluster have relatively higher monitoring power with their short distances. With this feature, we assume that any node can monitor and obtain public keys of the nodes in the same group accurately unless they are compromised in a sudden attack. Apart from the clustering model, we define trust value as an authenti-

cation metric for indicating the assurance with which a requesting node s can obtain the correct public key of a target node t. The chance for obtaining a correct public key certification increases if the node signing the certificate with high trusts value. Simply a clustering model and trust value are not enough in prohibiting dishonest users because a node with high trust value can still be malicious suddenly when it is attacked. Therefore, we design each public key request on new node with multiple replies, so that conclusion can be made with the majority votes. This operation improves the security for obtaining a correct public key and helps to discover dishonest user in the network. Trust value of the dishonest user will be reduced, so malicious nodes will be isolated in our authentication service.

Figure 4.1 shows the architecture of our authentication service. There are totally 4 layers in this architecture, including the mobile hosts, network model, trust model, and the security operations. Wireless ad hoc network contains large amount of mobile hosts, each with a transmission range that is small relative to the network size. We divide the network into different region and nodes in the same region form a cluster. A cluster, or we call a group, is a connected sub-network usually with a smaller diameter. We define two kinds of trust relationship in the clustered network, including the trust relationship of two nodes within the same group and the trust relationship of two nodes in different groups. The security operations are performed on top of the lower layers. These operations include public key certification and trust value update, which will be presented in Section 4.

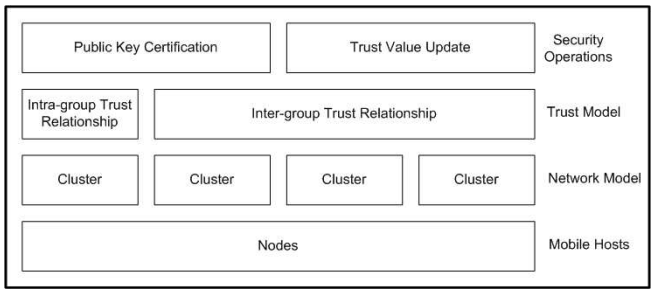

Figure 4.1: Architecture of Our Authentication Service

4.2 The Network Model

Since a wireless ad hoc network is an infrastructureless network that requires only mobile units to form the network, it can involve a large number of mobile units and each with a short transmission range. An important feature in wireless ad hoc networks is multi-hopping, which is the ability of the mobile units to relay packets through radios from one another without the use of base stations. Obtaining a hierarchical organization has been proven effective in minimizing the amount of storage for communication information, and in optimizing the use of network bandwidth.

Apart from the view of efficiency, we believe clustering improves the security of a network. Since wireless ad hoc network lacks of a centralized server for management, security measure relies on individual nodes to monitor each other. However, the monitoring capability of a node is normally limited to its neighboring nodes. On the other hand, nodes clustering together allow the monitoring work to proceed more naturally, so as to improve the overall network security. In this thesis, we propose an authentication service in wireless ad hoc network by trust management and clustering techniques.

A number of existing solutions have been proposed for clustering

in wireless ad hoc networks. In our design, we divide the network into different regions with similar number of hosts in each of them like in Figure 4.2. Nodes clustering together in the same region form a group and are assigned with a unique group ID. The group ID is assigned to be the clusterhead ID in this thesis. However, it is also possible for the clusterhead to suggest a group ID which is agreed by its group member. We adopt the zonal algorithm for clustering ad hoc network [18] in our network model. The zonal distributed algorithm partitions the network into different regions by an asynchronous distributed algorithm for finding minimum spanning tree (MST). It is assumed that the MST algorithm can finish before the nodes being moved around. The execution of the MST algorithm terminates when the size of components in the tree reaches a value x, which is the maximum group size in our network model. Once the network is divided into regions and a spanning tree has been determined for each region. It computes the weakly connected dominating sets of the regions. Finally, it fixes the borders of different regions by including some additional nodes from the borders of the regions. We assume that nodes in the network can know the group, which another belongs to by exchanging messages.

Our work aims at generating the clusters with similar sizes. There is a clusterhead in each cluster for management purpose. Similar sizes among the clusters balance the workload of the clusterheads. It also avoids the existence of small cluster that may not have enough members to provide public key certificates for new nodes. A node in the network is either a clusterhead or at most d' hops away from the clusterhead.

Due to the mobile nature in ad hoc networks, nodes can join, leave, and move in the network freely. The clustering algorithm should be able to adapt to the dynamic network topology. Most of the existing pa-

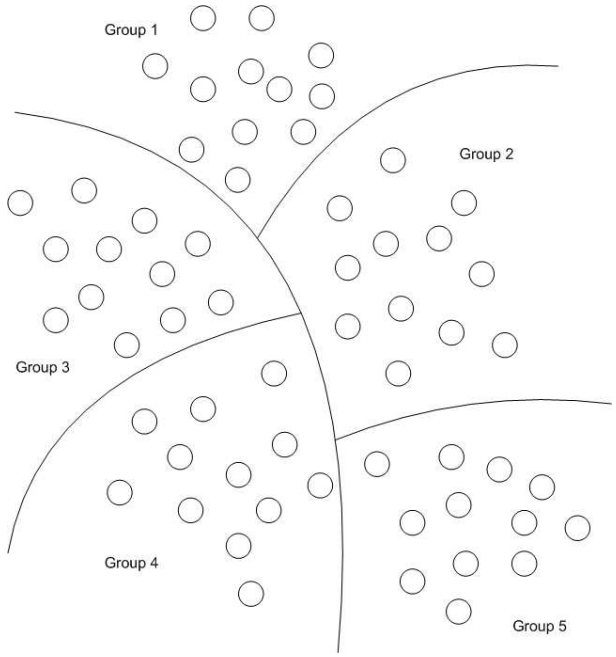

Figure 4.2: The Network Model

pers on clustering focus on the algorithm to create a cluster structures when it is invoked. They are not handling node dynamics after the cluster structures are formed [18]. Some of the current work assumes the network topology remains unchanged throughout the execution of the clustering algorithm [24]. Some of them did not determine the appropriate time to trigger the clustering algorithm [5].

Although re-clustering the entire network is possible, we have to determine the appropriate time to invoke the clustering algorithm. The clustering algorithm can be run either at regular intervals, or whenever the network configuration changes. If re-clustering is carried out on each node-entry or exit, it overloads the network with many clustering messages. The network can hardly be stabilized if the entry and exit of nodes are too frequent. Another approach is invoking the clustering algorithm periodically, but it is hard to determine how long it should

be between two invocations. Moreover, the invocation of the clustering algorithm brings many clustering messages and consumes the computation time and power of the nodes. Handling the node dynamics with appropriate maintaining scheme is important to reduce the network load.

4.2.1 Clustering-Based Structure

Mobile ad hoc network is a collection of mobile nodes with wireless communication. Before building the clustering structures, we assume that a collection of nodes already exist in the network. To build the clustering-based structures, the clustering algorithm is invoked. We adopt the Max-Min D-Cluster Formation approach [33] with several modifications to form the clusters. It ensures a node is either a clustehead or at most d hops away from a clusterhead. Max-Min runs asynchronously eliminating the need and overhead of highly synchronized clocks. The number of messages is a multiple of d rounds. The main difference between our approach and the original Max-Min approach is that we use the trust value, instead of node ID, as the major factor in clustering. In the original approach, node IDs are compared in the Floodmax and Floodmin steps, such that the node with the largest node ID among its members becomes the clusterhead. In our approach, nodes are able to rate the level of trust of its neighboring nodes and store their trust values. The Floodmax and Floodmin steps compare the trust values among the nodes, such that nodes with higher trust value will have higher probability to become clusterheads. If two nodes are having same trust value in the process of Floodmax, the node with higher node ID will be propagated. The Floodmax and Floodmin each runs for d rounds. Each node maintains a winning pair $\langle t, ID \rangle$ for

information exchange. The value t and ID in the winning pair represents the trust value and the ID of the winning node of a particular round.

Our algorithm works as follow:

1. Each node i obtain their trust values $t_{j,i}$ from its neighboring nodes j. Then, node i calculate its trust value by averaging the values $t_{j,i}$ that it received.

$$t_i = \frac{\sum_{j=1}^{n} t_{j,i}}{n}$$

Each node initializes the winning pair to be its trust value and node ID.

2. Each node broadcasts its winning pair $\langle t, ID \rangle$ to its 1-hop neighbors. After all neighboring nodes have been heard from, for a single round, the node chooses the pair with largest trust value as the new winning pair. If there are more than one pairs of value having equal highest trust value, the pair with highest node ID will be selected. This FloodMax mechanism lasts for d rounds.

3. The FloodMin is similar to FloodMax, but each node chooses the pair with smallest trust value as the new winning pair. This FloodMin mechanism also lasts for d rounds.

4. Nodes who received its own node ID in the floodmin stage declare itself a clusterhead. Other nodes join the clusterhead whose node id occurs at least once as a winning pair in both the Floodmax and Floodmin rounds of flooding.

The cluster formation has four logical stages: Firstly, each node collects its trust values from the view of its neighboring nodes and calculates its trust value. Secondly, the FloodMax runs for d rounds. Thirdly, the FloodMin runs for another d rounds. Finally, nodes are selected as clusterheads or join the corresponding clusters as members.

This algorithm relies on individual node to calculate its trust value based on the information from its neighbors. Malicious node is able to broadcast its trust and id pair with an incorrect trust value. To make the mechanism more secure, a node can broadcast the trust values to its neighboring nodes to its 2-hop neighbors. This allows a node to collect more information on the trust values of its neighboring nodes. When it receives the 1st FloodMax message from its neighbor, it can use the trust values from the view of itself and some other nodes to estimate the correctness of that FloodMax message.

4.2.2 Clusterhead Selection Criteria and Role

Clusterhead selection criteria are related to the levels of trust among the nodes. Each of the nodes gives trust value to each of its neighboring nodes in terms of its view. Normally, this value is based on its observation and past experience from one to another. The view of a node is independent to that of another. Different nodes may not have the same trust value to the same node. To sum up the views from all neighboring nodes to a specific node, trust values from neighboring nodes are normalized and averaged to get the resulting trust value.

A clusterhead coordinates the activities in its cluster. It also stores the list of members and keeps exchanging information with other clusterhead. The clusterhead maintains the number of members k to a defined level, where $S \leq k \leq L$. When $k > L$, the clusterhead invoke the division of the cluster. When $k < S$, the clusterhead invoke the merging of it with another cluster. If the clusterhead leaves the current cluster, it will invoke the re-selection of clusterhead. It will pass the member list to the new clusterhead for cluster management. Since the authentication protocol proposed in this report requires informa-

tion exchange among the clusters, each clusterhead has to maintain its member list and pass the information to the clusterhead of other clusters for updating.

Network Maintenance

As mentioned in the above paragraphs, invocation of the clustering algorithm leads to messages that overload the network. We try to reduce the frequency for running the clustering algorithm by introducing some network management techniques. With proper network management, a node can leave from the current cluster and join a new cluster automatically. This avoids the invocation of the clustering algorithm in every change of network topology. The merge and division can maintain the size of cluster within an acceptable level. Re-selection of clusterhead is necessary when the clusterhead is leaving the cluster or is found to be not trust-worthy anymore. Although network maintenance reduces the invocation of cluster algorithm, it is still necessary to run the algorithm at certain moment.

Move to a New Cluster

A clusterhead is responsible for the management of the cluster. It periodically broadcasts "hello" messages for d hops, so all cluster members received the messages. Members will then reply to the clusterhead to confirm their existence in the cluster. If the clusterhead does not receive any reply from a member, it indicates that that cluster member is no longer in the cluster. It will then update its member list.

When a cluster member moves from one cluster to another, it may not know its leaving from the original cluster unless it does not receives the "hello" message from the original clusterhead for a period of time.

If this is the case, the node has to join a new cluster. If it receives "hello" message from a new cluster, then it will know which its new cluster. Otherwise, it has to ask its new neighbors for their clusterhead. Then, it should send a "join" message to the new clusterhead. The clusterheads in the network exchange their lists of memberships periodically, so they keep update about the members each cluster. This information is important for selecting introducing nodes in our trust- and clustering-based authentication protocol.

Cluster Merge and Division

When the number of nodes in a cluster A becomes too small, it can merge with one of its neighboring clusters. A clusterhead who maintains the member list will discover the decrease on the number of members, k. If k is smaller then the threshold S, it will check the member lists of its neighboring clusters. It selects a neighboring cluster B with smallest number of members, k' to consider the possibility on the merge. If $(k + k') \leq L$, it will prepare for the merge. To complete the merge, the clusterhead of cluster A sends a "request merge" message to the clusterhead of cluster B. If clusterhead of cluster B agrees for the merge, the clusterhead of cluster A will send it the member list in cluster A. Clusterhead of cluster B will then update its member list by including the member list of cluster A. Clusterhead of cluster A will also broadcast the new cluster ID to the members in its old clusters. After that, the original clusterhead and members of cluster A will join cluster B.

Similarly, when the number of nodes in a cluster A becomes too large, it can be divided into two clusters. If clusterhead of cluster A finds the number of members $k > L$, it will prepare for the division. In

this case, the clusterhead will broadcast a "division request" message to all its members. Members who receive the request message will reply to the clusterhead. With this mechanism, the clusterhead can find out the number of hops between itself and each member. After collecting this information, it selects one of the cluster members with maximum hops away from itself to be the clusterhead of the newly created cluster. Then, the cluster members decide to join the nearest clusterhead among the two and send "join" message to the corresponding clusterhead. After that, the two clusterheads update their member lists.

Clusterhead Re-selection

Since the clusterhead may move to other clusters or leave the network, the re-selection of clusterhead is necessary in some cases. If a clusterhead knows it would soon leave the cluster or the network, it can invoke the process for re-selection of clusterhead before its leave. In this case, the clusterhead can select a neighboring node with highest trust value to be the new clusterhead. It will then pass the member list to the new clusterhead and inform its members on the result of clusterhead reselection. In some cases, the clusterhead is unable to invoke the reselection of clusterhead, for example, a clusterhead does not notice it has already leave the original cluster while moving, or it is found to be malicious. If this is the case, the node who firstly discovers the leave of the clusterhead invokes the re-selection of clusterhead. In the re-selection of clusterhead, members agree on a new clusterhead with the highest trust values or node ID.

Network re-clustering

Each clusterhead maintains the list of members and their distances to the clusterhead. The number of hops from a member to the clusterhead is at most D hops. With the dynamic nature of the network and the following network operations, the distance from a member to clusterhead d may become greater than D. We define D_t as the threshold on the number of hops that may lead to malformation of the network cluster, where $D_t > d$. A clusterhead records the number of nodes whose $d > D_t$ and distribute this information to other clusterhead. If the number of these cases is higher than the threshold M, then the clustering algorithm will be invoked by any of the clusterhead. Then, the clustering algorithm will be invoked, which is as same as the cluster formation algorithm at the initialization of the network.

4.3 The Trust Model

Authentication in a network usually requires participation of trusted entities. Wireless ad hoc network has no centralized server for trust and key management. We define a fully distributed trust management algorithm to maintain network security. In our trust model, any user can act as a certifying authority. Any node can sign public key certificate of another node in the same group upon request. As mentioned before, we assume a node is able to obtain and store the correct public keys of the same group. Also, a node can observe and give trust value to each of its group members by some monitoring components. We define a trust value as an authentication metric, which represents the assurance with which a requesting node s can obtain the correct public key of a target t. We adopts the fully distributed trust management

approach, such that each node has a trust table for storing the trust values and public keys of the nodes that they know.

In our authentication service, when a node s wants to obtain the public key of another node t. It checks which group node t belongs to. Then, it looks up its trust table to find the first k nodes that belong to the group of node t and with the highest trust values. Node s then selects these k nodes as introducers and sends them request messages on the public key of node t. Introducers are the nodes in the same group of the target node t and are trusted by the requesting node s. To evaluate the trusts from the recommendation of other reliable entities, relying node should be able to estimate their trustworthiness. Many metrics have been proposed to evaluate the confidence afforded by different paths. In our trust model, we define the authentication metric as a continuous value between 0.0 and 1.0. This authentication metric, or we call trust value is assigned and stored by a node to another in a subjective and localized way. A trust value $V_{i,j}$ represents the level of trust from node i to node j. The higher the value represents the more node i trusts node j, and vice versa.

This particular trust model is selected as it allows a distributed management of trust in the network. This property accommodate to the self-organized and fully-distributed nature of mobile ad hoc networks. The authentication metrics formalizes the security levels of the nodes and allow the exchange of trust information among the nodes. The ideas of trust chains and multiple paths will be presented in the following paragraph. Normally, a node is able to observe and directly communicate with its neighboring nodes. Trust-worthy intermediate nodes on a trust path allow a node to obtain the trust information of farther away nodes by its multi-hop nature. The usage of multiple

paths reduce the harms from the incorrect information provided by malicious nodes.

The network model and trust model have certain dependency to each other. In order to determine a good cluster size, the architecture design has to take into account for enough number of introducers and secure results. The network size should not be too small, so it guarantees a requesting node can find enough introducers for public key certification. However, it should not be too large due to the difficulties in managing a large cluster. Managing a large cluster may overload its clusterhead. Also, the monitoring capability and the transmission range is relatively small in mobile ad hoc network, so it is hard to provide security in a large cluster involving a lot of nodes and many hops in communications.

Regarding to our network model, we present two types of trust relationships, including the direct trust relationship and recommendation trust relationship as shown in Figure 4.3. The direct trust relationship is the trust relationship between two nodes in the same group, while the recommendation trust is the trust relationship between nodes of different groups. We apply the formulae for combination of values from the direct trust and recommendation trust approach [10]. From [10], direct trust means to trust an entity directly means to believe in its capabilities with respect to the given trust class. Recommendation trust expresses the belief in the capability of an entity to decide whether another entity is reliable in the given trust class and in its honesty when recommending third entities.

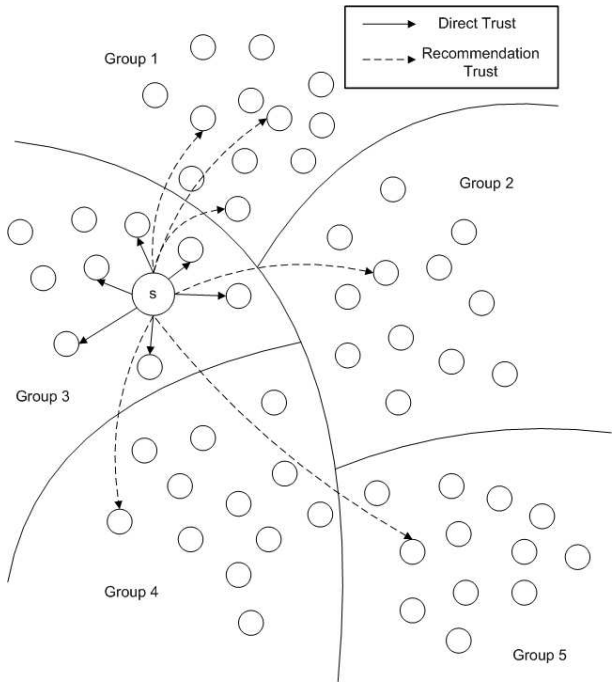

Figure 4.3: The Trust Model

4.3.1 Direct Trust

$$\mathrm{P} \xrightarrow{v_d} \mathrm{Q}$$

A direct trust relationship exists if all trust experiences with Q which P knows about are positive experience. It is a value of the trust relationship which is an estimation of the probability that Q behaves well when being trusted and is based on the number of positive experiences with Q which P knows about. The value v_d of these experiences can be computed by:

$$v_d = 1 - \alpha^p, \tag{4.1}$$

where p is the number of positive experiences which P knows about Q. It is the probability that Q has a reliability of more than α, found on the information P possesses about Q. The reliability is the probability that Q turns out to be reliable when being entrusted with a single

task. α should be chosen reasonably high to ensure sufficiently safe estimations.

4.3.2 Recommendation Trust

$$\mathrm{P} \xrightarrow{v_r} \mathrm{Q}$$

A recommendation trust relationship exists if P is willing to accept reports from Q about experiences with third parties with respect to trust. It represents the portion of offered experiences that P is willing to accept from Q and is based on the experiences P has had with the entities recommended by Q. The recommendation trust value v_r can be computed by:

$$v_r = \begin{cases} 1 - \alpha^{p-n} & if\ p \leq n \\ 0 & else \end{cases} \tag{4.2}$$

The numbers of positive and negative experiences are represented by p and n, respectively. This value can be regarded as a degree of similarity between P and Q, taking into account that different entities may have different experiences with a third party.

4.3.3 Deriving Direct Trust

The first formula computes the trust relationship:

$$V_1 \bigodot V_2 = 1 - (1 - V_2)^{V_1} \tag{4.3}$$

This formula can be used to calculate value of the new recommendation path. It is a result of the computation of the direct trust values and the semantics of the recommendation values. If V_2 is based on p positive experiences, the following equation holds:

$$V_1 \bigodot V_2 = 1 - (1 - (1 - \alpha^p))^{V_1} = 1 - \alpha^{V_1 \cdot p} \tag{4.4}$$

Thus, it is equivalent of having "$p \cdot V_1$" experiences.

In our model, a new recommendation path involve a recommendation trust relationship between a relying node and an introducer, and a direct trust relationship between an introducer and a new node. Based on the above relationships, the formula is appropriate for our occasion.

Combination of Direct Trust

Another formula combines values of direct trust relationships:

$$V_{com} = 1 - \Pi_{i=1}^{m}(\Pi_{j=1}^{n_i}(1 - V_{i,j}))^{\frac{1}{n_i}} \tag{4.5}$$

This formula is used for drawing a consistent conclusion when there are several derived trust relationships of same trust class between two entities. This can be applied in our model as well. It is because a relying node asks for multiple introducers, instead of one for signing public key certificates of a new node.

□ **End of chapter.**

Chapter 5

Trust- and Clustering-Based Authentication Service

This chapter covers the details of the trust- and clustering-based authentication service proposed. It includes the description of the clustering structure formation and maintenance and the security operations provided by the authentication service.

5.1 Clustering Structure Formation and Maintenance

In this section, it discusses how the clustering structure is formed when a collection of mobile nodes are present on a flat surface without any infrastructure. Then, it explains how to adapts to the mobile nature of nodes and keep a balance clustering structure in the network.

5.1.1 Clustering Structure Formation

When a mobile ad hoc network forms, there is only a collection of nodes without any infrastructure. The clustering structure formation

algorithm is then run, so that the network will be divided into several clusters. The clustering structure is a logical structure which provides the basic environment for our trust model and authentication service. It supports two kinds of trust relationships, including the intra-group trust relationship and inter-group trust relationship. The clustering structure is adopted as we believe that nodes in the same clustering can build up stronger trust relationship with the direct monitoring power among the neighboring nodes in mobile ad hoc network.

We adopt the Max-Min D-Cluster Formation algorithm with some modifications. In the original approach, clusters are formed by diffusing only the node ID along the wireless links. When the heuristic terminates, a node either becomes a clusterhead, or is at most d wireless hops away from its clusterhead. Nodes with higher node ID usually have higher chance to be the clusterheads. However, node ID actually does not give special meaning in protecting the network security, so we decide to use trust value, instead of node ID, to be the criteria in cluster formation. In our approach, clusters are formed by diffusing not only the node ID, but also the trust value of itself. The clusterheads are usually found to have high value in comparing with its cluster members.

Algorithm 1 shows the clustering formation algorithm we use in our authentication service. First, the trust value of a node is obtained from its neighboring nodes. Each node will broadcast a request to its neighboring nodes, its neighbors will then reply with the trust value of the requesting node. After collecting all the replies, a node calculates its trust value by averaging the received values. Then, each of the node initialize a winning pair $< WINNER_{ID}, WINNER_{TRUST} >$ as its node ID and trust value. After that, the FloodMax algorithm is

run for d rounds. In each round, each node broadcasts its winning pari to its neighboring nodes. After they have received the messages, each of them chooses the pair with highest trust value as its own winning pair. In the case with same trust values received in the same round, the pair with higher ID will be selected as the winning pair. This process repeats for d rounds. The Floodmin algorithm follows the FloodMax for another d rounds. It is similar to the FloodMax algorithm unless a node chooses the smallest trust value instead of the largest value as the winning pair. Finally, nodes can determine the clusterheads when the Floodmin completes. A node declares itself as cluterhead if its node ID is as same as the $WINNER_{ID}$ in its winning pair. Otherwise, it identifies all the node pairs in the MaxMin algoithm and selects the pair with minimum trust value to be the clusterhead and join it. If it still cannot select a clusterhead, it join the cluster with the maximum trust value in the 1st d rounds of flooding as its clusterhead. A node sends message to its clusterhead to indicate its joining of the cluster.

5.1.2 Network Maintenance

The clusters formed by the clustering formation algorithm are not in balance sizes. The number of nodes in each cluster is not similar to each other. In our trust- and clustering-based authentication service, however, a balance clustering structure benefits to the performance and the security of the network. With similar number of members in the clusters, the clusterheads share almost the same workload to maintain their own clusters. Also, it balance the intra-cluster trust relationship and inter-cluster trust relationship among the nodes in the network. It avoids nodes with too great size, such that nodes in the same clusters do not gain from the neighboring monitoring power due to the large

Algorithm 1 Clustering structure formation

for each node n **do**
  Obtain trust values $t_{neighbor_k,n}$ from its neighboring nodes $neighbor_k$, where $k = 1, \ldots, N$:
  $v_n \xrightarrow{b} v_{neighbor_k} : \langle v_n, REQ_{TRUST} \rangle$;
  $v_{neighbor_k} \rightarrow v_n : \langle v_{neighbor_k}, v_n, t_{neighbor_k,n} \rangle$;
  Calculates its trust value by averaging the values $t_{neighbor_k,n}$ received:

$$t_n = \frac{\sum_{k=1}^{N} t_{neighbor_k,n}}{N} \tag{5.1}$$

  Initializes the winning pair $\langle WINNER_{TRUST}, WINNER_{ID} \rangle$ to be its trust value t_n and node ID;
end for
for each node n **do**
  Broadcasts its winning pair $\langle WINNER_{ID}, WINNER_{TRUST} \rangle$ to its 1-hop neighbors for d-rounds in this Floodmax mechanism:
  for $i = 1$ to d **do**
    $v_n \xrightarrow{b} v_{neighbor_k} : \langle v_n, WINNER_{ID}, WINNER_{TRUST} \rangle$;
    $v_{neighbor_k} \rightarrow v_n : \langle v_{neighbor_k}, WINNER_{ID}, WINNER_{TRUST} \rangle$;
    Updates the winning pair by selecting the one with maximum trust value;
  end for
end for
for each node n **do**
  Broadcasts its winning pair $\langle WINNER_{ID}, WINNER_{TRUST} \rangle$ to its 1-hop neighbors for d-rounds in this Floodmin mechanism:
  for $i = 1$ to d **do**
    $v_n \xrightarrow{b} v_{neighbor_k} : \langle v_n, WINNER_{ID}, WINNER_{TRUST} \rangle$
    $v_{neighbor_k} \rightarrow v_n : \langle v_{neighbor_k}, WINNER_{ID}, WINNER_{TRUST} \rangle$;
    Updates the winning pair by selecting the one with minimum trust value;
  end for
end for
for each node n **do**
  if $WINNER_{ID} == ID$ **then**
    Declares itself as a clusterhead;
  else
    Identifies all node pairs and selects the node pairs with minimum trust value as its clusterhead;
  else
    Selects the node with maximum trust value in the 1st d rounds of flooding its clusterhead;
  end if
end for

distances. Furthermore, it prevents the cluster with the target node from not providing enough number of introducing nodes.

Apart from the imbalance structure after the clustering formation, actually, the mobile nature of host in ad hoc network has to be handled property. Nodes are moving from one location to another in the network. They are leaving one cluster and joining to another one frequently in a highly mobile environment. We have designed three approach to handle the change of memberships among the clusters. We also consider how they perform in maintaining a balance clustering structures.

Each node requests for the cluster ID of its neighboring nodes periodically to know its neighboring clusters. In each cycle, a node broadcasts request to its neighboring nodes and collects the replies. In Algorithm 2, a node updates its cluster ID by joining the neighboring cluster with minimum size. This approach can maintain a uniform cluster sizes in the network, but the it brings overhead on the frequent change of memberships. In Algorithm 3, a node joins the neighboring cluster with minimum size only if it leaves the original cluster. This approach effectively reduces the changes of memberships. However, the network is found to converge to a one cluster eventually. It may due to the imbalance cluster sizes after the cluster formation algorithm was run. In Algorithm 4, a node joins the neighboring cluster with minimum size only if it leaves the original cluster or the sizes of the neighboring clusters are not within certain range. We defined two parameters S and L which represents the minimum and maximum cluster size in the network. A cluster whose number of nodes is smaller than S or greater than L are claimed to be not within a certain range. If any of the neighboring cluster is this case, the node will leave its original cluster

and join the neighboring cluster with minimum number of nodes even it still receives the original clustering ID. This approach was found to maintain a balance clustering structure in the network, which is similar to the first approach and it leads to less frequent changes in the memberships from one cluster to another among the nodes. Since the second approach does not provide a balance clustering structure, so we can only adopt either approach 1 or approach 3. Consider the number of changes in memberships, approach 3 performs better than approach 1. It is because approach 3 involves fewer changes in membership, so it reduces the overhead due to the move. Therefore, we adopt approach 3 in our authentication service.

Experiments on three approaches have been conducted and the results are shown in the section of Simulations and Results.

Algorithm 2 Clustering Structure Maintaining - Approach 1

1: **for** each cycle **do**
2: **for** each node n **do**
3: $v_n \xrightarrow{b} v_{neighbor_k} : \langle v_n, REQ_{ClusterID} \rangle$;
4: $v_{neighbor_k} \rightarrow v_n : \langle v_n, v_{neighbor_k}, ClusterID_{neighbor_k} \rangle$;
5: $min_{size} = size\ of\ ClusterID_{neighbor_k}$;
6: $min_{cluster} = ClusterID_{neighbor_k}$;
7: **for** $\forall\ ClusterID_{neighbor_k}$ **do**
8: **if** $min_{size} < size\ of\ ClusterID_{neighbor_k}$ **then**
9: $min_{size} = size\ of\ ClusterID_{neighbor_k}$;
10: $min_{cluster} = ClusterID_{neighbor_k}$;
11: **end if**
12: **end for**
13: Joins the $min_{cluster}$;
14: **end for**
15: **end for**

Algorithm 3 Clustering Structure Maintaining - Approach 2

for each cycle **do**
  for each node n **do**
    $v_n \xrightarrow{b} v_{neighbor_k} : \langle v_n, REQ_{ClusterID} \rangle$;
    $v_{neighbor_k} \rightarrow v_n : \langle v_n, v_{neighbor_k}, ClusterID_{neighbor_k} \rangle$;
    if $ClusterID_n \neq \forall\ ClusterID_{neighbor_k}$ **then**
      $min_{size} = size\ of\ ClusterID_{neighbor_k}$;
      $min_{cluster} = ClusterID_{neighbor_k}$;
      for $\forall\ ClusterID_{neighbor_k}$ **do**
        if $min_{size} < size\ of\ ClusterID_{neighbor_k}$ **then**
          $min_{size} = size\ of\ ClusterID_{neighbor_k}$;
          $min_{cluster} = ClusterID_{neighbor_k}$;
        end if
      end for
    end if
    Joins the $min_{cluster}$;
  end for
end for

Algorithm 4 Clustering Structure Maintaining - Approach 3

for each cycle **do**
  for each node n **do**
    $v_n \xrightarrow{b} v_{neighbor_k} : \langle v_n, REQ_{ClusterID} \rangle$;
    $v_{neighbor_k} \rightarrow v_n : \langle v_n, v_{neighbor_k}, ClusterID_{neighbor_k} \rangle$;
    if $ClusterID_n \neq \forall\ ClusterID_{neighbor_k}$ or $\exists!(S \leq size\ of\ ClusterID_{neighbor_k} \leq L)$ **then**
      $min_{size} = size\ of\ ClusterID_{neighbor_k}$;
      $min_{cluster} = ClusterID_{neighbor_k}$;
      for $\forall\ ClusterID_{neighbor_k}$ **do**
        if $min_{size} < size\ of\ ClusterID_{neighbor_k}$ **then**
          $min_{size} = size\ of\ ClusterID_{neighbor_k}$;
          $min_{cluster} = ClusterID_{neighbor_k}$;
        end if
      end for
    end if
    Joins the $min_{cluster}$;
  end for
end for

5.2 Security Operations

The authentication protocol we propose takes a certificate-based approach [58] [57]. If a user i believes a given key belongs to a given user t, it can issue a public key certificate of t. When a node s wants to get the public key of a node t, it requests for the public key certificates of node t from some trustable nodes. Node s sends request messages to some nodes that belong to the group of node t and with high trust values in the view of s. These nodes which sign the public key certificates of node t are called introducers. They reply to the requesting node with the public key certificate of the target node and also the trust value of the target node.

The security operations are divided into three parts, including the public key certification, identification of malicious nodes and the trust value update. Figure 5.1 shows the flow of the major security operations. In public key certification, a node requests the public key certificates of the target node, collects and concludes the public key of the target node by majority votes. During the comparison among the received certificates, identification of malicious nodes can be done. Finally, trust values of the target node can be calculated and updated.

Table 5.1 shows the security operations of a requesting node s. When node s wants to obtain the public key of a node t, it selects a certain number of nodes that it trusts as introducers. These introducers should be in the same group of node t, so they can provide the public key and trust value of node t accurately. Then, node s sends the request of public key certificate to all the selected introducers. After node s collects all the replies, it compares the public key certificates received and concludes the public key of node t with the majority votes. If a malicious introducer providing a false public key certificate of node

t is discovered, it will be isolated by reducing its trust value to zero. Finally, trust value of node t will be calculated and inserted into the trust table of node s. Details operations on public key certification and trust value update will be presented in the following subsections.

It shows the operations of s on obtaining public key certificates of t. To request the public key of t, s first looks up the group ID φ_t of node t. Then, it sorts the trust values that belong to φ_t and selects the nodes with the highest trust value as introducer $i_1, i_2, \ldots, i_n$ and sends them request messages. After collecting the reply messages encrypted with introducers' secret keys, s decrypts the messages with the corresponding public key. Next, it compares the public keys obtained from the reply messages and concludes the public key of t as the one with majority votes. It reduces the trust values of the nodes which do not agree with that public key, so to avoid selecting these dishonest nodes as introducers in the future. Finally, s will calculate and update the trust value of t, V_t.

5.2.1 Public Key Certification

Authentication in our network relies on the public key certificates signed by some trustable nodes. Let s be the node requests for public key of a target node t. Node s has to ask for public key certificates signed by some introducing nodes, $i_1, i_2, \ldots, i_n$, as shown in Figure 5.2. Every node is able to request for public key certificates of any other new nodes. However, nodes in the same group are assumed to know each other by means of their monitoring components and the short distances among them. With the above assumptions, we focus on the public key certification where s and t belong to different groups. Nodes which are in the same group with t and have already built up

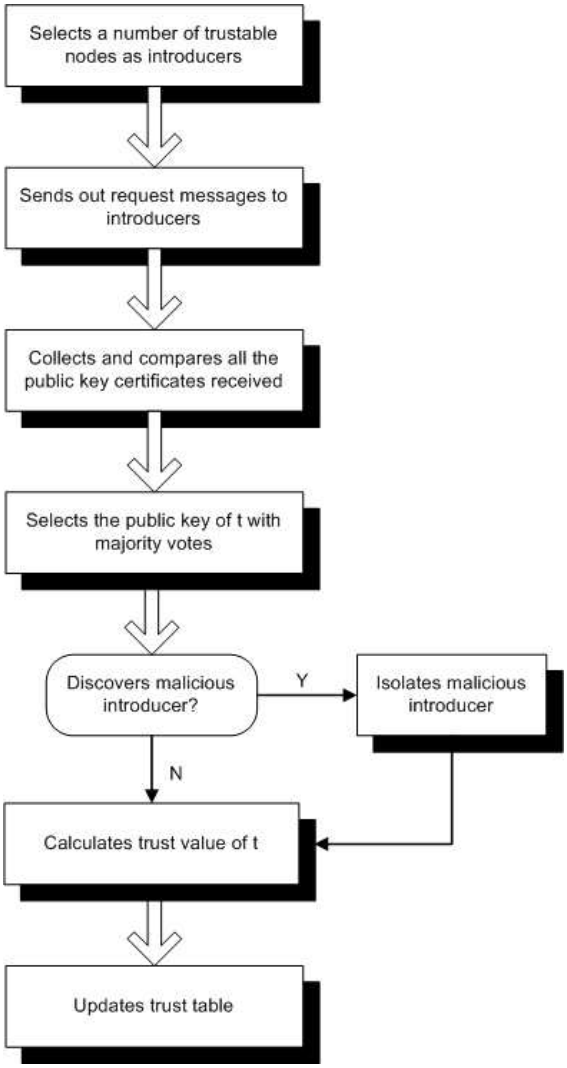

Figure 5.1: Security Operations

Table 5.1: Operations of Node s in Public Key Certification

1. Looks up the group ID of t,φ_t.
2. Sorts the trust values of nodes belonging to group φ_t in the trust table. Let i_1, i_2,..., $i_n \in I$, where i_1, i_2,..., i_n denote nodes with the highest trust values in group φ_t.
3. Sends request messages to nodes in I.
4. Collects the reply messages $m \in M$ from i_1, i_2,..., i_n, where $m = \{Pk_t, V_{i_k,t}, ...\}Sk_{i_k}$. Pk_t denotes the public key of node t, $V_{i_k,t}$ denotes the trust value from i_k to t, and Sk_{i_k} denotes the secret key of i_k. The reply message is signed by the secret key of i_k, Sk_{i_k}.
5. Compares the public keys received and concludes with the majority votes. Let $i_{good} \in I_{good}$ and $i_{bad} \in I_{bad}$, where i_{good} are the nodes that thought to be honest (agree on Pk_t with the majority) and i_{bad} are the remaining nodes that thought to be dishonest.
6. Reduces the trust values of i_{bad} to zero. Computes and updates the trust value of t,V_t , with this formulae: $$V_{s,i_k,t} = V_{s,i_k} \bigodot V_{i_k,t} = 1 - (1 - V_{i_k,t})^{V_{s,i_k}} \quad (5.2)$$ and $$V_t = 1 - \Pi_{k=1}^{n}(1 - V_{s,i_k,t}), \quad (5.3)$$ where i_k denote the nodes in I_{good} and n denotes the number of nodes in I_{good}.

trust relationship with s can be introducers. The requesting node s selects certain number of nodes with the highest trust values as introducers and sends them request messages. The introducers i_1, i_2 ,..., i_n, after receiving the messages will reply with the public key of the target node t. Apart from the public key of t, it includes the trust value of t as well. These values from i_1, i_2, ..., i_n, will be used for calculating the final trust value of t in s when all the reply messages are received. The reply message should be signed with the introducers' private keys to make the certificate valid.

Algorithm 5 shows the procedure on the request for public key certificates of a target node. In this algorithm, node v_i is requesting for the

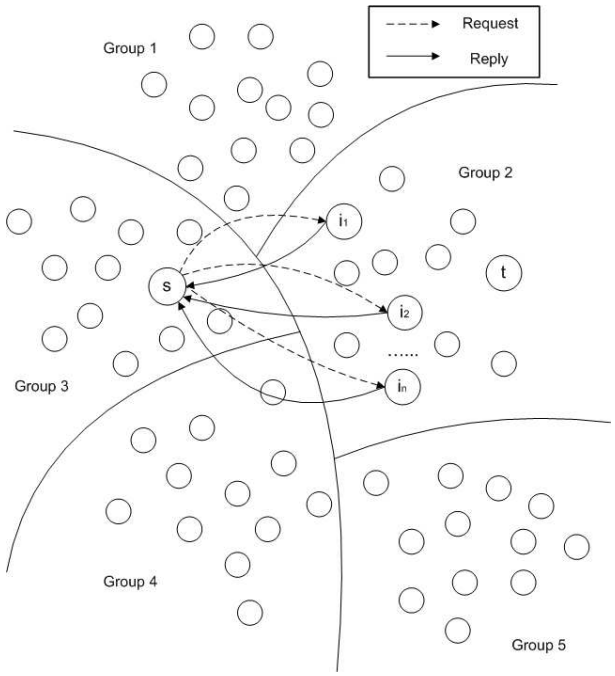

Figure 5.2: Public Key Certification

public key certificates of node v_j. Given that node v_i belongs to cluster $CLUST_A$ and node v_j belongs to cluster $CLUST_B$. Before sending out the request message, node v_i first check wether it is in the same cluster with v_j. If it is the case, it send request message to its neighboring nodes and it is believed that some of its neighboring nodes should have build up direct trust relationship with v_j by themselves or their neighboring nodes. After receiving the reply messages, v_i simply updates the trust value and public key of v_j. It is because nodes in the same cluster are believed to know each other other. They are able to discover the malicious nodes in their own cluster, so the neighboring nodes that they are communicating with are always trust-worthy. On the other hand, if v_i and v_j are in different clusters, then the problem become more complicated. Node v_i has to selects some trust-worthy nodes in the target cluster to be the introducing nodes, or so called introducers.

These are nodes with high trust values in the view of v_i and in the same cluster with v_j. Similar to the previous case, v_i sends the request message to the introducers and wait for the replies. However, it is possible for the introducers to be malicious and v_i has not yet discovered due to the long distance between them. Therefore, a voting will be carried out to conclude the correct public key of the target node with majority votes. This is the algorithm in making public key certificate requests.

Algorithm 5 Request for public key certificates

Given v_i belongs to $CLUST_A$ and v_j belongs to $CLUST_B$. A node v_i requests for the public key certificate of a node v_j:
if $(CLUST_A == CLUST_B)$ **then**
  v_i sends request to neighbors v_k:
  $v_i \xrightarrow{b} v_k : \langle v_i, v_j, REQ_{CERT} \rangle$;
  $v_k \rightarrow v_i : \langle v_j, T_{v_k \rightarrow j}, PK_j, ... \rangle_{SK_{v_k}}$;
  v_i updates PK_j and T_j;
else
  v_i selects trust-worthy nodes in $CLUST_B$ as introducers i_k;
  $v_i \xrightarrow{b} i_k : \langle v_i, v_j, REQ_{CERT} \rangle$;
  $i_k \rightarrow v_i : \langle v_j, T_{i_k \rightarrow j}, PK_j, ... \rangle_{SK_{i_k}}$;
  v_i compares the PK_j from the received certificates and update PK_j in their repository;
  v_i calculates and updates T_j;
end if

5.2.2 Identification of Malicious Nodes

As mentioned before, mobile ad hoc network is a collection of nodes connected with wireless communications and it is vulnerable to security attacks. Mobile ad hoc network does not provide any centralized servers for security or management purposes. The lack of infrastructure and organizational environment of mobile ad-hoc networks offer special opportunities to attackers. To protect the network security, it relies on the capability of individual nodes. In a hostile network en-

vironment, there exists a number of malicious nodes in the network. These malicious nodes can harm the network by dropping the packets, hijacking the communications between the nodes, etc. In terms of our public key certification service, a malicious node can reply a public key certificate request with an incorrect public key of the target node. In order to prevent the malicious nodes harming the network, we provide three ways to identify suspicious nodes in the network. After the identification, the trust values of malicious nodes will be lowered, so they will be isolated from public key certifications. Nodes with lower trust values will not be chosen as introducers in the public key certification in the future.

The first method is to identify malicious neighboring nodes by direct monitoring power of individual nodes. Nodes in mobile ad hoc network is able to observe the behavior of its 1-hop neighbors directly. This can be done by listening the traffic via wireless communications with some monitoring component, like watchdog. A number of researches [50, 12, 56, 67] have been carried out on detecting and isolating misbehaving nodes in the network with cooperation between nodes. All of the researches proposed in this area agree on the importance of cooperation between nodes and the work for monitoring the networks should be distributed and carried out by every node. Generally, a monitoring device will be implemented on every node for detecting misbehavior, then mechanisms for the exchange of misbehavior information and isolating the misbehave nodes will be developed. In our authentication service, we assume each of the nodes in the network are equipped with this capability.

The second method to isolate malicious nodes by identifying suspicious introducers who provide public key certificates different from the

others. In each of the public key certificates request, a node finds more than one introducers to collect multiple reply messages. After decrypting the public key certificates by using the introducers' public keys, it can read the public key of the target node provided by the introducers. The replied public key should be the same if all the introducers are honestly replying the correct answer. If there exists some introducers who provide public key of the target node which is different from the others, then these introducers is suspected to be malicious. It should be noted that using majority vote may not be able to identify malicious nodes when there are colluding nodes in the network. A set of nodes can collude to provide an incorrect public key, and these nodes may represent the majority opinion. To deal with this case, nodes have to update the trust values of the others throughout their experience in using the public keys. They can learn about the correctness of the keys by using them.

The third method allows the requesting node to identify the target node as malicious if the trust values provided from the introducers indicate that. After the requesting node sends the message asking for public key of the target node, its introducers reply with the public key certificates. In each of these public key certificates, it includes not only the ID and public key of the target node, but also the trust value from that particular introducer to the target node. With these values, the requesting node can analyze them and summarize the trust value of the target node. If the trust value of the target node is lower than a certain threshold, then the target node is indicated as dishonest. The requesting node will update the trust value of the target node and prevent from selecting them as introducers in public key certification in the future. Similar to the above case, a set of introducers can collude

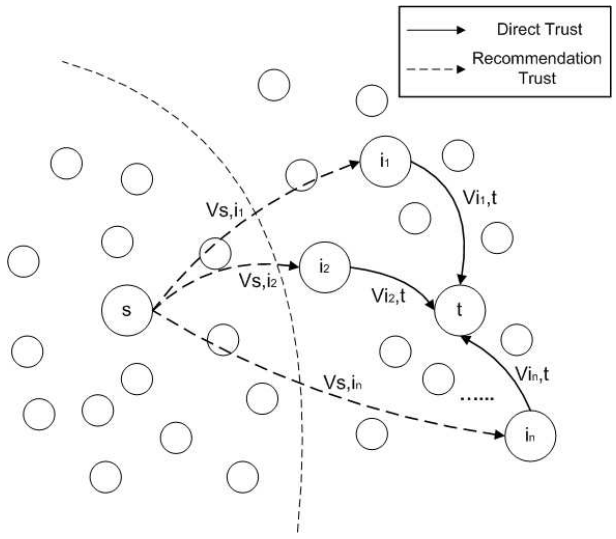

Figure 5.3: Trust Value Update

and provide an incorrect trust value of the target node. They may assign an honest node with very low trust value or assign a colluding node with very high trust value to increase their influence in the network. To make our approach more practical, the trust value update algorithm should not be limited to public key certification. It is more reasonable for a node to update the trust value of the other nodes that it has communicated with.

5.2.3 Trust Value Update

After collecting and decrypting the reply messages, the relying node obtains the trust values from different introducers i_k to t. These values can be used to calculate the ultimate trust value V_t of t in the view of s as shown in Figure 5.3.

In this figure, s denotes the requesting node; t denotes the target node, whose public key is requested by s. Nodes i_1, i_2,..., i_n are the introducers that reply to s with consistent public keys of t. $V_{i_1,t}$,

$V_{i_2,t}$, ...,$V_{i_n,t}$ denote trust values from introducers i_1, i_2,..., i_n to t; while$V_{i_1,t}$, $V_{i_2,t}$, ...,$V_{i_n,t}$ denote trust values from s to introducers i_1, i_2,..., i_n. Each $V_{s,i*}$ and $V_{i*,t}$ form a pair to make up a single trust path from s to t. To compute a new trust relationship from s to t of a single path, we apply the following formula:

$$V_{s,i_k,t} = V_{s,i_k} \bigodot V_{i_k,t} = 1 - (1 - V_{i_k,t})^{V_{s,i_k}} \tag{5.4}$$

It calculates the new recommendation trust relationship from s to t via an introducer i_k. With this formula, we can calculate the three different trust values from s to t via these three introducers on different path separately. The result values are usually different, so one has to find a way to draw a consistent conclusion. Actually, the different values do not imply a contradiction. In contrary, it can be used as collective information to compute a combined value. The following formula can be applied:

$$V_t = 1 - \Pi_{k=1}^{n}(1 - V_{s,i_k,t}), \tag{5.5}$$

where n denotes the number of paths.

This formula combines trust values $V_{s,i_k,t}$ of different paths to give the ultimate trust value V_t of t. This ultimate trust value V_t represents the trust value of t in the view of s after the public key certification. This value contains information of trust relationships from s to different introducers, and that from introducers to t. Finally, this value will be inserted to the trust table of s. If V_t is high, it indicates that t can be a possible introducer when s requests for public keys of other nodes that belong to the same group of t in the future. Apart from the trust value of the target, the trust value of the introducers will also be updated.

The introducers whom were found to be malicious, their trust value will be lowered and be isolated. In contrary, the requesting node will gradually increase the trust values of the introducers whom provide correct public key certificates of the target nodes.

As mentioned before, the trust value update algorithm can be applied not only when a node receives replies in public key certification. A node can update trust values of the any other nodes in their daily communications. It is more reliable for a node to adjust the trust values of other nodes according to its experiences in using their public keys or communicating with them. The localized trust value update approach strongly relies on a node's experience to make judgement about the network security. Its continuous update nature prevents colluding nodes from dominating the network upon a single event, like public key certification.

5.3 Special Scenarios

5.3.1 Join the network

When a node first joined into the network, it can only communicate with its neighboring nodes. It broadcasts the joining message to its neighboring nodes and build up intragroup trust relationship with the nodes in the same cluster. Since it is new to the network, it has no experience in communicating with the nodes in other groups. When a new node requests for the public key certificate of nodes in other groups, it collects the information from its group members as intergroup trust relationship has not yet been built up. In the early stage of a node joining into the network, it relies on the intragroup relationship in communicating with the others, including the nodes in different

groups. After several communications are made to the nodes in different groups, the new node can build up intergroup trust relationship gradually. Then, the node can rely on intergroup trust relationship for requesting the public key certificates of nodes in different cluster and it use intragroup trust relationship mainly for communication within the local group.

Algorithm 6 shows the procedure when a new node joins the network. A new node firsts sends a "hello" message to its neighboring nodes and asks for the clusterhead ID. After receiving the reply, it sends a "join" message to the clusterhead it selected. The clusterhead will then update its member list to include the new node. Update on the member list will be sent to the cluster members and other clusterheads periodically for update purpose. The new node will also generate its own pair of public key and private key. Then, it will exchange its public key with its neighboring nodes. The new node and its neighboring cluster members will initialize the trust values of each other as 0.5. It is the startup of their trust relationship, these trust values will be updated gradually later with the proceeding of their behavior monitoring work.

5.3.2 Move to another cluster

As mentioned before, nodes in mobile ad hoc network are free to move around. It is normal for them to leave the original cluster and join another one during its move. Algorithm 7 shows the procedure of the move. Given a node v_n moves from cluster $CLUST_A$ to cluster $CLUST_B$. In our authentication service, clusterheads in the network will broadcast "hello" message to its cluster members periodically. This allows the nodes to know if they are still staying in the same cluster. If

Algorithm 6 Joining of a new node

Given a graph $G = (V, E)$ with $n = |V|$ with nodes $v_i \in V$ and a new node v_{n+1}. Also, the graph with several clusters.

1: v_{n+1} sends a message to its neighboring nodes to look for the ID of the clusterhead:
$v_{n+1} \xrightarrow{b} v_{neighbor_k} : \langle v_{n+1}, REQ_{CLUSTHEAD_{ID}} \rangle$;
2: Neighboring nodes reply with the ID of the clusterhead:
$v_{neighbor_k} \rightarrow v_{n+1} : \langle v_{n+1}, v_{Head_A} \rangle$;
3: v_{n+1} sends a joining message to the clusterhead:
$v_{n+1} \rightarrow v_{Head_A} : \langle v_{n+1}, JOIN \rangle$;
4: v_{Head_A} updates the member list, where $V_A := V_A \cup \{v_{n+1}\}$;
5: v_{n+1} generates a pair of public key PK_{n+1} and private key SK_{n+1};
6: **for** each $v_{neighbor_k}$ **do**
7: v_{n+1} exchanges its public key PK_{n+1} with those of $v_{neighbor_k}$:
$v_{n+1} \xrightarrow{b} v_{neighbor_k} : \langle v_{n+1}, PK_{n+1} \rangle$;
$v_{neighbor_k} \rightarrow v_{n+1} : \langle v_{neighbor_k}, PK_{neighbor_k} \rangle$;
8: v_{n+1} and $v_{neighbor_k}$ initialize the trust value of each other to be 0.5;
9: **end for**
10: v_{Head_A} broadcasts the joining of v_{n+1} to other clusterheads at certain time later:
$v_{Head_A} \xrightarrow{b} v_{Head_k} : \langle V_A := V_A \cup v_{n+1} \rangle$;

a node does not receive message from clusterhead for a period of time, it will know its leave from the original cluster. When a node discovers such a leave, it will broadcast a request message to its neighboring nodes to obtain the cluster ID of them. After receiving the replies, similar to the situation when a new node joins, it joins one of the clusterhead by sending it joining message. Then, it exchanges its public key with its new neighbors. They update the trust values with each other. Finally, the update of the membership will be broadcasted to the cluster members and the other clusterheads.

5.3.3 Not Enough Introducer

A node requests for public key certificates of the target nodes that are new to them and in different groups via some introducers. Introducers

Algorithm 7 Move from a cluster to another

Given a node v_n moves from $CLUST_A$ to $CLUST_B$. Provided that each clusterhead constantly broadcast "hello" message in the cluster:

1: v_n discover its leave from $CLUST_A$ as it does not receive "Hello" message from $Head_A$;
2: v_n sends a message to its neighboring nodes to look for the ID of the clusterhead:
$v_n \xrightarrow{b} v_{neighbor_k} : \langle v_n, REQ_{CLUSTHEAD_{ID}} \rangle$;
3: New neighbors reply with the ID of the clusterhead:
$v_{neighbor_k} \rightarrow v_n : \langle v_n, v_{Head_B} \rangle$;
4: v_n sends a joining message to the $Head_B$:
$v_n \rightarrow v_{Head_B} : \langle v_n, JOIN \rangle$;
5: $Head_B$ updates the member list, where $V_B := V_B \cup \{v_n\}$;
6: **for** each $v_{neighbor_k}$ **do**
7: **if** v_n not know $v_{neighbor_k}$ **then**
8: v_n exchanges its public key PK_n with those of $v_{neighbor_k}$:
$v_n \xrightarrow{b} v_{neighbor_k} : \langle v_n, PK_n \rangle$;
$v_{neighbor_k} \rightarrow v_n : \langle v_{neighbor_k}, PK_{neighbor_k} \rangle$;
9: v_n and $v_{neighbor_k}$ initialize the trust value of each other to be 0.5;
10: **end if**
11: **end for**
12: v_{Head_B} broadcasts the joining of v_n to other clusterheads at certain time later:
$v_{Head_B} \xrightarrow{b} v_{Head_k} : \langle V_B := V_B \cup v_n \rangle$;

are the nodes in the same group of the target node and have intergroup trust relationship with the relying node. In some situations, the relying node may find not enough introducers to request for public key certificates of the nodes in other groups. These situations may be at the early stage of a node in the network or a node finds that most of the nodes in another group that it built up intergroup trust relationships become malicious. If there are not enough introducers in the target group, the relying node will choose nodes with high values from its own group to be introducers. It should be noted that a node request public key certificates of the node in another group always find introducers from the target group as the first choice. It finds introducers from its local only if it is unable to find enough number of introducers from the target group. A node chooses introducers from the target group with higher priority than from the local group. It is because nodes in the same group with the target nodes are able to collect more information on the trust of the target node with the relatively shorter distances. In contrary, introducer in the same group with the relying node only provides information of its past communication with the target node and the not up-to-date trust information collected when it requested for the public key certificate of the target node.

□ **End of chapter.**

Chapter 6

Simulations and Results

In this chapter, we evaluate the performance of the authentication service proposed in terms of security by extensive simulations.

6.1 Authentication Service Based on Trust and Network Models

6.1.1 Experiments Set-Up

We implemented our design in network simulator Glomosim [72].Our main objective in the security evaluation is to investigate whether our authentication service provides effective measurement results in public key certifications with the presence of malicious nodes. We imitate the malicious nodes by selecting certain percentage of the nodes in the network randomly and assign them to reply with false public key certificates. A false public key certificate may contain an incorrect public key and trust value of the target node.

The base settings that apply for most of the experiments are summarized in Table 6.1. The settings represent a wireless ad hoc network with the size of 600m x 600m. It contains 100 nodes and is divided

into 5 groups. The network size and the number of nodes are set in these values to make sure the network density is high enough to build a connected network. The number of introducers per request is three. It is selected as an odd number to bring a conclusion based on majority votes. This number should be large enough to avoid incorrect conclusion due to malicious nodes, but it should not be too large such that a requesting node can find enough number of introducers for public key certification. In this thesis, we use 802.11 as the MAC protocol in the experiments. IEEE 802.11 has a higher data rate and transmission range in compare with bluetooth, so it provides more bandwidth for communications and public key certifications in our authentication service. Also, its transmission range allows a network to be formed in a larger area and with high number of nodes. A certain percentage of nodes p is regarded as trustable at initialization and certain percentage of nodes m becomes malicious when the simulation begins. We are particularly interested in the successful rate, fail rate, unreachable rate, and type I and type II error rate in our protocol. We vary different parameter in each of the experiment, including the percentage of trustable nodes at initialization, percentage of malicious nodes, and the mobility of the nodes. In the last experiment, we compare the successful rate, fail rate, and unreachable rate between our protocol and the PGP approach with distributed certificate repository. Yet, our experiments indicate that our scheme works well even in a hostile environment.

Table 6.1: Simulation Parameters

Network	
Network size	600m x 600m
No. of nodes	100
No. of groups	5
% of trustable nodes at initialization	p
% of malicious nodes	m
Mobility	
Mobility	Random-Waypoint
Pause Time	20s
Maximum speed	10m/s
PublicKeyCertification	
Max. no. of introducers for each request	3
Min. no. of reply for each request	1
No. of query cycles	80
No. of requests per cycles	100
Simulation Time	100000s

6.1.2 Simulation Results

Evaluation on Ratings to Malicious Nodes

In this experiment, we evaluate the successful rate, fail rate, unreachable rate, false-positive error rate, and false-negative error rate of the authentication service proposed. Successful rate is the percentage of public key requests that lead to a conclusion of the new node's public key. Fail rate is the percentage of public key requests that are unable to make a conclusion of the new node's public key or the conclusion drawn is incorrect. Unreachable rate is the percentage of public key requests that are unable to be sent out or the requests have no reply. A request unable to be sent out may due to no trustable introducer is available, or the request messages cannot reach the introducers. It is also possible that the request messages are sent, but the messages are dropped or unreachable to the requesting node in the reply.

Apart from the successful rate, fail rate, and unreachable rate discussed above, we also carry out the Type I and Type II error tests. We

evaluate the false-negative error rate on identifying malicious nodes in the Type I error test and the false-positive error rate in identifying malicious nodes in the Type II error test. In the authentication service we propose, nodes requesting for the public key of a new node compare the public key certificates it received from introducers and try to make a conclusion by the majority votes. If it discovers certain replies of the public key are different from that of the majority, then it suspects the nodes as malicious and lowers their trust values. With this voting algorithm, it is possible for it to incorrectly identify trustable nodes as malicious. We assume that the malicious nodes are not forming malicious peer in the network, so they have low probability to reply with a consistent false public key in the certificates. The following examples illustrate how false-positive and false-negative errors may occur, where "O" indicates a certificate replied by a good node and "X" indicates a certificate replied by a malicious node:

Examples of false-positive error:

"O X" Two public key certificates are received from the replies and they are different from each other. The relying node can make no conclusion on the new node's public key in this case and it concludes that either both or any one of the replies are come from malicious nodes. To put the authentication service in the safest place, it lowers the trust values of both nodes to avoid any malicious node to be selected as introducers in the future. If one of the reply nodes is indeed trustable in this situation, then a false-positive error occurs as it falsely suggests that a node as malicious which it is actually not.

"O X X" Similar situation occurs when three different public key certificates are received in the replies. The requesting node can make no conclusion on the new node's public key again in this case and it concludes that either all or any one of the replies are come from malicious nodes. To keep the network safe, it lowers the trust values of all the nodes to avoid any malicious to be selected as introducers in the future. If one of the introducer is actually a good node, then a false-positive error occurs again in this case.

Not only false-positive errors may occur in the system, but false-negative errors also. The following example shows how false-negative error that may occur in public key certification:

Example of false-negative error:

"X" The relying node receives only one reply message, so it has no chance to make comparison and conclude the new node's public key by majority votes. In this situation, the relying node may believe the reply is trustable as there is no evidence showing inconsistency of the received public key. It may assume this public key certificate is correct to allow its communication with the new node. Unfortunately, if the replying node is indeed malicious, then a false-negative error occurs.

Figure 6.1 shows the successful rate, fail rate, unreachable rate, false-positive error rate, and false-negative error rate in the authentication service we propose with the percentage of malicious nodes varies from 0% to 100%. The percentage of trustable at initialization is fixed at 40% in Figure 6.1a and at 70% in Figure 6.1b respectively. In both

figures, the successful rate drops with the percentage of malicious nodes increases. It is because more false public key certificates are received with the increased number of malicious nodes in the network. With the above reason, it is hard for the requesting node to draw a conclusion on the public key of the new node, so the successful rate decreases. The fail rate on the hand increases gradually with the percentage of malicious nodes. It has the same reason as the drop of the successful rate. The unreachable rate increases dramatically with the percentage of malicious nodes. It is due to large amount of nodes initially trustable becomes malicious in the network. These malicious nodes can no longer be introducers upon being discovered and isolated. Some requesting nodes may not be able to contact any introducer as none of them remains trustable on its list, so public key certificate requests cannot be sent.

From the above figures, we can observer that false-positive error rate and false-negative error rate increase with the percentage of malicious nodes as well. The false-positive error rate of both graphs begin from zero and rise gradually from 30% to 70% and then drops to zero gradually afterwards. In our experiment, the number of introducers is three, which means a relying node sends request messages to three introducers in each public key request. The rise and drop of the false-positive rate is related to the probability of having the two cases of false-positive errors ("OX" and "OXX") from the replies. The false-negative rate rises as there is a higher probability to receive only a single reply from a malicious node when the percentage of malicious nodes increases. The reason is that the higher the percentage of malicious nodes leads to smaller number of trustable introducers left in the network, so a node has a higher chance to find only one introducer

to sign valid public key certificate. However, this remaining introducer also has a higher probability to be a malicious node.

In comparing the two figures, we find that Figure 6.1a has a lower successful rate, higher unreachable rate, lower failure rate, and lower false-positive rate than Figure 6.1b. The lower successful rate and the higher unreachable rate in Figure 6.1a are because of the less trustable introducers are available in public key certification with the face that the percentage of trustable node at initialization in Figure 6.1a is much lower than that of Figure 6.1b. The lower failure of Figure 6.1a is due to smaller number of malicious nodes has to be discovered. Since only trustable nodes will be selected as introducers, the higher the percentage of trustable nodes at initialization leads to the greater number of malicious nodes have to be discovered to avoid false public key certification. The malicious node discovering algorithm is based on majority voting in our authentication service. Normally, the more public key certificate request made, the higher number of malicious nodes can be identified. In this experiment, both figures run for 80 cycles and the experiment results are the average of each rating during the whole simulation. It is reasonable that Figure 6.1b receive more false certificates than Figure 6.1a, so it has higher failure rate and false-positive rate than Figure 6.1a.

Evaluation on Ratings to Trustable Nodes at Initialization

Similar to the above experiment, the successful rate, fail rate, unreachable rate, false-positive error rate, and false-negative error rate are evaluated. However, we fix the percentage of malicious nodes and vary the percentage of trustable nodes at initialization in this experiment. We set the percentage of malicious nodes at 40% in Figure 6.2a and

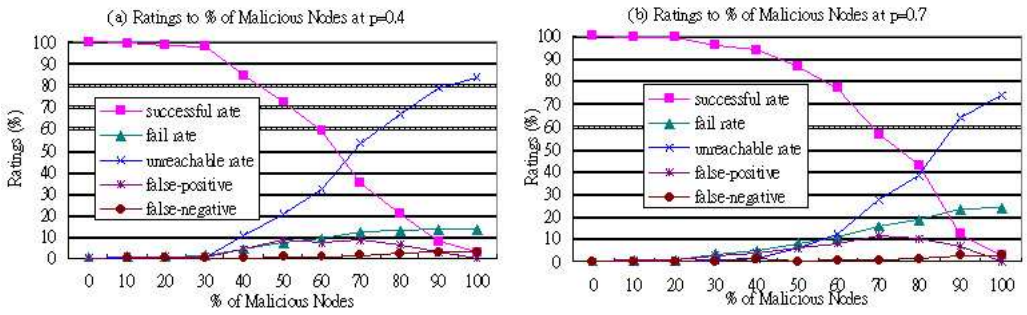

Figure 6.1: Ratings to Percentage of Malicious Nodes

at 70% in Figure 6.2b, then vary the percentage of trustable nodes at initialization from 0% to 100%. Both figures show the successful rate increases and the unreachable rate decreases with the increase on the percentage of trustable nodes at initialization. It is because greater number of nodes can be selected as introducers for public key certifications if there is more trustable nodes at initialization. The increase of fail rate is due to more number of malicious nodes need to be discovered as greater number of nodes appear to be trustable initially become malicious later.

In comparing the two figures, Figure 6.2a has a higher successful rate, lower fail rate, unreachable rate, false-positive rate, and false-negative rate in compare with Figure 6.2b. The performance in terms of security of Figure 6.2a is better than that of Figure 6.2b overall. It is reasonable that a network with lower percentage of malicious nodes to be more secure in public key authentication.

Evaluation on Convergence Time

We investigate the convergence time of our authentication service in this experiment. Again, the same ratings, including the successful rate,

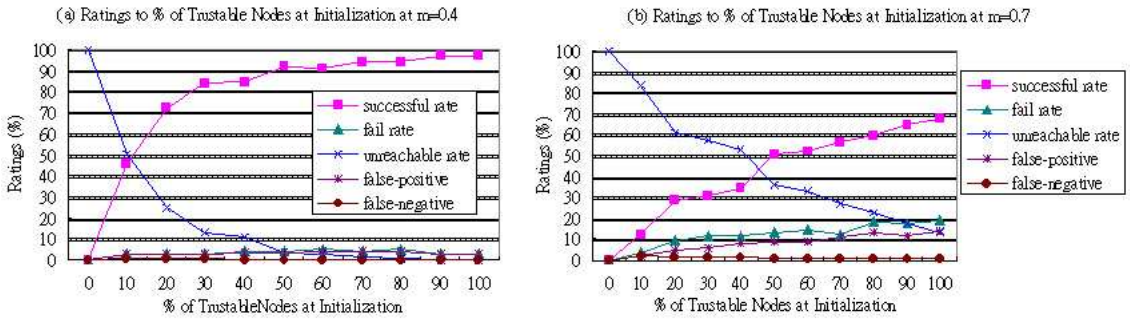

Figure 6.2: Ratings to Percentages of Trustable Nodes at Initialization

fail rate, unreachable rate, false-positive error rate, and false-negative error rate are evaluated. We plot different ratings every five cycles in this experiment to get the time of convergence. At the time of convergence, all the ratings are expected to become steady. Since malicious nodes are assigned randomly at the beginning of the experiment, malicious nodes will be discovered gradually and the ratings will vary during this period of time. The convergence time represents the moment that most of the malicious nodes in the network are discovered, so all the ratings become steady upon it.

In Figure 6.3a, the percentage of malicious nodes and the percentage of trustable nodes at initialization are both fixed at 40%. From the experiment results, we observe that each rating converges to a certain limit value s after certain number of cycles. For example, it shows that the successful rate converges to around 85.4%, fail rate converges to 0.6%, unreachable rate converges to 14%, false-positive rate converges to 0.6%, and false-negative rate converges to 0% in Figure 6.3a. We define n as the number of cycles, s as the limit value, x as one of the rating at certain cycle. There exists a positive integer N such that when $n > N$, we have $|x_n - s| < \xi$. If we set ξ to be 2% for the

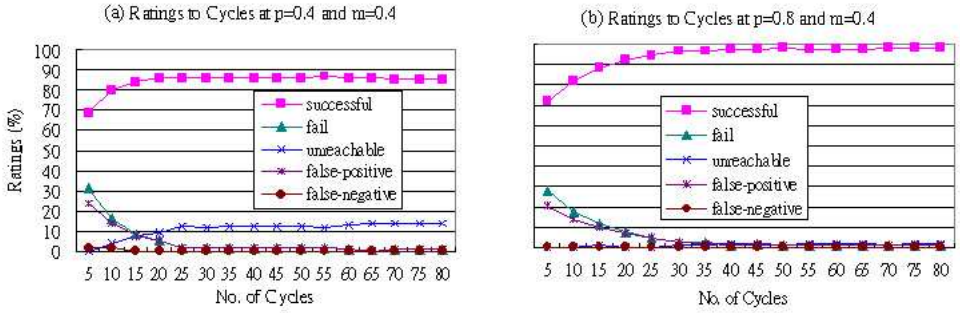

Figure 6.3: Ratings to No. of Cycles

successful rate x_n, N is equal to 25 in Figure 6.3a, while N is equal to 30 in Figure 6.3b.

Evaluation on Ratings to Mobility

In this experiment, we investigate the influence of mobility to the authentication protocol we propose. Throughout all simulations, a relying node sends out public key certificate request to three introducers and gets back their replies. The network size is 600m x 600m with 100 nodes, which allow most of the request and reply messages to reach their destinations. Figure 6.4 shows the distribution of the ratings under different mobility of nodes. It evaluates the successful rate, fail rate, unreachable rate, false-positive error rate, and false-negative error rate with the percentage of trustable nodes at initialization to be fixed at 40% and the percentage of malicious nodes to be fixed at 60%. We vary the mobility of nodes by setting the maximum speeds of nodes at 0m/s, 5m/s, 10m/s, 15m/s, and 20m/s respectively. The authentication service we propose maintains almost constant distribution under different mobility conditions as shown in Figure 6.4. Since the network size is not large in compare with the number of nodes, the transmission

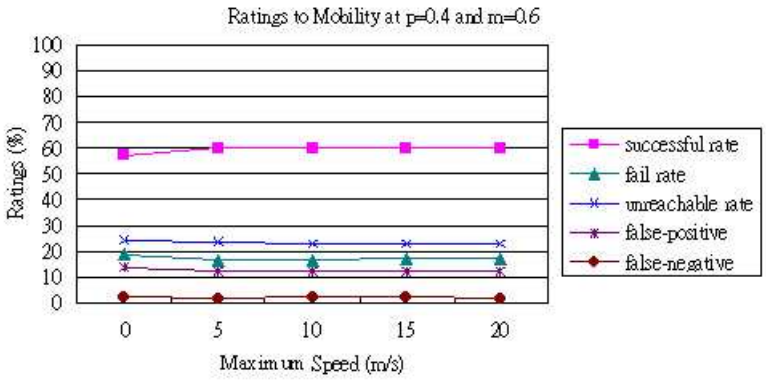

Figure 6.4: Ratings to Mobility

range of a node normally can cover any of its neighboring nodes. Similar result showing the mobility independent with the successful rate has been appeared in another paper. This paper employed a simple flooding protocol to implement a practical key management framework for ad hoc wireless network [68]. It believes independency of the mobility is because of the effectiveness of flooding as the reliable data dissemination method.

Comparison with the PGP Approach

In this sub-section, we compare our authentication service with the web of trust model in Pretty Good Privacy (PGP). We judge against their performance in protecting network security during public key certification. A fully self-organizing public key management system using certificate graph, which is similar to PGP, was proposed in ad hoc wireless network [16]. It proposed an algorithm for the construction of the local certificate repositories to help users to find certificate chains to each other in their merged repository. The certificates of this approach are stored and distributed by the nodes and unlike in PGP,

where certificates are stored in centralized. It implies that the web of trust model of PGP is applicable to wireless ad hoc networks with certain adjustment.

In a PGP environment, any user can act as a certifying authority. A PGP user validates another PGP user's public key certificate if the relying party recognizes the validator as a trusted introducer. Usually, a keyring stores the validity of a particular key and the level of trust it placed on the key that the key's owner can serve as certifier of other's key. There are three levels of validity in PGP, including Valid, Marginally Valid, and Invalid. PGP requires one Completely trusted signatures or two Marginally trusted signature to establish a key as valid. Although PGP involves a trust model with three levels of trust and three levels of validity in public key certification, it does not have any measurement in handling malicious nodes that issue false certificates. It assumes that the public key certificate and the level of trust of a node are valid during its validity period, but this does not reflect the reality. It is because attackers may compromise a node suddenly without being discovered, so it is important to protect authentication against malicious nodes. To deal with the problem of false certificates signed by undiscovered malicious nodes, we propose a novel public key authentication approach based on the trust and clustering techniques.

In comparing our trust- and clustering-based approach with the original PGP approach, our approach is different in distributing repository on certificates among all the nodes. In the original PGP approach, it just defines three levels of trust for a node. In our approach, the trust is defined as a continuous value between 0.0 and 1.0. Therefore, a more accurate trust level can be expressed in our approach than in the original PGP approach. Moreover, the original PGP approach relies on

a single trust chain with multiple intermediate nodes to acquire the public key certificate of a new node. In our approach, a trust chain only involves on one intermediate node to reduce the probability for obtaining an invalid trust chain, which involves any malicious nodes. The only intermediate node on a trust chain is in the same cluster as the target node. The close distance between the intermediate node and the target node enhance the performance of the monitoring component on the intermediate node. This increases the correctness for the intermediate node to introduce the target node and estimate its trust value. Also, it relies on multiple trust chains instead of single trust chain in our approach. The public key certificates of the target node signed by different introducers will be compared. Certificates different from the majority votes will be identified and the introducer who signs these suspicious certificates will be isolated gradually. The trust values from different introducers on the target node will be gathered and summarized, and finally be updated to the trust table of the relying node. In summary, our approach makes use the behavior monitoring advantage and the hierarchical architecture brought by the clustering techniques to develop an authentication procedure that involves multiple trust chains and single intermediate node in each chain. The security is further enhanced by the idea of majority voting and the combination and calculation of continuous trust values among the nodes. It promotes the identification and isolation of malicious nodes, and provides a highly secure public key authentication service in mobile ad hoc network.

The PGP approach we implemented in this experiment distributes certificate repository among all the nodes to fit the characteristics of wireless ad hoc networks. Similar to the fully self-organizing public key management system using certificate graph proposed in [16], a relying

node has to look for a certificate chain to perform authentication. It shows from our experiment results that a relying node is able to find a trust chain usually with only one intimidate node. It is probably because the density of nodes in our network is pretty hight. Due to this reason, the PGP approach with distributed certificate repository we implemented is fairly simple as complicate algorithm on finding a trust chain is not required. This experiment focuses on the security evaluation, instead of performance evaluation, between our new authentication protocol and the PGP approach with distributed certificate repository, which is different from the work of the others.

In Figure 6.5, it shows the successful rate, failure rate, and unreachable rate of our authentication service and the PGP approach. We fix the percentage of trustable nodes at initialization to be 40% and 70% respectively and vary the percentage of malicious nodes m from 0% to 100%. With certain percentage of nodes p is initialized as trustable in the network, a node finds it generally easy to find a valid introducer in PGP. However, there is a probability m for those nodes to become malicious in public key certification. Since there is no mechanism to handling the malicious nodes in PGP, it has a pretty high fail rate in public key certification especially when the percentage of malicious nodes is high. The rise of the fail rate in PGP leads to the drop of its successful rate when the percentage of malicious nodes increases. In contrast, our authentication service has a more sophisticated trust model with a well defined quantitative authentication metric in compare with the PGP approach. Also, its public key certification involves request to multiple introducers, so a relying node is able to identify the malicious nodes by comparing the certificates in the replies. A malicious node in authentication can issue false certificates that are

different from the majority. After these malicious are discovered, they will be isolate from public key certification in the future. This leads to the higher successful rate and lower fail rate in our approach than the PGP approach. It should be noted that the unreachable rate of our scheme increase with the percentage of malicious nodes as the increased number of malicious nodes decreases the number of trustable introducers available. However, the unreachable rate keeps zero in the PGP approach as there is no mechanism to detect and isolate malicious nodes during authentication.

Figure 6.6 shows the same comparison of our approach with the PGP approach as above. The main difference is that it fixes the percentage of malicious nodes instead of the percentage of trustable nodes at initialization in this experiment. The percentage of malicious nodes is fixed at 40% and 70% respectively with the percentage of trustable nodes at initialization varies from 0% to 100%. It shows that our scheme out perform the PGP approach by having a higher successful rate and lower fail rate in average. This is mainly due to the success of our authentication service in identifying and isolating malicious nodes in public key certification as we discussed before. A special phenomenon occurs when the percentage of trustable nodes at initialization p is equal to 10%, we find that the PGP approach performs better than our approach. This may due to the fail rate of the PGP approach keeps at m and its fail rate keeps at $(1-m)$ constantly upon the percentage of trustable nodes is greater than zero. On the other hand, the malicious introducers are identified in our authentication service, so there may not be enough number of introducers in the network when the percentage of trustable nodes at initialization is only 10%. The increase of unreachable rate leads to the decrease of successful

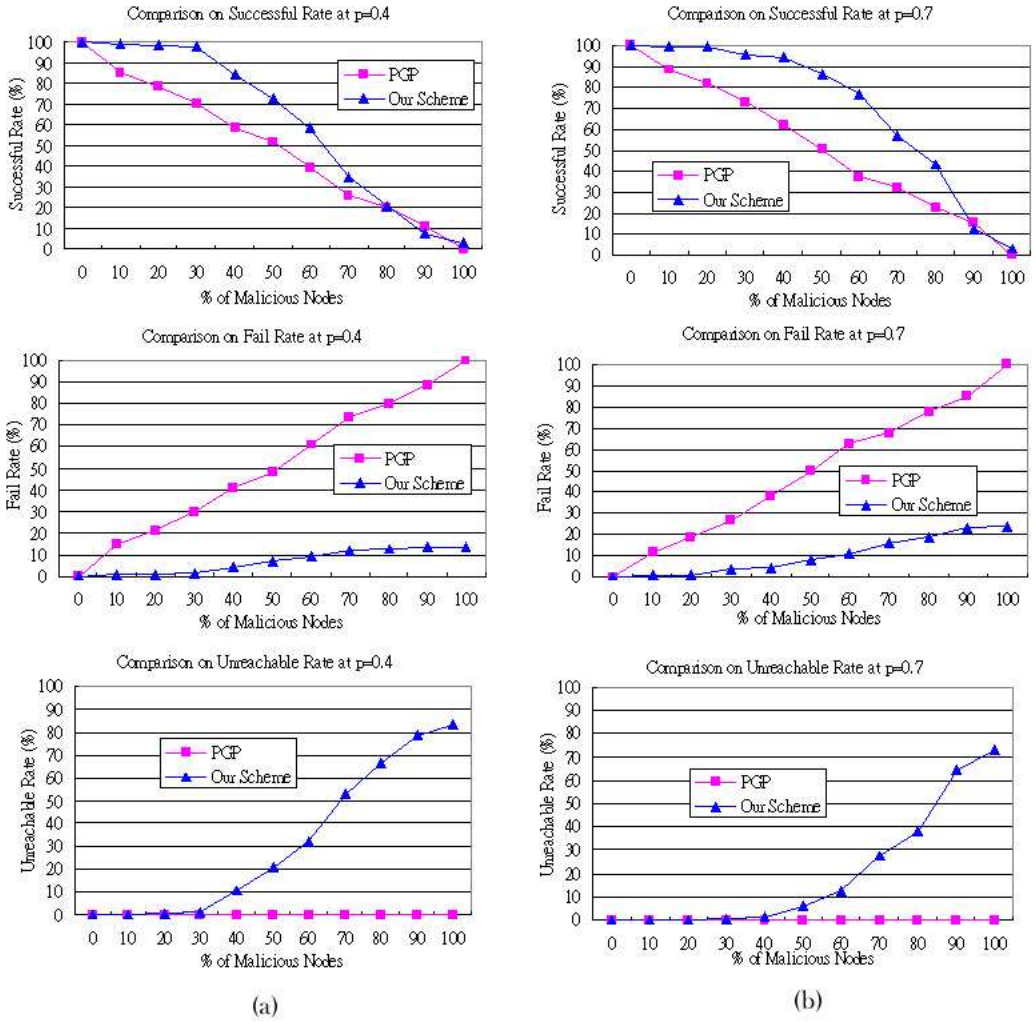

Figure 6.5: Comparison Between Our Scheme and PGP with p is Fixed

rate in the authentication service we propose subsequently. Though the PGP approach has a higher successful rate when p is equal to 10%, it gives a higher fail rate at the same time that is more harmful than our protocol.

In this part, we analysis the successful rate and fail rate of the our authentication service and the PGP approach with distributed certificate repository base on the setting of our experiment. In our analysis, the relying nodes under the PGP approach can always find an introducer with Complete trust due to certain percentage of nodes are regarded as trustable at initialization and some of them are assigned with high trust level in our network. We assume that all of the request in the PGP approach are handled by a Complete trust introducer in the following analysis. Let m_t be the percentage of malicious nodes in the set of trustable nodes at certain time t. It should be noted that the set size of the trustable nodes may vary with time.

The successful rate of PGP at time t is:

$$1 - m_t \tag{6.1}$$

The successful rate of the authentication service we propose at time t is:

$$P_1*(1-m_t)+P_2*[C_0^2*(1-m_t)^2]+P_3*[C_0^3*(1-m_t)^3+C_1^3*m_t*(1-m_t)^2], \tag{6.2}$$

where P_k is the probability of receiving k certificate replies, for $1 \leq k \leq 3$.

The fail rate of PGP at time t is:

$$m_t \tag{6.3}$$

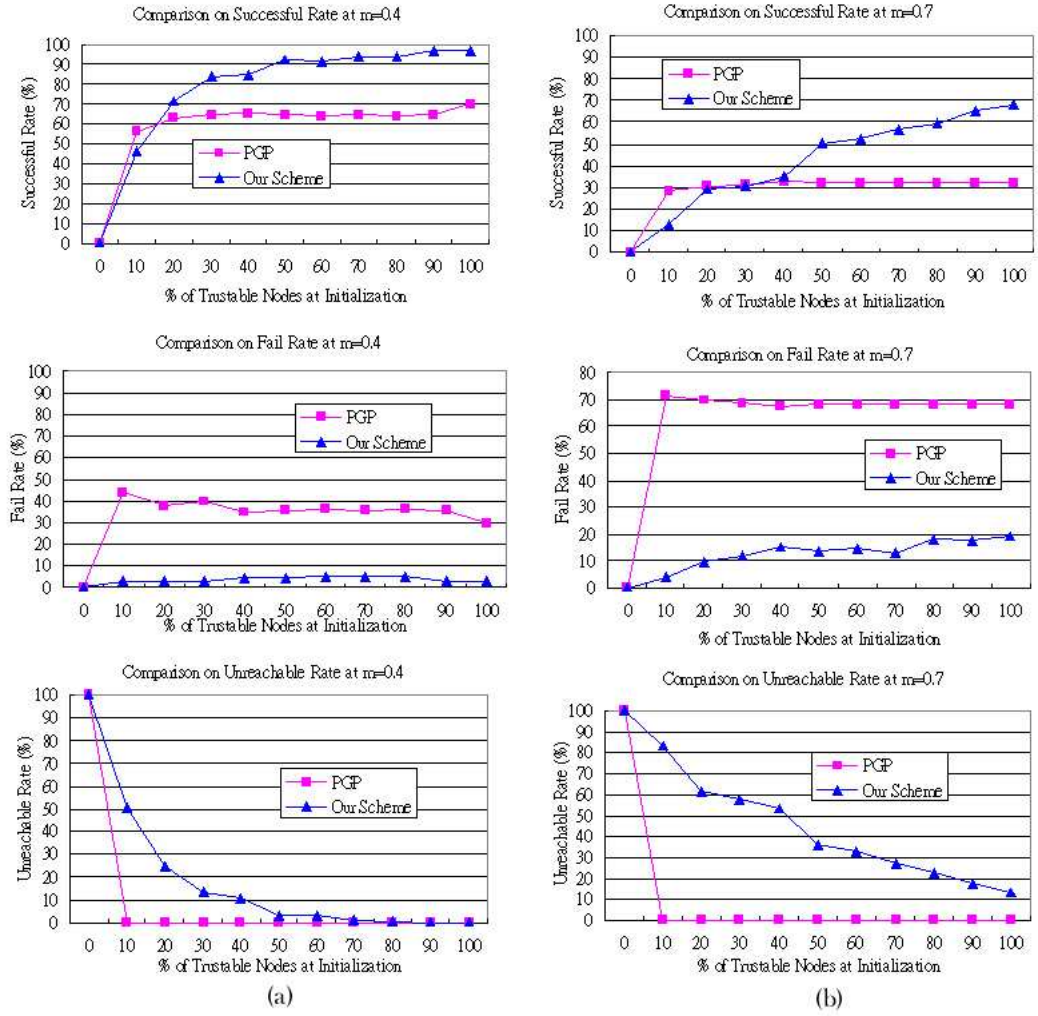

Figure 6.6: Comparison Between Our Scheme and PGP with m is Fixed

The fail rate of the authentication service we propose at time t is:

$$P_1 * m_t + P_2 * [C_0^2 * m_t^2 + C_1^2 * m_t * (1 - m_t)] + P_3 * [C_3^3 * m_t^3 + C_2^3 * m_t^2 * (1 - m_t)], \tag{6.4}$$

where P_k is the probability of receiving k certificate replies, for $1 \leq k \leq 3$.

In the PGP approach, this value m_t is equal to the percentage of malicious nodes m that we fix at the beginning of the experiment as it has no algorithm to isolate malicious nodes. However, this value m_t decreases as the number of requests made increases in the authentication service we propose as its security operations help to discover and isolate malicious nodes.

It appears that our authentication service performs better than the PGP approach in protecting network security on public key authentication. Nevertheless, it consumes more network bandwidth and CPU resources than the PGP approach. In the PGP approach, normally only one request and reply message pair are required in the case of involving introducer with Complete trust. Even there is no Complete trust introducer, two Marginally introducers take only two message pairs per request only. In our authentication service, the number of message pairs per request is as same as the number of introducers, n. Therefore, it generates more network traffic than PGP.

Message pairs per request in PGP approach is formulated as follow:

$$P_1 * 1 + P_2 * 2 = O(1),$$

where P_1 indicates the probability for having 1 Complete trust introducer and P_2 indicates the probability for having 2 Marginally trust

introducers.
Message pairs per request in our authentication service is formulated as follow:

$$O(n)$$

Also, our approach requires the relying node to compare all the certificate replies and conclude with the majority votes, which takes the amount of time:

$$O(n \log n + n) = O(n \log n)$$

In addition, the relying node has to calculate the quantitative trust value of the target node and update the trust table, which is $O(n)$. All these operations consume more CPU resources of the relying node than the PGP approach though it seems to be necessary in order to protect the network security.

The CPU cost per request in the authentication service proposed is:

$$O(n \log n)$$

The CPU cost per request in PGP approach is:

$$O(1)$$

Furthermore, the authentication service we propose assume an underlying clustering algorithm in the network. Messages for exchanging grouping information are required among the nodes, which increases the network overhead in the system as well.

6.2 Clusters Formation and Maintenance

6.2.1 Experiments Set-Up

In this experiment, we implemented the algorithms for network clusters formation and maintenance. We adopted the max-min clustering formation algorithm with some modification. In our approach, a node has to gather information from the neighboring nodes for the trust of itself. Also, the criteria for winning in a max round or min round does not depend on the node ID, but the trust value of the node from its neighbors. This experiment studies the formation of clustering based on the trust values among the nodes. Table 6.2 shows the parameter settings of this experiment. The network size is set to 1500m X 1500m and the number of nodes is set to 40. This experiment studies the network behavior on clustering structure formation and maintenance. It emulates a network with a normal node density. A node is able to contact a few neighborhoods directly in this network density. Its neighborhoods will change accordingly after moving to new locations. The number of introducer is selected to be three to make sure a node can find enough number of introducers in a cluster. This value also provides an conclusion by majority votes effectively.

Apart from the the clustering formation algorithm which is invoked at the initialization of the network, we propose some algorithms for maintaining a balance clustering structures in a highly mobile network. We believe a balance clustering structure, which means the sizes of each cluster are similar, benefits to our trust model and hence the authentication service we proposed. These algorithms adapt to the change of network topology. It provides strategy for a node to choose a suitable cluster to join while it is moving around. A node may leave the

original cluster and join another cluster after moving from one location to another. To maintain the clustering structure up-to-date, such that every nodes are joining the right clusters, each node broadcast a request to collect the cluster ID of its neighboring nodes periodically. To make it simple, we call such a period, a cycle. After receiving the reply from it neighbors, a node run the network maintenance algorithm to select and join the right cluster. We introduce three approaches and compare their performance in this experiment. The simulation is run for 40 cycles to study the behavior of the network in terms of the sizes of the clusters and the number of cycle that a node usually spends in the same cycle continuously.

Table 6.2: Simulation Parameters

Network	
Network size	1500m x 1500m
No. of nodes	40
% of malicious nodes	m
Clustering	
D-hops	3
Min. cluster size	S
Max. cluster size	L
Mobility	
Mobility	Random-Waypoint
Pause Time	20s
Maximum speed	10m/s
Public Key Certification	
Max. no. of introducers for each request	3
Min. no. of reply for each request	1
No. of query cycles	40
Simulation Time	4000s

6.2.2 Simulation Results

Evaluation on Cluster Sizes

In the first approach, a node joins the neighboring cluster with minimum size every cycle after it collected the cluster ID of its neighboring

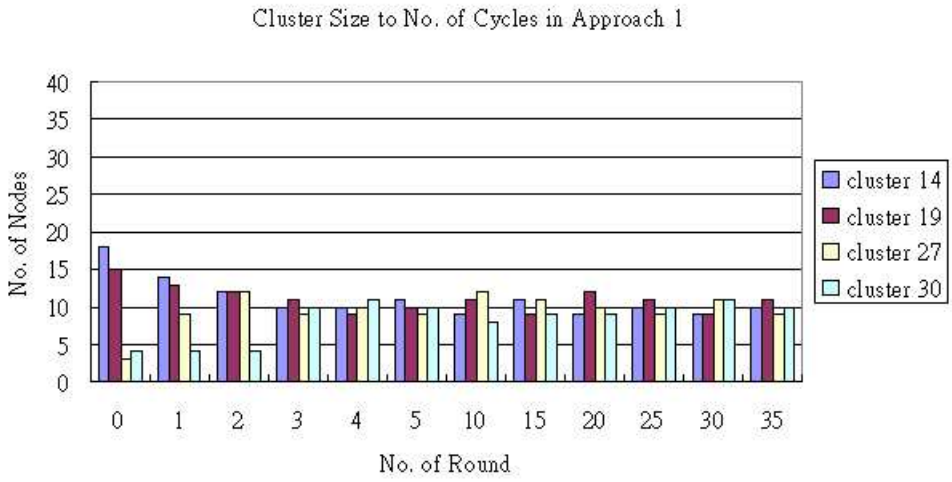

Figure 6.7: Clusters Sizes to No. of Cycles in Approach 1

nodes. Figure 6.7 shows the number of nodes of each cluster in different cycles in the network life. There are totally 40 nodes in the network. After the running of the max-min clustering algorithm, there are four clusters being formed. The cluster IDs are 40, 19, 27 and 30. It shows that the sizes of the clusters are not balance immediately after the formation of the clustering structure in cycle 0. However, each node requests for the cluster ID of its neighbors and update its cluster ID by joining the neighboring cluster with minimum size. The sizes of the clusters become balance after only several cycles. This condition was keep afterwards with this cluster maintenance algorithm.

In the second approach, a node joins the neighboring cluster with minimum only if it leaves the original cluster. A node collects the cluster IDs of its neighboring nodes and it changes to a new cluster only if the neighboring cluster IDs do not include its original cluster ID. This algorithm sounds effective, but it leads to a serious problem.

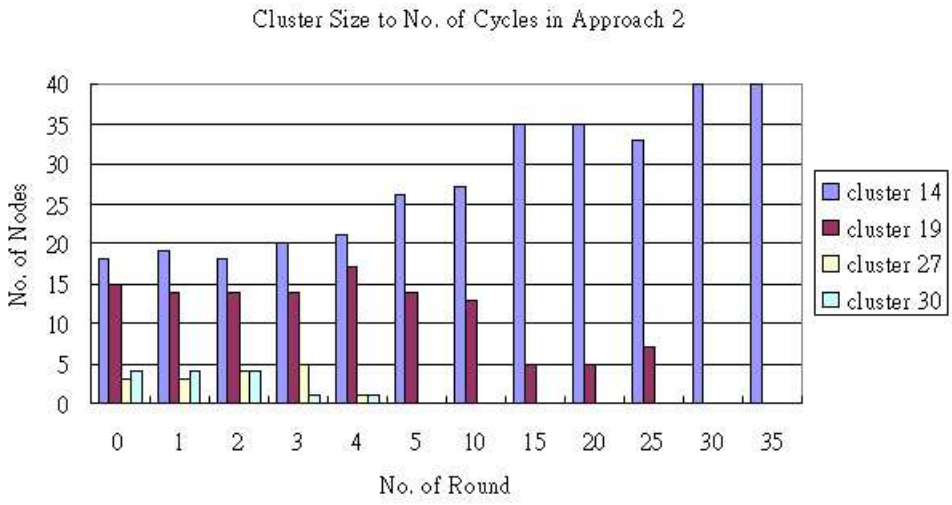

Figure 6.8: Clusters Sizes to No. of Cycles in Approach 2

From Figure 6.8, we found that all the nodes will be converged to one cluster gradually. This algorithm does not lead to a balance clustering structures in the network.

In the third approach, a node joins the neighboring cluster with minimum size if it leaves the original cluster or if the sizes of its neighboring clusters are not with a certain threshold, such as $S \leq size \leq L$. When a node received the cluster IDs of it neighbors, it checks whether its current cluster ID was included. If it is included, it means the node does not leave its original cluster. Otherwise, it joins the neighboring with minimum size, similar to the first two approaches. However, even a node does not leave its original cluster, it may still have to join another cluster if its any of its neighboring clusters is found to have a network size exceeding the defined range S and L. Figure 6.9 shows the cluster sizes are pretty balance.

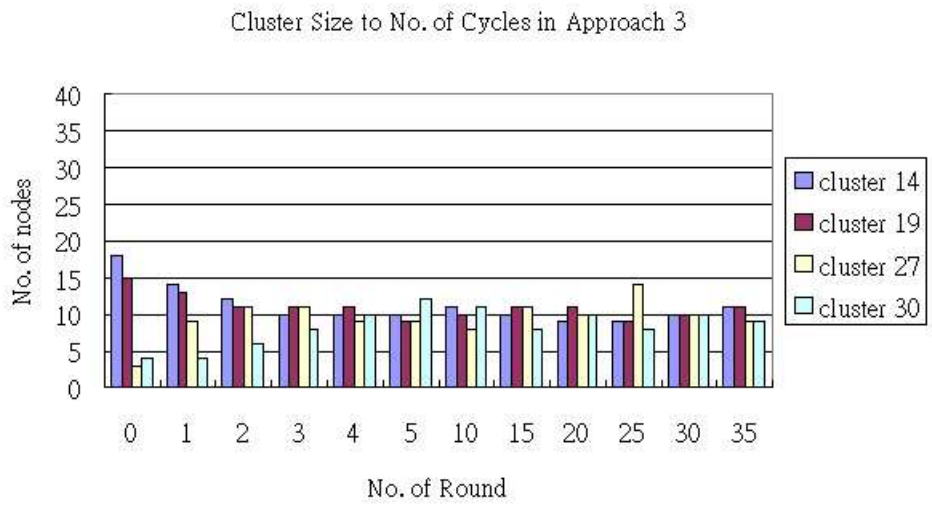

Figure 6.9: Clusters Sizes to No. of Cycles in Approach 3

Evaluation on No. of Cycles that a Node Stays in the Same Cluster

Up to now, we have studied the effectiveness in maintaining a balance clustering structures among the three approaches. From the above experiment results, both approach 1 and approach 3 brings to satisfactory result. In Figure 6.10, it further studies the performance of the three approach. It shows the number of changes in membership in each of the approaches. The number of nodes which join a new cluster means a frequent change of clustering memberships. From this figure, we found that the approach 2 out perform the others, but it can not be adopted since it does not lead to a balance network model. In comparison between approach 1 and approach 3, it shows that approach 3 involves few number of membership changes among the clusters. Therefore, approach 3 is adopted as our network maintenance algorithm.

Figure 6.11 and Figure 6.12 show the number of cycles that a node stays in the cluster. It shows that most of the nodes stay for less than

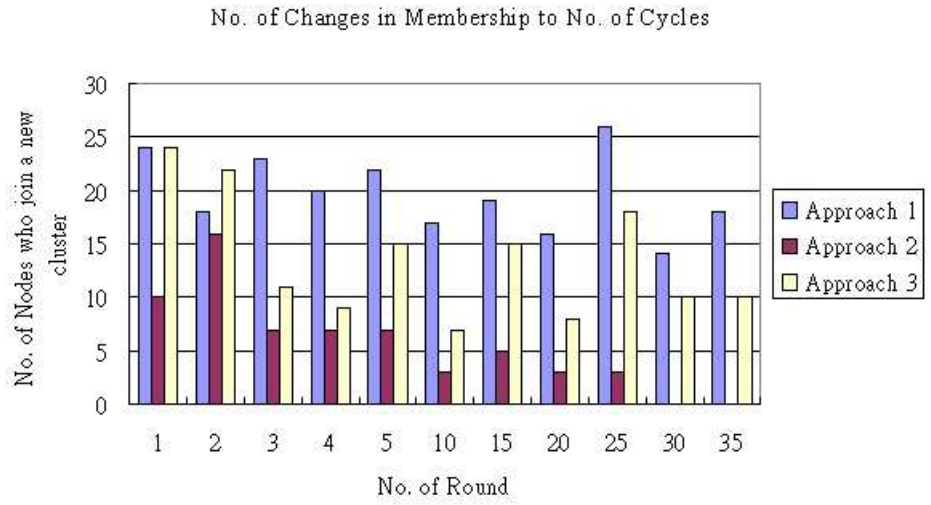

Figure 6.10: Compare the changes of memberships among the three approaches

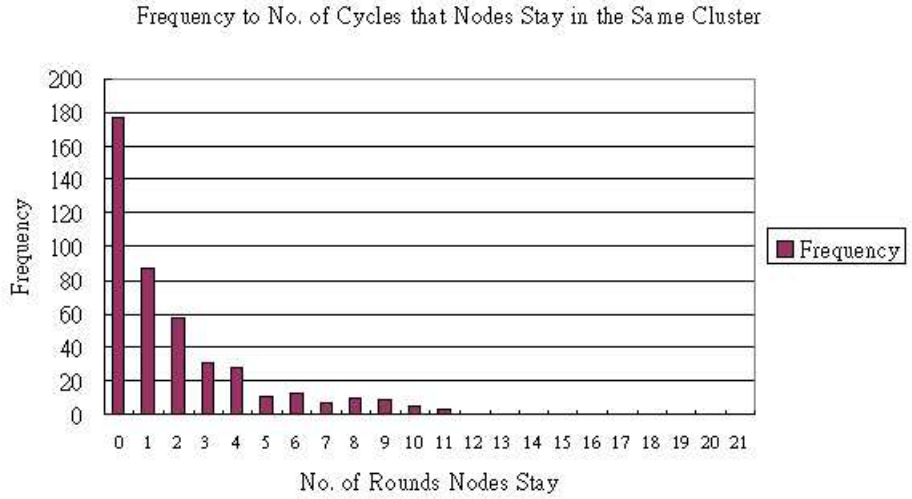

Figure 6.11: Frequency to Number of Rounds that a node stay in the same clusters

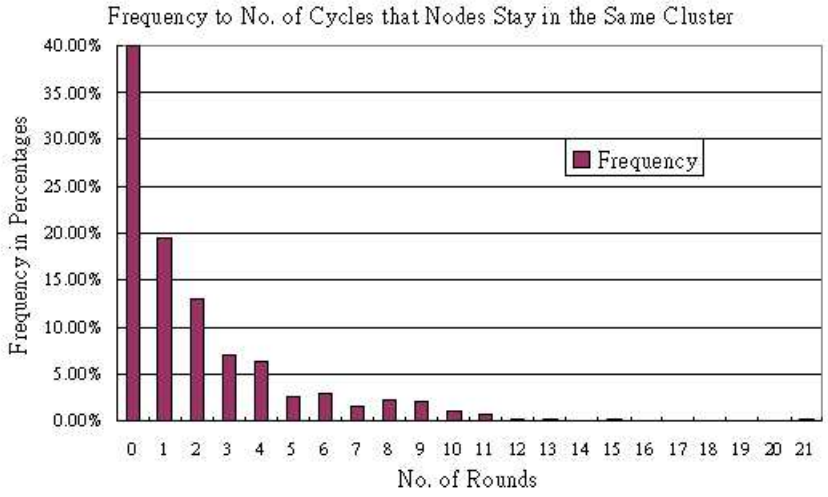

Figure 6.12: Frequency in Percentage to Number of Rounds that a node stay in the same clusters

1 round. It is because the mobility of nodes is high.

6.3 Authentication Service Based on Trust and Network Models with Clusters Formation and Maintenance

6.3.1 Experiments Set-Up

In the following experiments, it simulates a group of nodes in a network without any infrastructure. At the beginning of the experiment, the clustering algorithm will run and divide the network into several clusters. Also, the nodes will gradually build up trust relationship with each other by neighbor monitoring and public key certifications. To balance the trust relationship among the nodes, it is desired to maintain the clusters in a similar size. The nodes move from one location to

another in the mobile ad hoc network. They need to collect information about their neighboring nodes and decide joining which of the clusters in the network. It is important to update the node and maintain the balance size of the clusters. This avoids frequent clustering algorithm revocations and the merge and divisions among the clusters. Table 6.3 shows the parameter settings in the following experiments. This experiment involves 40 nodes with network size of 1500m X 1500m, or 100 nodes with network size of 3000m X 3000m. A node usually have several neighboring nodes under this density. This settings emulate real scenario that include the leave and join of the nodes from one cluster to another. The size of each cluster is also maintained within a certain range for balance and efficient operations in the network. The number of introducers is selected to be three again to give a high chance of successful public key certification by majority votes.

Table 6.3: Simulation Parameters

Network	
Network size	1500m x 1500m or 3000m x 3000m
No. of nodes	n
% of malicious nodes	m
Mobility	
Mobility	Random-Waypoint
Pause Time	20s
Max. speed	10m/s
Clustering	
D-hops	3
Min. cluster size	S
Max. cluster size	L
NeighborMonitoring	
No. of cycles required to identify malicious neighbors	2
PublicKeyCertification	
Max. no. of introducers for each request	3
Min. no. of reply for each request	1
No. of cycles	r
Simulation Time per cycle	110-120s

There are totally 11 cases described in Table 6.4. It contains all

the possible cases in public key certification with 3 introducers. Case 0 represents the situation that there is not enough number of introducers for supporting this request, so the request message will not be sent. It leads to the increase of the unreachable rate. Case 1 to case 10 represent the cases that the request messages are send and some or all public key certificates are received from the introducers. The symbol 'O' indicates a received certificate which is providing the correct public key certificate of the target node, while the symbol 'X' indicates a received certificate which is providing an incorrect public key certificate of the target node. According to the definitions of the symbols, case 1, 5, and 8 means the requesting node receives three, two, and one correct public key certificates from the introducers respectively. They all lead to a successful public key certification request. Requesting node in case 2 receives two correct and one incorrect public key certificates. It can still conclude with a correct public key successfully by majority votes. In contrary, the number of incorrect public key certificates in case 3 and 6 are less than or equal to the number of correct public key certificates. They are unable to conclude with a correct public key certificates by majority votes. Similar situations in cases 4, 7, and 9, where all the received public key certificates are incorrect. All of them lead to a failure public key certification on the target node. Although the request messages are sent to the introducers, it does not guarantee the requesting node can receive all the replies from these introducers. In some cases, they receive less than three replies or even receive 0 replies, just like in case 10. In case 10, the public key of the target node is regarded as unreachable as no public key certificates can be obtained from the introducers.

In a cycle, each of the nodes asks for the neighboring information

and updates the clusterhead that it belongs to. It also makes request on the public key certificates of one other node in each cycle. The requesting node conclude the correct public key of the target node by majority votes. At the same time, it may identify the suspicious introducers who provide incorrect public keys of the target node. Since the requesting node relies on majority vote to identify the malicious introducers, it does not always lead to an accurate identification. For example, in case 3 and 6, the number of correct certificate is only one, it is impossible for the requesting node to identify which is true and which is false among the certificates received. In its view, all the introducers providing different public key certificates are suspicious. It the requesting node isolates all of these introducers, a good node in each case will be isolated accidentally. This brings to the false-positive error in identifying malicious nodes in the network. In contrary, the requesting node only receive one certificate reply in case 9. Even the received public key of the target node is incorrect, it is still unable to figure it out. The failure in identifying the dishonest introducer leads to the false negative error in the identification of malicious nodes in the network.

6.3.2 Simulation Results

Evaluation on Ratings with Neighbor Monitoring

In this experiment, it implements the neighbor monitoring algorithm to facilitate the identification of malicious nodes in the network. When a node stay in the same cluster for a certain period of time, it may have ability to detect and identify the malicious nodes in the network.

Experiment result is shown in Figure 6.13 with n=40, m=0.3, and r=35. At the initialization, there are a group of nodes in the network

Table 6.4: Possible Cases in Public Key Certifications with 3 introducers

Network						
ID	State	Successful	Fail	Unreachable	False+	False-
0	Not enough Introducers			√		
1	OOO	√				
2	OOX	√				
3	OXX		√		√	
4	XXX		√			
5	OO	√				
6	OX		√		√	
7	XX		√			
8	O	√				
9	X		√			√
10	No reply			√		

without any infrastructure. The clustering formation algorithm is run, and then a number of clusters are formed. The nodes do not know each other unless they build up trust relationship with their neighboring nodes or they ask for public key certificates via some introducing nodes. During the first few rounds of the operations, the high unreachable rate is because the trust relationships with other nodes are very limited. The number of trust worthy nodes will increase with the time when a node moves around and meet new neighbors or they ask for public key certificates of other nodes. If there is not enough number of trust worthy nodes, a node can hardly find any nodes to be the introducers in public key certification. The unreachable rate will drop gradually after more rounds.

Figure 6.14 shows the same experiment running for 100 number of rounds. The identification of malicious nodes heavily relies on the monitoring power of the neighboring nodes. A node can discover its neighbor is malicious only if it has observed that node for a period of time (say, for a few rounds). However, the nodes in the network are highly mobile, a node usually does not stay in the same cluster for

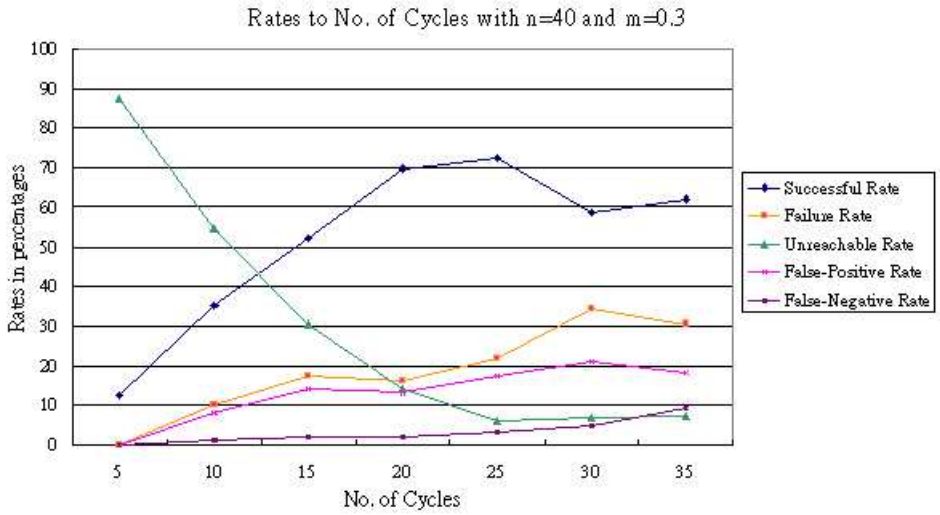

Figure 6.13: Rates to No. of Cycles with n=40, m=0.3, and r=35

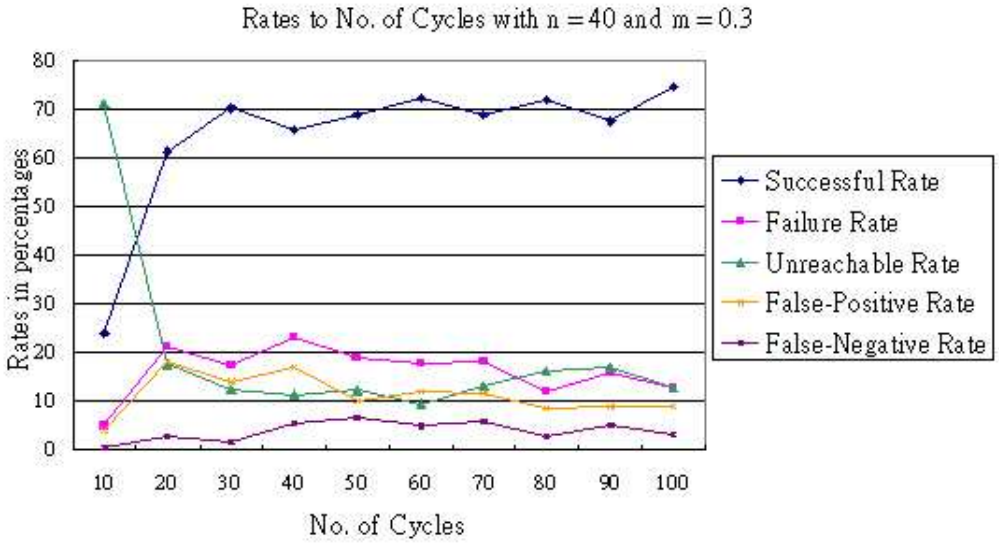

Figure 6.14: Rates to No. of Cycles with n=40, m=0.3, and r=100

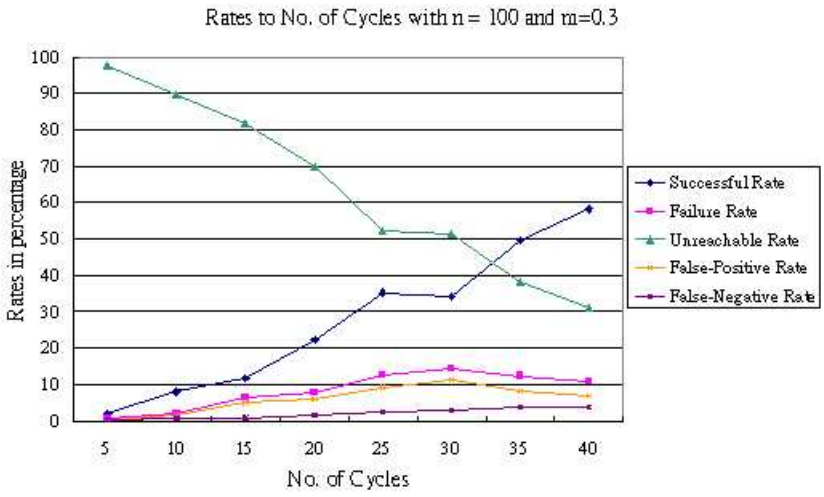

Figure 6.15: Rates to No. of Cycles with n=100, m=0.3, and r=40

very long time. The monitoring work is therefore quite difficult to be carried out. The performance is not very impressive as the successful rate is only around 70% which is equal to the percentage of honest nodes in the network. Of course, the nodes in the network do not know each other at the beginning, so it takes time to build up the trust relationships among them. The failure rate is improved to 10% which is lower than 30% of malicious nodes in the network. The performance of the operations can be improved by identifying malicious nodes in public key certification.

Figure 6.15 shows the same experiment with 100 number of nodes. The successful rate is pretty low and the unreachable is quite high even the number of cycles reach 40. In comparing with the experiment in the network with only 40 nodes, it shows that a network with high number of nodes takes longer time to build up trust relationships among nodes in order to reach a stable condition in public key certification.

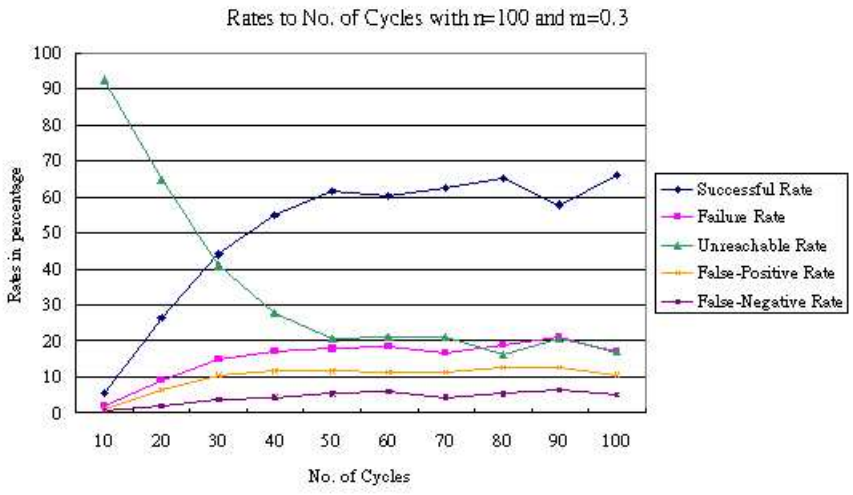

Figure 6.16: Rates to No. of Cycles with n=100, m=0.3, and r=100

Figure 6.16 shows the same experiment with 100 number of nodes and running for 100 rounds. This experiment shows that the successful rate is not so satisfactory with almost 70%. However, the failure rate is keep lower than 20%, which is less than the percentage of malicious nodes in the network. It shows that the unreachable rate gradually drops to around 20% after around 50 cycles.

Figure 6.17 shows the experiment result with the parameters n=40 and r=100. The percentage of malicious m is defined as 70%, which represents a hostile network condition. The successful rate is above 40% and the failure rate is around 30%. The unreachable rate is around 30%. This result is a bit better than randomly select a node as introducer from the network with the assumption that an introducer is reachable. Since this assumption is not very realistic, so normally a random algorithm is expected to perform even worse.

Figure 6.18 shows the experiment result with the same parameters

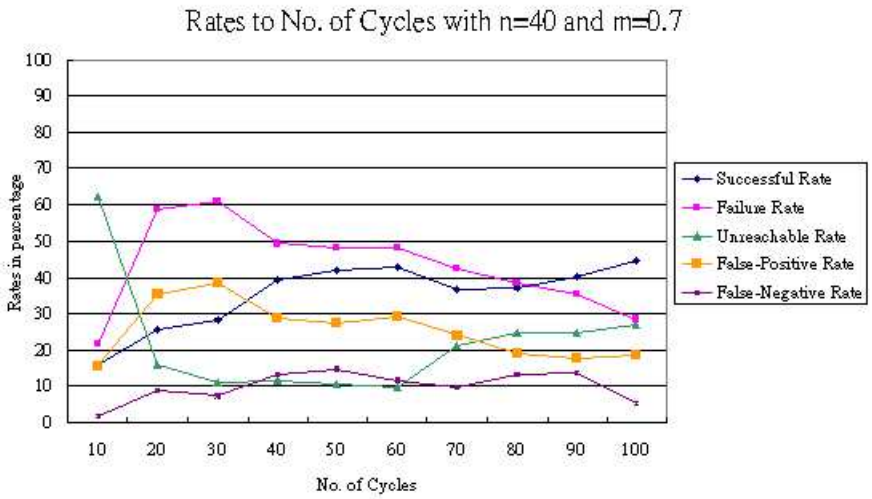

Figure 6.17: Rates to No. of Cycles with n=40, m=0.7, and r=100

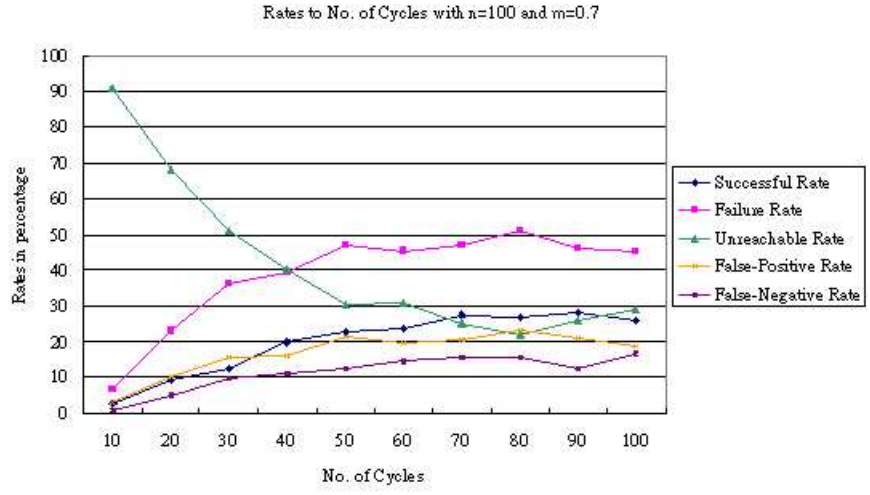

Figure 6.18: Rates to No. of Cycles with n=100, m=0.7, and r=100

setting as the above one, but the network contains more number of nodes. It shows that the successful rate does not have any improvement in compare with the single and random introducer searching algorithm. However, the failure rate is maintained lower than 50% which is lower than the 70% of malicious nodes.

In summary, the above experiment shows the neighboring monitoring power does not lead to great improvement in the successful rate of public key certifications. It is probably because the mobility of nodes is too high, such that the neighboring monitoring power does not perform as well as expected. We believe that it takes time for a node to collect enough data and identify its neighboring nodes as malicious. From the experiment result about mobility in the previous experiment, we found that most of the nodes stay in the same cluster for less than one cycle. In this experiment, we have set the number of cycles required to identify malicious neighbors to be 2. Therefore, it is reasonable that many of the nodes are unable to discover its malicious neighbors while they move around. It shows that the neighbor monitoring power does not perform so effectively in highly mobile network environment. In order to protect the network security, it is necessary to rely on other security operations, like the identification of suspicious nodes in public key certifications. The experiments result will be presented in the following sub-sections.

Evaluation on Ratings with Neighbor Monitoring and Isolation of Suspicious Nodes

To improve the security of the network, we include the identification of suspicious nodes in the operations of public key certification. In this experiment, suspicious nodes are not only be identified by neigh-

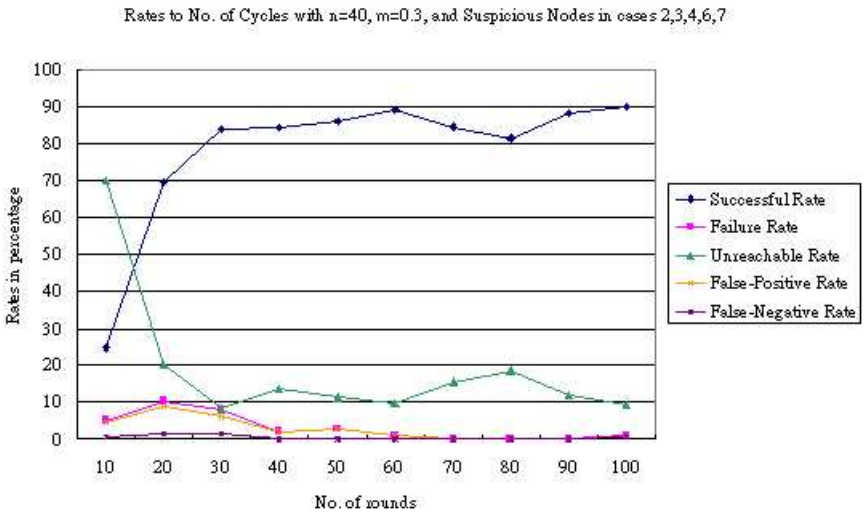

Figure 6.19: Rates to No. of Cycles with n=40, m=0.3, r=100, and Suspicious Nodes in cases 2,3,4,6,7

bor monitoring and the trust value of the target node in public key certifications. Also, they can be identified by analyzing the received public key certificates. The introducers who provide certificates different from the others are identified as suspicious and be isolated from being selected as introducers. Suspicious nodes are present in cases 2, 3, 4, 6, and 7. It should be noted that case 2 and 6 lead to false positive error. It means that a honest node may be falsely identified as malicious. Case 9 leads to false negative rate as well as the single reply can makes no comparisons with other certificates, so it is always thought to be correct.

Figure 6.19 shows the experiment result with n=40, m=0.3 and r=100. The successful rate is greatly improved with the identification of suspicious nodes during the process of public key certification. The failure rate is very low. It indicates that the identification and isola-

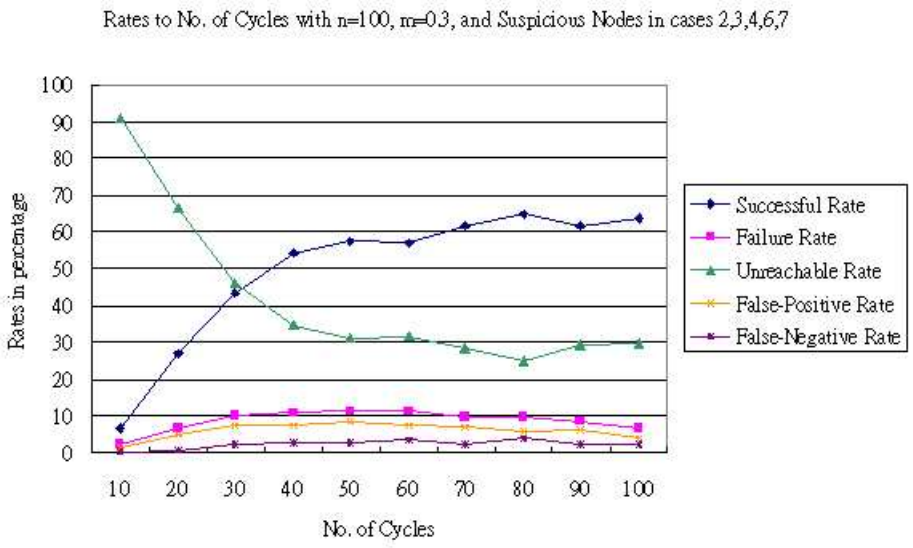

Figure 6.20: Rates to No. of Cycles with n=100, m=0.3, r=100, and Suspicious Nodes in cases 2,3,4,6,7

tion of suspicious nodes in cases 2, 3, 4, 6, and 7 effectively avoid the number of incorrect public certificates replied by malicious introducers. The high successful rate and the low failure rate bring satisfactory authentication results in the network.

Figure 6.20 shows the experiment result with n=100, m=0.3 and r=100. The successful rate is not so high in this case, though the failure rate is greatly reduced to only 10%. The successful rate was found to be lower than 70%, but the failure rate is much lower than the percentage of malicious nodes 30% in the network. The unsatisfactory successful rate is mainly due to the high unreachable rate. We believe that it is because cases 3 and 6 lead to false positive error in identification of suspicious nodes.

Figure 6.21 shows the experiment result with n=40, m=0.7 and r=100. The successful rate is just between 60% and 70% which is

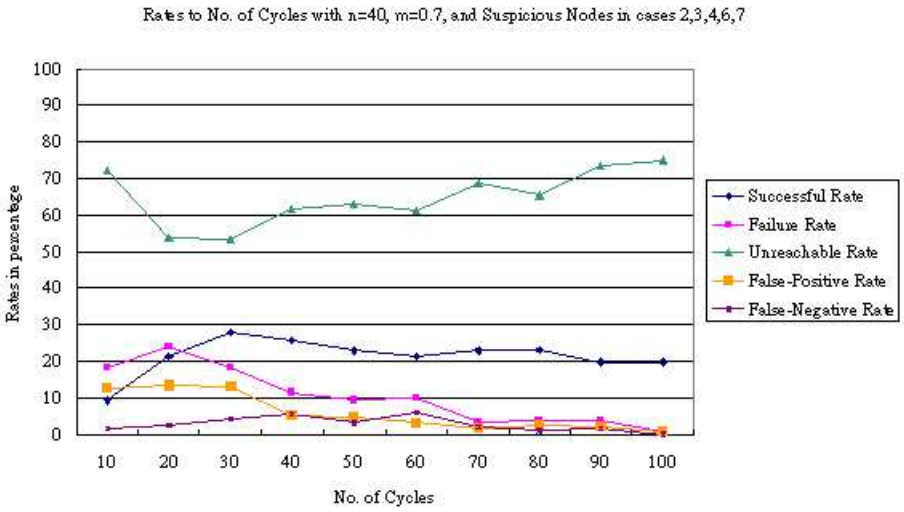

Figure 6.21: Rates to No. of Cycles with n=40, m=0.7, r=100, and Suspicious Nodes in cases 2,3,4,6,7

not so satisfactory. The failure is pretty low on the other hand. We notice that the unreachable rate is pretty high, which leads to the low successful rate. It is because many of the honest nodes are falsely identified as suspicious and isolated from taking the role of introducers. In a hostile environment, where the number of honest nodes is low, it is easy to lead to the problem of not enough introducers. This is the major reason for the high unreachable rate as well.

Figure 6.22 shows the experiment result with n=100, m=0.7 and r=100. Similar to the previous figure, the successful rate is low and the unreachable is high. Under a network with high malicious rate, the situation becomes worse as the number of case 3 and 6 increases. The honest nodes in the network are not many under a network with 70% of malicious nodes. If they are falsely identified as suspicious and not be selected as introducers, then a requesting may not find enough number

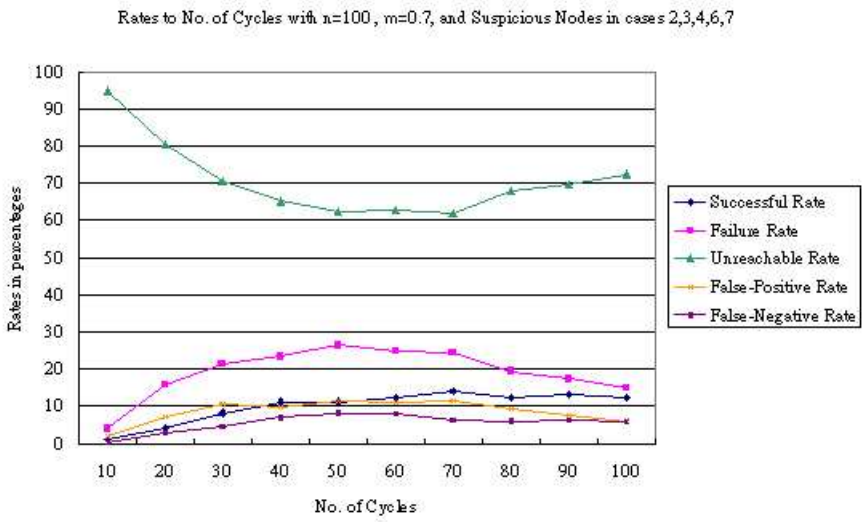

Figure 6.22: Rates to No. of Cycles with n=100, m=0.7, r=100, and Suspicious Nodes in cases 2,3,4,6,7

of introducers in the request of public key certificates. To improve the successful rate and reduce the unreachable rate, we have to reduce the false-positive error rate. The experiment in the next sub-section will focus on this problem.

Evaluation on Ratings with Neighbor Monitoring and Isolation of Malicious Nodes

Similar to the previous experiment, suspicious nodes can be identified by neighbor monitoring and update the trust value of the target node in public key certification in this experiment. Unlike the pervious experiment, the requesting node will avoid to identify suspicious nodes in some cases where it expects trust-worthy introducers still exist in that reply. It means in the cases that the requesting node is able to identify malicious introducers confidently, like in case 3 and 6, the node may

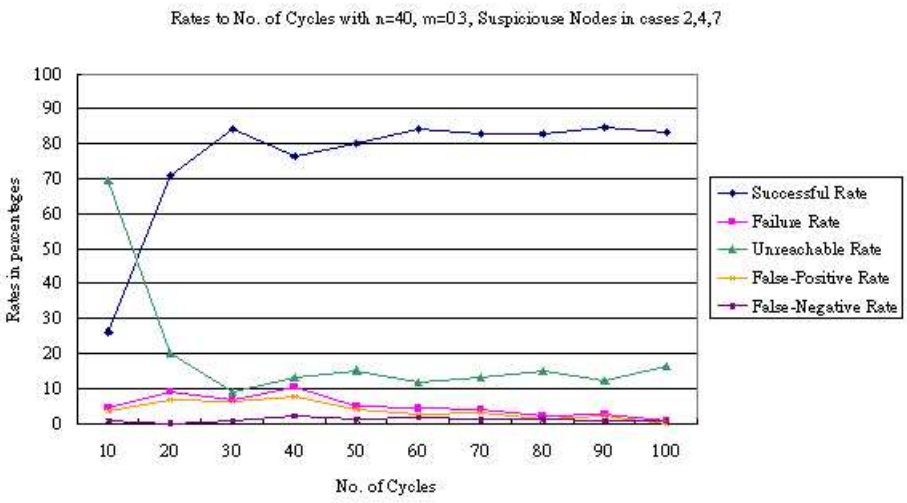

Figure 6.23: Rates to No. of Cycles with n=40, m=0.3, r=100, and Suspicious Nodes in cases 2,4,7

consider to keep all the suspicious introducers instead of isolating all of them. Although it is hard to make the judgement on keep the introducers or not, a node still can try to do it by considering the trust values of the introducers and their past records in public key certifications. If this policy is carried out, it is possible to avoids the false-positive errors brought by cases 3 and 6. However, the reduce of false-positive error also prevents some malicious nodes to be isolated immediately in cases 3 and 6. Anyway, these malicious nodes can still be discovered in the future when they are requested to be introducers again.

Figure 6.23 shows that the successful rate is quite high and the failure rate is almost zero after running for 100 cycles. It indicates that the new policy on the identification of suspicious nodes brings satisfactory result in public key certifications. However, when compares with Figure 6.19 in the previous, the successful rate in this figure is not

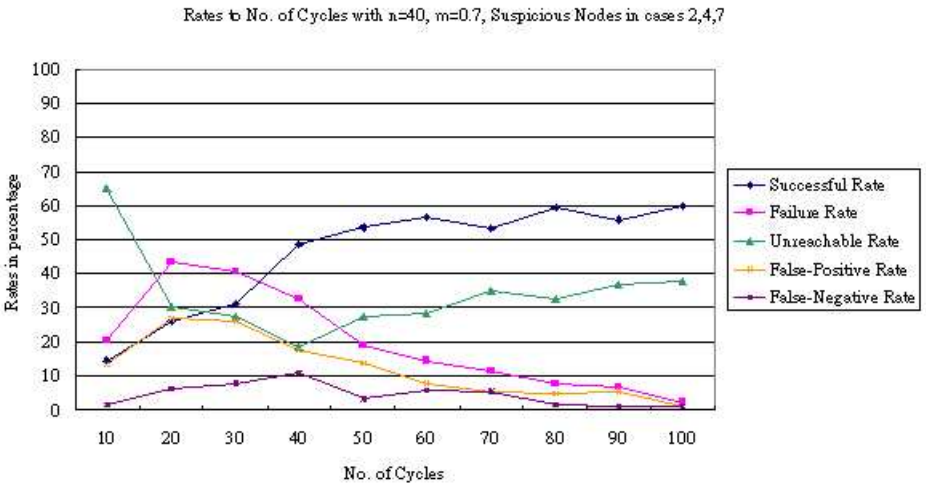

Figure 6.24: Rates to No. of Cycles with n=40, m=0.7, r=100, and Suspicious Nodes in cases 2,4,7

as good as that. The main reason is that some of the malicious nodes are not isolated in cases 3 and 6, so it takes longer time for a node to discover the malicious nodes in the network. When the malicious nodes are not isolated, they are still possible to be selected as introducers and provide with false certificates which may decrease the successful rate.

Figure 6.24 shows the successful rate is greatly improved with this strategy. It maintains a high successful rate under a hostile situation and with extremely low failure rate. It indicates that the new strategy on the identification of suspicious nodes brings satisfactory result in public key certifications in a hostile network environment. In comparison with Figure 6.21, the successful rate is greatly improved with the unreachable rate greatly reduced as well. In an environment full of malicious nodes, it is hard to find enough number of trust-worthy introducers for public key certifications. Therefore, the keeping of honest

introducer in cases 3 and 6 can make a great different in the results as it provides the requesting more choices on selecting trust-worthy nods as introducers for certification. It effectively increases the successful rate and decreases the unreachable rate in the network.

In conclusion, the new strategy on keeping some of the introducers in a failure public key certification avoids the false-positive errors in the network. This effectively reduces the unreachable rate especially in a hostile network environment where trust-worthy nodes are extremely valuable. It is understandable that deciding isolate or not is sometimes an uneasy task to the requesting node. A node is suggested to make the decision by referencing not only the public key certificates received, but also the trust value and past records of the introducers. We believe that the simple approach on isolating all the suspicious nodes which provide with different certificates, which presented in the previous experiment, can be adopted in a less hostile environment. In such an environment, higher number of trust-worthy nodes are usually available to be potential introducers, so the false-positive errors do not give too much bad effects to the network. It also provides a faster way to isolate malicious nodes then the new approach. However, in an environment with high percentage of malicious nodes, it is recommended to adopted the approach with more relax strategy on identification of suspicious nodes. This approach leaves the malicious nodes serving the network, rather then removing the small portions of trust-worthy nodes. If we select the appropriate strategy according to different network conditions, our authentication service is expected to perform better.

□ **End of chapter.**

Chapter 7

Conclusion

In conclusion, this work aims at providing a secure, scalable and distributed authentication service that assures the correctness of public key certification in wireless ad hoc networks with the presence of malicious nodes. Our system does not rely on any trusted-third party, such that authentication is performed in a distributed manner. New nodes are introduced by other trust-worthy nodes in the same group. Nodes in the network monitor the behavior of each other and update their trust tables accordingly. We suggest a well-defined trust model and a network model to develop our public key authentication services. The trust model allows nodes to monitor and update trust values of each other in a distributed manner. The network model is clustering-based which makes it convenient to behavior monitoring and provides high available on public key certification. Based on the above models, we propose a new mechanism to perform public key authentication in wireless ad hoc networks. The security operations proposed include carrying out public key certification and update of trust tables in a novel way. Also, there are operations enable a node to discover and isolate malicious nodes which sign false public key certificates.

Extensive experiments are completed to evaluate the performance of our authentication protocol in the security perspective. A number of metrics, including the successful rate, fail rate, unreachable rate, false-positive and false-negative error rates are evaluated. Parameters like the percentage of trust-worthy nodes and percentage of malicious nodes in the network are fixed at different values in certain experiments. In addition, comparison is made between the authentication service we propose and the PGP approach with distributed certificate repository. The experiment results show that our authentication service performs well in protecting the network security in a hostile environment. Apart from that, a number of experiments are run to demonstrate the network formation and maintenance algorithms are adaptive to mobility of nodes and able to keep the network in a balance clustering structure. The neighbor monitoring power and different strategies on the identification of suspicious nodes are implemented as well. Experiment results show the effectiveness in providing secure authentication service using different strategies in various adversary levels. In conclusion, our approach provides a secure and highly available authentication service in wireless ad hoc network.

In the future, the work may be extended to handle the existence of malicious peers in the network. More advance security operations will be included to prevent the harms from the colluding nodes.

□ **End of chapter.**

Bibliography

[1] A. Abdul-Rahman. The PGP trust model. *EDI-Forum: the Journal of Electronic Commerce*, Apr. 1997.

[2] A. Abdul-Rahman and S. Halles. A distributed trust model. In *New Security Paradigms Workshop '97*, pages 48–60, 1997.

[3] K. Aberer. P-grid: A self-organizing access structure for p2p information systems. In *Proc. of the 9th International Conference on Cooperative Infomation Systems (CoopIS'01)*, pages 179–194, 2001.

[4] K. Aberer and Z. Despotovic. Managing trust in a peer-2-peer information system. In *Proc. of the 10th International Conference on Information and Knowledge Management (CIKM'01)*, pages 310–317, 2001.

[5] A. D. Amis, R. Prakash, T. H. P. Vuong, , and D. T. Huynh. Max-min d-cluster formation in wireless ad hoc network. In *Proc. of the 19nd Annual Joint Conference of the IEEE Computer and Communications Societies (Infocom'00)*, pages 32–41, Israel, Mar. 2000.

[6] N. Asokan and P. Ginzborg. Key agreement in ad hoc networks. *Computer Communications*, 23(17):1627–1637, 2000.

[7] D. Balfanz, D. Smetters, P. Stewart, and H. Wong. Talking to stranger: Authentication in ad-hoc wireless networks. In *The 9th Annual Symposium on Network and Distributed System Security (NDSS'02)*. Internet Society, 2002.

[8] S. Basagni. Distributed clustering for ad hoc networks. In *Proc. of ISPAN'99 International Symposium On Parallel Architectures, Algorithms, and Networks*, pages 310–315, 1999.

[9] S. Basagni, K. Herrin, E. Rosti, and D. Bruschi. Secure pebblenets. In *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'01)*, pages 156–163, 2001.

[10] T. Beth, B. Malte, and K. Birgit. Valuation of trust in open networks. In *Proc. of the Conference on Computer Security*, pages 3–18, New York, 1994. Sprintger-Verlag.

[11] J.-Y. L. Boudec and S. Buchegger. Nodes bearing grudges: Towards routing security, fairness, and robustness in mobile ad hoc networks. In *Workshop on Parallel, Distributed Network-based Processing*, pages 403–410, Canary Islands, Spain, Jan. 2002.

[12] J.-Y. L. Boudec and S. Buchegger. Performance analysis of the confidant protocol (cooperation of nodes: Fairness in dynamic ad-hoc networks). In *Proc. of the 3rd ACM Interational Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'02)*, pages 80–91, Lausanne, Switzerland, June 2002.

[13] J. Broch, D. A. Maltz, D. B. Johnson, Y. C. Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *The 4th Annual International Conference on Mobile Computing and Networking (MobiCom'98)*, pages 85–97, Oct. 1998.

[14] S. Capkun, L. Buttyan, , and J.-P. Hubaux. Small worlds in security systems: an analysis of the PGP certificate graph. In *Proc. of the ACM New Security Paradigms Workshop*, pages 28–35, 2002.

[15] S. Capkun, L. Buttyan, and J.-P. Hubaux. Self-organized public-key management for mobile ad hoc networks. *ACM Mobile Computing and Communications Review (MC2R'02)*, 6(4):55–62, 2002.

[16] S. Capkuny, L. Buttyan, and J.-P. Hubaux. Self-organized public-key management for mobile ad hoc networks. Swiss Federal, Institute of Technology Lausanne (EPFL) Techical Report, June 2002.

[17] Y. P. Chen and A. L. Liestman. Approximating minimum size weakly-connected dominating sets for clustering mobile ad hoc networks. In *The 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computer (MobiHoc'02)*, pages 165–172, June 2002.

[18] Y. P. Chen and A. L. Liestman. A zonal algorithm for clustering ad hoc networks. In *International Journal of Foundations of Computer Science*, pages 305–322, Apr. 2002.

[19] Y. P. Chen, A. L. Liestman, and J. Liu. Clustering algorithms for ad hoc wireless networks. In *Ad Hoc and Sensor Networks*. Nova Science Publisher, 2004.

[20] S. Corson and J. Macker. Approximating minimum size weakly-connected dominating sets for clustering mobile ad hoc networks. *Network Working Group rfc 2501*, Jan. 1999.

[21] C.-K. T. Elizabeth M. Royer, Santa Babara. A review of current routing protocols for ad hoc mobile wireless networks. *IEEE Personal Communications*, 6(2):46–55, April 1999.

[22] C. Elliott and B. Heile. Self-organizing, self-healing wireless networks. In *Proc. 2000 IEEE Aerospace Conference*, pages 149–156, 2000.

[23] C. Ellison, W. Frantz, B. Lampson, R. Rivest, B. Thnomas, and T. Ylonen. SPKI certificate theory. Internet RFC 2693, 1999.

[24] Y. Fernandess and D. Malkhi. K-clustering in wireless ad hoc networks. In *Workshop on Principles of Mobile Computing*, pages 31–37, Toulouse, France, Oct. 2002. ACM.

[25] W. Ford. Public-key infrastructure interoperation. In *Proc. 1998 IEEE Aerospace Conference*, pages 329–333, 1998.

[26] Y. Frankel and Y. G. Desmedt. Parallel reliable threshold multisignature. Technical Report TR-92-04-02, Dept. of EECS, University of Wisconsin-Milwaukee, 1992.

[27] Y. Frankel, P. Gemmel, P. MacKenzie, and M. Yung. Optimal-resilience proactive public-key cryptosystems. In *38th Annual Symposium on Foundations of Computer Science (FOCS'97)*, pages 384–393, Miami Beach, Florida, Oct. 1997.

[28] Y. Frankel, P. Gemmell, P. Mackenzie, and M. Yung. Proactive RSA. In *Proc. of the 17th Annual International Cryptology Conference on Advances in Cryptology*, pages 440–454, 1997.

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

[30] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Robust and efficient sharing of RSA functions. *Journal of Cryptology*, 13(2):273–300, 2000.

[31] M. Gerla and J. T. C. Tsai. Multicluster, mobile, multimedia radio network. *ACM-Baltzer Journal of Wireless Networks*, 1(3):255–256, 1995.

[32] S. Giordano. Mobile ad-hoc networks. *Handbook of Wireless Networks and Mobile Computing, Wiley*, 2000.

[33] L. Gong. Increasing availability and security of an authentication service. *IEEE Journal on Selected Areas in Communications*, 11(5):657–662, June 1993.

[34] M. Hietalahti. Key establishment in ad-hoc networks. Laboratory for Theoretical Computer Science, Helsinki University of Technology, 2001.

[35] A. P. Y.-C. Hu and D. B. Johnson. Sead: Secure efficient distance vector routing for mobile wireless ad hoc network. In *Proc. of the 4th IEEE Workshop on Mobile Computing Systems & Applications (WMCSA'02)*, pages 3–13, Calicoon, NY, June 2002.

[36] J.-P. Hubaux, L. Buttyan, and S. Capkun. The quest for security in mobile ad hoc networks. In *Proc. of the 2nd ACM International Symposium on Mobile ad hoc networking & computing (MobiHoc'01)*, pages 146–155, Long Beach, CA, USA, Oct. 2001.

[37] IETF. Internet x.509 public key infrastructure. draft-ietf-pkix-roadmap-06.txt, 2002.

[38] N. A. Inc. and its Affiliated Companies. How pgp works. *Chapter 1 of the document Introduction to Cryptography in the PGP 6.5.1 documentation*, 1990–1999.

[39] A. N. S. Institute. PKI practices and policy framework. ANSI X9.79, 2000.

[40] Q. Jiang, D. S. Reeves, , and P. Ning. Improving robustness of pgp keyrings by conflict detection. *RSA-CT Cryptographer's Track*, 2004.

[41] J.Kohl and B.Neuman. The kerberos network authentication service (version 5). RFC-1510, Sept. 1993.

[42] D. B. Johnson, Y.-C. Hu, and A. Perrig. Ariadne: A secure on-demand routing protocol for ad hoc networks. In *The 8th ACM International Conference on Mobile Computing and Networking (MobiCom'02)*, pages 12–23, Sept. 2002.

[43] S. D. Kamvar, M. T. Schlosser, and H. G.-Mollina. The eigentrust algorithm for reputation management in P2P networks. In *The Twelfth International World Wide Web Conference*, pages 640–651, Budapest, Hungary, May 2003.

[44] V. Karpijoki. Security in ad hoc networks. Helsinki University of Technology, Tik-110.501 Seminar on Network Security, Telecommunications Software and Multimedia Laboratory, 2000.

[45] S. Kent. Evaluating certification authority security. In *Proc. 1998 IEEE Aerospace Conference*, volume 4, pages 319–327, May 1998.

[46] J. Kohl and B. Neuman. The kerberos network authentication service (version 5). RFC-1510, June 1991.

[47] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang. Providing robust and ubiquitous security support for mobile ad-hoc networks. In *Proc. of the 9th International Conference on Network Protocols (ICNP'01)*, pages 251–260, Riverside, California, USA, Nov. 2001.

[48] K. Lai, S. Marti, T. Giuli, and M. Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *The 6th annual international conference on mobile computing and networking (MobiCom'00)*, pages 255–265, 2000.

[49] S. Lee, R. Sherwood, and B. Bhattacharjee. Cooperative peer groups in nice. In *The 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (Infocom'03)*, volume 2, pages 1272–1282, Apr. 2003.

[50] W. Lee and Y. Zhang. Intrusion detection in wireless ad-hoc networks. In *The 6th annual international conference on mobile computing and networking (MobiCom'00)*, pages 275–283, 2000.

[51] J. Liu, X. Zhang, B. Li, Q. Zhang, and W. Zhu. Distributed distance estimation for large-scale networks. *Elsevier Computer Networks*, 41(2):177–193, Feb. 2003.

[52] H. Luo and S. Lu. Ubiquitous and robust authentication services for ad hoc wireless networks. Technical Report UCLA-CSD-TR-200030, 2000.

[53] H. Luo, P. Zerfos, J. Kong, S. Lu, and L. Zhang. Self-securing ad hoc wireless networks. In *The 7th IEEE Symposium on Computers and Communications (ISCC'02)*, pages 567–574, Italy, July 2002.

[54] A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1997.

[55] P. Michiardi and R. Molva. Making greed work in mobile ad hoc networks. Research Report RP-02-069, Mar. 2002.

[56] P. Michiardi and R. Molva. Prevention of denial of service attacks and selfishness in mobile ad hoc network. Research Report, Jan. 2002.

[57] E. Ngai and M. Lyu. Trust- and clustering-based authentication services in mobile ad hoc networks. In *Proc. 2nd International Workshop on Mobile Distributed Computing (MDC'04)*, pages 582–587, Tokyo, Japan, Mar. 2004.

[58] E. Ngai, M. Lyu, and R. Chin. An authentication service against dishonest users in mobile ad hoc networks. In *Proc. 2004 IEEE Aerospace Conference*, Big Sky, Montana, USA, Mar. 2004.

[59] P. Papadimitratos and Z. J. Haas. Secure routing for mobile ad hoc networks. In *Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS'02)*, San Antonio, TX, Jan. 2002.

[60] R. Perlman. An overview of PKI trust models. *IEEE Network Magazine*, 3(6):38–43, Nov. 1999.

[61] M. K. Reiter. Authentication metric analysis and design. *ACM Transactions on Information and System Security*, 2(2):138–158, May 1999.

[62] M. K. Reiter and S. G. Stubblebine. Resilient authentication using path independence. *IEEE Transactions on Computers*, 47(12):1351–1362, Dec. 1998.

[63] R. Rivest and B. Lampson. SDSI - a simple distributed security infrastructure. Working document from http://theory.lcs.mit.edu/cis/sdsi.html.

[64] F. Stajano and R. Anderson. The resurrecting duckling: Security issues for ad-hoc wireless networks. In *Proc. of the 7th International workshop on Security Protocol*, pages 172–182, Apr. 1999.

[65] W. Stallings. *Protect Your Privacy: A Guide for PGP Users*. Prentice-Gall, Inc., Uppder Saddle River, NJ, 1995.

[66] T. Wu, M. Malkin, and D. Boneh. Building intrusion tolerant applications. In *The 8th USENIX Security Symposium*, pages 79–92, Washington, D.C, Aug. 1999.

[67] H. Yang, X. Meng, and S. Lu. Self-organised network-layer security in mobile ad hoc networks. In *Proc. of the ACM workshop on Wireless security (Wise'02)*, pages 11–20, Atlanta, GA, USA, 2002.

[68] S. Yi and R. Kravets. Practical PKI for ad hoc wireless networks. Department of Computer Science, University of Illinois, Urbana-Champaign, Technical Report No. UIUCDCS-R-2002-2273, Aug. 2001.

[69] S. Yi and R. Kravets. Key management for heterogeneous ad hoc wireless networks. Department of Computer Science, University of Illinois, Urbana-Champaign, Technical Report UIUCDCS-R-2002-2290, 2002.

[70] S. Yi and R. Kravets. Moca: Mobile certificate authority for wireless ad hoc networks. In *The 2nd Annual PKI Research Workshop Program (PKI'03)*, Gaithersburg, Maryland, Apr. 2003.

[71] M. G. Zapata and N. Asokan. Securing ad hoc routing protocol. In *Proc. of the ACM workshop on Wireless security (Wise'02)*, pages 1–10, 2002.

[72] X. Zeng, R. Bagrodia, and M. Gerla. Glomosim: a library for parallel simulation of large-scale wireless networks. In *Proc. of the 12th Workshop on Parallel and Distributed Simulations (PADS'98)*, pages 154–161, Banff, Alberta, Canada, May 1998.

[73] L. Zhou and Z. J. Hass. Securing ad hoc networks. *IEEE Networks Magazine*, 13:24–30, 1999.

[74] P. Zimmermann. The official PGP user's guide. MIT Press, Cambridge, MA, June 1995.